Applications of Information Extraction for Information Retrieval

Binu P Chacko
Prajyoti Niketan College, Pudukad, Thrissur

Abstract—**Large language models have emerged as transformative forces across various research fields, such as natural language processing (NLP), recommender systems, finance, and molecule discovery. They are primarily based on the Transformer architecture and undergo extensive pre-training on diverse textual sources, including web pages, research articles, books, and codes. The information is the most valuable resource for this kind of research works. It is necessary to get apt information in response to user query. In this respect, different tasks in information retrieval are elaborated in this article. Also, module-wise description is given to information retrieval, one of the applications of information extraction. For personalisation of LLM, RAG and PEFT are included. This article gives an outline of the information extraction/retrieval processes with personalisation ability.**

Index Terms—**Information extraction, Information retrieval, RAG, PEFT**

I. INTRODUCTION

Information Extraction (IE) refers to the automatic extraction of implicit information from unstructured or semi-structured data sources. In IE methods, focus is on the extraction and/or linking of three main elements from an input source. 1. Entities: anything with named identity, 2. Concepts: a conceptual grouping of elements. Two types of concepts are classes (a named set of individuals), and topics (categories to which individuals or documents relate), 3. Relations: an n-ary tuple of entities ($n \geqslant 2$) with a predicate term denoting the type of relation. The extraction process identifies mentions referring to such entities/concepts/relations in the unstructured or semi-structured input, while linking process associates a disambiguated identifier in a knowledge-base (KB) for a mention [1].

The main applications of IE are knowledge discovery, information retrieval, etc. Information access is one of the fundamental daily needs of human beings. To fulfill the need for rapid acquisition of desired information, various information retrieval (IR) systems such as Google, Bing, and Baidu have been developed. The functions of an IR system are web page retrieval, which aims to determine the relevance between a user query and the content to be retrieved, dialogue systems [2] such as MS Xiaoice, Apple Siri, and Google Assistant to retrieve appropriate responses to user utterance, question answering systems [3] to select relevant clues essential for addressing user questions, and image search engines [4] to return images aligned with user queries. IR systems operate on extensive repositories. To improve the user experience, the retrieval performance is enhanced from both the upstream (query reformulation) and downstream (reranking and reading) perspectives. As an upstream technique, query reformulation is designed to refine user queries so that they are more effective at retrieving relevant documents [5]. On the downstream side, reranking approaches are developed to further adjust the document ranking [6]. It is performed only on a limited set of relevant documents, already retrieved by the retriever. Additionally, reranking can accommodate other specific requirements, such as personalization [7] and diversification [8]. Following the retrieval and reranking stages, a reading component is incorporated to summarize the retrieved documents and deliver a concise document to users [9]. The trajectory of IR has traverses through term-based methods and Boolean logic [10], vector space models [11], statistical language models [12] and neural models [13].
This paper is organized in seven different sections. Section 2 presents related research works in

information extraction/retrieval. Information extraction and the different tasks involved in it are explained in section 3 and 4 respectively. Section 5 describes different processes carried out during information retrieval. A comparison between RAG and PEFT are given in section 6, followed by conclusion in section 7.

II. RELATED WORKS

Computer vision models trained with the latest contrastive learning methods led to features well suited to retrieval [14]. Izcard et al. explored the limits of contrastive learning as a way to train unsupervised dense retrievers and show that it leads to strong performance in various retrieval settings. Contrastive learning is an approach that relies on the fact that every document, in some way, is unique. This signal is the only information available in the absence of manual supervision. A contrastive loss is used to learn by discriminating between documents. This loss compares either positive (from the same document) or negative (from different documents) pairs of document representations. Formally, given a query q with an associated positive document $k+$, and a pool of negative documents $(k_i)_{i=1..K}$, the contrastive InfoNCE loss is defined as:

$$\mathcal{L}(q,k+) = -\frac{exp(s(q,k_+)/\tau)}{\exp\,(s(q,k_+)/\tau) + \sum_{i=1}^{K} exp(s(q,k_+)/\tau)}$$

where τ is a temperature parameter. This loss encourages positive pairs to have high scores and negative pairs to have low scores. Authors evaluated this approach for multi-lingual retrieval as well as cross-lingual transfers.

Recently, information retrieval has seen the emergence of dense retrievers, using neural networks. The objective of a retriever is to find relevant documents in a large collection for a given query. Given a query q and document d, each of them is encoded independently using bi-encoder f_θ, parameterized by θ. The relevance score s(q,d) between a query q and a document d is then the dot product of the resulting representations:

$s(q, d) = \langle f_\theta(q), f_\theta(d)\rangle$

Izacard et al. used a transformer network for f_θ to embed both queries and documents. The representation $f_\theta(q)$ (resp. $f_\theta(d)$) for a query (resp. document) is obtained by averaging the hidden representations of the last layer.

