Retrieving and Reading : A Comprehensive Survey on Open-domain Question Answering

Fengbin Zhu, Wenqiang Lei*, Chao Wang, Jianming Zheng, Soujanya Poria, Tat-Seng Chua,

Abstract—Open-domain Question Answering (OpenQA) is an important task in Natural Language Processing (NLP), which aims to answer a question in the form of natural language based on large-scale unstructured documents. Recently, there has been a surge in the amount of research literature on OpenQA, particularly on techniques that integrate with neural Machine Reading Comprehension (MRC). While these research works have advanced performance to new heights on benchmark datasets, they have been rarely covered in existing surveys on QA systems. In this work, we review the latest research trends in OpenQA, with particular attention to systems that incorporate neural MRC techniques. Specifically, we begin with revisiting the origin and development of OpenQA systems. We then introduce modern OpenQA architecture named "Retriever-Reader" and analyze the various systems that follow this architecture as well as the specific techniques adopted in each of the components. We then discuss key challenges to developing OpenQA systems and offer an analysis of benchmarks that are commonly used. We hope our work would enable researchers to be informed of the recent advancement and also the open challenges in OpenQA research, so as to stimulate further progress in this field.

Index Terms—Textual Question Answering, Open Domain Question Answering, Machine Reading Comprehension, Information Retrieval, Natural Language Understanding, Information Extraction

✦

1 INTRODUCTION

Question Answering (QA) aims to provide precise answers in response to the user's questions in natural language. It is a long-standing task dating back to the 1960s [1]. Compared with a search engine, the QA system aims to present the final answer to a question directly instead of returning a list of relevant snippets or hyperlinks, thus offering better user-friendliness and efficiency. Nowadays many web search engines like Google and Bing have been evolving towards higher intelligence by incorporating QA techniques into their search functionalities [2]. Empowered with these techniques, search engines now have the ability to respond precisely to some types of questions such as

— Q: *"When was Barack Obama born?"*
— A: *"4 August 1961".*

The whole QA landscape can roughly be divided into two parts: textual QA and Knowledge Base (KB)-QA, according to the type of information source where answers are derived from. Textual QA mines answers from unstructured text documents while KB-QA from a predefined structured KB that is often manually constructed. Textual QA is generally more scalable than the latter, since most of the unstructured text resources it exploits to obtain answers are fairly common and easily accessible, such as Wikipedia [3], news

articles [4] and science books [5], etc. Specifically, textual QA is studied under two task settings based on the availability of contextual information, i.e. Machine Reading Comprehension (MRC) and Open-domain QA (OpenQA). MRC, which originally took inspiration from language proficiency exams, aims to enable machines to read and comprehend specified context passage(s) for answering a given question. In comparison, OpenQA tries to answer a given question without any specified context. It usually requires the system to first search for the relevant documents as the context w.r.t. a given question from either a local document repository or the World Wide Web (WWW), and then generate the answer, as illustrated in Fig. 1. OpenQA therefore enjoys a wider scope of application and is more in line with real-world QA behavior of human beings while MRC can be considered as a step to OpenQA [6]. In fact, building an OpenQA system that is capable of answering any input questions is deemed as the ultimate goal of QA research.

In literature, OpenQA has been studied closely with research in Natural Language Processing (NLP), Information Retrieval (IR), and Information Extraction (IE) [7], [8], [9], [10]. Traditional OpenQA systems mostly follow a pipeline consisting of three stages, i.e. *Question Analysis*, *Document Retrieval* and *Answer Extraction* [6], [9], [11]. Given an input question in natural language, *Question Analysis* aims to reformulate the question to generate search queries for facilitating subsequent *Document Retrieval* and classify the question to obtain its expected answer type(s) that would guide *Answer Extraction*. In the *Document Retrieval* stage, the system searches for question-relevant documents or passages with the generated search queries, usually using existing IR techniques like TF-IDF and BM25, or specific techniques developed for Web search engines like Google.com and Bing.com. After that, in the *Answer Extraction* stage, the final answer is extracted from related documents received from

- *Corresponding author: Wenqiang Lei
- Fengbin Zhu, Wenqiang Lei and Tat-Seng Chua are with National University of Singapore (NUS) E-mail: zhfengbin@gmail.com, wenqianglei@gmail.com, dcscts@nus.edu.sg
- Fengbin Zhu and Chao Wang are with 6ESTATES PTE LTD, Singapore E-mail: wangchao@6estates.com
- Jianming Zheng is with National University of Defense Technology, China E-mail: zhengjianming12@nudt.edu.cn
- Soujanya Poria is with Singapore University of Technology and Design (SUTD) E-mail: sporia@sutd.edu.sg

Q: When was Barack Obama born? → [WIKIPEDIA] [NEWS] ... → A: 4 August, 1961

Fig. 1: An illustration of OpenQA. Given a natural language question, the system infers the answer from a collection of unstructured text documents.

the previous stage.

Deep learning techniques, which have driven remarkable progress in many fields, have also been successfully applied to almost every stage of OpenQA systems [12]. For example, [13] and [14] develop the question classifier using a CNN-base model and an LSTM-based model respectively. In [15], [16], [17], they propose neural retrieval models to search for relevant documents in a latent space. In recent years, with the emergence of some large-scale QA datasets [18], [19], [20], [21], [22], [23], neural MRC techniques have been greatly advanced [18], [24], [25], [26], [27]. By adopting the popular neural MRC methods to extract the answer to a given question from the relevant document(s), traditional OpenQA systems have been revolutionized [3], [28], [29], [30] and evolved to a modern *"Retriever-Reader"* architecture. *Retriever* is responsible for retrieving relevant documents w.r.t. a given question, which can be regarded as an IR system, while *Reader* aims at inferring the final answer from the received documents, which is usually a neural MRC model. A handful of works [3], [31], [32] even rename OpenQA as Machine Reading Comprehension at Scale (MRS). Following this architecture, extensive research has been made along various directions, such as re-ranking the retrieved documents before feeding them into a neural MRC model [28], [33], [34], retrieving the relevant documents iteratively given a question [29], [35], [36], and training the entire OpenQA system in an end-to-end manner [15], [30], [37], [38], etc.

Based on above observations and insights, we believe it is time to provide a comprehensive literature review on OpenQA systems, with particular attention to techniques that incorporate neural MRC models. Our review is expected to acknowledge the advancement that has been made thus far and summarize the current challenges to stimulate further progress in this field. In the rest of this survey, we will present the following contents. In Section 2, we review the development of OpenQA systems, including the origin, traditional architecture, and recent progress in using deep neural networks. In Section 3, we summarize and elaborate a *"Retriever-Reader"* architecture for OpenQA followed by detailed analysis on the various techniques adopted. In Section 4, we first discuss some salient challenges towards OpenQA, identifying the research gaps and hoping to enhance further research in this field, and subsequently provide a summary and analysis of QA benchmarks that are applicable to either MRC or OpenQA. Finally, we draw our conclusions based on the presented contents above in Section 5.

2 DEVELOPMENT OF OPENQA

In this section, we first briefly introduce the origin of Open-domain Question Answering (OpenQA) and then review the traditional and deep learning approaches to OpenQA sequentially to describe its remarkable advancement in the past two decades.

2.1 Origin of OpenQA

Pioneering research on Question Answering (QA) system was conducted within the scope of Information Retrieval (IR), with the focus on restricted domain or closed-domain settings. The earliest QA system, as is widely acknowledged, is the Baseball [1], which was designed in 1961 to answer questions about American baseball games, such as game time, location and team name. Within this system, all the relevant information is stored in a well-defined dictionary, and user questions are translated into query statements using linguistic methods to extract final answers from the dictionary. In 1973, another famous QA system LUNAR [39] was developed as a powerful tool to assist research work of lunar geologists, where chemical analysis data about lunar rocks and soil obtained from Apollo moon missions are stored in one data base file provided by NASA MSC for each scientist to conveniently view and analyze. In 1993, MURAX [40] was designed to answer simple academic questions based on an English academic encyclopedia which mainly employs linguistic analysis and syntactic pattern matching techniques.

In 1999, OpenQA was first defined as extracting top 5 probable snippets containing the correct answer from a collection of news articles in the QA track launched by the Text REtrieval Conference (TREC) [4]. Compared to the previous research on QA, in the open-domain setting, a large number of unstructured documents is used as the information source from which the correct answer to a given question would be extracted. In the later years, a series of TREC QA Tracks have remarkably advanced the research progress on OpenQA [41], [42], [43]. It is worth noting that

systems are required to return exact short answers to given questions starting from TREC-11 held in 2002 [42].

The TREC campaign provides a local collection of documents as the information source for generating answers, but the popularity of World Wide Web (WWW), especially the increasing maturity of search engines, has inspired researchers to build Web-based OpenQA systems [40], [44], [45], [46] obtaining answers from online resources like Google.com and Ask.com, using IR techniques. Web search engines are able to consistently and efficiently collect massive web pages, therefore capable of providing much more information to help find answers in response to user questions. In 2001, a QA system called MULDER [44] was designed to automatically answer open-domain factoid questions with a search engine (e.g., Google.com). It first translates users' questions to multiple search queries with several natural-language parsers and submits them to the search engine to search for relevant documents, and then employs an answer extraction component to extract the answer from the returned results. Following this pipeline, a well-known QA system AskMSR [45] was developed, which mainly depends on data redundancy rather than sophisticated linguistic analysis of either questions or candidate answers. It first translates the user's question into queries relying on a set of predefined rewriting rules to gather relevant documents from search engines and then adopts a series of n-gram based algorithms to mine, filter and select the best answer. For such OpenQA systems, the search engines are able to provide access to an ocean of information, significantly enlarging the possibility of finding precise answers to user questions. Nevertheless, such an ample information source also brings considerable noisy content that challenges the QA system to filter out.

2.2 Traditional Architecture of OpenQA

The traditional architecture of OpenQA systems is illustrated in Fig. 2, which mainly comprises three stages: *Question Analysis*, *Document Retrieval*, and *Answer Extraction* [6], [11]. Given a natural language question, *Question Analysis* aims to understand the question first so as to facilitate document retrieval and answer extraction in the following stages. Performance of this stage is found to have a noticeable influence upon that of the following stages, and hence important to the final output of the system [47]. Then, *Document Retrieval* stage searches for question-relevant documents based on a self-built IR system [4] or Web search engine [44], [45] using the search queries generated by *Question Analysis*. Finally, *Answer Extraction* is responsible for extracting final answers to user questions from the relevant documents received in the preceding step. In the following, we will analyze each stage one by one.

2.2.1 Question Analysis

The goals of *Question Analysis* stage are two-fold. On one hand, it aims to facilitate the retrieval of question-relevant documents, for which a Query Formulation module is often adopted to generate search queries. On the other hand, it is expected to enhance the performance of *Answer Extraction* stage by employing a Question Classification module to predict the type of the given question, which leads to a set of expected answer types. A simple illustration of this stage is given in the leftmost grey box of Fig. 2.

In Query Formulation, linguistic techniques such as POS tagging [40], [44], stemming [40], parsing [44] and stop word removal [45], [48] are usually utilized to extract keywords for retrieving. However, the terms used in questions are often not the same as those appearing in the documents that contain the correct answers. This problem is called "term mismatch" and is a long-standing and critical issue in IR. To address this problem, query expansion [49], [50] and paraphrasing techniques [51], [52], [53], [54] are often employed to produce additional search words or phrases so as to retrieve more relevant documents.

Question Classification, the other module that is often adopted for *Question Analysis* stage, aims to identify the type of the given question based on a set of question types (e.g., where, when, who, what) or a taxonomy [55], [56] manually defined by linguistic experts. After obtaining the type of the question, expected answer types can be easily determined using rule-based mapping methods [9]. For example, given a question *"When was Barack Obama born?"*, the answer type would be inferred as *"Date"* when knowing the question type is *"When"*. Identifying the question type can provide constraint upon answer extraction and significantly reduce the difficulty of finding correct answers. Question Classification has attracted much interest in literature [44], [55], [57], [58], [59]. For instance, [59] proposed to extract relevant words from a given question and then classify the question based on rules associating these words to concepts; [57] trained a list of question classifiers using various machine learning techniques such as Support Vector Machines (SVM), Nearest Neighbors and Decision Trees on top of the hierarchical taxonomy proposed by [55].

2.2.2 Document Retrieval

This stage is aimed at obtaining a small number of relevant documents that probably contain the correct answer to a given question from a collection of unstructured documents, which usually relies on an IR engine. It can significantly reduce the search space for arriving at the final answer.

In the past decades, various retrieval models have been developed for *Document Retrieval*, among which some popular ones are the Boolean model, Vector Space Models, Probabilistic Models, Language Models [60], etc., which are briefly revisited as follows.

