LLMs for Information Retrieval

Main Content Derived from:

Large Language Models for Information Retrieval: A Survey, Yutao Zhu, et al.

With Modifications and Additions

Roozbeh Sanaei

Evolution of Information Retrieval

- **Early Phase:**
  - Focusing primarily on keyword matching for document retrieval.
  - Utilizes Boolean logic operators to combine query terms and retrieve documents meeting specific conditions.

- **Intermediate Development:**
  - Progressed to statistical language models, including the influential BM25 algorithm, which enhanced relevance ranking by accounting for term frequency and document length.

- **Current Advancements:**
  - Transitioned to the integration of neural models, adept at capturing complex contextual and semantic nuances, though facing challenges like data scarcity and interpretability.

  - This balances traditional efficiency with advanced semantic capabilities.

Large Language Models

Generative Likelihood Calculation: LMs are designed to calculate the likelihood of word sequences, using contextual information from preceding words to predict subsequent words.

Text Generation Capabilities: By employing word selection strategies like greedy decoding or random sampling, LMs proficiently generate natural language texts.

Text-to-Text Problem Reformulation: Recent studies have shown that many natural language processing problems can be effectively reformulated into a text-to-text format, making them solvable through text generation.

Evolution Stages of LMs:

- *Statistical Language Models:* Initially, LMs used statistical learning techniques, focusing on word prediction based on the Markov assumption.
- *Neural Language Models:* The introduction of neural networks, especially recurrent neural networks (RNNs), advanced text sequence likelihood calculation.
- *Contextualized Word Representations:* ELMo and BERT introduced the concept of learning contextualized word representations through pretraining on large corpora, followed by fine-tuning.
- *Pre-trained Language Models (PLMs):* Marked a new era with the "pre-training then fine-tuning" paradigm, leading to models like GPT-2, BART, and T5, which excel in various text generation tasks.

Scaling Law and Emergent Abilities: Increasing the scale of PLMs improves performance on downstream tasks. Large-sized PLMs exhibit emergent abilities in complex tasks, leading to the term Large Language Models (LLMs).

LLMs and Impact of Scale

- **Versatile Impact:**
  - LLMs are influential in fields like NLP, recommender systems, finance, and molecule discovery.
  - They are based on the Transformer architecture and are extensively pre-trained on diverse texts.
- **Impacts of Scale**

  As LLMs grow in size and data volume, their capabilities in language tasks improve.

  - *Emergent Capabilities:* Larger LLMs show emergent abilities in complex tasks, such as generalization and reasoning.
  - *Adaptability:* They can tackle new tasks with minimal specific instructions.
  - *In-Context Learning:* Techniques like in-context learning enhance generalization performance without extensive fine-tuning.
  - *Explainability:* Using strategies like chain-of-thought, LLMs provide step-by-step reasoning in outputs.

LLM Common Categories and Techniques

- **LLM Categories Based on Architecture:**
    - The evolution of LLMs has been marked by significant models like GPT-3 and its successors, which are based on the Transformer decoder structure and trained on various datasets.
        - ***Encoder-Decoder Models:*** These models, like T5, transform input text into vectors and then produce output texts. They solve natural language processing problems by converting them into text-to-text formats.
        - ***Decoder-Only Models:*** Exemplified by GPT models, these rely on the Transformer decoder architecture and generate word sequences using a self-attention mechanism. Models like GPT-3, GPT-J, BLOOM, OPT, and LLaMA follow this structure.
- **Advanced LLM techniques utilized in IR:**
    - ***In-Context Learning (ICL):*** An emergent ability of LLMs, allowing them to understand and respond based on the input context without additional parameter tuning. Chain-of-thought prompting can augment ICL's efficacy.
    - ***Parameter-Efficient Fine-Tuning:*** Techniques like LoRA and QLoRA aim to reduce trainable parameters while maintaining performance. While widely applied in various NLP tasks, their use in IR tasks is still emerging and represents a future research direction.
- **Challenges in Fine-Tuning LLMs:** Due to the vast number of parameters, fine-tuning LLMs for specific tasks like IR is often considered impractical, leading to the development of alternative methods like ICL and parameter-efficient fine-tuning.

Rewriter

- **Traditional IR System Process:** Users input queries, and the system returns a list of documents that match these terms.
- **Vocabulary Mismatch and Retrieval Challenges:** Queries that are often short or ambiguous lead to a vocabulary mismatch, exemplified by terms like “automobile” vs. “vehicle”, resulting in challenges in retrieving the most relevant documents.
  Role in Modern IR: Query rewriting is especially crucial in contemporary systems like conversational search.

- **Refining User Queries:**
  - The Query Rewriter's primary function is to refine or modify the initial user query to more closely match the user's information needs.
  - It's the first step in the search process, with overall effectiveness highly dependent on the quality of query rewriting.
  - A core aspect of query rewriting involves using techniques like query expansion, notably pseudo relevance feedback, to significantly improve search effectiveness in various contexts.

- **Traditional Rewriting Methods:**
  - Iterative refinement methods, including RM3, LCE, KL expansion, and relevance modeling, are used to improve query accuracy. However, they often rely on predefined rules, limiting their ability to capture the nuances of user intent.

- **Application in Specialized Contexts Demonstrating its Significance:**
  - Beyond general search scenarios, the Query Rewriter is vital in specialized retrieval areas like personalized search and conversational search, underscoring its wide-ranging importance in the field of information retrieval.

Rewriter - Rewriting Scenarios

- **Ad-hoc Retrieval:** Focuses on resolving vocabulary mismatches between user queries and document terms.
- **Conversational Search:** Adapts and refines system responses during ongoing conversational interactions.

Rewriter - Rewriting Scenarios : Adhoc Retrieval

- **Better Semantic Understanding:**
  LLMs offer a deeper understanding of language semantics, effectively capturing the meaning and context of queries, unlike traditional methods that rely on statistical term frequency analysis.

- **Broad Knowledge Base:**
  Possessing extensive knowledge, LLMs can draw from a wide range of concepts and information to generate contextually appropriate query rewrites.

- **Independence from First-pass Retrieval:**
  LLMs can rewrite queries directly based on the original user input, without relying on the results from an initial document retrieval. This approach reduces reliance on possibly irrelevant initial search results, making the process more efficient and potentially more accurate

Rewriter - Rewriting Knowledge - Supplementary Corpora

- Common query rewriting methods often need additional corpora to enrich original queries.
- LLMs contain world knowledge in their parameters but may lack specificity for certain domains.
- For specialized areas, domain-specific corpora are necessary to provide comprehensive, relevant information.

- **Two Approaches:**
  - *LLM-Only Methods:* Depend solely on the knowledge already existing within the LLM.
  - *Corpus-Enhanced LLM-Based Methods:* Utilize domain-specific corpora to enhance the LLM's capabilities.