Jernite et al. introduced different objective functions to learn sentence representations, including next sentence prediction and sentence order prediction. These objectives were later used in pre-trained models based on transformers, such as BERT [17] and AlBERT [18]. Guu et al. integrated a bi-encoder retriever model in a BERT pre-training scheme. The retrieved documents are used as additional context in the BERT task, and the whole system is trained end-to-end in an unsupervised way. Lewis et al. proposed to jointly learn a retriever and a generative seq2seq model, using self-supervised training. Fang et al. proposed to apply MoCo in NLP where positive pairs of sentences are obtained using back-translation. SBERT [22] uses a Siamese network similar to contrastive learning to learn a BERT-like model that is adapted to matching sentence embeddings. Spider [23] uses spans appearing multiple times in a document to create pseudo examples for contrastive learning in order to train unsupervised retrievers.

The inverse Cloze task (ICT), proposed by Lee et al. to pre-train retrievers, uses a given sentence as a query and predicts the context surrounding it. Nogu & Cho introduced a cross-encoder model, based on the BERT model [17], which jointly encodes queries and documents. Gillick et al. first studied whether continuous retrievers, based on bi-encoder neural models, could be viable alternative to re-ranking. In the context of question answering, Karpukhin et al. introduced a dense passage retriever (DPR) based on the bi-encoder architecture. This model is initialized with a BERT network, and trained discriminatively using pairs of queries and relevant documents, with hard negatives from BM25. Xiong et al. further extended this work by mining hard negatives with the model itself during optimization, and trained on MS MARCO dataset. Once a collection of documents is encoded, retrieval is performed with a fast k-nearest neighbors library such as FAISS [29]. A limitation of bi-encoders is that queries and documents are represented by a single vector, preventing the model to capture fine-grained interactions between terms. To alleviate the limitations of bi-encoders, Humeau et al. introduces the poly-encoder architecture, where documents are encoded by multiple vectors. Similarly, Khattab et al. proposes the ColBERT model, which keeps a vector representation for each term of the queries and documents. To make the retrieval tractable, the term-level function is approximated to first retrieve an initial set of candidates, which are then reranked with the true score. In the context of question answering, knowledge distillation has been used to train retrievers, either using the attention scores of the reader of the downstream task as synthetic labels [32], or the relevance score from a cross encoder [33]. Luan et al. compares, theoretically and empirically, the performance of sparse and dense retrievers, including bi-, cross- and poly-encoders.

III. INFORMATION EXTRACTION

Information extraction is highly diversified due to its varying targets, heterogeneous structures, and demand-specific schemas. Most of the IE approaches are task specialized, which leads to dedicated architectures, isolated models, and specialized knowledge sources for different IE tasks. These task specialized solutions greatly hinder the rapid architecture development, effective knowledge sharing, and quick cross-domain adaptation of IE systems. It is very complicated to develop dedicated architectures for a large amount of IE tasks. Moreover, learning isolated models severely restricts the knowledge sharing between related tasks and settings. Apart from these, it is costly and time-consuming to construct data sets and knowledge sources specialized for different IE tasks. Therefore, it will be of great benefit to develop a universal IE architecture. Lu et al. proposed a unified text-to-structure generation framework, namely universal information extraction (UIE), which can universally model different IE tasks, adaptively generate targeted structures, and collaboratively learn general IE abilities from different knowledge sources. UIE uniformly encodes different extraction structures via a structured extraction language (SEL), adaptively generates target extractions via a schema-based prompt mechanism – structural schema instructor (SSI), and captures the common IE abilities via a large-scale pre-trained text-to-structure model. Given a specific pre-defined schema s and text x, a universal IE model needs to generate a structure that contains the desirable structural information in the text x indicated by the schema s.