- *Boolean Model*: The Boolean Model is one of the simplest retrieval models. The question is transformed into the form of a Boolean expression of terms, which are combined with the operators like "AND", "OR" and "NOT" to exactly match with the documents, with each document viewed as a set of words.
- *Vector Space Model*: The Vector Space Models represent the question and each document as word vectors in a d-dimensional word space, where d is the number of words in the vocabulary. When searching for relevant documents to a given question, the relevance score of each document is computed by computing the similarity (e.g., the cosine similarity) or distance (e.g., the euclidean distance) between its vector and the question vector. Compared to the Boolean model, this approach returns documents to the question even if the
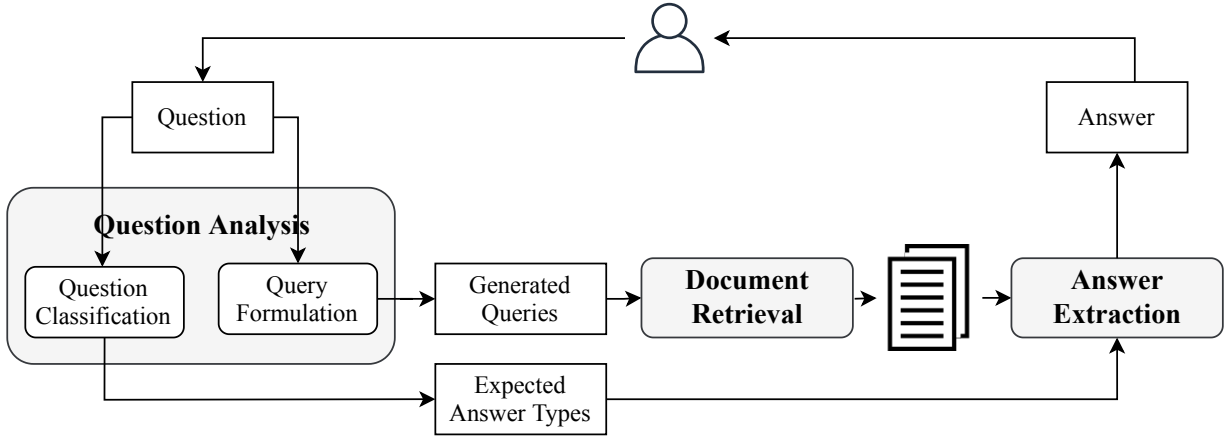
Fig. 2: An illustration of traditional architecture of OpenQA system

constraints posed by the question are only partially met, with precision sacrificed.

- *Probabilistic Model*: The Probabilistic Models provide a way of integrating probabilistic relationships between words into a model. Okapi BM25 [61] is a probabilistic model sensitive to term frequency and document length, which is one of the most empirically successful retrieval models and widely used in current search engines.

- *Language Model*: The Language Models [62] are also very popular, among which the Query Likelihood Model [60] is the most widely adopted. It builds a probabilistic language model LM_d for each document d and ranks documents according to the probability $P(q \mid LM_d)$ of the language model generating the given question q.

In practice, the documents received often contain irrelevant ones, or the number of documents is so large that the capacity of the *Answer Extraction* model is overwhelmed. To address the above issues, post-processing on the retrieved documents is very demanded. Widely used approaches on processing retrieved documents include document filtering, document re-ranking and document selection [9], etc. Document filtering is used to identify and remove the noise w.r.t. a given question; document re-ranking is developed to further sort the documents according to a plausibility degree of containing the correct answer in the descending order; document selection is to choose the top relevant documents. After post-processing, only the most relevant documents would be remained and fed to the next stage to extract the final answer.

2.2.3 Answer Extraction

The ultimate goal of an OpenQA system is to successfully answer given questions, and *Answer Extraction* stage is responsible for returning a user the most precise answer to a question. The performance of this stage is decided by the complexity of the question, the expected answer types from *Question Analysis* stage, the retrieved documents from *Document Retrieval* stage as well as the extraction method adopted, etc. With so many influential factors, researchers need to take a lot of care and place special importance on this stage.

In traditional OpenQA systems, factoid questions and list questions [63] have been widely studied for a long time. Factoid questions (e.g., When, Where, Who...) to which the answers are usually a single text span in in the documents, such as such as an entity name, a word token or a noun phrase. While list questions whose answers are a set of factoids that appeared in the same document or aggregated from different documents. The answer type received from the stage of *Question Analysis* plays a crucial role, especially for the given question whose answers are named entities. Thus, early systems heavily rely on the Named Entity Recognition (NER) technique [40], [46], [64] since comparing the recognised entities and the answer type may easily yield the final answer. In [65], the answer extraction is described as a unified process, first uncovering latent or hidden information from the question and the answer respectively, and then using some matching methods to detect answers, such as surface text pattern matching [66], [67], word or phrase matching [44], and syntactic structure matching [40], [48], [68].

In practice, sometimes the extracted answer needs to be validated when it is not confident enough before presenting to the end-users. Moreover, in some cases multiple answer candidates may be produced to a question and we have to select one among them. Answer validation is applied to solve such issues. One widely applied validation method is to adopt an extra information source like a Web search engine to validate the confidence of each candidate answer. The principle is that the system should return a sufficiently large number of documents which contain both question and answer terms. The larger the number of such returned documents is, the more likely it will be the correct answer. This principle has been investigated and demonstrated fairly effective, though simple [9].

2.3 Application of Deep Neural Networks in OpenQA

In the recent decade, deep learning techniques have also been successfully applied to OpenQA. In particular, deep learning has been used in almost every stage in an OpenQA system, and moreover, it enables OpenQA systems to be

end-to-end trainable. For *Question Analysis*, some works develop neural classifiers to determine the question types. For example, [13] and [14] respectively adopt a CNN-based and an LSTM-based model to classify the given questions, both achieving competitive results. For *Document Retrieval*, dense representation based methods [16], [29], [30], [35] have been proposed to address "term-mismatch", which is a long-standing problem that harms retrieval performance. Unlike the traditional methods such as TF-IDF and BM25 that use sparse representations, deep retrieval methods learn to encode questions and documents into a latent vector space where text semantics beyond term match can be measured. For example, [29] and [35] train their own encoders to encode each document and question independently into dense vectors, and the similarity score between them is computed using the inner product of their vectors. The Sublinear Maximum Inner Product Search (MIPS) algorithm [69], [70], [71] is used to improve the retrieval efficiency given a question, especially when the document repository is large-scale. For *Answer Extraction*, as a decisive stage for OpenQA systems to arrive at the final answer, neural models can also be applied. Extracting answers from some relevant documents to a given question essentially makes the task of Machine Reading Comprehension (MRC). In the past few years, with the emergence of some large-scale datasets such as CNN/Daily Mail [18], MS MARCO [20], RACE [21] and SQuAD 2.0 [22], research on neural MRC has achieved remarkable progress [24], [25], [26], [27]. For example, BiDAF [24] represents the given document at different levels of granularity via a multi-stage hierarchical structure consisting of a character embedding layer, a word embedding layer, and a contextual embedding layer, and leverages a bidirectional attention flow mechanism to obtain a question-aware document representation without early summarization. QANet [26] adopts CNN and the self-attention mechanism [72] to model the local interactions and global interactions respectively, which performs significantly faster than usual recurrent models.

Furthermore, applying deep learning enables the OpenQA systems to be end-to-end trainable [15], [30], [37]. For example, [37] argue it is sub-optimal to incorporate a standalone IR system in an OpenQA system, and they develop an ORQA system that treats the document retrieval from the information source as a latent variable and trains the whole system only from question-answer string pairs based on BERT [27]. REALM [30] is a pre-trained language model that contains a knowledge retriever and a knowledge augmented encoder. Both its retriever and encoder are differentiable neural networks, which are able to compute the gradient w.r.t. the model parameters to be back propagated all the way throughout the network. Similar to other pre-training language models, it also has two stages, i.e., pre-training and fine-tuning. In the pre-training stage, the model is trained in an unsupervised manner, using masked language modeling as the learning signal while the parameters are fine-tuned using supervised examples in the fine-tuning stage.

In early OpenQA systems, the success of answering a question is highly dependent on the performance of *Question Analysis*, particularly Question Classification, that provides expected answer types [47]. However, either the types or the taxonomies of questions are hand-crafted by linguists, which are non-optimal since it is impossible to cover all question types in reality, especially those complicated ones. Furthermore, the classification errors would easily result in the failure of answer extraction, thus severely hurting the overall performance of the system. According to the experiments in [47], about 36.4% of errors in early OpenQA systems are caused by miss-classification of question types. Neural models are able to automatically transform questions from natural language to representations that are more recognisable to machines. Moreover, neural MRC models provide an unprecedented powerful solution to *Answer Extraction* in OpenQA, largely offsetting the necessity of applying the traditional linguistic analytic techniques to questions and bringing revolutions to OpenQA systems [3], [28], [29], [37]. The very first work to incorporate neural MRC models into the OpenQA system is DrQA proposed by [3], evolving to a "Retriever-Reader" architecture. It combines TF-IDF based IR technique and a neural MRC model to answer open-domain factoid questions over Wikipedia and achieves impressive performance. After [3], lots of works have been released [28], [30], [33], [34], [37], [73], [74], [75]. Nowadays, to build OpenQA systems following the "Retriever-Reader" architecture has been widely acknowledged as the most efficient and promising way, which is also the main focus of this paper.

3 MODERN OPENQA: RETRIEVING AND READING

In this section, we introduce the *"Retriever-Reader"* architecture of the OpenQA system, as illustrated in Fig. 3. *Retriever* is aimed at retrieving relevant documents w.r.t. a given question, which can be regarded as an IR system, while *Reader* aims at inferring the final answer from the received documents, which is usually a neural MRC model. They are two major components of a modern OpenQA system. In addition, some other auxiliary modules, which are marked in dash lines in Fig. 3, can also be incorporated into an OpenQA system, including *Document Post-processing* that filters and re-ranks retrieved documents in a fine-grained manner to select the most relevant ones, and *Answer Post-processing* that is to determine the final answer among multiple answer candidates. The systems following this architecture can be classified into two groups, i.e. *pipeline systems* and *end-to-end systems*. In the following, we will introduce each component with the respective approaches in the pipeline systems, then followed by the end-to-end trainable ones. In Fig. 4 we provide a taxonomy of the modern OpenQA system to make our descriptions better understandable.

3.1 Retriever

Retriever is usually regarded as an IR system, with the goal of retrieving related documents or passages that probably contain the correct answer given a natural language question as well as ranking them in a descending order according to their relevancy. Broadly, current approaches to *Retriever* can be classified into three categories, i.e. *Sparse Retriever*, *Dense Retriever*, and *Iterative Retriever*, which will be detailed in the following.
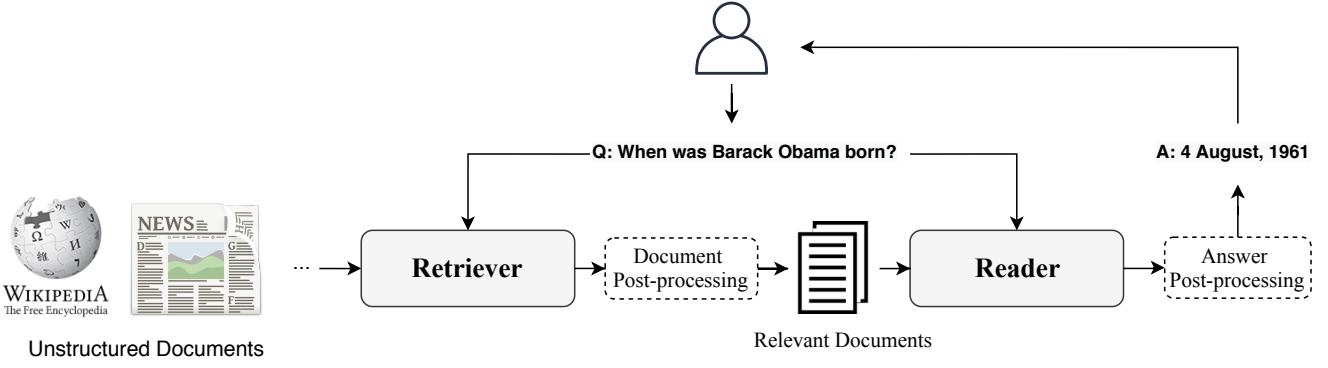
Fig. 3: An illustration of "Retriever-Reader" architecture of OpenQA system. The modules marked with dash lines are auxiliary.

3.1.1 Sparse Retriever

It refers to the systems that search for the relevant documents by adopting classical IR methods as introduced in Section 2.2.2, such as TF-IDF [3], [34], [76], [77] and BM25 [78], [79]. DrQA [3] is the very first approach to modern OpenQA systems and developed by combining classical IR techniques and neural MRC models to answer open-domain factoid questions. Particularly, the retriever in DrQA adopts bi-gram hashing [80] and TF-IDF matching to search over Wikipedia, given a natural language question. BERTserini [78] employs Anserini [81] as its retriever, which is an open-source IR toolkit based on Lucene. In [78], different granularities of text including document-level, paragraph-level and sentence-level are investigated experimentally, and the results show paragraph-level index achieves the best performance. Traditional retrieval methods such as TF-IDF and BM25 use sparse representations to measure term match. However, the terms used in user questions are often not the same as those appearing in the documents. Various methods based on dense representations [16], [29], [30], [35] have been developed in recent years, which learn to encode questions and documents into a latent vector space where text semantics beyond term match can be measured.