Rewriter - Rewriting Knowledge - LLM Only methods

- **Knowledge Storage in LLMs:**
  - LLMs store extensive knowledge, ideal for query rewriting tasks.
- **Specific Methods Using LLMs:**
  - *HyDE Approach:*
    - Generates a hypothetical document based on a query.
    - Uses a dense retriever to find relevant documents.
  - *Query2Doc Approach:*
    - Creates pseudo documents through LLM prompting.
    - Expands the query with the generated content.
  - *Use of Small Language Models:*
    - A small model aids in rewriting, trained via reinforcement learning.
    - Works in conjunction with an LLM-based reader.
- **Reliance on LLM Knowledge:**
  - These methods utilize the knowledge stored in LLMs, not external corpora.
- **Comparative Performance:**
  - Demonstrates significant improvements over traditional methods like RM3.

Rewriter - Rewriting Knowledge - LLMs + Document Corpus

- LLMs can produce hallucinatory or irrelevant queries due to unfamiliarity with specific domains.
- Recent studies suggest enhancing LLM-based query rewriting with a document corpus.
- **Advantages of Incorporating a Document Corpus**
  - *Domain-Specific Knowledge:*
    - Fine-tunes the query generation for specific subjects, offering targeted and specialized IR
  - *Grounded in Facts:*
    - Integrates factual information, grounding queries in reliable knowledge.
  - *Up-to-Date Concepts:*
    - Enriches queries with contemporary information, beyond the scope of LLMs.
  - *Enhanced Relevance:*
    - Utilizes relevant documents to refine query generation, improving contextually relevant outputs.

Rewriter - Rewriting Knowledge - LLMs + Document Corpus Inclusion Methods

- ***Late Fusion Approach:*** combines LLM-only rewriting and traditional PRF retrieval results.
  - Traditional PRF methods use documents from a corpus to rewrite queries, limiting to the corpus's information.
  - LLM-only rewriting methods provide diverse external context not in the corpus.
  - A weighted fusion method on retrieval results is an effective strategy for combining these approaches.
- ***Incorporating Documents in LLM Prompts:*** a flexible method for specific functionalities in the era of LLMs.
  - *QUILL's Approach for Query Intent Classification:* QUILL demonstrates that augmenting queries with retrieval information provides LLMs with context for better understanding and proposes a two-stage distill pipeline for efficient query rewriting.
  - *LameR's Retrieve-Rewrite-Retrieve Framework:* LameR suggests a framework using BM25 for retrieval without annotated query-document pairs, showcasing LLMs' potential in zero-shot retrieval. There are no strict rules on the sequence of rewriters and retrievers in the process.
  - *InteR's Multi-Turn Interaction Framework:* InteR proposes a framework for multi-turn interactions between search engines and LLMs, allowing search engines to expand queries with LLM knowledge and LLMs to refine prompts with relevant documents from search engines.
- ***Enhancing Generative Relevance Feedback (GRF):***
  - *Generative vs. Traditional Documents:* Generative documents often present relevance and diversity but may include hallucinatory content, whereas traditional documents are viewed as reliable and factual.
  - *GRM's Proposal:* GRM introduces a novel technique called relevance-aware sample estimation (RASE).
  - *RASE Technique:* This method utilizes relevant documents from a collection to assign weights to generated documents. By applying RASE, GRM ensures that the relevance feedback is both diverse and factually accurate.

Rewriter - Generation Approach

- **Prompting Methods:**
  - The most common method for query rewriting with LLMs.
  - Uses specially designed prompts to direct the language model
  - Providing flexibility and clarity in the output generation process.

- **Fine-Tuning in Domain-Specific Areas:**
  - Fine-tuning involves training a pre-trained LLM on a specific dataset or task
  - Enhancing its effectiveness in domain-specific applications.

- **Knowledge Distillation:**
  - This method addresses the increased complexity of LLM inference due to retrieval augmentation by transferring LLM knowledge to lighter models, simplifying the process.

Rewriter - Generation Approaches - Prompting Methods

- Prompts can range from questions and incomplete sentences to specific instructions.
- The success of prompting hinges on the prompt's quality and design.

- **Zero-Shot Prompting:**
  - Zero-shot prompting instructs a model to generate text on unfamiliar topics.
  - The model uses its existing knowledge for coherent outputs.
  - Proven to be a simple and effective method for query rewriting.

- **Few-Shot Prompting:**
  - the model is given a few examples for specific tasks, also called in-context learning.
  - Serve as direct instructions to help the model adapt its language to the task.
  - Uses LLMs to write documents answering queries, utilizing example pairs from datasets like MSMARCO and NQ, focusing on single-prompt experiments.

- **Chain of Thought Prompting:**
  - Chain-of-Thought prompting uses iterative instructions or partial outputs to guide the model.
  - Aligns with conversational search's multi-turn query rewriting, refining queries through ongoing user-engine interaction.
  - *Conversational search:* Engineers implement the CoT process in conversational search by adding instructions during each turn, like "Based on all previous turns, xxx".
  - *Ad-hoc search:* In the one-round format of ad-hoc search, CoT prompting is simplified and more direct, often involving instructions like "Give the rationale before answering" to encourage deeper LLM analysis in single-turn queries.

Rewriter - Generation Approaches - Fine Tuning Methods

- Employing fine-tuning to adapt LLMs to specific domains, involving further training of a pre-trained language model (like GPT-3) on domain-specific data.
- Dataset, which may include queries, rewrites, labels, and could be enhanced with human expertise and data augmentation.
- Adjusts the model's parameters to optimize performance for domain-specific tasks.
- Resource-intensive, requiring substantial computing power.
- Fine-tuning a 770M FiD model is feasible on a single Tesla V100 32GB GPU, but a 3B T5 model needs a bigger cluster of eight Tesla V100 or A100 GPUs.

Rewriter - Generation Approaches - Knowledge Distillation Methods

- LLM-based methods improve query rewriting but face practical deployment issues due to high computational latency.
- To reduce latency, knowledge distillation is used, a prominent technique in the industry.

- **QUILL Framework's Distillation Method:**
    - QUILL proposes a two-stage distillation with three models:
        - A retrieval-augmented LLM as the 'professor'
        - A vanilla LLM as the 'teacher'.
        - A lightweight BERT model as the 'student'.
    - The professor model is trained on Orcas-I and EComm datasets, with knowledge sequentially transferred to the teacher and then the student models.
    - Outperforms simple model scaling, leading to greater improvements.
- **"Rewrite-Retrieve-Read" Framework:**
    - Involves an LLM rewriting queries, followed by a retrieval-augmented reading process.
    - Adds a small language model as a rewriter to align queries with the retriever and LLM reader's needs.
    - Begins with supervised warm-up training using pseudo data, followed by a reinforcement learning-based retrieve-then-read pipeline to optimize performance.

Rewriter - Limitations

- **Concept Drift in LLM-Based Query Rewriting:**
  - Unrelated information due to their extensive knowledge base, a phenomenon known as concept drift.
  - Can enrich the query but also risks generating irrelevant or off-target results.
  - It is essential to strike a balance in LLM-based query rewriting to ensure that the query's essence and focus are preserved while enhancing and clarifying it.
- **Strategic Approach to Expansion:**
  - It is advised to employ expansion techniques with weaker models or in scenarios where the target dataset substantially differs in format from the training corpus.
  - In other cases, expansions should be avoided to maintain the clarity of the relevance signal.