All IE tasks can be modeled as text-to-structure transformations, with different tasks correspond to different structures. These text-to-structure transformations in IE can be further decomposed into atomic transformation operations such as spotting (locates the desirable spans concerning to given specific semantic types), associating (connects spans by assigning them with semantic roles in pre-defined schemas). All IE models share the same underlying spotting and associating abilities. Lu et al. designed a unified SEL, which encodes different IE structures via spotting-associating structure. Because different IE tasks have different schemas, a challenge here is to adaptively control the information generation during extraction. To this end, authors proposed structural schema instructor, a schema based prompt mechanism that controls which kinds of information need to be spotted and associated. Each SEL expression contains three types of semantic units: 1) SPOTNAME represents there is a specific information piece with the type of spot name existing in the source text; 2) ASSONAME indicates there exists a specific information piece in the source text that is with the ASSONAME association to its upper-level spotted information in the structure; 3) INFOSPAN represents the text span corresponding to the specific spotting or associating information piece in the source text. The schema-based prompt can: 1) effectively guide the SEL generation of UIE, so that the general IE ability can be transferred to new IE tasks; 2) adaptively control to spot, associate, and generate so that semantic knowledge across different labels and tasks can be better shared.

The successful application of deep learning methods in natural language processing leads to the wide spread use of neural models in IE subtasks. Approaches based on neural networks allow learning beyond lexical similarities, resulting in state-of-the-art performance on question answering. The strong retrieval results of neural networks have been possible for domains and applications where large training datasets are available [15]. Lin et al. proposed a joint neural framework, ONEIE, to perform end-to-end IE with global constraints. ONEIE aims to extract a globally optimal information network for the input sentence. When comparing candidate information networks during the decoding process, authors considered individual label scores for each knowledge element, and evaluated cross-subtask and cross-instance interactions in the network.

ONEIE framework extracts the information network from a given sentence in four steps: (1) Encoding a given sentence as contextualized word representations; (2) Identifying entity mentions and event triggers as nodes; (3) Computing label scores for all nodes and their pairwise links using local classifiers; (4) During decoding, explore possible information networks for the input sentence using beam search and return the one with the highest global score. A limitation of local classifiers is that they are incapable of capturing interdependencies between knowledge elements in an information network. There are two types of interdependencies in the framework: cross subtask interactions (between entities, relations, and events), and cross instance interactions (between multiple event and/or relation instances in the sentence). At the decoding stage, authors incorporated global features to capture the cross-subtask and cross-instance interactions. ONEIE makes a joint decision for all nodes and their pairwise edges to obtain the globally optimal graph. The basic idea is to calculate the global score for each candidate graph and select the one with the highest score. [6].

IV. TASKS IN INFORMATION EXTRACTION

The main tasks in an IE pipeline considering textual input are given below [1].

Extraction and cleaning: An initial pre-processing step is required to extract plain text from diverse formats like formatted text documents, plain text files, or documents with markup and to clean that text in preparation for subsequent processing. This step involves, recognizing and extracting text from graphics or scanned documents, stripping the sources of control strings and presentational tags, unescaping special characters, etc.

Text tockenization: Here, text is broken down into a sequence of atomic tokens encoding words, phrases, punctuation, etc. This process is employed to identify linguistic units, and to leave intact indivisible or compound words. The result is a sequence of tokens that preserves the order in which they are extracted from the input.

Part-of-speech (POS) tagging: A POS tagger in a given sentence identifies verbs, nouns, adjectives, etc. The grammatical category assigned to a word often depends not only on the meaning of the word, but also its context. A wide variety of techniques have been used for POS tagging. Rule-based approaches use hand-crafted rules for tagging or correcting tags. Supervised stochastic approaches learn statistics and patterns from a corpus of tagged text. Unsupervised approaches try to identify terms that are used frequently in similar contexts. Hybrid approaches can also be used for POS tagging.

Structural parsing (constituency): The constituency parser organizes adjacent elements of a text into groups or phrases using a context-free grammar. Its output is an ordered syntactic tree that denotes a hierarchical structure extracted from the text where non terminal nodes are types of phrases, and terminal nodes are individual words.

Structural parsing (dependency): Dependency parsers are used as an alternative/complement to constituency parsers. The output of a dependency parser will be an ordered tree denoting the structure of a sentence. The dependency parse tree does not use hierarchical nodes to group words and phrases, but rather builds a tree in a bottom-up fashion from relations between words called dependencies.

Named entity recognition (NER): It refers to the identification of strings in a text that refer to various types of entities such as person, organisation, and location. It can be seen as a classification problem, where an NER algorithm assigns a category to an entity. The context is important to disambiguate the correct entity type. Two kinds of approaches can be used for classification: rule based and machine learning based. The first uses hand-crafted rules or patterns based on regular expressions to detect entities. The machine learning based approaches are supervised, unsupervised and semi-supervised.

Terminology extraction: The goal of Terminology Extraction is to identify domain-specific phrases representing concepts and relationships. Applications of Terminology Extraction include the production of domain-specific glossaries and manuals, machine translation of domain specific text, domain-specific modeling tasks, etc.