3.1.2 Dense Retriever

Along with the success of deep learning that offers remarkable semantic representation, various deep retrieval models have been developed in the past few years, greatly enhancing retrieval effectiveness and thus lifting final QA performance. According to the different ways of encoding the question and document as well as of scoring their similarity, dense retrievers in existing OpenQA systems can be roughly divided into three types: *Representation-based Retriever* [16], [30], [37], [73], *Interaction-based Retriever* [15], [32], and *Representation-interaction Retriever* [17], [82], [83], as illustrated in Fig. 5.

Representation-based Retriever: Representation-based Retriever, also called Dual-encoder or Two-tower retriever, employs two independent encoders like BERT [27] to encode the question and the document respectively, and estimates their relevance by computing a single similarity score between two representations. For example, ORQA [37] adopts two independent BERT-based encoders to encode a question and a document respectively and the relevance score between them is computed by the inner product of their vectors. In order to obtain a sufficiently powerful retriever, they pre-train the retriever using Inverse Cloze Task (ICT), i.e., to predict its context given a sentence. DPR [16] also employs two independent BERT encoders like ORQA but denies the necessity of the expensive pre-training stage. Instead, it focuses on learning a strong retriever using pairwise questions and answers sorely. DPR carefully designs the ways to select negative samples to a question, including any random documents from the corpus, top documents returned by BM25 that do not contain the correct answer, and in-batch negatives which are the gold documents paired with other questions in the same batch. It is worth mentioning that their experiments show the inner product function is optimal for calculating the similarity score for a dual-encoder retriever. Representation-based method [16], [16], [30], [37] can be very fast since the representations of documents can be computed and indexed offline in advance. But it may sacrifice the retrieval effectiveness because the representations of the question and document are obtained independently, leading to only shallow interactions captured between them.

Interaction-based Retriever: Such a kind of retrievers take a question together with a document at the same time as input, and are powerful by usually modeling the token-level interactions between them, such as transformer-based encoder [27], [72]. [15] propose to jointly train *Retriever* and *Reader* using supervised multi-task learning [24]. Based on BiDAF [24], a retrieval layer is added to compute the relevance score between question and document while a comprehension layer is adopted to predict the start and end position of the answer span in [15]. [32] develop a paragraph-level dense *Retriever* and a sentence-level dense *Retriever*, both based on BERT [27]. They regard the process of dense retrieval as a binary classification problem. In particular, they take each pair of question and document as input and use the embedding of *[CLS]* token to determine whether they are relevant. Their experiments show that both paragraph-level and sentence-level retrieval are necessary for obtaining good performance of the system. Interaction-based method [15], [32] is powerful as it allows for very rich interactions between question and document. However,
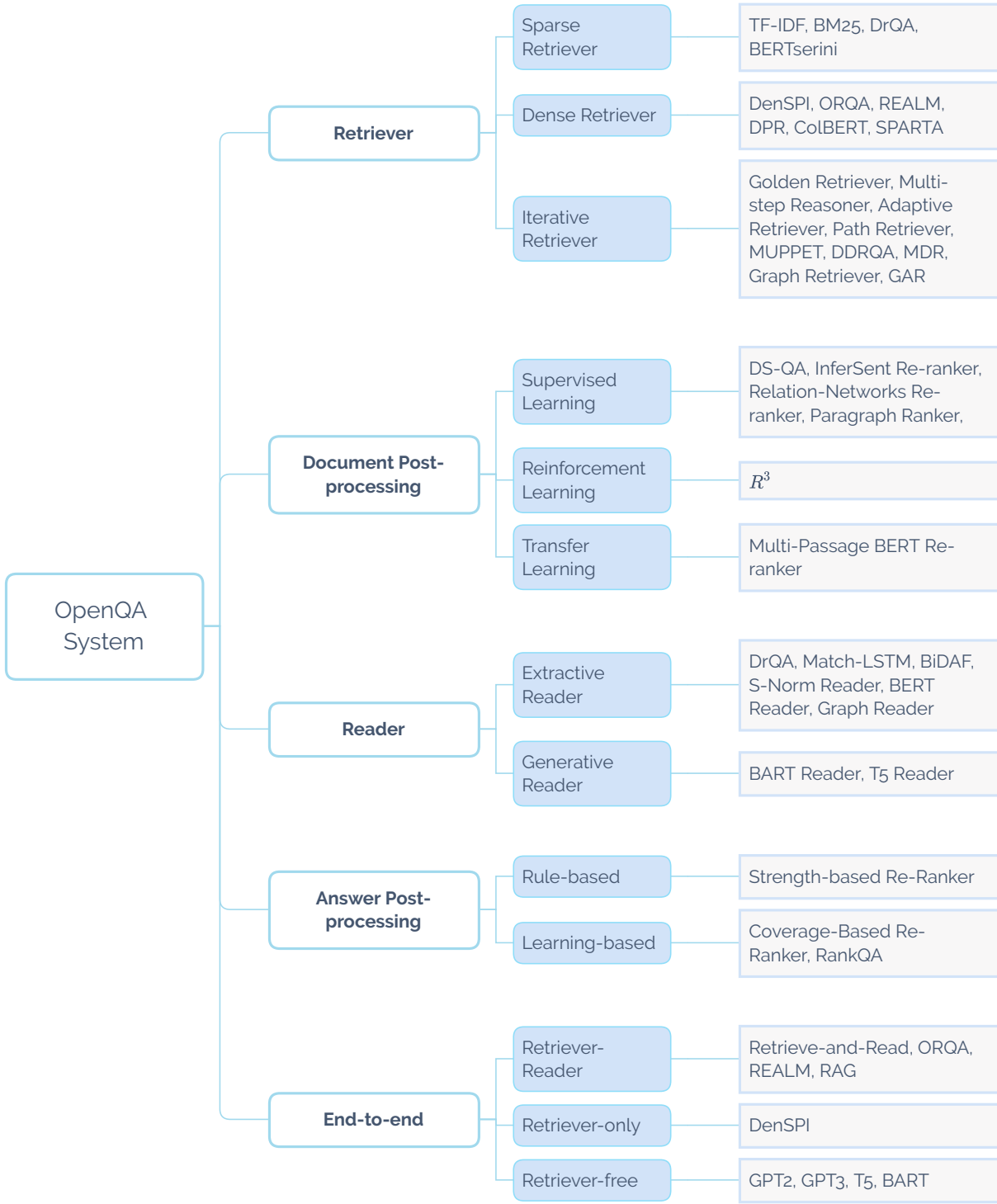
Retriever

- Sparse Retriever — TF-IDF, BM25, DrQA, BERTserini
- Dense Retriever — DenSPI, ORQA, REALM, DPR, ColBERT, SPARTA
- Iterative Retriever — Golden Retriever, Multi-step Reasoner, Adaptive Retriever, Path Retriever, MUPPET, DDRQA, MDR, Graph Retriever, GAR

Document Post-processing

- Supervised Learning — DS-QA, InferSent Re-ranker, Relation-Networks Re-ranker, Paragraph Ranker,
- Reinforcement Learning — R^3
- Transfer Learning — Multi-Passage BERT Re-ranker

Reader

- Extractive Reader — DrQA, Match-LSTM, BiDAF, S-Norm Reader, BERT Reader, Graph Reader
- Generative Reader — BART Reader, T5 Reader

Answer Post-processing

- Rule-based — Strength-based Re-Ranker
- Learning-based — Coverage-Based Re-Ranker, RankQA

End-to-end

- Retriever-Reader — Retrieve-and-Read, ORQA, REALM, RAG
- Retriever-only — DenSPI
- Retriever-free — GPT2, GPT3, T5, BART

OpenQA System

Fig. 4: A taxonomy of *"Retriever-Reader"* OpenQA system.

such a method usually requires heavy computation, which is sometimes prohibitively expensive, making it hardly applicable to large-scale documents.

Representation-interaction Retriever: In order to achieve both high accuracy and efficiency, some recent systems [17], [82], [83] combine representation-based and interaction-based methods. For instance, ColBERT-QA [17] develops its retriever based on ColBERT [84], which extends the dual-encoder architecture by performing a simple token-level interaction step over the question and document representations to calculate the similarity score. Akin to DPR [16], ColBERT-QA first encodes the question and document independently with two BERT encoders. Formally, given a question q and a document d, with corresponding vectors denoted as E_q (length n) and E_d (length m), the relevance
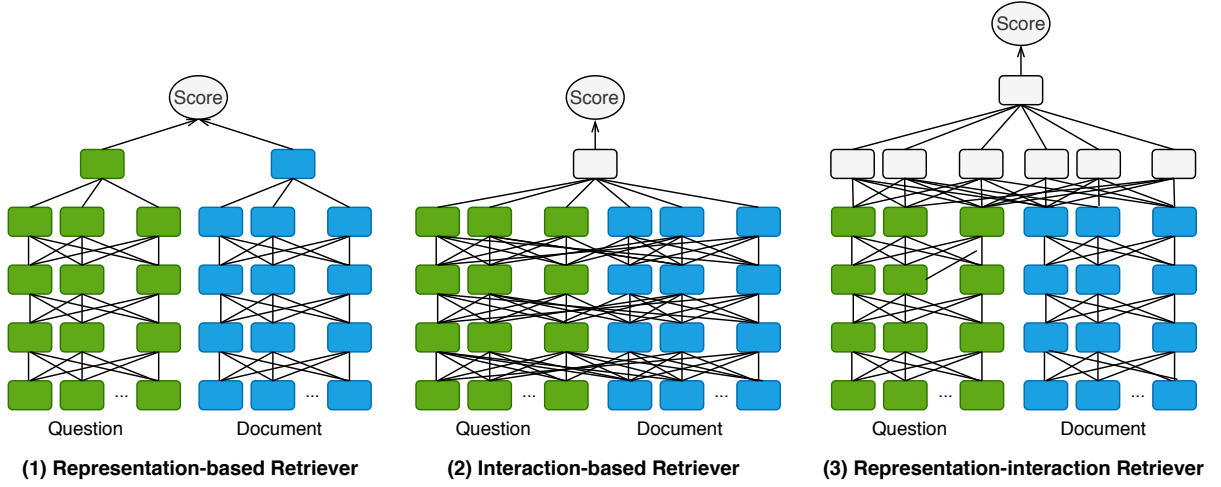
Fig. 5: Three types of dense retrievers.

score between them is computed as follows:

$$S_{q,d} = \sum_{i=1}^{n} \max_{j=1}^{m} E_{q_i} \cdot E_{d_j}^T. \qquad (1)$$

Then, ColBERT computes the score of each token embedding of the question over all those of the document first, and then sums all these scores as the final relevance score between q and d. As another example, SPARTA [82] develops a neural ranker to calculate the token-level matching score using dot product between a *non-contextualized encoded* (e.g., BERT word embedding) question and a *contextualized encoded* (e.g., BERT encoder) document. Concretely, given the representations of the question and document, the weight of each question token is computed with max-pooling, ReLU and log sequentially; the final relevance score is the sum of each question token weight. The representation-interaction method is a promising approach to dense retrieval, due to its good trade-off between effectiveness and efficiency. But it still needs to be further explored.

Though effective, Dense Retriever often suffers heavy computational burden when applied to large-scale documents. In order to speed up the computation, some works propose to compute and cache the representations of all documents offline in advance [16], [29], [30], [35], [37]. In this way, these representations will not be changed once computed, which means the documents are encoded independently of the question, to some extent sacrificing the effectiveness of retrieval.

3.1.3 Iterative Retriever
Iterative Retriever aims to search for the relevant documents from a large collection in multiple steps given a question, which is also called Multi-step Retriever. It has been explored extensively in the past few years [29], [35], [36], [85], [86], [87], [88], [89], especially when answering complex questions like those requiring multi-hop reasoning [90], [91]. In order to obtain a sufficient amount of relevant documents, the search queries need to vary for different steps and be reformulated based on the context information in the previous step. In the following, we will elaborate on Iterative

Retriever based on its workflow: 1) Document Retrieval: the IR techniques used to retrieve documents in every retrieval step; 2) Query Reformulation: the mechanism used to generate a query for each retrieval; 3) Retrieval Stopping Mechanism: the method to decide when to terminate the retrieval process.

Document Retrieval: We first revisit the IR techniques used to retrieve documents in every retrieval step given a query. Some works [36], [86], [89] apply Sparse Retriever iteratively, and some [29], [35], [85], [88] use Dense Retriever interatively. Among the works using Sparse Retriever, GOLDEN Retriever [36] adopts BM25 retrieval, while Graph Retriever [89] and DDRQA [86] retrieve top K documents using TF-IDF. For those with Dense Retriever, most prior systems including Multi-step Reasoner [29], MUPPET [35] and MDR [85] use MIPS retrieval to obtain the most semantically similar documents given a representation of the question; Path Retriever [88] develops a Recurrent Neural Network (RNN) retrieval to learn to retrieve reasoning paths for a question over a Wikipedia graph, which is built to model the relationships among paragraphs based on the Wikipedia hyperlinks and article structures.

Query Reformulation: In order to obtain a sufficient amount of relevant documents, the search queries used for each step of retrieval are usually varied and generated based on the previous query and the retrieved documents. The generated queries take each from the two forms: 1) explicit form, i.e. a natural language query [36], [86], [87]; and 2) implicit form, i.e. a dense representation [29], [35], [85].