Rewriter - Specific Challenges and Approaches

- **Ranking Performance**
  - LLMs rewrite queries based on several ground-truth cases and a task description but lack awareness of the retrieval quality of rewritten queries.
  - Reinforcement learning can be utilized to adjust the rewriting process based on generation results, but more research is needed on integrating ranking results.
- **Conversational Search**
  - Efforts to improve rewriting primarily focus on ad-hoc searches.
  - Conversational search offers broader applications, adapting responses based on user history and preferences for a tailored experience.
  - Potential for simulating user behavior in conversational scenarios to generate training data is underexplored.
- **Personalized Query Rewriting**
  - Utilizes LLMs for personalized search by analyzing user-specific data such as search history, click-through behaviors, and dwell time.
  - This analysis aids in building user profiles, leading to personalized query rewriting for enhanced information retrieval.

Retriever

- **Role of Retriever in IR Systems**
  - Acts as the first-pass document filter.
  - Collects broadly relevant documents based on user queries.
  - Efficiency is crucial due to the vast amount of documents.
  - High recall is important to feed documents to the ranker, influencing search engine ranking quality.
- **Evolution of Retrieval Models**
  - Shift from statistical algorithms to neural models.
  - Neural models offer superior semantic capabilities and excel at understanding complex user intents.
- **Key Factors for Success of Neural Retrievers**
  - **Data:**
    - Large volume of high-quality training data is essential.
    - Comprehensive knowledge and accurate matching patterns are derived from quality training data.
    - Quality of search data (queries and document corpus) significantly impacts performance.
  - **Model:**
    - Strongly representational neural architecture is necessary.
    - This architecture enables effective storage and application of knowledge from training.
- **Challenges Facing Retrieval Models**
  - *User Queries:*
    - Often short and ambiguous, complicating the understanding of user intents.
  - *Document Characteristics:*
    - Typically lengthy and noisy, making it hard to encode and extract relevant information.
  - *Data Annotation:*
    - Collection of human-annotated relevance labels is time-consuming and expensive.
    - Limits the knowledge boundaries and generalization capabilities of retrievers.
  - *Model Architecture Limitations:*
    - Primarily based on BERT, with inherent limitations that restrict performance potential.
- **Utilizing LLMs to Overcome Challenges**
  - LLMs demonstrate extraordinary capabilities in language understanding, text generation, and reasoning.
  - *Research is driven towards using LLMs for:*
    - Generating search data.
    - Enhancing model

Retriever - LLM for Generating Data

- **Search Data Refinement Methods**
  - Concentrates on reformulating input queries.
  - Aims to more precisely present user intents, aligning queries closer to their underlying needs.
- **Training Data Augmentation Methods**
  - Utilizes LLMs' generation capabilities.
  - Expands the training dataset for dense retrieval models.
  - Particularly valuable in zero- or few-shot learning scenarios, where data scarcity is an issue.

Retriever - LLM for Generating Data- Search Data Refinement

- **Characteristics of Input Queries**
  - Consist of short sentences or keyword-based phrases.
  - Often ambiguous with multiple possible user intents.
  - Accurate determination of specific user intent is crucial.
- **Challenges with Document Content**
  - Documents usually contain redundant or noisy information.
  - Retrievers struggle to extract relevance signals between queries and documents.
- **Utilizing LLMs for Enhanced Text Understanding**
  - LLMs have strong text understanding and generation capabilities.
  - Offer a promising solution to the challenges posed by ambiguous queries and noisy documents.
- **Current Research Focus**
  - Primarily using LLMs as query rewriters to refine input queries.
  - Aims for more precise expression of user’s search intent.
- **Potential Future Research Avenues**
  - Exploring the use of LLMs to refine lengthy documents.
  - Enhances the effectiveness of retrieval processes.
  - Represents an open area for investigation and development.

Retriever - LLM for Generating Data - Training Data Augmentation

- **Challenge of Training Data Scarcity**
  - Neural retrieval models often face a lack of training data due to the high costs of human-annotated labels.
- **Potential of LLMs in Text Generation**
  - LLMs' excellent text generation capabilities present a potential solution to data scarcity.
- **Research Focus**
  - Developing strategies to use LLMs for generating pseudo-relevant signals and augmenting training datasets for retrieval tasks.

Retriever - LLM for Generating Data - Training Data Augmentation
Why do we need data augmentation?

1. **Limited Scope of Labeled Data**:
   - Datasets like MS MARCO, while extensive, are confined to the domains they represent. They provide labeled data suitable for training models in a supervised manner, but this limits the model's ability to generalize to unseen domains.
2. **Generalization Challenges**:
   - Retrieval models trained on specific domains often struggle to perform well with out-of-distribution data. For example, a model trained on web document retrieval may not perform well in biomedical information retrieval without additional training.
3. **Cost of Annotation**:
   - Acquiring domain-specific labeled data for different applications is often costly and time-consuming.
   - This makes it impractical to retrain models from scratch for every new domain or application.
4. **Emerging Learning Paradigms**:
   - There is a growing interest in zero-shot and few-shot learning models, which can operate effectively with minimal or no labeled data from the target domain.
   - These models rely on data augmentation techniques to enhance their ability to adapt to new domains.
5. **Enhancing Model Effectiveness**:
   - Data augmentation methods, such as synthesizing new data examples or modifying existing ones, help in expanding the diversity of training examples.
   - Enhances the model's robustness and its ability to handle different types of queries and documents in practice.

Retriever - LLM for Generating Data - Training Data Augmentation
Pseudo query generation

1. In IR, it's relatively straightforward to accumulate a large number of documents. The real challenge, however, lies in obtaining real user queries that align with these documents. Gathering authentic user queries and labeling documents as relevant or not is both difficult and expensive. This process requires significant resources and time.
2. **Utilizing LLMs for Data Generation**: Given the advanced text generation capabilities of Large Language Models (LLMs), researchers have proposed using these models to generate synthetic data for training. This includes pseudo queries and relevance labels.
3. **Methods of Data Augmentation**:
   - **Pseudo Query Generation**: LLMs are used to generate queries that might plausibly be asked about given documents in the dataset. This helps in creating more training data for models.
   - **Relevance Label Generation**: Instead of traditional binary labels, LLMs can generate probabilities (soft relevance labels) based on the likelihood of a query being relevant to a given document. This nuanced approach can improve the model's understanding of query-document relevance.
4. **Specific Implementations and Tools**:
   - Examples include "InPairs", using models like GPT-J to generate relevant queries, and "PROMPTAGATOR" which uses FLAN for generating queries and applies round-trip filtering.
   - Techniques range from fixed LLM tuning to soft prompt tuning, adapting the training approach based on the augmentation method used.