Keyphrase extraction: This task identifies keyphrases that characterize the subject of a document. Typically such keyphrases are noun phrases that appear frequently in a given document relative to other documents in a given corpus. Keyphrase extraction is used in IR to summarize documents, organize documents based on a taxonomy or tagging system that users can leverage to refine their search needs. There are two general settings under which Keyphrase Extraction can be performed [36]: assignment assumes an input set of keywords that are assigned to input documents, whereas extraction resolves keywords from the documents themselves.

Topic modeling: In Topic Modeling [37], a topic is a cluster of keywords that is viewed intuitively as representing a latent semantic theme present in a document. Such topics are computed based on probability distributions over terms in a text. Topic Modeling is a semantic process that applies a higher level clustering of keywords based on statistical models that capture the likelihood of semantically related terms appearing in a given context. A number of Topic Modeling variants have been proposed. A seminal approach for modeling topics in text is Latent Semantic Analysis (LSA). The key idea of LSA is to compute a low-dimension representation of the words in a document by “compressing” words that frequently co-occur. The probabilistic LSA (pLSA) [38] variant applies a probabilistic model to compute topics. It assumes that documents are sequences of words associated with certain probabilities of being generated. However, which words are generated is assumed to be governed by a given latent (hidden) variable: a topic. Likewise, a document has a certain probability of being on a given topic. The third popular variant of a topic model is Latent Dirichlet Allocation (LDA) [39], which also assumes that a document is associated with potentially multiple latent topics, and that each topic has a certain probability of generating a particular word. The main novelty of LDA is to assume that topics are distributed across documents, and words distributed across topics, according to a sparse Dirichlet prior with two parameters.

Coreference resolution: While entities are directly named, they can also be referred to by pronouns or more general forms of noun phrase in subsequent mentions. The aim of coreference resolution is to identify all such expressions that mention a particular entity in a given text. Various approaches for coreference resolution have been introduced – supervised methods like decision tree, hidden markov model, or deep reinforcement learning, and unsupervised method like Markov logic.

Relation extraction (RE): It is the task of identifying semantic relations from text, where a semantic relation is a tuple of arguments (entities, things, concepts) with a semantic fragment acting as predicate (noun, verb, preposition) [40]. Depending on the number of arguments, a relation may be unary (one argument), binary (two arguments), or n-ary (n>2 arguments). The RE process may follow one of three strategies [41]: knowledge-based methods, supervised methods, and self-supervised methods. KB methods are those that rely on manually-specified pattern-matching rules; supervised methods require a training dataset with labelled examples of sentences containing positive and negative relations; self-supervised methods learn to label their own training datasets.

V. INFORMATION RETRIEVAL

Language models (LMs) are designed to understand or generate human language by taking into account the contextual information from word sequences. IR aims to efficiently retrieve information relevant to user queries from a large repository. Generally, users interact with an IR system by submitting their queries in textual form. Subsequently, IR systems undertake the task of matching and ranking these user-supplied queries against an indexed database, thereby facilitating the retrieval of the most pertinent results [42]. This IR process consists of four modules – query rewriter, retriever, reranker, and reader.

Query rewriter functioning as an essential preprocessing component for search engines, increases the accuracy of retrieval systems through the refinement of initial queries. It seeks to improve the precision and expressiveness of user queries. A query rewriter is primarily designed to serve two distinct scenarios: ad-hoc retrieval and conversational search. Ad-hoc retrieval aims to bridge the semantic gap between a user’s query and the potential documents. For conversational search, query rewriters aim to refine a query within a conversation’s context, transforming it into isolated queries based on historical dialogues. The format of rewritten queries includes questions, keywords, and answer-incorporated passages. Query rewriters modify original queries to new questions to make it more precise, understandable, and aligned with the user’s actual search intent. It can involve rephrasing, expanding, or simplifying the query. Keywords serve as a high-level abstraction of the concepts contained within a query. Rewriting queries into keywords proves particularly effective when the downstream retriever is a sparse retriever. The advent of large language models (LLMs) with their inherent question-answering capabilities has introduced a novel approach to query rewriting. This approach involves initially utilizing LLMs to generate comprehensive answers to the given queries. These detailed answers are then employed to retrieve relevant passages from the corpus, thereby effectively bridging the semantic divide between short queries and long candidate documents. The utilization of LLMs in query rewriting can be categorized into three primary methodologies: prompting, supervised fine-tuning, and reinforcement learning. The prompting approaches employ specific prompts to guide the LLM's output. The supervised fine-tuning techniques adapt pre-trained LLMs to the specific task of query rewriting. The reinforcement learning methods utilize feedback from downstream applications, thereby improving the performance of query rewriters [42].