Some works produce a new query taking the form of natural language. For example, GOLDEN Retriever [36] recasts the query reformulation task as an MRC task because they both take a question and some context documents as input and aim to generate a string in natural language. Instead of pursuing an answer in MRC, the target for query reformulation is a new query that helps obtain more supporting documents in the next retrieval step. GAR [87] develops a query expansion module using a pretrained Seq2Seq model BART [92], which takes the initial question as input and generates new queries. It is trained by taking various gener-

ation targets as output consisting of the answer, the sentence where the answer belongs to, and the title of a passage that contains the answer.

Some other works produce dense representations to be used for searching in a latent space. For example, Multi-step Reasoner [29] adopts a Gated Recurrent Unit (GRU) [93] taking token-level hidden representations from *Reader* and the question as input to generate a new query vector, which is trained using Reinforcement learning (RL) by measuring how well the answer extracted by *Reader* matches the ground-truth after reading the new set of paragraphs retrieved with the new question. MUPPET [35] applies a bidirectional attention layer adapted from [94] to a new query representation $\tilde{q}$, taking each obtained paragraph P and the initial question Q as input. MDR [85] uses a pre-trained masked language model (such as RoBert) as its encoder, which encodes the concatenation of the representation of the question as well as all the previous passages as a new dense query.

Comparably, the explicit query is easily understandable and controllable to humans but is constrained by the terms in the vocabulary, while the implicit query is generated in a semantic space, which can get rid of the limitation of the vocabulary but lacks interpretability.

Retrieval Stopping Mechanism: The iterating retrieval manner yields greater possibilities to gather more relevant passages, but the retrieval efficiency would drop dramatically along with the increasing number of iterations. Regarding the mechanism for stopping an iterative retrieval, most existing systems choose to specify a fixed number of iterations [29], [36], [85], [86], [89] or a maximum number of retrieved documents [35], [87], which can hardly guarantee the retrieval effectiveness. [77] argue that setting a fixed number of documents to be obtained for all input questions is sub-optimal and instead they develop an Adaptive Retriever based on the Document Retriever in DrQA [3]. They propose two methods to dynamically set the number of retrieved documents for each question, i.e. a simple threshold-based heuristic method as well as a trainable classifier using ordinal ridge regression. Since for the questions that require arbitrary hops of reasoning, it is difficult to specify the number of iterations, Path Retriever [88] terminates its retrieval only when the end-of-evidence token (e.g. *[EOE]*) is detected by its Recurrent Retriever. This allows it to perform adaptive retrieval steps but only obtains one document at each step. To the best of our knowledge, it is still a critical challenge to develop an efficient iterative retriever while not sacrificing accuracy.

In addition, typical IR systems pursue two optimization targets, i.e. precision and recall. The former computes the ratio of relevant documents returned to the total number of documents returned while the latter is the number of relevant documents returned out of the total number of relevant documents in the underlying repository. However, for OpenQA systems, recall is much more important than precision due to the post-processing usually applied to returned documents [95], as described below.

3.2 Document Post-processing

Post-processing over the retrieved documents from *Retriever* is often needed since the retrieved documents inevitably contain irrelevant ones, and sometimes, the number of returned documents is extremely large that overwhelms the capability of *Reader*. *Document Post-processing* in the modern OpenQA architecture is similar with that in the traditional one, as introduced in Section 2.2.2. It aims at reducing the number of candidate documents and only allowing the most relevant ones to be passed to the next stage.

In the past few yeas, this module has been explored with much interest [28], [33], [34], [79], [96], [97], [98], [99]. For example, R^3 [28] adopts a neural Passage Ranker, and trains it jointly with *Reader* through Reinforcement Learning (RL). DS-QA [33] adds a Paragraph Selector to remove the noisy ones from the retrieved documents by measuring the probability of each paragraph containing the answer among all candidate paragraphs. [96] explore two different passage rankers that assign scores to retrieved passages based on their likelihood of containing the answer to a given question. One is InferSent Ranker, a forward neural network that employs InferSent sentence representations [100], to rank passages based on semantic similarity with the question. The other one is Relation-Networks Ranker that adopts Relation Networks [101], focusing on measuring word-level relevance between the question and the passages. Their experiments show that word-level relevance matching significantly improves the retrieval performance and semantic similarity is more beneficial to the overall performance. [34] develop a Paragraph Ranker using two separate RNNs following the dual-encoder architecture. Each pair of question-passage is fed into the Ranker to obtain their representations independently and inner product is applied to compute the relevance score. [98] propose a time-aware re-ranking module that incorporates temporal information from different aspects to rank the candidate documents over temporal collections of news articles.

The focus of research on this module is learning to further re-rank the retrieved documents [28], [33], [34], [79]. However, with the development of Dense Retriever, recent OpenQA systems tend to develop a trainable retriever that is capable of learning to rank and retrieving the most relevant documents simultaneously, which would result in the absence of this module.

3.3 Reader

Reader is the other core component of a modern OpenQA system and also the main feature that differentiates QA systems against other IR or IE systems, which is usually implemented as a neural MRC model. It is aimed at inferring the answer in response to the question from a set of ordered documents, and is more challenging compared to the original MRC task where only a piece of passage is given in most cases [18], [19], [90], [102], [103]. Broadly, existing Readers can be categorised into two types: Extractive Reader that predicts an answer span from the retrieved documents, and Generative Reader that generates answers in natural language using sequence-to-sequence (Seq2Seq) models. Most prior OpenQA systems are equipped with an Extractive Reader [3], [16], [28], [29], [30], [33], [35], [78], [89], while some recent ones develop a Generative Reader [38], [85], [104].

3.3.1 Extractive Reader

Extractive Reader is based on the assumption that the correct answer to a given question definitely exists in the context, and usually focuses on learning to predict the start and end position of an answer span from the retrieved documents. The approaches can be divided into two types according to whether the retrieved documents are processed independently or jointly for answer extraction.

Many prior systems [16], [33], [86], [88], [89] rank the retrieved documents by the probability of including the answer and extract the answer span from the concatenation of the question and the most probable document(s). For example, DS-QA [33] extracts an answer span from the paragraph selected by a dedicated Paragraph Selector module through measuring the probability of each paragraph containing the answer among all candidate paragraphs. DPR [16] computes the probability of a passage containing the answer and that of a token being the starting and ending position of an answer span using BERT Reader, and selects the answer with the highest probability after combining them. Some systems develop graph-based Reader [88], [89] to learn to extract an answer span from a retrieved graph. For example, Graph Reader [89] takes the graph as input and learns the passage representation mainly using Graph Convolution Networks [105] first, and then extracts the answer span from the most probable one. Path Retriever [88] leverages BERT Reader to simultaneously re-rank the reasoning paths and extract the answer span from the one with the highest probability of containing the correct answer using multi-task learning. However, with retrieved documents processed independently, the model fails to take advantage of different evidences from the long narrative document or multiple documents for answer extraction, harming the performance especially in cases where the given questions require multiple hop reasoning.

In contrast, some systems [3], [78], [82], [94] extract an answer span based on all retrieved documents in a joint manner. For example, DrQA [3] decomposes the retrieved documents into paragraphs and extracts various features consisting of Part-of-Speech (POS), Named Entity (NE) and Term-Frequency (TF), etc. Then the DrQA Reader, which is implemented with a multi-layer Bi-LSTM, takes as input the question and the paragraphs, and predicts an answer span. In this process, to make answer scores comparable across paragraphs, it adopts the unnormalized exponential function and takes argmax over all answer spans to obtain the final result. BERTserini [78] develops its Reader based on BERT by removing the softmax layer to enable answer scores to be compared and aggregated among different paragraphs. [94] argue using un-normalized scores (e.g., exponential scores or logits score) for all answer spans is suboptimal and propose a Shared-Normalization mechanism by modifying the objective function to normalize the start and end scores across all paragraphs, achieving consistent performance gain. After that, many OpenQA systems [17], [29], [35], [36], [79], [82] develop their readers by applying this mechanism based on original MRC models like BiDAF [24], BERT [27] and SpanBERT [106].

3.3.2 Generative Reader

Generative Reader aims to generate answers as natural as possible instead of extracting answer spans, usually relying on Seq2Seq models. For example, S-Net [107] is developed by combining extraction and generation methods to complement each other. It employs an evidence extraction model to predict the boundary of the text span as the evidence to the answer first and then feeds it into a Seq2Seq answer synthesis model to generate the final answer. Recently, some OpenQA systems [38], [85], [104] adopt pretrained Seq2Seq language models to develop their Readers, like BART [92] and T5 [108]. For example, RAG [38] adopts a pretrained BART model as its reader to generate answers by taking the input question as well as the documents retrieved by DPR [16]. FID [104] first encodes each retrieved document independently using T5 or BART encoder and then performs attention over all the output representations using the decoder to generate the final answer. However, the current generation results often suffer syntax error, incoherence, or illogic [109]. Generative Reader needs to be further explored and advanced.

3.4 Answer Post-processing

Neural MRC techniques have been advancing rapidly in recent years, but most existing MRC models still specialise in extracting answers only from a single or several short passages and tend to fail in cases where the correct answer comes from various evidences in a narrative document or multiple documents [110]. The *Answer Post-processing* module is developed to help detect the final answer from a set of answer candidates extracted by the *Reader*, taking into account their respective supporting facts. The methods adopted in existing systems can be classified into two categories, i.e. rule-based method [34], [110] and learning-based method [76], [110] For example, [110] propose two answer re-rankers, a "strength-based re-ranker" and a "coverage-based re-ranker", to aggregate evidences from different passages to decide the final answer. The "strength-based re-ranker" is a rule-based method that simply performs counting or probability calculation based on the candidate predictions and does not require any training. The "coverage-based re-ranker" is developed using an attention-based match-LSTM model [111]. It first concatenates the passages containing the same answer into a pseudo passage and then measures how well this pseudo passage entails the answer for the given question. The experiments in [110] show that a weighted combination of the outputs of the above different re-rankers achieves the best performance on several benchmarks. RankQA [76] develops an answer re-ranking module consisting of three steps: feature extraction, answer aggregation and re-ranking. Firstly, taking the top k answer candidates from *Reader* as input, the module extracts a set of features from both *Retriever* such as document-question similarity, question length and paragraph length, and *Reader* such as original score of answer candidate, part-of-speech tags of the answer and named entity of the answer. Secondly, the module groups all answer candidates with an identical answer span to generate a set of aggregation features like the sum, mean, minimum, and maximum of the span scores, etc. Based on these features, a re-ranking network such as a

feed-forward network or an RNN is used to learn to further re-rank the answers to select the best one as the final answer.

3.5 End-to-end Methods

In recent years, various OpenQA systems [15], [30], [37] have been developed, in which *Retriever* and *Reader* can be trained in an end-to-end manner. In addition, there are some systems with only Retriever [73], and also some that are able to answer open questions without the stage of retrieval, which are mostly pre-trained Seq2Seq language models [92], [108], [112], [113]. In the following, we will introduce these three types of systems, i.e. *Retriever-Reader*, *Retriever-only* and *Retriever-free*.

3.5.1 Retriever-Reader

Deep learning techniques enable *Retriever* and *Reader* in an OpenQA system to be end-to-end trainable [15], [30], [37], [38]. For example, [15] propose to jointly train *Retriever* and *Reader* using multi-task learning based on the BiDAF model [24], simultaneously computing the similarity of a passage to the given question and predicting the start and end position of an answer span. [37] argue that it is sub-optimal to incorporate a standalone IR system in an OpenQA system and develop ORQA that jointly trains *Retriever* and *Reader* from question-answer pairs, with both developed using BERT [27]. REALM [30] is a pre-trained masked language model including a neural *Retriever* and a neural *Reader*, which is able to compute the gradient w.r.t. the model parameters and backpropagate the gradient all the way throughout the network. Since both modules are developed using neural networks, the response speed to a question is a most critical issue during inference, especially over a large collection of documents.

3.5.2 Retriever-only

To enhance the efficiency of answering questions, some systems are developed by only adopting a *Retriever* while omitting *Reader* which is usually the most time-consuming stage in other modern OpenQA systems. DenSPI [73] builds a question-agnostic phrase-level embedding index offline given a collection of documents like Wikipedia articles. In the index, each candidate phrase from the corpus is represented by the concatenation of two vectors, i.e. a sparse vector (e.g., tf-idf) and a dense vector (e.g., BERT encoder). In the inference, the given question is encoded in the same way, and FAISS [114] is employed to search for the most similar phrase as the final answer. Experiments show that it obtains remarkable efficiency gains and reduces computational cost significantly while maintaining accuracy. However, the system computes the similarity between each phrase and the question independently, which ignores the contextual information that is usually crucial to answering questions .

3.5.3 Retriever-free

Recent advancement in pre-training Seq2Seq language models such as GPT-2 [112], GPT-3 [113], BART [92] and T5 [108] brings a surge of improvements for downstream NLG tasks, most of which are built using Transformer-based architectures. In particular, GPT-2 and GPT-3 adopt Transformer left-to-right decoder while BART and T5 use Transformer encode-decoder closely following its original form [72]. Prior studies [115], [116] show that a large amount of knowledge learned from large-scale textual data can be stored in the underlying parameters, and thus these models are capable of answering questions without access to any external knowledge. For example, GPT-2 [112] is able to correctly generate the answer given only a natural language question without fine-tuning. Afterwards, GPT-3 [113] achieves competitive performance with few-shot learning compared to prior state-of-the-art fine-tuning approaches, in which several demonstrations are given at inference as conditioning [112] while weight update is not allowed. Recently, [116] comprehensively evaluate the capability of language models for answering questions without access to any external knowledge. Their experiments demonstrate that pre-trained language models are able to gain impressive performance on various benchmarks and such Retrieval-free methods make a fundamentally different approach to building OpenQA systems.