Retriever - LLM for Generating Data - Training Data Augmentation
Relevance label generation

- **Context of Use**:
  - While there might be an ample collection of questions available, there is often a scarcity of relevance labels that link these questions with appropriate supporting evidence (passages).
- **Relevance Label Generation via LLMs**:
  - Leveraging the capabilities of LLMs for relevance label generation is seen as a viable method to augment the training data for retrieval systems.
- **Implementation Example (ART Method)**:
  - **Step 1**: Retrieve the top-relevant passages for each question.
  - **Step 2**: Use an LLM to calculate the generation probabilities of the question based on these passages.
  - **Step 3**: Normalize these probabilities to serve as soft relevance labels.
  - **Purpose**: These soft relevance labels are used for training retrievers, providing a graded measure of relevance rather than a binary one.
- **Comparative Analysis**:
  - **Comparative Factors**: Includes the number of examples, the generator used (specific LLMs), the type of synthetic data produced, the filtering method applied, and the fine-tuning status of the LLMs.
  - **Utility of Table**: Helps in understanding the diversity of approaches and the effectiveness of different configurations in relevance label generation.
- **Significance**:
  - This method provides a more nuanced and scalable way to create training data, crucial for improving the performance of retrieval systems in real-world applications where direct relevance labels may not be abundantly available.

Retriever - LLM for enhanced model architecture

1. **Leveraging the capabilities of LLMs:**
   - Large Language Models (LLMs) exhibit enhanced text encoding and decoding capabilities, allowing for a more precise understanding of queries and documents compared to smaller-sized models.
2. **Foundation for retrieval models:**
   - Researchers have utilized LLMs as a basis for developing advanced retrieval models, broadly categorized into:
     i. Dense retrievers
     ii. Generative retrievers

Retriever - LLM for enhanced model architecture - Dense Retriever

Researchers leverage LLMs as retrieval encoders and explore how the size of these models affects their performance. They discover that bigger models usually perform better.

General Retriever

- **Text Embedding Importance**: The effectiveness of retrievers primarily relies on the capability of text embedding.
- **Evolution of Embedding Models**: In the era of LLMs, OpenAI made pioneering efforts by unsupervised pre-training of text embedding models ranging from 300M to 175B parameters using adjacent text segments as positive pairs.
- **Experiments and Findings**: Conducted experiments on MS MARCO and BEIR datasets show that larger models enhance performance in unsupervised learning and transfer learning for text search tasks.
- **Cost Challenges and Solutions**: Pre-training LLMs from scratch is cost-prohibitive for most; studies use pre-trained LLMs to initialize the bi-encoder of dense retrievers (e.g., GTR with T5 models, RepLLaMA with LLaMA model).

Task-aware Retriever

- **Integration of Specific Instructions**: Retrieval performance can be greatly enhanced by integrating task-specific instructions into the retrieval process.
- **TART Model Implementation**: TART introduces a task-specific instruction before the question, which includes details about the task's intent, domain, and desired retrieved unit, effectively capturing the user's search intents.
- **Model Efficiency**: Begins with fine-tuning a TART-full model with a cross-encoder architecture, followed by a TART-dull model learned by distillating knowledge from the TART-full, optimizing the efficiency of retrievers.

Retriever - LLM for enhanced model architecture - Generative Retriever

- **Traditional IR System Paradigm**:
  - **"Index-Retrieval-Rank" Paradigm**: Traditional information retrieval (IR) systems follow this paradigm to locate relevant documents based on user queries, which has been effective in practice.
  - **Three Separate Modules**: Consists of the index module, the retrieval module, and the reranking module.
  - **Optimization Challenges**: Optimizing these modules collectively can be challenging, often resulting in sub-optimal retrieval outcomes.
  - **Storage Demands**: This paradigm requires additional space for storing pre-built indexes, which burdens storage resources.
- **Model-Based Generative Retrieval Methods**:
  - **Shift from Traditional Paradigm**: Recently, methods have moved away from the traditional paradigm and instead use a unified model to directly generate document identifiers (DocIDs) relevant to the queries.
  - **Storage Efficiency**: The knowledge of the document corpus is stored within the model parameters, eliminating the need for additional index storage space.
  - **Implementation via LLMs**: Existing methods explore generating document identifiers through fine-tuning and prompting of Large Language Models (LLMs).

Retriever - LLM for enhanced model architecture - Generative Retriever
Fine-tuning LLMs

- **Utilization of World Knowledge**: LLMs contain a vast amount of world knowledge, making them intuitive choices for building model-based generative retrievers.
- **DSI Methodology**: DSI (Document Semantic Identifier) fine-tunes pre-trained T5 models on retrieval datasets to encode queries and decode document identifiers directly for retrieval.
- **Semantically structured identifiers**: DSI explores multiple techniques and finds that constructing semantically structured identifiers yields optimal results.
- **Semantic Structuring**:
  - **Hierarchical Clustering**: Documents are grouped according to their semantic embeddings.
  - **Assignment of DocIDs**: Each document is assigned a semantic DocID based on its hierarchical group.
- **Validation of DocIDs**:
  - **Trie Construction**: A trie is constructed using all DocIDs to ensure that decoded identifiers are valid and represent actual documents in the corpus.
  - **Constraint Beam Search**: Used during the decoding process to ensure the output DocIDs are accurate.
- **Scaling Law Observance**: The approach observes that the scaling law, which suggests larger LMs lead to improved performance, also applies to generative retrievers.

Retriever - LLM for enhanced model architecture - Generative Retriever

Prompting LLM

- **Prompting Large Language Models (LLMs)**: LLMs, such as those in the GPT series, have shown the ability to directly generate relevant web URLs in response to user queries. This capability is enhanced through fine-tuning and providing a few in-context examples.
- **Training Exposure**: The unique ability of LLMs to serve as generative retrievers is attributed to their extensive training on diverse HTML resources, which enables them to directly generate document identifiers.
- **LLM-URL Model**: A specific model, referred to as LLM-URL, utilizes the GPT-3 text-davinci-003 model to generate candidate URLs. This model incorporates regular expressions to filter and extract valid URLs from the candidates for document retrieval.

Retriever - Future Directions

- **Reducing Latency**
  - LLMs, due to their massive parameters, often exhibit high latency, impacting practical applications for real-time search engine responses.
  - Potential solutions include transferring LLM capabilities to smaller models and exploring quantization techniques.
- **Simulating Realistic Queries for Data Augmentation**
  - High latency limits online applications, so LLMs are used offline to augment training data, which may introduce noise due to misalignment with real user queries.
  - Techniques like reinforcement learning could improve the simulation of realistic queries, enhancing the training data quality.
- **Incremental Indexing for Generative Retrieval**
  - Face challenges in updating indexes when new documents are added due to static LLM parameters and high fine-tuning costs.
  - Investigating methods for constructing incremental indexes could allow for more efficient updates in generative retrievers.
- **Supporting Multi-Modal Search**
  - While LLMs enhance text-based retrieval, web pages often contain multi-modal information (texts, images, audios, videos).
  - A potential approach is integrating multi-modal large models like GPT-4, although this could increase deployment costs.
  - Combining LLMs with existing multi-modal retrieval models could leverage their language understanding capabilities across different content types.