Retriever is employed in early stages of IR for document recall. It serves as the first-pass document filter to collect broadly relevant documents for user queries. Given the enormous amounts of documents in an IR system, the retriever's efficiency in locating relevant documents is essential for maintaining search engine performance. In light of the quality and quantity of search data, there are two prevalent perspectives on how to improve retrieval performance via LLMs. The first perspective revolves around search data refinement methods, which concentrate on reformulating input queries to precisely present user intents. The second perspective involves training data augmentation methods, which leverage LLMs' generation ability to enlarge the training data for dense retrieval models. The application of LLMs brings about two major impacts on dense retrieval. On one hand, it advances the on-going progress of the existing methods, making substantial improvements in terms of both in-domain accuracy and out-of-domain generalizability. On the other hand, it extends the boundaries of current methods, introducing new capabilities such as instruction-following and in-context learning [42].

Reranker focuses on fine-grained reordering of documents within the retrieved document set. Moreover, it facilitates the adoption of specialized ranking strategies tailored to meet distinct user requirements, such as personalized and diversified search results. Reranker, as the second-pass document filter in IR, aims to rerank a document list retrieved by the retriever based on the query-document relevance. The existing LLM-based reranking methods can be divided into three paradigms: utilizing LLMs as supervised rerankers, utilizing LLMs as unsupervised rerankers, and utilizing LLMs for training data augmentation. Based on the backbone model structure, we can categorize existing supervised rerankers as: (1) encoder-only, (2) encoder-decoder, and (3) decoder-only. As the size of 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, listwise, and pairwise methods. The pointwise methods measure the relevance between a query and a single document, and can be categorized into two types: relevance generation and query generation. Listwise methods insert the query and a document list into the prompt and instruct the LLMs to output the reranked document identifiers. Due to the limited input length of LLMs, it is not feasible to insert all candidate documents into the prompt. To alleviate this issue, these methods employ a sliding window strategy to rerank a subset of candidate documents (only the documents within the window) each time [42]. In pairwise methods, LLMs are given a prompt that consists of a query and a document pair. Then, they are instructed to generate the identifier of the document with higher relevance. To rerank all candidate documents, aggregation methods like AllPairs are used [43].

Reader has the ability to comprehend real-time user intent and generate dynamic responses based on the retrieved text. It has been introduced to generate answers based on the document corpus in IR systems. By integrating a reader module, IR systems can directly present conclusive passages to users. The integration of references into generated responses has been an effective technique of the reader module.

According to the way LLMs utilize IR systems in the reader module, we can categorize them into passive readers and active readers [42].

VI. RAG vs PEFT

Personalization has been an area of much interest in the information retrieval community and has been extensively studied for searching [44], recommendation [45], and question answering [46]. With the rise of LLMs, personalizing them has emerged as an important topic due to its applications in various real-world systems, such as personalized recommender systems [47], virtual assistants [48], and content generation [49]. Personalization allows LLMs to better understand and predict user needs, offering more relevant and contextually appropriate content or responses. . Currently, the most reliable approach to personalizing LLMs is through RAG personalization. Retrieval-augmented generation (RAG) has emerged as an effective approach for enhancing various aspects of machine learning tasks. It integrates information retrieval with natural language generation, thereby improving the relevance and factual accuracy of the generated outputs. It utilizes a retriever to access external information at inference time, enabling contextually grounded and factually consistent generation. This capability allows RAG to produce more informed and relevant outputs [50].

Another potential approach to personalizing LLMs involves fine tuning the LLM on user-specific data, such as documents authored by the user or other relevant information about the user. However, training a separate LLM for each user is computationally expensive and requires a large amount of storage. To solve this, parameter-efficient fine-tuning (PEFT) techniques such as low-rank adaptation (LoRA) [51], can be used to adjust a subset of the model's parameters based on each user's data. LaMP comprises three classification tasks and four text generation tasks, each designed to evaluate different aspects of LLM personalization. It has demonstrated significant reductions in memory consumption and storage requirements while increasing performance for tasks such as language modeling, reasoning, text generation, and question answering [52]. Thus, using LoRA to train LLM on individual user data personalizes the LLM while avoiding the need to store a full set of parameters for each user [50]. PEFT enables the adaptation of large language models (LLMs) to specific tasks without the need for full model retraining. Among various PEFT methods, Low-Rank Adaptation (LoRA) is more effective for efficiently fine-tuning LLMs. It introduces low-rank decomposition into weight matrices and injects trainable low-rank matrices into otherwise frozen model weights [51]. This technique minimizes the number of trainable parameters while preserving the model's expressiveness.