In Table 1, we summarize existing modern OpenQA systems as well as the approaches adopted for different components.

4 CHALLENGES AND BENCHMARKS

In this section, we first discuss key challenges to building OpenQA systems followed by an analysis of existing QA benchmarks that are commonly used not only for OpenQA but also for MRC.

4.1 Challenges to OpenQA

To build an OpenQA system that is capable of answering any input questions is regarded as the ultimate goal of QA research. However, the research community still has a long way to go. Here we discuss some salient challenges that need be addressed on the way. By doing this we hope the research gaps can be made clearer so as to accelerate the progress in this field.

4.1.1 Distant Supervision

In the OpenQA setting, it is almost impossible to create a collection containing "sufficient" high-quality training data for developing OpenQA systems in advance. Distant supervision is therefore popularly utilized, which is able to label data automatically based on an existing corpus, such as Wikipedia. However, distant supervision inevitably suffers from wrong label problem and often leads to a considerable amount of noisy data, significantly increasing the difficulty of modeling and training. Therefore, the systems that are able to tolerate such noise are always demanded.

4.1.2 Retrieval Effectiveness and Efficiency

Retrieval effectiveness means the ability of the system to separate relevant documents from irrelevant ones for a given question. The system often suffers from "term-mismatch", which results in failure of retrieving relevant documents; on the other hand the system may receive noisy documents that contain the exact terms in the question or even the correct answer span, but are irrelevant to the

TABLE 1: Approaches adopted for different components of existing modern OpenQA systems.

System	Category	Retriever	Document Post -Processing	Reader	Answer Post-processing
DrQA [3]	Pipeline	Sparse	-	Extractive	-
R3 [28]	Pipeline	Sparse	RL[1]	Extractive	-
DS-QA [33]	Pipeline	Sparse	SL[2]	Extractive	-
[110]	Pipeline	Sparse	-	Extractive	Rule-based Learning-based
[96]	Pipeline	Sparse	SL	Extractive	-
Paragraph Ranker [34]	Pipeline	Sparse	SL	Extractive	Rule-based
RankQA [76]	Pipeline	Sparse	-	Extractive	Learning-based
BERTserini [78]	Pipeline	Sparse	-	Extractive	-
Multi-Passage BERT [79]	Pipeline	Sparse	TL[3]	Extractive	-
[15]	Pipeline	Dense	-	Extractive	-
DPR [16]	Pipeline	Dense	-	Extractive	-
ColBERT-QA [17]	Pipeline	Dense	-	Extractive	-
SPARTA [82]	Pipeline	Dense	-	Extractive	-
FID [104]	Pipeline	Sparse Dense	-	Generative	-
Adaptive Retrieval [77]	Pipeline	Iterative	-	Extractive	-
Multi-step Reasoner [29]	Pipeline	Iterative	-	Extractive	-
GOLDEN Retriever [36]	Pipeline	Iterative	-	Extractive	-
MUPPET [35]	Pipeline	Iterative	-	Extractive	-
Path Retriever [88]	Pipeline	Iterative	-	Extractive	-
Graph Retriever [89]	Pipeline	Iterative	-	Extractive	-
DDRQA [86]	Pipeline	Iterative	-	Extractive	-
GAR [87]	Pipeline	Iterative	-	Extractive	Rule-based
MDR [85]	Pipeline	Iterative	-	Extractive Generative	-
DenSPI [73]	End-to-end	Retriever	-	-	-
Retrieve- -and-Read [32]	End-to-end	Dense	-	Extractive	-
ORQA [37]	End-to-end	Dense	-	Extractive	-
REALM [30]	End-to-end	Dense	-	Extractive	-
RAG [38]	End-to-end	Dense	-	Generative	-

[1] RL: Reinforcement Learning;
[2] SL: Supervised Learning;
[3] TL: Transfer Learning.

question. Both issues increase the difficulty of accurately understanding the context during answer inference. Some neural retrieval methods [15], [16], [30], [37], [73], [117] are proposed recently for improving retrieval effectiveness. For example, [37] and [30] jointly train the retrieval and reader modules, which take advantage of pre-trained language models and regard the retrieval model as a latent variable. However, these neural retrieval methods often suffer from low efficiency. Some works [15], [16], [37], [117] propose to pre-compute the question-independent embedding for each document or phrase and construct the embedding index only once. Advanced sub-linear *Maximum Inner Product Search (MIPS)* algorithms [69], [70], [71] are usually employed to obtain the top K related documents given a question. However, the response speed still has a huge gap from that of typical IR techniques when the system faces a massive set of documents.

Retrieval effectiveness and efficiency are both crucial factors for the deployment of an OpenQA system in practice, especially when it comes to the real-time scenarios. How to

consistently enhance both aspects (also with a good trade-off) will be a long-standing challenge in the advancement of OpenQA.

4.1.3 Knowledge Incorporation

To incorporate knowledge beyond context documents and given questions is a key enhancement to OpenQA systems [7], e.g. world knowledge, commonsense or domain-specific knowledge. Before making use of such knowledge, we need to first consider how to represent them. There are generally two ways: `explicit` and `implicit`.

For the `explicit` manner, knowledge is usually transformed into the form of triplets and stored in classical KBs such as DBPedia [118], Freebase [119] and Yago2 [120], which are easily understood by humans. Some early QA systems attempt to incorporate knowledge to help find the answer in this way. For example, IBM Watson DeepQA [58] combines a Web search engine and a KB to compete with human champions on the American TV show "Jeopardy"; QuASE [121] searches for a list of most prominent sentences

from a Web search engine (e.g, Google.com), and then utilizes entity linking over Freebase [119] to detect the correct answer from the selected sentences. In recent years, with the popularity of Graph Neural Network (GNN), some works [89], [122], [123] propose to gain relevant information not only from a text corpus but also from a KB to facilitate evidence retrieval and question answering. For example, [122] construct a question-specific sub-graph containing sentences from the corpus, and entities and relations from the KB. Then, graph CNN based methods [105], [124], [125] are used to infer the final answer over the sub-graph. However, there also exist problems for storing knowledge in an explicit manner, such as incomplete and out-of-date knowledge. Moreover, to construct a KB is both labor-intensive and time-consuming.

On the other hand, with the `implicit` approach, a large amount of knowledge [115] can be stored in underlying parameters learned from massive texts by pre-trained language models such as BERT [27], XLNet [126] and T5 [108], which can be applied smoothly in downstream tasks. Recently, pre-trained language models have been popularly researched and applied to developing OpenQA systems [16], [30], [32], [37], [78], [87], [88]. For example, [32], [78], [88] develop their *Reader* using BERT [27] while [16], [37] use BERT to develop both *Retriever* and *Reader*. In addition, pre-trained language models like GPT-2 [112] are able to generate the answer given only a natural language question. However, such systems act like a "black box" and it is nearly impossible to know what knowledge has been exactly stored and used for a particular answer. They lack interpretability that is crucial especially for real-world applications.

Knowledge enhanced OpenQA is desired not only because it is helpful to generating the answer but also because it serves as the source for interpreting the obtained answer. How to represent and make full use of the knowledge for OpenQA still needs more research efforts.

4.1.4 Conversational OpenQA

Non-conversational OpenQA is challenged by several problems that are almost impossible to resolve, such as the lengthy words for a complex question (e.g. *Who is the second son of the first Prime Minister of Singapore?*), ambiguity resulting in incorrect response (e.g. *When was Michael Jordan born?*) and insufficient background knowledge from the user that leads to unreasonable results (e.g. *Why do I have a bad headache today?*). These problems would be well addressed under the conversational setting.

Conversational systems [150], [151] are equipped with a dialogue-like interface that enables interaction between human users and the system for information exchange. For the complex question example given above, it can be decomposed into two simple questions sequentially: *"Who is the first Prime Minister of Singapore?"* followed by *"Who is the second son of him?"*. When ambiguity is detected in the question, the conversational OpenQA system is expected to raise a follow-up question for clarification, such as *"Do you mean the basketball player?"*. If a question with insufficient background knowledge is given, a follow-up question can also be asked to gather more information from human users

for arriving at the final answer. To achieve these goals, three major challenges need to be addressed.

First, conversational OpenQA should have the ability to determine if a question is unanswerable, such as to detect if ambiguity exists in the question or whether the current context is sufficient for generating an answer. Research on unanswerable questions has attracted a lot of attention in the development of MRC over the past few years [20], [22], [128], [144], [152], [153]. However, current OpenQA systems rarely incorporate such a mechanism to determine unanswerability of questions, which is particularly necessary for conversational OpenQA systems.

Second, when the question is classified as unanswerable due to ambiguity or insufficient background knowledge, the conversational OpenQA system needs to generate a follow-up question [154]. Question Generation (QG) can then be considered as a sub-task of QA, which is a crucial module of conversational OpenQA. In the past few years, research on automatic question generation from text passages has received growing attention [155], [156], [157], [158]. Compared to the typical QG task targeting at generating a question based on a given passage where the answer to the generated question can be found, the question generated in conversational OpenQA should be answered by human users only.

The third challenge is how to better model the conversation history not only in *Reader* but also in *Retriever* [159]. The recently released conversational MRC datasets like CoQA [133] and QuAC [134] are aimed at enabling a *Reader* to answer the latest question by comprehending not only the given context passage but also the conversation history so far. As they provide context passages in their task setting, they omit the stage of document retrieval which is necessary when it comes to OpenQA. Recently, in [159] the QuAC dataset is extended to a new OR-QuAC dataset by adapting to an open-retrieval setting, and an open-retrieval conversational question answering system (OpenConvQA) is developed, which is able to retrieve relevant passages from a large collection before inferring the answer, taking into account the conversation QA pairs. OpenConvQA tries to answer a given question without any specified context, and thus enjoys a wider scope of application and better accords with real-world QA behavior of human beings. However, the best performance (F1: 29.4) of the system on OR-QuAC is far lower than the state-of-the-art (F1: 74.4[1]) on QuAC, indicating that it is a bigger challenge when it comes to an open-retrieval setting.

4.2 Benchmarks

A large number of QA benchmarks have been released in the past decade, which are summarized in Table 2. Here we provide a brief analysis of them with the focus on their respective characteristics, dataset distributions w.r.t. background information domain, number of questions, year of release. As aforementioned in this paper, the success of the MRC task is a crucial step to more advanced OpenQA and we believe the future advancement of MRC methods will significantly promote the OpenQA systems. Thus, we

1. stated on June 2020 https://quac.ai/

TABLE 2: **Dataset:** The name of the dataset. **Domain:** The domain of background information in the dataset. **#Q (k):** The number of questions contained in the dataset, with unit(k) denoting "thousand". **Answer Type:** The answer types included in the dataset. **Context in MRC:** The context documents or passages that are given to generate answers in MRC tasks. **OpenQA:** This column indicates whether the dataset is applicable for developing OpenQA systems, with the tick mark denoting yes.

Dataset	Domain	#Q (k)	Answer Type	Context in MRC	OpenQA
MCTest [102]	Children's story	2.0	Multiple choices	A children's story	
CNN/Daily Mail [18]	News	1,384.8	Entities	A passage from one CNN or Daily Mail news	✔
CBT [127]	Children's story	687.3	Multiple choices	A children's story	
SQuAD [19]	Wikipedia	108.0	Spans	A passage from Wikipedia	✔
MS MARCO [20]	Web search	1,010.9	Free-form Boolean Unanswerable	Multiple passages from Bing Search	✔
NewsQA [128]	News	119.6	Spans Unanswerable	A news article from CNN news	✔
SearchQA [129]	Web search	140.4	Spans	Multiple passages from Google Search	✔
TriviaQA [130]	Trivia	95.9	Spans Free-form	One or multiple passages	
RACE [21]	Science	97.6	Multiple choices	A passage from mid/high school exams	
Quasar-T [91]	Reddit	43.0	Free-form	Multiple documents from Reddit	✔
Quasar-S [91]	Technical	37.0	Entities	A passage from Stack Overflow	✔
NarrativeQA [131]	Others	46.7	Free-form	A summary and a full story from movie scripts	
DuReader [132]	Web search	200.0	Free-form Boolean	Multiple passages from Baidu Search or Baidu Zhidao	✔
SQuAD 2.0 [22]	Wikipedia	158.0	Spans Unanswerable	A passage from Wikipedia	✔
CoQA [133]	Others	127.0	Free-form Boolean Unanswerable	A passage and conversation history	
QuAC [134]	Wikipedia	98.4	Spans Boolean Unanswerable	A passage from Wikipedia and conversation history	✔
ARC [135]	Science	7.7	Multiple choices	No additional context	
ShARC [136]	Others	32.4	Boolean	A rule text, a scenario and conversation history	
CliCR [137]	Medical	104.9	Spans	A passage from clinical case reports	
HotpotQA [90]	Wikipedia	113.0	Spans Boolean Unanswerable	A pair of paragraphs from Wikipedia	✔
MultiRC [138]	Others	6.0	Multiple choices	Multiple sentences	
SWAG [139]	Commonsense	113.0	Multiple choices	A piece of video caption	
DuoRC [140]	Others	186.0	Free-form Spans Unanswerable	A movie plot story	
WikiHop [91]	Wikipedia	51.3	Multiple choices	Multiple passages from Wikipedia	✔
MedHop [91]	Medical	2.5	Multiple choices	Multiple passagee from MEDLINE	
ReCoRD [141]	News	120.7	Multiple choices	A passage from CNN/Daily Mail News	
OpenBookQA [5]	Science	5.9	Multiple choices	Open book	
CommonsenseQA [142]	Commonsense	12.2	Multiple choices	No additional context	
CODAH [143]	Commonsense	2.8	Multiple choices	No additional context	
DROP [103]	Wikipedia	96.5	Free-form	A passage from Wikipedia	
Natural Questions [144]	Wikipedia	323.0	Spans Boolean Unanswerable	An article from Wikipedia	✔
Cosmos QA [145]	Commonsense	35.6	Multiple choices	A passage	
BoolQ [146]	Wikipedia	16.0	Boolean	An article from Wikipedia	✔
ELI5 [147]	Reddit	272.0	Free-form	A set of web documents	
TWEETQA [148]	Social media	13.7	Free-form	A tweet from Twitter	
XQA [149]	Wikipedia	90.6	Entities	A passage from Wikipedia in a target language	✔

include not only the datasets for OpenQA but also those solely for MRC to make our survey more comprehensive.