ReRanker

Rerankers act as second-pass filters to reorder document lists retrieved by the primary retriever based on query-document relevance.

- **Distinction of Objective:**
  Unlike the Retriever, which aims for a balance of efficiency and effectiveness, the Reranker emphasizes the quality of document ranking.

- **Paradigms of LLM-Based Reranking**
  - **Supervised Rerankers:** Utilize LLMs with supervised learning techniques to enhance reranking.
  - **Unsupervised Rerankers:** Employ LLMs without labeled data for reranking tasks.
  - **Training Data Augmentation:** Use LLMs to generate additional training data to improve reranking performance.

ReRanker

- **Utilizing LLMs as Supervised Rerankers**
  - LLMs are employed to rerank documents based on their relevance to a query, using labeled training data that provides examples of relevant and non-relevant documents.
  - **Challenges:** The need for high-quality, labeled training data and the potential for overfitting to the training data characteristics, which might not generalize well to unseen queries.

- **Utilizing LLMs as Unsupervised Rerankers**
  - This approach uses LLMs to rerank documents without relying on labeled data.
  - Techniques might involve semantic similarity measurements between queries and documents.
  - **Challenges:** Difficulty in capturing query-document relevance accurately without explicit relevance signals from labeled data, leading to potentially lower performance compared to supervised methods.

- **Utilizing LLMs for Training Data Augmentation**
  - LLMs generate synthetic data that mimics real query-document interactions, used to augment existing datasets and improve the model’s exposure to diverse scenarios.
  - **Challenges:** Ensuring that the synthetic data is realistic and relevant, avoiding the introduction of biases or irrelevant information which could degrade model performance.

ReRanker - Supervised Fine-Tuning of LLMs

- To adapt pre-trained LLMs to reranking tasks by training them on task-specific datasets that include both relevance and irrelevance signals.
- MS MARCO passage ranking dataset, which provides real-world examples of query-document pairs along with relevance annotations.
- Fine-tuning allows LLMs to adjust their parameters specifically for measuring query-document relevance, which they were not originally trained to do during pre-training.
- **Categories of Supervised Rerankers Based on Model Structure**
  - **Encoder-Only Models**
    - These models use a single encoder architecture to process input and produce output, focusing on understanding and encoding the content effectively.
    - Typically used when the task involves understanding text or context without the need for generating new text.
  - **Encoder-Decoder Models**
    - These models use an encoder to understand the input and a decoder to generate the output, suitable for tasks requiring a transformation of input into a different output.
    - Useful in reranking tasks where both understanding the query and generating a response (ranking) are necessary.
  - **Decoder-Only Models**
    - These models rely solely on a decoder, often pre-trained with a masked language modeling objective to generate text based on the partial inputs.
    - Can be adapted for reranking by using the decoder to generate scores or classifications based on the input query and document content.

ReRanker - Encoder-Based

- **Encoder-Based Rerankers**
  - These rerankers have significantly advanced the application of LLMs to document ranking tasks, showing how pre-trained models can be effectively fine-tuned.
  - BERT, known for its ability to be adapted to various tasks including relevance prediction for document ranking.
- **MonoBERT:** ***A Representative Approach***
  - MonoBERT fine-tunes BERT for reranking by processing a query-document pair formatted as "[CLS] query [SEP] document [SEP]". The relevance score is calculated from the representation of the "[CLS]" token.
  - The relevance score is derived by passing the "[CLS]" token representation through a linear layer, with the model being optimized using cross-entropy loss to assess the accuracy of relevance predictions.

ReRanker - Encoder-Decoder

- These rerankers view document ranking as a generation task, using encoder-decoder models to optimize the ranking process.
- Typically, these models are fine-tuned to generate a binary token—like "true" or "false"—indicating the relevance of a document to a query.
- **Example Models and Approaches**
  - *T5 Model:* Fine-tuned to classify query-document pairs as relevant or irrelevant by generating "true" or "false" tokens. During inference, the relevance score is calculated using softmax on these tokens to determine the probability of the "true" token.
  - *Multi-view Learning with T5:* Combines token generation for relevance classification with generating a query conditioned on the provided document, enhancing context understanding.
  - *DuoT5 Model:* Considers a pair of documents with a query and determines which document is more relevant by generating a "true" or "false" token. Uses global aggregation to compute relevance scores across multiple documents, enhancing comparative analysis.
- **Challenges and Limitations**
  - *Generative Loss vs. Ranking Loss:* While generative methods have shown success, they are not always optimal due to the generation of textual tokens instead of direct numerical relevance scores. Ranking losses like RankNet may be more effective for optimizing rerankers.
  - *RankT5 Model:* Addresses these issues by directly calculating numerical relevance scores and optimizing with "pairwise" or "listwise" ranking losses, potentially offering a more direct and effective measurement of document relevance.
- **Potential Enhancements**
  - *Scaling Model Size:* Performance improvements might be achievable by replacing the base-sized T5 model with a larger variant, leveraging increased computational power and model capacity to enhance ranking accuracy.

ReRanker- Decoder Only

Recent attempts have focused on using decoder-only models like LLaMA for document reranking by fine-tuning them specifically for this task.

RankLLaMA: Formats query-document pairs into prompts such as "query: {query} document: {document} [EOS]" and utilizes the representation of the last token to calculate relevance scores.

RankingGPT: Employs a two-stage training process to adapt LLMs for document ranking:

- *Stage 1:* Continuously pretrains LLMs using a large corpus of relevant text pairs from web resources to enhance the model's ability to generate queries related to input documents.
- *Stage 2:* Fine-tunes the model using high-quality supervised data and carefully designed loss functions to optimize text ranking performance.

Rank-without-GPT: Proposes an alternative by using a listwise reranking approach, which directly outputs a reranked list of documents. This method addresses the limitations of pointwise training datasets by using the ranking outputs from existing ranking systems as training data for a listwise reranker, focusing on achieving a more comprehensive assessment of document lists rather than individual document relevance.

ReRanker- Decoder Only

Challenges and Limitations

- *Data Adequacy for Listwise Reranking:* Pointwise datasets like MS MARCO, which primarily contain binary relevance labels, are often insufficient for training listwise rerankers that require more detailed information about the relative rankings of multiple documents.
- *Training Data Acquisition:* The use of existing ranking systems' outputs as "gold" rankings introduces dependencies on the accuracy and biases of these systems, which could impact the training and final performance of the reranker.

Innovations and Future Directions

- *Expanding Model Training:* The approaches illustrate a move towards more dynamic and context-aware reranking strategies by incorporating broader and more complex training mechanisms.
- *Potential for Enhanced Accuracy:* The shift towards listwise reranking and the integration of complex training stages may yield rerankers that are better tuned to the nuances of real-world document ranking scenarios.

ReRanker - Unsupervised

- As the size of Large Language Models (LLMs) scales up (e.g., exceeding 10 billion parameters), it becomes increasingly difficult to fine-tune the reranking model.
- Addressing this challenge, recent efforts have attempted to prompt LLMs to directly enhance document reranking in an unsupervised way.
- In general, these prompting strategies can be divided into three categories:
  - Pointwise methods
  - Listwise methods
  - Pairwise methods

ReRanker - Unsupervised - Pointwise

- Measures relevance between a query and a single document.
  - Relevance Generation
  - Query Generation

- **Relevance Generation**
  - Utilizes a binary labeling system ("Yes" or "No") to determine relevance.
  - *Relevance score:*

$$f(q,d) = \frac{\exp(SY)}{\exp(SY) + \exp(SN)}$$

  - Zhuang et al. [147] expanded the label options to include "highly relevant", "somewhat relevant", and "not relevant".

ReRanker - Unsupervised - Pointwise - *Query Generation*

Score calculated by averaging log-likelihoods of generating actual query tokens from the document:

$$\text{score} = \frac{1}{|q|} \sum_i \log p(q_i | q_{<i}, d, P)$$

- Where $|q|$ is the token number of query, d denotes the document, and P the provided prompt.
- Notable performance in zero-shot document reranking observed in some LLMs (e.g., T0, LLaMA).

2. **Prompt Optimization**
   - The effectiveness of these methods heavily relies on the construction of the prompt.
     i. *Co-Prompt [150]:* Proposes a method for optimizing prompt generation in reranking tasks.
     ii. *PaRaDe [151]:* Introduces a difficulty-based method for selecting few-shot demonstrations to enhance prompt performance.
3. **Limitations**
   - Pointwise methods depend on access to the output logits from LLMs.
   - Not applicable to closed-sourced LLMs, where API results do not include logits.

ReRanker - Unsupervised - Listwise

Listwise methods aim to directly rank a list of documents by inserting the query and document list into the prompt of an LLM.

- **Challenge Due to LLM Limitations:**
  - The limited input length of LLMs makes it infeasible to insert all candidate documents into the prompt.
  - To manage this, a sliding window strategy is employed to rerank a subset of documents in each iteration, ranking from back to front.
- **Performance Insights:**
  - While listwise methods have shown promising results, their effectiveness varies:
    - GPT-4-based methods are competitive.
    - Smaller models like FLANUL2 with 20B parameters often underperform, producing few usable results compared to many supervised methods.
- **Sensitivity to Document Order:**
  - The performance of listwise methods is highly sensitive to the order of documents in the prompt.
  - Random shuffling of document order leads to poorer performance, sometimes even worse than traditional methods like BM25, indicating positional bias.
  - To counteract positional bias, Tang et al. introduced a permutation self-consistency method. This involves shuffling the list in the prompt and aggregating the results to achieve a more accurate and unbiased ranking.

ReRanker - Unsupervised - PairWise

- Pairwise methods involve LLMs receiving a prompt that includes a query and a pair of documents.
- The LLM is then instructed to generate the identifier of the more relevant document within the pair.
- **Aggregation Method**:
  - The AllPairs method is typically used to aggregate relevance scores across all possible document pairs, determining a final relevance score for each document.
- **Utilization of Sorting Algorithms**:
  - To enhance the efficiency of ranking, sorting algorithms like heap sort and bubble sort are used.
  - These algorithms leverage efficient data structures to selectively compare document pairs and prioritize the most relevant documents, which is beneficial for top-k ranking.
- **Performance Insights**:
  - Pairwise methods have demonstrated state-of-the-art performance in standard benchmarks using moderately sized LLMs (e.g., Flan-UL2 with 20B parameters), which are smaller than those typically used in listwise methods (e.g., GPT3.5).
- **Challenges with Time Complexity**:
  - Despite their effectiveness, pairwise methods suffer from high time complexity.
  - **Innovative Solution - Setwise Approach**:
    - To improve efficiency, a setwise approach has been introduced, which compares a set of documents at each step rather than just pairs.
    - This method reduces the total number of comparisons needed and speeds up the sorting process by allowing algorithms like heap sort to handle more than two documents at a time.

ReRanker - Unsupervised - Overall Comparison

- **Pointwise Methods (Query Generation and Relevance Generation)**:
    - *Advantages:* Lower time complexity and the ability to perform batch inference.
    - *Disadvantages:* Generally lower performance compared to other ranking methods.

- **Listwise Methods**:
    - *Advantages:* Significant performance improvements, especially when using GPT-4.
    - *Disadvantages:* High costs associated with API usage and issues with reproducibility.

- **Pairwise Methods**:
    - *Advantages:* Competitive results even with smaller models like FLAN-UL2 (20B).
    - *Disadvantages:* Low efficiency due to the need to compare a large number of document pairs.

ReRanker - Limitations

Cost and Efficiency: Minimizing the number of calls to LLM APIs is a significant challenge due to the associated costs and efficiency concerns.

Adaptability Challenges: Most research currently focuses on using LLMs for open-domain datasets, such as MSMARCO , and relevance-based text ranking tasks.The adaptability of LLMs to in-domain datasets and non-standard ranking datasets is an area requiring further exploration to understand how well these models perform across diverse data types and tasks.

ReRanker - Future Directions

- **Enhancing Online Availability**
    - Many LLMs, due to their large size, pose challenges in online deployment.
    - Costs associated with LLM API calls necessitate exploring effective methods like model distillation to improve online applicability.
- **Improving Personalized Search**
    - Current LLM-based reranking methods are often limited to ad-hoc tasks.
    - Incorporating user-specific information, such as search history, allows LLMs to construct accurate user profiles and deliver personalized reranking, enhancing user satisfaction.
- **Adapting to Diverse Ranking Tasks**
    - Beyond document reranking, LLMs could also excel in tasks like response, evidence, and entity ranking within the universal information access system.
    - Specialized methodologies like instruction tuning can broaden LLMs' capabilities across various ranking tasks, representing a promising research direction.

Reader

The Reader has significantly evolved alongside the rapid development of large language model technologies. It is capable of comprehending real-time user intent and generating dynamic responses based on retrieved texts, transforming how IR results are presented.

- **Intuitive Presentation of Information:**
  Unlike the traditional method of presenting a list of documents, the Reader organizes answer texts in a way that simulates natural human information access.

- **Credibility Enhancement with References:**
  To improve the credibility of its generated responses, the Reader effectively integrates references into these responses.

Reader

- ***Conclusive Passage Presentation:*** Directly presents users with the most relevant passages, simplifying information consumption.
- ***Comparison with Traditional Methods*****:** Offers a more straightforward approach than analyzing ranked lists of documents.
- ***Enhanced Accuracy and Richness*****:** Repeated interactions with documents improve the accuracy and detail of the answers generated.

- **Implementation Strategies for Reader Modules**
  - *Naive Strategy*:
    - Heuristically provides LLMs with documents relevant to user queries or previously generated texts.
    - Restricts LLMs to merely collecting documents without engaging actively with the IR systems.
  - *Proactive Interaction with Search Engines*:
    - Training LLMs to formulate queries and interact with search engines independently.
    - *Passive Readers*: Limited to using documents provided based on external queries.
    - *Active Readers*: Capable of independently generating queries for more dynamic interaction.

- **Document Handling by LLMs**
  - *Compression Modules*: Proposed solutions for managing lengthy documents that are impractical for direct LLM input.
    - *Extractive Compression*: Selects important portions of text to present to LLMs.
    - *Abstractive Compression*: Summarizes lengthy documents into concise, manageable texts for LLMs. Facilitates better understanding and efficient answer generation by LLMs.

Reader - Passive Reader

- Utilizing documents retrieved based on user queries or previously generated texts as inputs for LLMs.
- Serve as passive recipients, creating passages from the supplied documents without engaging further with the IR systems.
- LLMs and IR systems operate independently within this approach.
- Categorized according to the frequency of retrieving documents for LLMs.

Reader - Passive Reader

- **Initial Document Retrieval for Answer Prediction:**
  - **REALM**:
    - Adopts direct attention of document contents to the original queries.
    - Predicts answers using masked language modeling.
  - **RAG**:
    - Similar to REALM but uses generative language modeling paradigm.
- **Leveraging Large Language Models (LLMs):**
  - **Recent Approaches (REPLUG and Atlas)**:
    - Improved older methods by utilizing GPTs, T5s, and LLaMAs.
    - Focus on fine-tuning LLMs on QA tasks for enhanced performance.
- **Resource-Efficient Strategies:**
  - **Using Prompting with LLMs:**
    - Due to limited computing resources, many methods opt for prompting LLMs.
    - Allows for the use of larger language models without extensive computational overhead.
- **Enhancing Answer Quality:**
  - Several approaches train or prompt LLMs to generate citations or notes alongside answers.
  - Aims to improve LLMs' understanding and relevance assessment of retrieved passages.
  - Some methods evaluate the importance of each retrieved reference through policy gradients to enhance answer generation.
- **Instruction Tuning for Knowledge-Based Outputs:**
  - Researchers explore instruction tuning to improve LLMs' ability to generate conclusive passages based on retrieved knowledge.

Reader - Periodic Retrieva Reader

Initial Retrieval Limitations:

- Utilizing only the initially retrieved references, as seen with once-retrieval readers. Often falls short when generating detailed responses.
- For instance, an initial query about "Barack Obama" might require additional context, such as his university background, which may not be captured in the initial search results.

Periodic Retrieval Techniques:

- **RETRO** : Incorporates a method where additional documents are periodically collected. This system uses cross-attention within the Transformer model, integrating generating texts with newly retrieved references, rather than embedding these references directly into the input.

- **RALM** : Similar to RETRO, this approach triggers a retrieval for every set number of tokens generated, ensuring that fresh references inform the ongoing text generation.

- **IRCoT:** Focuses on retrieving documents at the end of every generated sentence, aiming to maintain semantic coherence and reduce the noise from irrelevant documents.

- **Comprehensive Passage Retrieval:** These methods go further by using the entire generated passages as context for retrieving new references. This iterative process continues until reaching a predefined limit, enhancing the relevance and comprehensiveness of the references used in generation.

Trade-offs and Efficiency

- While these advanced retrieval methods provide more accurate and contextually relevant answers, they inherently increase computational demands due to the continuous retrieval and integration processes.

Reader - Active Reader

Self-Ask and DSP Approaches

Trigger Mechanism: Utilizes few-shot prompts to enable LLMs to autonomously issue search queries when necessary.

Example Scenario: For a question like "When was the existing tallest wooden lattice tower built?", the LLM may first query "What is the existing tallest wooden lattice tower" to collect initial data.

Iterative Querying: After gathering basic information, the LLM continues to query for more details until it can provide a final answer without further queries.

MRC Approach

Multi-Patch Reasoning:
Prompts LLMs to explore multiple reasoning chains rather than constructing a singular reasoning pathway.

Combination of Answers:
Integrates all generated responses to formulate a comprehensive answer using the capabilities of LLMs.

Reader - Compressor

Length Limitations

- Large Language Models (LLMs) like LLaMA and Flan-T5 typically support input lengths of 4,096 or 8,192 tokens. This limitation poses challenges when dealing with longer documents retrieved by upstream Information Retrieval (IR) systems, which often exceed these token limits.

Conventional Solution

- Current approaches aggregate answers from each retrieved document to form a final answer. However, this method often overlooks the relationships between the different passages.

Direct Compression Strategies

- A more direct solution involves compressing lengthy documents into shorter inputs or dense vectors, aiming to maintain essential information while reducing token count.

Compression Techniques for LLMs

Extractive Compression

- **LeanContext**: Uses reinforcement learning to select the top K sentences most relevant to the query.
- **RECOMP**: Utilizes answer matching probabilities to tune compression, focusing on sentences that align closely with correct answers.
- **FILCO**: Employs "hindsight" methods, aligning predicted sentence importance with actual relevance determined by correct answers.

Abstractive Compression

- Unlike extractive methods, abstractive techniques summarize documents into concise overviews, reducing the potential loss of intent across multiple references.
- Examples include leveraging models like GPT-3.5-turbo to create datasets for abstractive compression training for models like MT5.

Reader - Analysis

The integration of retrieval systems with Large Language Models (LLMs) has advanced significantly, prompting researchers to examine how these retrieval-augmented approaches impact the functionality and effectiveness of LLMs.

Influence of Reference Positioning (Liu et al) : They find that the position of a relevant reference (either at the beginning or the end) significantly impacts generation performance, highlighting the need for a ranking module to optimize the order of retrieved information.

Awareness of Knowledge Boundaries (Ren et al.) : Their show that retrieval-augmented generation strategies enhance LLMs' awareness of their knowledge limitations, allowing for more informed and accurate responses.

Integration Strategies (Liu et al.) : They analyze various methods of combining retrieval systems with LLMs, including concatenation of all references for answer generation and post-fusion aggregation of answers from each reference. This study also explores hybrid approaches that blend these strategies.

Trade-offs in Retrieval-Augmentation (Aksitov et al.) : They demonstrate a trade-off between attribution and fluency; as more references are incorporated, the accuracy of answer attribution improves, but the fluency of the generated text suffers.

Impact of Excessive Retrieval (Mallen et al.) : They argue that excessive retrieval can degrade question-answering performance because LLMs often possess sufficient inherent knowledge for common topics.

They suggest a strategy of selective retrieval based on the query's entity popularity, which enhances both the effectiveness and efficiency of retrieval-augmented generation.

Reader - Applications

- **Clinical, Medical, and Financial QA**: Researchers have applied retrieval-augmented generation to enhance Large Language Models (LLMs) with external knowledge for domain-specific applications in clinical QA, medical QA, and financial QA.
- **ATLANTIC (Atlas in Science Domain)**: Adaptation of Atlas to the scientific domain to create a specialized science QA system.
- **Privacy Protection Techniques**: Implementation of federated learning techniques such as multi-party computation to perform personal retrieval-augmented generation with enhanced privacy measures.
- **RETA-LLM**: A framework that simplifies complex generation tasks by breaking them into simpler modules within the reader pipeline:
    - **Query Rewriting Module**: Refines query intents.
    - **Passage Extraction Module**: Aligns reference lengths to accommodate LLM limitations.
    - **Fact Verification Module**: Ensures generated answers do not contain fabricated information.

Reader - Applications

Effective Query Reformulation:
Reformulating user queries accurately to fetch relevant information from large databases.

Optimal Retrieval Frequency:
Determining how often the system should retrieve data to balance performance and relevance.

Correct Document Comprehension:
Ensuring the system understands the content of the documents it retrieves.

Accurate Passage Extraction:
Extracting the most relevant passages from documents accurately.

Effective Content Summarization:
Summarizing the content succinctly and effectively for the end user.

Reader - Future Directions

- **Improving Reference Quality for LLMs**
  - Current methods feed entire documents as references, which can introduce noise due to irrelevant content.
  - Exploring techniques to extract relevant snippets from documents could enhance the performance of retrieval-augmented generation.
- **Enhancing Answer Reliability of LLMs**
  - Incorporating retrieved references helps mitigate LLMs' tendency to "hallucinate," but their reliance on these materials during query answering is still uncertain.
  - Studies indicate that LLMs can still provide unfaithful answers even with additional references, suggesting a need to assess and improve the influence of these references on the generation process to enhance the credibility of reader-based IR systems.

Search Agents

- **Introduction of Intelligent Agents**
    - Focus on mimicking human browsing patterns.
    - Enhances model capabilities for complex retrieval tasks.

- **Capabilities Empowered by LLMs**
    - Advanced natural language understanding and generation.
    - Autonomous search, interpretation, and synthesis from diverse sources.

- **Methods to Implement Intelligent Agents**
    - **Pre-defined Pipeline Approach**
        - Mimics user behaviors on the web through sub-tasks.
        - Static nature may struggle with complex user behavior sequences.
        - Limited interaction with real-world environments.
    - **Autonomous Exploration Approach**
        - LLMs freely explore the web and interact autonomously.
        - Decisions based on feedback from the environment or humans.
        - Higher flexibility, resembling human behavior more closely.

Search Agents - Static vs Dynamic Agents

- **Static Agents**
  - Designs a static system to browse the web and synthesize information.
  - Breaks down the information-seeking process into multiple subtasks.
  - Utilizes a pipeline containing various LLM-based modules assigned to different subtasks.
  - **LaMDA**
    - Early example of a static agent.
    - Consists of Transformer-based models specialized for dialogue.
    - Pre-trained on large-scale data, followed by strategic fine-tuning to enhance dialogue quality, safety, and groundedness.
    - Integrates external IR systems for factual grounding.
- **Dynamic Agents**
  - **WebGPT**
    - Trains LLMs to use search engines automatically through a reinforcement learning framework.
    - Constructs a simulated environment for GPT-3 models.
    - Employs special tokens for actions like querying, scrolling through rankings, and quoting references on search engines.
    - Enhances the reliability and real-time capability of text generation using search engines.
  - **Study on Chinese Question Answering**
    - Applies search-based question answering techniques to the Chinese language.
  - **Benchmarks for Web-Based Agents**
    - **WebShop:** Provides a scalable, interactive environment for online shopping, focusing on language understanding and decision-making.
    - **ASH Prompting :** Enhances LLMs' capabilities in the WebShop benchmark by processing raw observations into actionable insights and predicting subsequent actions based on past interactions.
  - **Challenges with Dynamic Agents**
    - **Limited Research:** Dynamic search agents are less studied compared to static ones, with ongoing research gaps.
    - **Real-Time Fact-Checking :** Some agents lack mechanisms for real-time verification against authoritative sources, raising concerns about misinformation.
    - **Bias in Training Data:** Training on Internet data may perpetuate existing biases, leading to biased or offensive outputs and unethical content collection.
    - **Privacy and Data Security:** Processing user queries raises concerns about privacy and security, particularly with sensitive or personal information.

Search Agents - Future Directions

- **Enhancing Trustworthiness**
  - Ensuring the validity of retrieved documents is crucial to prevent unfaithful information and reduce LLMs' “hallucination” issues.
  - Even if the gathered information is of high quality, it remains unclear whether it is actually used for synthesizing responses.
  - Enabling LLMs to autonomously validate documents, assessing their credibility and accuracy, can improve trustworthiness.

- **Mitigating Bias and Offensive Content**
  - Biases and offensive content in LLM outputs stem from biased training data and low-quality web information.
  - Addressing this requires a multi-faceted approach, including improvements in training data, algorithmic adjustments, and continuous monitoring for bias and inappropriate content.

Text Generation Evaluation

- **Traditional Evaluation Metrics:** Comparing the retrieval results of IR models with ground-truth (relevance) labels:
    - Precision
    - Recall
    - Mean Reciprocal Rank (MRR)
    - Mean Average Precision (MAP)
    - Normalized Discounted Cumulative Gain (nDCG)
- **Limitations of Previous Evaluation Metrics:**
    - **1. Dependency on Lexical Matching:**
        - **Methods such as BLEU and ROUGE:** These methods primarily evaluate the quality of generated outputs based on n-gram matching.
        - **Issues with Current Approach:** This approach cannot account for lexical diversity and contextual semantics. As a result, models may favor generating common phrases or sentence structures rather than producing creative and novel content.
    - **Insensitivity to Subtle Differences:**
        - **Impact on Evaluation:** Existing evaluation methods may be insensitive to subtle differences in generated outputs.
        - **Example Issue:** For example, if a generated output has minor semantic differences from the reference answer but is otherwise similar, traditional methods might overlook these nuanced distinctions.
    - **Lack of Ability to Evaluate Factuality:**
        - **Problem of "Hallucination" in LLMs:** LLMs are prone to generating "hallucination" problems.
        - **Challenges in Identifying Non-factual Content:** The hallucinated texts can closely resemble the oracle texts in terms of vocabulary usage, sentence structures, and patterns, while having non-factual content.
        - **Potential Solutions:** Existing methods are hard to identify such problems, while the incorporation of additional knowledge sources such as knowledge bases or reference texts could potentially aid in addressing this challenge.

Bias

- **Potential for Non-Factual Texts:**

  As LLMs may hallucinate and generate non-factual texts, the increasing number of LLM-generated contents also brings worries that these contents may provide fictitious information for users across IR systems.

  Preference of IR Systems for LLM-Generated Documents:

  More severely, researchers have shown that some modules in IR systems, such as retriever and reranker, especially those based on neural models, may prefer LLM-generated documents since their topics are more consistent and the perplexity of them is lower compared with human-written documents.

- **Source Bias:** This phenomenon is referred to as the “source bias” towards LLM-generated text.
- **Building Bias-Free Systems:** It is challenging but necessary to consider how to build IR systems free from this category of bias.