There are two main approaches to personalizing LLMs. The first approach involves personalizing the input prompt provided to the model. In this approach, employ a retrieval model to retrieve a set of personalized information from the user profile and use this information to construct a personalized prompt. The personalized prompt is then fed to the LLM to generate a tailored response. Here, we can use a wide range of retrieval models such as BM25 [53] as a lexical-matching retrieval model, Contriever [54] as a semantic matching retrieval model, Recency [55] as a time-aware retrieval model, and RSPG [56] as an ensemble model. This model ensures that each input is directed to the most suitable retrieval model for retrieving relevant documents.

The second approach focuses on changing the parameters of the LLM through training the LLM using user-specific data to help the model learn the user's preferences, writing style, and background knowledge. By using the same LLM backbone for all users and loading individual LoRA adapters per user, the system can operate more efficiently from a computational aspect. Therefore, using PEFT provides a more cost-effective solution compared to training an entire LLM for each user. While PEFT is more resource intensive than RAG due to the need for fine-tuning and storing user-specific parameters, it is generally faster at inference time since it does not require retrieving external information. In contrast, RAG is more efficient in terms of storage and adaptation to new information but incurs additional latency during retrieval [50].

The integration of PEFT and RAG can enhance LLM personalization by leveraging their complementary strengths. PEFT facilitates efficient adaptation by fine-tuning a small number of additional parameters, reducing computational and memory overhead while allowing the model to capture user-specific preferences. However, it has some limitations, as it relies on static updates and may struggle to learn less common user preferences effectively. RAG mitigates this issue by retrieving user-specific information at inference time, ensuring that personalization remains dynamic and contextually relevant without requiring extensive fine-tuning. By combining both approaches, the model benefits from long-term adaptation through fine-tuned parameters and real-time personalization through retrieval [50].

VII. CONCLUSION

In recent years, researchers have made significant advances in the application of natural language processing algorithms. The majority of these works have focused on NER. A key challenge in natural language processing is the development of robust, simple, and general relation extraction techniques to accurately extract the relationships between named entities. LLMs such as GPT-3, LLaMA 1 and FLAN have been shown remarkable ability to leverage semantic information between tokens in natural language sequences of varying length.

REFERENCES

[1] Martinez-Rodriguez, Jose L., Aidan Hogan, and Ivan Lopez-Arevalo. "Information extraction meets the semantic web: a survey." Semantic Web 11, no. 2 (2020): 255-335.

[2] Y. Wu, W. Wu, C. Xing, M. Zhou, and Z. Li, “Se quential matching network: A new architecture for multi-turn response selection in retrieval-based chat bots,” in Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30- August 4, Volume 1: Long Papers, R. Barzilay and M. Kan, Eds. Association for Computational Linguistics, 2017, pp. 496–505.

[3] V. Karpukhin, B. Oguz, S. Min, P. S. H. Lewis, L. Wu, S. Edunov, D. Chen, and W. Yih, “Dense passage retrieval for open-domain question answering,” in 23 Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020, B. Webber, T. Cohn, Y. He, and Y. Liu, Eds. Association for Computational Linguis tics, 2020, pp. 6769–6781.

[4] R. Datta, D. Joshi, J. Li, and J. Z. Wang, “Image re trieval: Ideas, influences, and trends of the new age,” ACMComput. Surv., vol. 40, no. 2, pp. 5:1–5:60, 2008.

[5] Y. Arens, C. A. Knoblock, and W. Shen, “Query re formulation for dynamic information integration,” J. Intell. Inf. Syst., vol. 6, no. 2/3, pp. 99–130, 1996.

[6] R. F. Nogueira, W. Yang, K. Cho, and J. Lin, “Multi stage document ranking with BERT,” CoRR, vol. abs/1910.14424, 2019.

[7] J. Teevan, S. T. Dumais, and E. Horvitz, “Personalizing search via automated analysis of interests and activ ities,” in SIGIR 2005: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Salvador, Brazil, August 15-19, 2005, R. A. Baeza-Yates, N. Ziviani, G. Marchionini, A. Moffat, and J. Tait, Eds. ACM, 2005, pp. 449–456

[8] J G Carbonell and J. Goldstein, “The use of mmr, diversity-based reranking for reordering documents and producing summaries,” in SIGIR ’98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 24-28 1998, Melbourne, Australia, W. B. Croft, A. Moffat, C. J. van Rijsbergen, R. Wilkinson, and J. Zobel, Eds. ACM, 1998, pp. 335–336.