The major criterion for judging the applicability of a QA dataset to develop OpenQA systems is whether it involves a separate document set (usually large-scale) [90], or whether it has relatively easy access to such an information source [18], [22] where the answers to questions can be inferred. For example, HotpotQA [90] provides a full-wiki setting itself to require a system to find the answer to a question in the scope of the entire Wikipedia. [3] extend SQuAD [19] to $SQuAD_{open}$ by using the entire Wikipedia as its information source. We summarize and illustrate the distributions of datasets listed in Table 2 w.r.t. year of release in Fig. 7, background information domain in Fig.8 and

number of questions in Fig. 6. Also, we summarize the information source type of the datasets that are applicable to developing OpenQA systems in Table 3.

5 CONCLUSION

In this work we presented a comprehensive survey on the latest progress of Open-domain QA (OpenQA) systems. In particular, we first reviewed the development of OpenQA and illustrated a "Retriever-Reader" architecture. Moreover, we reviewed a variety of existing OpenQA systems as well as their different approaches. Finally, we discussed some salient challenges towards OpenQA followed by a summary of various QA benchmarks, hoping to reveal the research gaps so as to push further progress in this field. Based on

Fig. 6: Number of questions in each dataset
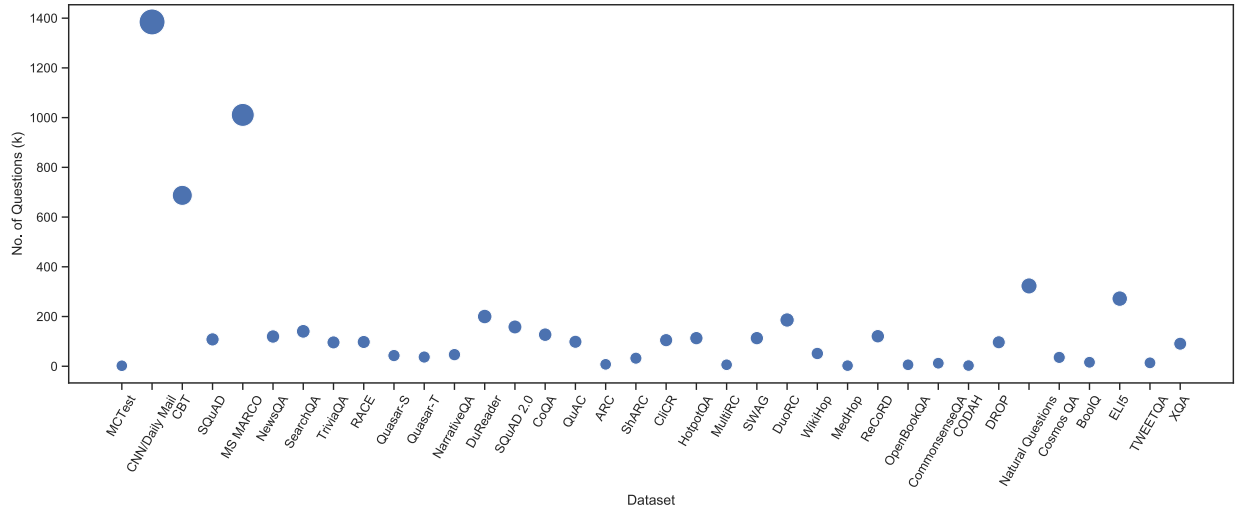
TABLE 3: The information source of the datasets that are applicable for developing OpenQA system. **Source Type:** The type of background information source. **Source**: The background information source in OpenQA setting.

Source Type	Source	Dataset
Wikipedia	Full Wikipedia	$SQuAD_{open}$ [3] HotpotQA [90] QuAC [134] WikiHop [91] Natural Questions [144] BoolQ [146] XQA [149]
Search Engine	Bing Search Google Search Baidu Search	MS MARCO [20] SearchQA [129] DuReader [132]
Online News	News from CNN/Daily News from CNN	CNN/Daily Mail [18] NewsQA [128]
Internet Forum	Reddit Stack Overflow	Quasar-T [91] Quasar-S [91]

Fig. 7: Distribution of popular datasets w.r.t. release year
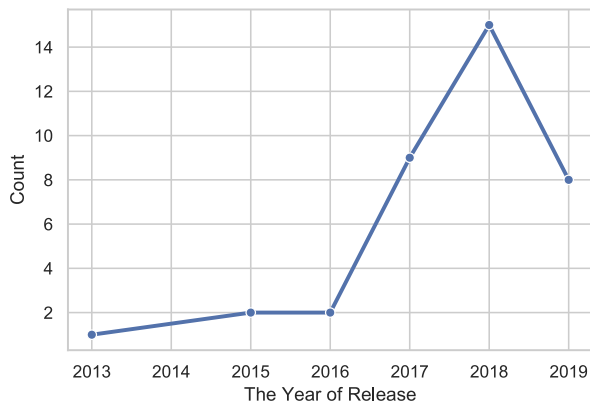
our review of prior research, we claim that OpenQA would continue to be a research hot-spot. In particular, single-step and multi-step neural retrievers will attract increasing attention due to the demand for more accurate retrieval of related documents. Also, more end-to-end OpenQA systems will be developed with the advancement of deep learning techniques. Knowledge enhanced OpenQA is very promising not only because it is helpful to generating the answer but also because it serves as the source for interpreting the obtained answer. However, how to represent and make full use of the knowledge for OpenQA still needs more research efforts. Furthermore, to equip OpenQA with a dialogue-like interface that enables interaction between human users and the system for information exchange is expected to attract increasing attention, which well aligns with real world application scenarios.

6 ACKNOWLEDGEMENTS

Fig. 8: Datasets distribution w.r.t. background information domain
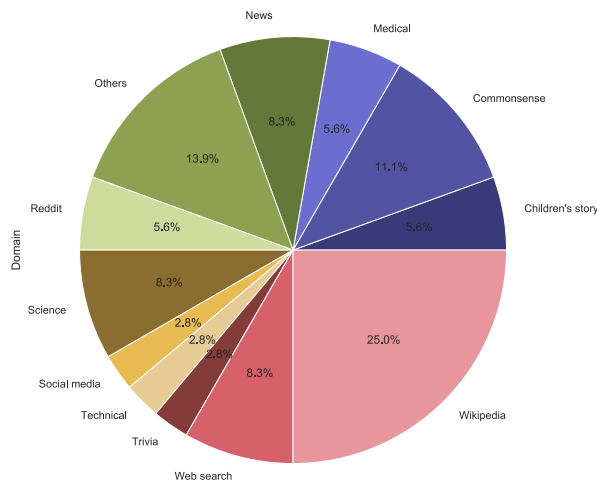
REFERENCES

[1] B. F. Green, Jr., A. K. Wolf, C. Chomsky, and K. Laughery, "Baseball: An automatic question-answerer," in *Papers Presented at the May 9-11, 1961, Western Joint IRE-AIEE-ACM Computer Conference*. ACM, 1961, pp. 219–224.

[2] J. Falconer, "Google: Our new search strategy is to compute answers, not links," 2011. [Online]. Available: https://thenextweb.com/google/2011/06/01/google-our-new-search-strategy-is-to-compute-answers-not-links/

[3] D. Chen, A. Fisch, J. Weston, and A. Bordes, "Reading Wikipedia to answer open-domain questions," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2017, pp. 1870–1879.

[4] E. M. Voorhees, "The trec-8 question answering track report," NIST, Tech. Rep., 1999.

[5] T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal, "Can a suit of armor conduct electricity? A new dataset for open book question answering," *CoRR*, vol. abs/1809.02789, 2018.

[6] S. M. Harabagiu, S. J. Maiorano, and M. A. Paundefinedca, "Open-domain textual question answering techniques," *Nat. Lang. Eng.*, vol. 9, no. 3, p. 231–267, 2003.

[7] V. C. John Burger, Claire Cardie *et al.*, "Issues, tasks and program structures to roadmap research in question & answering (q &a," NIST, Tech. Rep., 2001.

[8] O. Kolomiyets and M.-F. Moens, "A survey on question answering technology from an information retrieval perspective," *Inf. Sci.*, vol. 181, no. 24, pp. 5412–5434, 2011.

[9] A. Allam and M. Haggag, "The question answering systems: A survey," *International Journal of Research and Reviews in Information Sciences*, pp. 211–221, 2012.

[10] A. Mishra and S. K. Jain, "A survey on question answering systems with classification," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 28, no. 3, p. 345–361, 2016.

[11] M. Paşca, "Open-domain question answering from large text collections," *Computational Linguistics*, vol. 29, no. 4, pp. 665–667, 2003.

[12] Z. Huang, S. Xu, M. Hu, X. Wang, J. Qiu, Y. Fu, Y. Zhao, Y. Peng, and C. Wang, "Recent trends in deep learning based open-domain textual question answering systems," *IEEE Access*, vol. 8, pp. 94 341–94 356, 2020.

[13] T. Lei, Z. Shi, D. Liu, L. Yang, and F. Zhu, "A novel cnn-based method for question classification in intelligent question answering," in *Proceedings of the 2018 International Conference on Algorithms, Computing and Artificial Intelligence*. Association for Computing Machinery, 2018.

[14] W. Xia, W. Zhu, B. Liao, M. Chen, L. Cai, and L. Huang, "Novel architecture for long short-term memory used in question classification," *Neurocomputing*, vol. 299, pp. 20–31, 2018.

[15] K. Nishida, I. Saito, A. Otsuka, H. Asano, and J. Tomita, "Retrieve-and-read: Multi-task learning of information retrieval and reading comprehension," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, ser. CIKM '18. Association for Computing Machinery, 2018, p. 647–656.

[16] V. Karpukhin, B. Oğuz, S. Min, L. Wu, S. Edunov, D. Chen, and W.-t. Yih, "Dense passage retrieval for open-domain question answering," *arXiv preprint arXiv:2004.04906*, 2020.

[17] O. Khattab, C. Potts, and M. Zaharia, "Relevance-guided Supervision for OpenQA with ColBERT," 2020. [Online]. Available: http://arxiv.org/abs/2007.00814

[18] K. M. Hermann, T. Kočiský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*. MIT Press, 2015, pp. 1693–1701.

[19] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ questions for machine comprehension of text," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2016, pp. 2383–2392.

[20] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, and L. Deng, "MS MARCO: A human generated machine reading comprehension dataset," 2016.

[21] G. Lai, Q. Xie, H. Liu, Y. Yang, and E. H. Hovy, "RACE: large-scale reading comprehension dataset from examinations," *CoRR*, vol. abs/1704.04683, 2017.

[22] P. Rajpurkar, R. Jia, and P. Liang, "Know what you don't know: Unanswerable questions for SQuAD," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, 2018, pp. 784–789.

[23] J. Li, M. Liu, M.-Y. Kan, Z. Zheng, Z. Wang, W. Lei, T. Liu, and B. Qin, "Molweni: A challenge multiparty dialogues-based machine reading comprehension dataset with discourse structure," 2020.

[24] M. J. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," in *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net, 2017.

[25] W. Wang, N. Yang, F. Wei, B. Chang, and M. Zhou, "Gated self-matching networks for reading comprehension and question answering," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL*. Association for Computational Linguistics, 2017, pp. 189–198.

[26] A. W. Yu, D. Dohan, M. Luong, R. Zhao, K. Chen, M. Norouzi, and Q. V. Le, "Qanet: Combining local convolution with global self-attention for reading comprehension," in *International Conference on Learning Representations, ICLR*. OpenReview.net, 2018.

[27] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018.

[28] S. Wang, M. Yu, X. Guo, Z. Wang, T. Klinger, W. Zhang, S. Chang, G. Tesauro, B. Zhou, and J. Jiang, "R3: Reinforced ranker-reader for open-domain question answering," in *AAAI*, 2018.