[9] S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. van den Driessche, J. Lespiau, B. Damoc, A. Clark, D. de Las Casas, A. Guy, J. Menick, R. Ring, T. Hennigan, S. Huang, L. Maggiore, C. Jones, A. Cassirer, A. Brock, M. Pa ganini, G. Irving, O. Vinyals, S. Osindero, K. Si monyan, J. W. Rae, E. Elsen, and L. Sifre, “Improv ing language models by retrieving from trillions of tokens,” in International Conference on Machine Learn ing, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA, ser. Proceedings of Machine Learning Research, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesv´ari, G. Niu, and S. Sabato, Eds., vol. 162. PMLR, 2022, pp. 2206–2240.

[10] G. Salton and M. McGill, Introduction to Modern Infor mation Retrieval. McGraw-Hill Book Company, 1984

[11] G. Salton, A. Wong, and C. Yang, "A vector space model for automatic indexing," Commun. ACM, vol. 18, no. 11, pp. 613–620, 1975.

[12] F. Song and W. B. Croft, "A general language model for information retrieval," in Proceedings of the 1999 ACMCIKM International Conference on Information and Knowledge Management, Kansas City, Missouri, USA, November 2-6, 1999. ACM, 1999, pp. 316–321

[13] J. Guo, Y. Fan, Q. Ai, and W. B. Croft, "A deep relevance matching model for ad-hoc retrieval," in Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, In dianapolis, IN, USA, October 24-28, 2016, S. Mukhopad hyay, C. Zhai, E. Bertino, F. Crestani, J. Mostafa, J. Tang, L. Si, X. Zhou, Y. Chang, Y. Li, and P. Sondhi, Eds. ACM, 2016, pp. 55–64.

[14] MathildeCaron,HugoTouvron, IshanMisra,HervéJégou,JulienMairal,PiotrBojanowski,andArmand Joulin.Emergingproperties inself-supervisedvisiontransformers. arXivpreprintarXiv:2104.14294,2021.

[15] Izacard, Gautier, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. "Unsupervised dense information retrieval with contrastive learning." arXiv preprint arXiv:2112.09118 (2021).

[16] Yacine Jernite, Samuel R Bowman, and David Sontag. Discourse-based objectives for fast unsupervised sentence representation learning. arXiv preprint arXiv:1705.00557, 2017

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Proc. NAACL, 2019

[18] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942, 2019

[19] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Realm: Retrieval-augmented language model pre-training. arXiv preprint arXiv:2002.08909, 2020

[20] Mike Lewis, Marjan Ghazvininejad, Gargi Ghosh, Armen Aghajanyan, Sida Wang, and Luke Zettlemoyer. Pre-training via paraphrasing. arXiv preprint arXiv:2006.15020, 2020

[21] Hongchao Fang, Sicheng Wang, Meng Zhou, Jiayuan Ding, and Pengtao Xie. Cert: Contrastive self-supervised learning for language understanding. arXiv preprint arXiv:2005.12766, 2020

[22] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084, 2019

[23] Ori Ram, Gal Shachaf, Omer Levy, Jonathan Berant, and Amir Globerson. Learning to retrieve passages without supervision, 2021.

[24] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. arXiv preprint arXiv:2007.00808, 2020.

[25] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with bert. arXiv preprint arXiv:1901.04085, 2019

[26] Daniel Gillick, Alessandro Presta, and Gaurav Singh Tomar. End-to-end retrieval in continuous space. arXiv preprint arXiv:1811.08008, 2018.

[27] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. arXiv preprint arXiv:2004.04906, 2020

[28] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. arXiv preprint arXiv:2007.00808, 2020

[29] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. IEEE Transactions on Big Data, 2019

[30] Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. Poly-encoders: Transformer architectures and pre-training strategies for fast and accurate multi-sentence scoring. arXiv preprint arXiv:1905.01969, 2019

[31] Omar Khattab, Christopher Potts, and Matei Zaharia. Relevance-guided supervision for openqa with colbert, 2020

[32] Gautier Izacard and Edouard Grave. Distilling knowledge from reader to retriever for question answering, 2020

[33] Sohee Yang and Minjoon Seo. Is retriever merely an approximator of reader? arXiv preprint arXiv:2010.10999, 2020

[34] Lu, Yaojie, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. "Unified structure generation for universal information extraction." arXiv preprint arXiv:2203.12277 (2022).