[29] R. Das, S. Dhuliawala, M. Zaheer, and A. McCallum, "Multi-step retriever-reader interaction for scalable open-domain question answering," in *International Conference on Learning Representations*, 2019.

[30] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M.-W. Chang, "Realm: Retrieval-augmented language model pre-training," *CoRR*, 2020.

[31] M. Ding, C. Zhou, Q. Chen, H. Yang, and J. Tang, "Cognitive graph for multi-hop reading comprehension at scale," in *Proceedings of the 57th Annual Meeting of the Association for Computational*

Linguistics. Association for Computational Linguistics, 2019, pp. 2694–2703.

[32] Y. Nie, S. Wang, and M. Bansal, "Revealing the importance of semantic retrieval for machine reading at scale," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, 2019, pp. 2553–2566.

[33] Y. Lin, H. Ji, Z. Liu, and M. Sun, "Denoising distantly supervised open-domain question answering," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2018, pp. 1736–1745.

[34] J. Lee, S. Yun, H. Kim, M. Ko, and J. Kang, "Ranking paragraphs for improving answer recall in open-domain question answering," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018, pp. 565–569.

[35] Y. Feldman *et al.*, "Multi-hop paragraph retrieval for open-domain question answering," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019, pp. 2296–2309.

[36] P. Qi, X. Lin, L. Mehr, Z. Wang, and C. D. Manning, "Answering complex open-domain questions through iterative query generation," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, 2019, pp. 2590–2602.

[37] K. Lee, M.-W. Chang, and K. Toutanova, "Latent retrieval for weakly supervised open domain question answering," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019, pp. 6086–6096.

[38] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," 2020. [Online]. Available: http://arxiv.org/abs/2005.11401

[39] W. A. Woods, "Progress in natural language understanding: An application to lunar geology," in *Proceedings of the June 4-8, 1973, National Computer Conference and Exposition*. ACM, 1973, pp. 441–450.

[40] J. Kupiec, "Murax: A robust linguistic approach for question answering using an on-line encyclopedia," in *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, 1993, p. 181–190.

[41] E. M. Voorhees, "Overview of the trec 2001 question answering track," in *In Proceedings of TREC-10*, 2001, pp. 42–51.

[42] ——, "Overview of the TREC 2002 question answering track," in *Proceedings of The Eleventh Text REtrieval Conference, TREC 2002, Gaithersburg, Maryland, USA, November 19-22, 2002*, ser. NIST Special Publication, vol. 500-251. National Institute of Standards and Technology (NIST), 2002.

[43] E. Voorhees, "Overview of the trec 2003 question answering track," NIST, Tech. Rep., 2003.

[44] C. Kwok, O. Etzioni, O. Etzioni, and D. S. Weld, "Scaling question answering to the web," *ACM Transactions on Information Systems*, vol. 19, no. 3, pp. 242–262, 2001.

[45] E. Brill, S. Dumais, and M. Banko, "An analysis of the AskMSR question-answering system," in *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*. Association for Computational Linguistics, 2002, pp. 257–264.

[46] Z. Zheng, "Answerbus question answering system," in *Proceedings of the Second International Conference on Human Language Technology Research*, ser. HLT '02. Morgan Kaufmann Publishers Inc., 2002, p. 399–404.

[47] D. Moldovan, M. Pasca, S. Harabagiu, and M. Surdeanu, "Performance issues and error analysis in an open-domain question answering system," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2002, pp. 33–40.

[48] R. Sun, J. Jiang, Y. F. Tan, H. Cui, T.-S. Chua, and M.-Y. Kan, "Using syntactic and semantic relation analysis in question answering," in *TREC*, 2005.

[49] J. Xu and W. B. Croft, "Query expansion using local and global document analysis," in *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, 1996, p. 4–11.

[50] C. Carpineto and G. Romano, "A survey of automatic query expansion in information retrieval," *ACM Computing Survey*, vol. 44, no. 1, 2012.

[51] C. Quirk, C. Brockett, and W. Dolan, "Monolingual machine translation for paraphrase generation," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2004, pp. 142–149.

[52] C. Bannard and C. Callison-Burch, "Paraphrasing with bilingual parallel corpora," in *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. Association for Computational Linguistics, 2005, pp. 597–604.

[53] S. Zhao, C. Niu, M. Zhou, T. Liu, and S. Li, "Combining multiple resources to improve SMT-based paraphrasing model," in *Proceedings of ACL-08: HLT*. Association for Computational Linguistics, 2008, pp. 1021–1029.

[54] S. Wubben, A. van den Bosch, and E. Krahmer, "Paraphrase generation as monolingual translation: Data and evaluation," in *Proceedings of the 6th International Natural Language Generation Conference*, 2010.

[55] X. Li and D. Roth, "Learning question classifiers," in *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002.

[56] J. Suzuki, H. Taira, Y. Sasaki, and E. Maeda, "Question classification using HDAG kernel," in *Proceedings of the ACL 2003 Workshop on Multilingual Summarization and Question Answering*. Association for Computational Linguistics, 2003, pp. 61–68.

[57] D. Zhang and W. S. Lee, "Question classification using support vector machines," in *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, ser. SIGIR '03. Association for Computing Machinery, 2003, p. 26–32.

[58] D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. Prager, N. Schlaefer, and C. Welty, "Building watson: An overview of the deepqa project," *AI Magazine*, vol. 31, no. 3, pp. 59–79, 2010.

[59] H. Tayyar Madabushi and M. Lee, "High accuracy rule-based question classification using question syntax and semantics," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. The COLING 2016 Organizing Committee, 2016, pp. 1220–1230.

[60] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. USA: Cambridge University Press, 2008.

[61] S. Robertson, H. Zaragoza *et al.*, "The probabilistic relevance framework: Bm25 and beyond," *Foundations and Trends in Information Retrieval*, vol. 3, no. 4, pp. 333–389, 2009.

[62] W. B. Croft and J. Lafferty, *Language modeling for information retrieval*. Kluwer Academic Publ., 2003.

[63] E. M. Voorhees, "Overview of the trec 2004 question answering track," in *In Proceedings of the Thirteenth Text REtreival Conference (TREC 2004)*, 2005, pp. 52–62.

[64] D. Mollá, M. van Zaanen, and D. Smith, "Named entity recognition for question answering," in *Proceedings of the Australasian Language Technology Workshop 2006*, 2006, pp. 51–58.

[65] M. Wang, "A survey of answer extraction techniques in factoid question answering," *Computational Linguistics*, vol. 1, no. 1, pp. 1–14, 2006.

[66] M. M. Soubbotin and S. M. Soubbotin, "Patterns of potential answer expressions as clues to the right answers," in *In Proceedings of the 10th Text REtrieval Conference (TREC-10)*, 2001.

[67] D. Ravichandran and E. Hovy, "Learning surface text patterns for a question answering system," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2002, pp. 41–47.

[68] D. Shen, G.-J. M. Kruijff, and D. Klakow, "Exploring syntactic relation patterns for question answering," in *Second International Joint Conference on Natural Language Processing: Full Papers*, 2005.

[69] P. Ram and A. G. Gray, "Maximum inner-product search using cone trees," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012, pp. 931–939.

[70] A. Shrivastava and P. Li, "Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips)," in *Advances in Neural Information Processing Systems*, 2014, pp. 2321–2329.

[71] F. Shen, W. Liu, S. Zhang, Y. Yang, and H. Tao Shen, "Learning binary codes for maximum inner product search," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4148–4156.

[72] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, 2017, pp. 5998–6008.

[73] M. Seo, J. Lee, T. Kwiatkowski, A. Parikh, A. Farhadi, and H. Hajishirzi, "Real-time open-domain question answering with dense-sparse phrase index," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019, pp. 4430–4441.

[74] M. Dehghani, H. Azarbonyad, J. Kamps, and M. de Rijke, "Learning to transform, combine, and reason in open-domain question answering," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, ser. WSDM '19. Association for Computing Machinery, 2019, p. 681–689.

[75] B. Dhingra, M. Zaheer, V. Balachandran, G. Neubig, R. Salakhutdinov, and W. W. Cohen, "Differentiable reasoning over a virtual knowledge base," in *International Conference on Learning Representations*, 2020.

[76] B. Kratzwald, A. Eigenmann, and S. Feuerriegel, "RankQA: Neural question answering with answer re-ranking," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019, pp. 6076–6085.

[77] B. Kratzwald *et al.*, "Adaptive document retrieval for deep question answering," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018, pp. 576–581.

[78] W. Yang, Y. Xie, A. Lin, X. Li, L. Tan, K. Xiong, M. Li, and J. Lin, "End-to-end open-domain question answering with BERTserini," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*. Association for Computational Linguistics, 2019, pp. 72–77.

[79] Z. Wang, P. Ng, X. Ma, R. Nallapati, and B. Xiang, "Multi-passage BERT: A globally normalized BERT model for open-domain question answering," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, 2019, pp. 5878–5882.

[80] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg, "Feature hashing for large scale multitask learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*. Association for Computing Machinery, 2009, p. 1113–1120.

[81] P. Yang, H. Fang, and J. Lin, "Anserini: Enabling the use of lucene for information retrieval research," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '17. Association for Computing Machinery, 2017, p. 1253–1256.

[82] T. Zhao, X. Lu, and K. Lee, "Sparta: Efficient open-domain question answering via sparse transformer matching retrieval," *arXiv preprint arXiv:2009.13013*, 2020.

[83] Y. Zhang, P. Nie, X. Geng, A. Ramamurthy, L. Song, and D. Jiang, "Dc-bert: Decoupling question and document for efficient contextual encoding," 2020.

[84] O. Khattab and M. Zaharia, "Colbert: Efficient and effective passage search via contextualized late interaction over bert," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '20. Association for Computing Machinery, 2020, p. 39–48.

[85] W. Xiong, X. L. Li, S. Iyer, J. Du, P. Lewis, W. Y. Wang, Y. Mehdad, W.-t. Yih, S. Riedel, D. Kiela *et al.*, "Answering complex open-domain questions with multi-hop dense retrieval," *arXiv preprint arXiv:2009.12756*, 2020.

[86] Y. Zhang, P. Nie, A. Ramamurthy, and L. Song, "Ddrqa: Dynamic document reranking for open-domain multi-hop question answering," *arXiv preprint arXiv:2009.07465*, 2020.

[87] Y. Mao, P. He, X. Liu, Y. Shen, J. Gao, J. Han, and W. Chen, "Generation-Augmented Retrieval for Open-domain Question Answering," 2020. [Online]. Available: http://arxiv.org/abs/2009.08553

[88] A. Asai, K. Hashimoto, H. Hajishirzi, R. Socher, and C. Xiong, "Learning to retrieve reasoning paths over wikipedia graph for question answering," in *International Conference on Learning Representations*, 2020.

[89] S. Min, D. Chen, L. Zettlemoyer, and H. Hajishirzi, "Knowledge Guided Text Retrieval and Reading for Open Domain Question Answering," 2019. [Online]. Available: http://arxiv.org/abs/1911.03868

[90] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. Cohen, R. Salakhutdinov, and C. D. Manning, "HotpotQA: A dataset for diverse, explainable multi-hop question answering," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018, pp. 2369–2380.

[91] J. Welbl, P. Stenetorp, and S. Riedel, "Constructing datasets for multi-hop reading comprehension across documents," *Transactions of the Association for Computational Linguistics*, pp. 287–302, 2018. [Online]. Available: https://www.aclweb.org/anthology/Q18-1021

[92] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020, pp. 7871–7880.

[93] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *EMNLP*. ACL, 2014, pp. 1724–1734.

[94] C. Clark and M. Gardner, "Simple and effective multi-paragraph reading comprehension," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL*. Association for Computational Linguistics, 2018, pp. 845–855.

[95] A. Lampert, "A quick introduction to question answering," *Dated December*, 2004.

[96] P. M. Htut, S. Bowman, and K. Cho, "Training a ranking function for open-domain question answering," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*. Association for Computational Linguistics, 2018, pp. 120–127.

[97] P. Banerjee, K. K. Pal, A. Mitra, and C. Baral, "Careful selection of knowledge to solve open book question answering," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019, pp. 6120–6129.

[98] J. Wang, A. Jatowt, M. Färber, and M. Yoshikawa, "Answering event-related questions over long-term news article archives," in *ECIR*, ser. Lecture Notes in Computer Science, vol. 12035. Springer, 2020, pp. 774–789.

[99] J. Wang, A. Jatowt, M. Färber, and M. Yoshikawa, "Improving question answering for event-focused questions in temporal collections of news articles," *Information Retrieval Journal*, vol. 24, no. 1, pp. 29–54, 2021.

[100] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2017, pp. 670–680.

[101] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap, "A simple neural network module for relational reasoning," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 4967–4976.

[102] M. Richardson, C. J. Burges, and E. Renshaw, "MCTest: A challenge dataset for the open-domain machine comprehension of text," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2013, pp. 193–203.

[103] D. Dua, Y. Wang, P. Dasigi, G. Stanovsky, S. Singh, and M. Gardner, "DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs," in *Proc. of NAACL*, 2019.

[104] G. Izacard and E. Grave, "Leveraging passage retrieval with generative models for open domain question answering," *arXiv preprint arXiv:2007.01282*, 2020.

[105] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017.

[106] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy, "SpanBERT: Improving pre-training by representing and predicting spans," *arXiv preprint arXiv:1907.10529*, 2019.

[107] C. Tan, F. Wei, N. Yang, B. Du, W. Lv, and M. Zhou, "S-net: From answer extraction to answer synthesis for machine reading comprehension," in *AAAI*. AAAI Press, 2018, pp. 5940–5947.

[108] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *arXiv e-prints*, 2019.

[109] S. Liu, X. Zhang, S. Zhang, H. Wang, and W. Zhang, "Neural machine reading comprehension: Methods and trends," *CoRR*, vol. abs/1907.01118, 2019.

[110] S. Wang, M. Yu, J. Jiang, W. Zhang, X. Guo, S. Chang, Z. Wang, T. Klinger, G. Tesauro, and M. Campbell, "Evidence aggregation for answer re-ranking in open-domain question answering," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. ICLR, 2018.

[111] S. Wang and J. Jiang, "Learning natural language inference with LSTM," in *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. The Association for Computational Linguistics, 2016, pp. 1442–1451.

[112] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[113] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, 2020.

[114] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with gpus," *CoRR*, vol. abs/1702.08734, 2017.

[115] F. Petroni, T. Rocktäschel, P. Lewis, A. Bakhtin, Y. Wu, A. H. Miller, and S. Riedel, "Language models as knowledge bases?" *arXiv preprint arXiv:1909.01066*, 2019.

[116] A. Roberts, C. Raffel, and N. Shazeer, "How much knowledge can you pack into the parameters of a language model?" *arXiv preprint arXiv:2002.08910*, 2020.

[117] M. Seo, T. Kwiatkowski, A. Parikh, A. Farhadi, and H. Hajishirzi, "Phrase-indexed question answering: A new challenge for scalable document comprehension," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018, pp. 559–564.

[118] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: A nucleus for a web of open data," in *The Semantic Web*. Springer Berlin Heidelberg, 2007, pp. 722–735.

[119] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*. ACM, 2008, pp. 1247–1250.

[120] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum, "Yago2: A spatially and temporally enhanced knowledge base from wikipedia," *Artif. Intell.*, vol. 194, pp. 28–61, 2013.

[121] H. Sun, H. Ma, W.-t. Yih, C.-T. Tsai, J. Liu, and M.-W. Chang, "Open domain question answering via semantic enrichment," in *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2015, pp. 1045–1055.

[122] H. Sun, B. Dhingra, M. Zaheer, K. Mazaitis, R. Salakhutdinov, and W. Cohen, "Open domain question answering using early fusion of knowledge bases and text," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018, pp. 4231–4242.

[123] H. Sun, T. Bedrax-Weiss, and W. Cohen, "PullNet: Open domain question answering with iterative retrieval on knowledge bases and text," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, 2019, pp. 2380–2390.

[124] Y. Li, D. Tarlow, M. Brockschmidt, and R. S. Zemel, "Gated graph sequence neural networks," in *ICLR*, 2016.

[125] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.

[126] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *CoRR*, vol. abs/1906.08237, 2019.

[127] F. Hill, A. Bordes, S. Chopra, and J. Weston, "The goldilocks principle: Reading children's books with explicit memory representations," *CoRR*, 2015.

[128] A. Trischler, T. Wang, X. Yuan, J. Harris, A. Sordoni, P. Bachman, and K. Suleman, "NewsQA: A machine comprehension dataset," in *Proceedings of the 2nd Workshop on Representation Learning for NLP*. Association for Computational Linguistics, 2017, pp. 191–200.

[129] M. Dunn, L. Sagun, M. Higgins, V. U. Güney, V. Cirik, and K. Cho, "Searchqa: A new q&a dataset augmented with context from a search engine," *CoRR*, vol. abs/1704.05179, 2017.

[130] M. Joshi, E. Choi, D. Weld, and L. Zettlemoyer, "TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2017, pp. 1601–1611.

[131] T. Kociský, J. Schwarz, P. Blunsom, C. Dyer, K. M. Hermann, G. Melis, and E. Grefenstette, "The narrativeqa reading comprehension challenge," *CoRR*, vol. abs/1712.07040, 2017.

[132] W. He, K. Liu, Y. Lyu, S. Zhao, X. Xiao, Y. Liu, Y. Wang, H. Wu, Q. She, X. Liu, T. Wu, and H. Wang, "Dureader: a chinese machine reading comprehension dataset from real-world applications," *CoRR*, vol. abs/1711.05073, 2017.

[133] S. Reddy, D. Chen, and C. D. Manning, "Coqa: A conversational question answering challenge," *CoRR*, vol. abs/1808.07042, 2018.

[134] E. Choi, H. He, M. Iyyer, M. Yatskar, W. Yih, Y. Choi, P. Liang, and L. Zettlemoyer, "Quac : Question answering in context," *CoRR*, vol. abs/1808.07036, 2018.

[135] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord, "Think you have solved question answering? try arc, the AI2 reasoning challenge," *CoRR*, vol. abs/1803.05457, 2018.

[136] M. Saeidi, M. Bartolo, P. Lewis, S. Singh, T. Rocktäschel, M. Sheldon, G. Bouchard, and S. Riedel, "Interpretation of natural language rules in conversational machine reading," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018, pp. 2087–2097.

[137] S. Šuster *et al.*, "CliCR: a dataset of clinical case reports for machine reading comprehension," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, 2018, pp. 1551–1563.

[138] D. Khashabi, S. Chaturvedi, M. Roth, S. Upadhyay, and D. Roth, "Looking beyond the surface: A challenge set for reading comprehension over multiple sentences," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, 2018, pp. 252–262.

[139] R. Zellers, Y. Bisk, R. Schwartz, and Y. Choi, "SWAG: A large-scale adversarial dataset for grounded commonsense inference," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018, pp. 93–104.

[140] A. Saha, R. Aralikatte, M. M. Khapra, and K. Sankaranarayanan, "Duorc: Towards complex language understanding with paraphrased reading comprehension," *CoRR*, vol. abs/1804.07927, 2018.

[141] S. Zhang, X. Liu, J. Liu, J. Gao, K. Duh, and B. V. Durme, "Record: Bridging the gap between human and machine commonsense reading comprehension," 2018.

[142] A. Talmor, J. Herzig, N. Lourie, and J. Berant, "Commonsenseqa: A question answering challenge targeting commonsense knowledge," *CoRR*, vol. abs/1811.00937, 2018.

[143] M. Chen, M. D'Arcy, A. Liu, J. Fernandez, and D. Downey, "CODAH: An adversarially-authored question answering dataset for common sense," in *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*. Association for Computational Linguistics, 2019, pp. 63–69.

[144] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, M. Kelcey, J. Devlin, K. Lee, K. N. Toutanova, L. Jones, M.-W. Chang, A. Dai, J. Uszkoreit, Q. Le, and S. Petrov, "Natural questions: a benchmark for question answering research," *Transactions of the Association of Computational Linguistics*, 2019.

[145] L. Huang, R. Le Bras, C. Bhagavatula, and Y. Choi, "Cosmos QA: Machine reading comprehension with contextual commonsense reasoning," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 2391–2401.

[146] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova, "BoolQ: Exploring the surprising difficulty of natural yes/no questions," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, 2019, pp. 2924–2936.

[147] A. Fan, Y. Jernite, E. Perez, D. Grangier, J. Weston, and M. Auli, "ELI5: Long form question answering," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019, pp. 3558–3567.

[148] W. Xiong, J. Wu, H. Wang, V. Kulkarni, M. Yu, S. Chang, X. Guo, and W. Y. Wang, "TWEETQA: A social media focused question answering dataset," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019, pp. 5020–5031.

[149] J. Liu, Y. Lin, Z. Liu, and M. Sun, "XQA: A cross-lingual open-domain question answering dataset," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019, pp. 2358–2368.

[150] J. Gao, M. Galley, and L. Li, "Neural approaches to conversational ai," *Foundations and Trends® in Information Retrieval*, vol. 13, no. 2-3, pp. 127–298, 2019.

[151] W. Lei, X. He, M. de Rijke, and T.-S. Chua, "Conversational recommendation: Formulation, methods, and evaluation," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '20. Association for Computing Machinery, 2020, p. 2425–2428.

[152] H. Zhu, L. Dong, F. Wei, W. Wang, B. Qin, and T. Liu, "Learning to ask unanswerable questions for machine reading comprehension," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019, pp. 4238–4248.

[153] M. Hu, F. Wei, Y. xing Peng, Z. X. Huang, N. Yang, and M. Zhou, "Read + verify: Machine reading comprehension with unanswerable questions," *ArXiv*, vol. abs/1808.05759, 2018.

[154] M. Aliannejadi, H. Zamani, F. Crestani, and W. B. Croft, "Asking clarifying questions in open-domain information-seeking conversations," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, 2019, p. 475–484.

[155] X. Du, J. Shao, and C. Cardie, "Learning to ask: Neural question generation for reading comprehension," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, 2017, pp. 1342–1352.

[156] N. Duan, D. Tang, P. Chen, and M. Zhou, "Question generation for question answering," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, 2017, pp. 866–874. [Online]. Available: https://www.aclweb.org/anthology/D17-1090

[157] Q. Zhou, N. Yang, F. Wei, C. Tan, H. Bao, and M. Zhou, "Neural question generation from text: A preliminary study," *CoRR*, vol. abs/1704.01792, 2017.

[158] L. Pan, W. Lei, T. Chua, and M. Kan, "Recent advances in neural question generation," *CoRR*, vol. abs/1905.08949, 2019.

[159] C. C. Chen Qu, Liu Yang *et al.*, "Open-retrieval conversational question answering," *CoRR*, vol. abs/2005.11364, 2020.
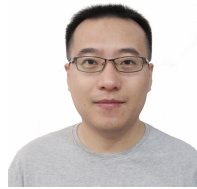
Fengbin Zhu received his B.E. degree from Shandong University, China. He is currently pursuing his Ph.D degree at the School of Computing, National University of Singapore (NUS). His research interests include natural language processing, machine reading comprehension and conversational question answering.

Wenqiang Lei is a Research Fellow with School of Computing, National University of Singapore (NUS). He received his Ph.D. in Computer Science from NUS in 2019. His research interests cover natural language processing and information retrieval, particularly on dialogue systems, conversational recommendations and question answering. He has published multiple papers at top conferences like ACL, IJCAI, AAAI, EMNLP and WSDM and the winner of ACM MM 2020 best paper award. He served as (senior) PC members on toptier conferences including ACL, EMNLP, SIGIR, AAAI, KDD and he is a reviewer for journals like TOIS, TKDE, and TASLP.

Chao Wang holds a PhD in Computer Science from Tsinghua University, where he was advised by Dr. Shaoping Ma and Dr. Yiqun liu. His work has primarily focused on nature language processing, information retrieval, search engine user behavior analysis. His work has appeared in major journals and conferences such as SIGIR, CIKM, TOIS, and IRJ.

Jianming Zheng is a PhD candidate at the School of System Engineering, the National University of Defense Technology, China. His research interests include semantics representation, few-shot learning and its applications in information retrieval. He received the BS and MS degrees from the National University of Defense Technology, China, in 2016 and 2018, respectively. He has several papers published in SIGIR, COLING, IPM, FITEE, Cognitive Computation, etc.

Soujanya Poria is an assistant professor of Information Systems Technology and Design, at the Singapore University of Technology and Design (SUTD), Singapore. He holds a Ph.D. degree in Computer Science from the University of Stirling, UK. He is a recipient of the prestigious early career research award called "NTU Presidential Postdoctoral Fellowship" in 2018. Soujanya has co-authored more than 100 research papers, published in top-tier conferences and journals such as ACL, EMNLP, AAAI, NAACL, Neurocomputing, Computational Intelligence Magazine, etc. Soujanya has been an area chair at top conferences such as ACL, EMNLP, NAACL. Soujanya serves or has served on the editorial boards of the Cognitive Computation and Information Fusion.

Tat-Seng Chua is the KITHCT Chair Professor at the School of Computing, National University of Singapore (NUS). He is also the Distinguished Visiting Professor of Tsinghua University. Dr. Chua was the Founding Dean of the School of Computing from 1998-2000. His main research interests include heterogeneous data analytics, multimedia information retrieval, recommendation and conversation systems, and the emerging applications in E-commerce, wellness and Fintech. Dr. Chua is the co-Director of NExT, a joint research Center between NUS and Tsinghua, focusing on Extreme Search.

Dr. Chua is the recipient of the 2015 ACM SIGMM Achievements Award for the Outstanding Technical Contributions to Multimedia Computing, Communications and Applications. He is the Chair of steering committee of ACM ICMR (2015-19), and Multimedia Modeling (MMM) conference series. He was the General Co-Chair of ACM Multimedia 2005, ACM CIVR (now ACM ICMR) 2005, ACM SIGIR 2008, and ACM Web Science 2015. He serves in the editorial boards of three international journals. He holds a PhD from the University of Leeds, UK.