[35] Lin, Ying, Heng Ji, Fei Huang, and Lingfei Wu. "A joint neural model for information extraction with global features." In Proceedings of the 58th annual meeting of the association for computational linguistics, pp. 7999-8009. 2020.

[36] S. Siddiqi and A. Sharan, Keyword and keyphrase extraction techniques: A literature review, International Journal of Com puter Applications 109(2) (2015)

[37] D.M. Blei, Probabilistic topic models, Commun. ACM 55(4) (2012), 77–84.

[38] T. Hofmann, Unsupervised learning by probabilistic latent se mantic analysis, Machine Learning 42(1) (2001), 177–196

[39] D.M. Blei, A.Y. Ng and M.I. Jordan, Latent Dirichlet alloca tion, Journal of machine Learning research 3 (2003), 993 1022

[40] N. Bach and S. Badaskar, A review of relation extraction, in: Literature Review for Language and Statistics II, 2, 2007.

[41] M. Banko, M.J. Cafarella, S. Soderland, M. Broadhead and O. Etzioni, Open information extraction from the Web, in: International Joint Conference on Artificial Intelligence (IJ CAI), M.M. Veloso, ed., 2007.

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

[43] Z. Qin, R. Jagerman, K. Hui, H. Zhuang, J. Wu, J. Shen, T. Liu, J. Liu, D. Metzler, X. Wang, and M. Bendersky, “Large language models are effective text rankers with pairwise ranking prompting,” CoRR, vol. abs/2306.17563, 2023.

[44] Qian Guo, Wei Chen, and Huaiyu Wan. 2021. AOL4PS: A Large-scale Data Set for Personalized Search. Data Intelligence 3, 4 (10 2021), 548–567

[45] CarsonK.Leung, EvanW.R. Madill, andQiWen.2023. APrivacy-PreservingSemi Decentralized Personalized Recommendation System. In 2023 IEEE International Conference on Big Data (BigData). 1336–1343

[46] Pranav Kasela, Marco Braga, Gabriella Pasi, and Raffaele Perego. 2024. SE-PQA: Personalized Community Question Answering. In Companion Proceedings of the ACMWebConference 2024 (Singapore, Singapore) (WWW ’24). Association for Computing Machinery, New York, NY, USA, 1095–1098

[47] Junyi Chen. 2023. A Survey on Large Language Models for Personalized and Explainable Recommendations

[48] Ahmet Baki Kocaballi, Shlomo Berkovsky, Juan C Quiroz, Liliana Laranjo, Huong Ly Tong, Dana Rezazadegan, Agustina Briatore, and Enrico Coiera. 2019. The Personalization of Conversational Agents in Health Care: Systematic Review. J Med Internet Res 21, 11 (7 Nov 2019)

[49] Bashar Alhafni, Vivek Kulkarni, Dhruv Kumar, and Vipul Raheja. 2024. Per sonalized Text Generation with Fine-Grained Linguistic Control. In Proceedings of the 1st Workshop on Personalization of Generative AI Systems (PERSONAL IZE 2024), Ameet Deshpande, EunJeong Hwang, Vishvak Murahari, Joon Sung Park, Diyi Yang, Ashish Sabharwal, Karthik Narasimhan, and Ashwin Kalyan (Eds.). Association for Computational Linguistics, St. Julians, Malta, 88–101

[50] Salemi, Alireza, and Hamed Zamani. "Comparing retrieval-augmentation and parameter-efficient fine-tuning for privacy-preserving personalization of large language models." In Proceedings of the 2025 International ACM SIGIR Conference on Innovative Concepts and Theories in Information Retrieval (ICTIR), pp. 286-296. 2025.

[51] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In International Conference on Learning Representations

[52] Rui Pan, Xiang Liu, Shizhe Diao, Renjie Pi, Jipeng Zhang, Chi Han, and Tong Zhang. 2024. LISA: Layerwise Importance Sampling for Memory-Efficient Large Language Model Fine-Tuning. In The Thirty-eighth Annual Conference on Neural Information Processing Systems

[53] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In Text Retrieval Conference

[54] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bo janowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised Dense Infor mation Retrieval with Contrastive Learning

[55] Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. 2024. LaMP: When Large Language Models Meet Personalization. In Proceed ings of the 62nd Annual Meeting of the Association for Computational Linguis tics (Volume 1: Long Papers), Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, Bangkok, Thailand, 7370–7392.

[56] Alireza Salemi, Surya Kallumadi, and Hamed Zamani. 2024. Optimization Meth ods for Personalizing Large Language Models through Retrieval Augmentation. In Proceedings of the 47th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval