IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received February 9, 2020, accepted March 22, 2020, date of publication April 20, 2020, date of current version June 2, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2988903

Recent Trends in Deep Learning Based Open-Domain Textual Question Answering Systems

ZHEN HUANG[1], SHIYI XU[1], MINGHAO HU[1], XINYI WANG[1], JINYAN QIU[2], YONGQUAN FU[1], YUNCAI ZHAO[3], YUXING PENG[1], AND CHANGJIAN WANG[1]

[1]Science and Technology on Parallel and Distributed Laboratory, College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, China
[2]H.R. Support Center, Beijing 100010, China
[3]Unit 31011, People's Liberation Army, Beijing 102249, China

Corresponding author: Shiyi Xu (xushiyi18@nudt.edu.cn)

This work was supported by the National Key Research and Development Program of China under Grant 2018YFB0204300.

ABSTRACT Open-domain textual question answering (QA), which aims to answer questions from large data sources like Wikipedia or the web, has gained wide attention in recent years. Recent advancements in open-domain textual QA are mainly due to the significant developments of deep learning techniques, especially machine reading comprehension and neural-network-based information retrieval, which allows the models to continuously refresh state-of-the-art performances. However, a comprehensive review of existing approaches and recent trends is lacked in this field. To address this issue, we present a thorough survey to explicitly give the task scope of open-domain textual QA, overview recent key advancements on deep learning based open-domain textual QA, illustrate the models and acceleration methods in detail, and introduce open-domain textual QA datasets and evaluation metrics. Finally, we summary the models, discuss the limitations of existing works and potential future research directions.

INDEX TERMS Open-domain textual question answering, deep learning, machine reading comprehension, information retrieval.

I. INTRODUCTION

A. BACKGROUND

Question answering (QA) systems have long been concerned by both academia and industry [1]–[3], where the concept of QA system can be traced back to the emergence of artificial intelligence, namely the famous Turing test [4]. Technologies with respect to QA have been constantly evolving over almost the last 60 years in the field of Natural Language Processing (NLP) [5]. Early works on QA mainly relied on manually-designed syntactic rules to answer simple answers due to constrained computing resources [6], such as Baseball in 1961, Lunar in 1977, Janus in 1989 and so on [5]. Around 2000, several conferences such as TREC QA [1] and QA@CLEF [7], have greatly promoted the development of QA. A large number of systems that utilize information retrieval (IR) techniques were proposed at that time. Then around 2007, with the development of knowledge bases (KBs), such as Freebase [8] and DBpedia [9], especially with the emergence of open-domain datasets on WebQuestions [10] and SimpleQuestions [11], KBQA technologies evolved quickly. In 2011, IBM Watson [12] won the Jeopardy! game show, which received a great deal of attention. Recently, due to the release of several large-scale benchmark datasets [13]–[15] and the fast development in deep learning techniques, large advancements have been made in the QA field. Especially, recent years have witnessed a research renaissance on deep learning based open-domain textual QA, an important QA branch that focuses on answering questions from large knowledge sources like Wikipedia and the web.

B. MOTIVATION

Despite the flourishing research of open-domain textual QA, there remains a lack of comprehensive survey that summarizes existing approaches&datasets as well as systemically analysis of the trends behind these successes. Although several surveys [16]–[19] were proposed to discuss the broad

The associate editor coordinating the review of this manuscript and approving it for publication was Jan Chorowski.

picture of QA, none of them have focused on the specific deep learning based open-domain textual QA branch. Moreover, there are several surveys [20]–[23] that illustrate recent advancements in machine reading comprehension (MRC) by introducing several classic neural MRC models. However, they only reported the approaches in close-domain single-paragraph settings, and failed to present the latest achievements in open-domain scenarios. So we write this paper to summarize recent literature of deep learning based open-domain textual QA for the researchers, practitioners, and educators who are interested in this area.

C. TASK SCOPE

In this paper, we conduct a thorough literature review on recent progress in open-domain textual QA. To achieve this goal, we first category previous works based on five characteristics described as below, then give an exact definition of open-domain textual QA that explicitly constrains its scope.

1) **Source:** Towards different data sources, QA systems can be classified into *structured*, *semi-structured* and *unstructured* categories. One the one hand, structured data are mainly organized in the form of knowledge graph (KG) [9], [24], [25], while semi-structured data are usually viewed as lists or tables [26]–[28]. On the other hand, unstructured data are typically plain text composed of natural language.
2) **Question:** The question type is defined as a certain semantic category characterized by some common properties. The major types include *factoid*, *list*, *definition*, *hypothetical*, *causal*, *relationship*, *procedural*, and *confirmation* questions [17]. Typically, factoid question is the question that starts with a Wh-interrogated word (What, When, Where, etc.) and requires an answer as fact expressed in the text [17]. The form of question can be *full question* [14], *key word/ phrase* [15] or *(item, property, answer) triple* [29].
3) **Answer:** Based on how the answer is produced, QA systems can be roughly classified into *extractive-based* QA and *generative-based* QA. Extractive-based QA selects a span of text [13], [15], [30], a word [31], [32] or an entity [10], [11] as the answer. Generative-based QA may rewrite the answer if it does not (i) include proper grammar to make it a full sentence, (ii) make sense without the context of either the query or the passage, (iii) have a high overlap with exact portions in context [33], [34].
4) **Domain:** *Closed-domain* QA system deals with questions under a specific field [35], [36] (e.g., law, education, and medicine), and can exploit domain-specific knowledge frequently formalized in ontologies. Besides, closed-domain QA usually refers to a situation where only a limited type of question is asked, and a small amount of context is provided. *Open-domain* QA system, on the other hand, deals with questions from a broad range of domains, and only rely on general text and knowledge base. Moreover, systems are usually required to find answers from large open-domain knowledge sources (e.g., Wikipedia, web), instead of a given document [37], [38].
5) **Methodology:** As for involved methodologies, QA systems can be categorized into *IR based* [39]–[41], *NLP based* [31] and *KB based* [42] approaches [5]. IR based models mainly return the final answer as a text snippet that is most relevant to the question. NLP based models aim to extract candidate answer strings from the context document and re-rank them by semantic matching. KBQA systems build a semantic representation of the query and transform it into a full predicate calculus statement for the knowledge graph.

TABLE 1. Question-answer pairs with sample excerpts from TriviaQA [14], which requires reasoning from multiple paragraphs.

Question	Who was the next British Prime Minister after Arthur Balfour?
Answer	Henry Campbell-Bannerman
Excerpt	The topic of Tariff Reform split Balfour's government and when he resigned in 1905, Edward VII invited **Henry Campbell-Bannerman** to form a government. Campbell-Bannerman accepted and in the 1906 General Election that followed the Liberal Party had a landslide victory. In November, the Conservative Prime Minister Arthur Balfour tried to expose the divisions within the Liberal opposition by resigning, but his rival **Henry Campbell-Bannerman** formed a Liberal government and then led it to a smashing success at the polls in January 1906.

Following the above categories, open-domain textual QA can be defined as: (1) unstructured data sources on text, (2) factoid questions or keyword/phrase as inputs, (3) extractive-based answer, (4) open-domain, and (5) NLP based technologies with auxiliary IR technologies. Table. 1 shows an example of deep learning based open-domain textual QA.

D. CONTRIBUTIONS

The purpose of this survey is to review the recent research progress of open-domain textual QA based on deep learning. It provides the reader with a panoramic view that allows the reader to establish a general understanding of open-domain textual QA and know how to build a QA model with deep learning technique. In conclusion, the main contributions of this survey are as follows: (1) we conducted a systematic review for open-domain textual QA system based on deep learning technique; (2) we introduced the recent models, discussed the pros and cons of each method, summarized method used in each components of model, and compared the models performance on each dataset; (3) we discussed the current challenges and problems to be solved, and explored new trends and future directions in the research on open domain textual QA system based on deep learning.

E. ORGANIZATION

After making the definition clear, we further give an overview of open-domain textual QA systems, including presenting a brief history, explaining the motivation of using deep learning techniques, and introducing a general open-domain textual QA architecture (Section II). Next, we illustrate several key components of open-domain textual QA including *ranking module*, *answer extraction*, and *answer selection*, summarize recent trends on acceleration techniques as well as public datasets and metrics (Section III). Last, we conclude the work with discussions on the limitations of existing works and some future research directions (Section IV).

II. OVERVIEW OF OPEN-DOMAIN TEXTUAL QA SYSTEMS

Before we dive into the details of this survey, we start with an introduction to the history, the reason why deep learning based method emerges and architecture regarding to open-domain textual QA systems based on deep learning.

A. HISTORY OF OPEN-DOMAIN TEXTUAL QA

In 1993, START became the first knowledge-based question-answering system on the Web [43], since then answered millions of questions from Web users all over the world. In 1999, the 8th TREC competitions [44] began to run the QA track. In the following year, at the 38th ACL conference, a special discussion topic ''Open-domain Question Answering'' was opened up. Since then, open-domain QA system has become a hot topic in the research community. With the development of structured KBs like Freebase [8], many works have proposed to construct QA systems with KBs, such as WebQuestions [10] and SimpleQuestions [11]. These approaches usually achieve high precision and nearly solve the task on simple questions [45], but their scope is limited to the ontology of the KBs. There are also some pipelined QA approaches that use a large number of data resources, including unstructured text collections and structured KBs. The landmark approaches are ASKMSR [3], DEEPQA [12], and YODAQA [2]. A landmark event in this filed is the success of IBM Watson [12], who won the Jeopardy! game show in 2011. This complicated system adopted a hybrid scheme including technologies brought from IR, NLP, and KB. In recent years, With the development of deep learning, NLP based QA systems emerge, which can directly carry out end-to-end processing of unstructured text sequences at the semantic level through neural network model [46]. Specifically, DrQA [37] was the first neural-network-based model for the task of open-domain textual QA. Based on this framework, some end-to-end textual QA models have been proposed, such as R^3 [47], DS-QA [48], DocumentQA [49], and RE^{3}QA [38].

B. WHY DEEP LEARNING FOR OPEN-DOMAIN TEXTUAL QA

It is beneficial to understand the motivation behind these approaches for open-domain textual QA. Specifically, why do we need to use deep learning techniques to build open-domain textual QA systems? What are the advantages of neural-network-based architectures? In this section, we would like to answer the above questions to show the strengths of deep learning-based QA models, which are listed as below:

1) **Automatically learn complex representation:** Using neural networks to learn representations has two advantages: (1) it reduces the efforts in hand-craft feature designs. Feature engineering is a labor-intensive work, deep learning enables automatically feature learning from raw data in unsupervised or supervised ways [50]. (2) contrary to linear models, neural networks are capable of modeling the non-linearity in data with activation functions such as Relu, Sigmoid, Tanh, etc. This property makes it possible to capture complex and intricate user item interaction patterns [50].
2) **End-to-end processing:** Many early years' QA systems heavily relied on the question and answer templates, which were mostly manually constructed and time-consuming. Later most of the QA research adopted a pipeline of conventional linguistically-based NLP techniques, such as semantic parsing, part-of-speech tagging, and coreference resolution. This could cause the error propagation during the entire progress. On the other hand, neural networks have the advantage that multiple building blocks can be composed into a single (gigantic) differentiable function and trained end-to-end. Besides, models of different stages can share learned representations and benefit from multi-task learning [51].
3) **Data-driven paradigm:** Deep learning is essentially a science based on statistics, one intrinsic property of deep learning is that it follows a data-driven paradigm. That is, neural networks can learn statistical distributions of features from massive data, and the performance of the model could be constantly improved as more data are used [52]. This is important for open-domain textual QA as it usually involves wide range of domains and large text corpus.

C. DEEP LEARNING BASED TECHNICAL ARCHITECTURE OF OPEN-DOMAIN TEXTUAL QA SYSTEMS

As shown in TABLE. 1, given a question, the QA system needs to retrieve several relevant documents, read and gather information across multiple text snippets, then extract the answer from raw text. Notably, not all given paragraphs contain the correct answer, and the exact location of the ground-truth answer is unknown. Such setting is usually referred to as *distant supervision*, which brings difficulties in designing supervised training signals. In summary, open-domain textual QA poses great challenges as it requires to: 1) filter out irrelevant noise context, 2) reason across

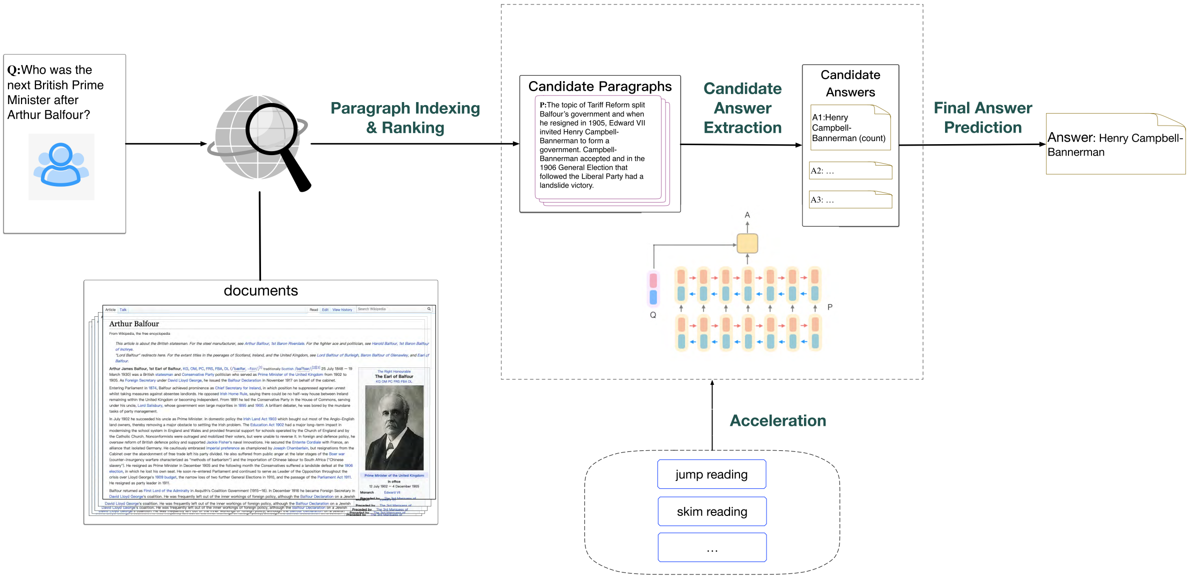

Q:Who was the next British Prime Minister after Arthur Balfour?
Paragraph Indexing & Ranking
documents
Arthur Balfour
Candidate Paragraphs
P:The topic of Tariff Reform split Balfour's government and when he resigned in 1905, Edward VII invited Henry Campbell-Bannerman to form a government. Campbell-Bannerman accepted and in the 1906 General Election that followed the Liberal Party had a landslide victory.
Candidate Answer Extraction
Candidate Answers
A1:Henry Campbell-Bannerman (count)
A2: ...
A3: ...
A
Q
P
Final Answer Prediction
Answer: Henry Campbell-Bannerman
Acceleration
jump reading
skim reading
...

FIGURE 1. The technical architecture of deep learning based open-domain textual QA systems. The paragraph index&ranking module first retrieves several related documents and then selects a few top-ranked paragraphs relative to the question, from which the extractive reading comprehension module extracts multiple candidate answers. Finally, the system picks the most promising prediction as the answer. Besides, to boost the processing speed while ensuring accuracy, several acceleration techniques are adopted.

multiple evidence snippets, and 3) train with distantly-supervised objectives.

In recent years, with the rapid development of deep learning technologies, significant technical advancements have been made in the field of open-domain textual QA. Specifically, Chen *et al.* proposed the DrQA system [37], which splits the task into two subtasks: paragraph retrieval and answer extraction. The paragraph retrieval module selects and ranks the candidate paragraphs according to the relevance between paragraph and question, while the answer extraction module predicts the start and end positions of candidate answers in the context. Later, Clark and Gardner [49] proposed a shared-normalization mechanism to deal with the distant-supervision problem in open-domain textual QA. Wang *et al.* [47] adopted reinforcement learning to joint train the ranker and the answer-extraction reader. Based on this work, Wang *et al.* [53] further proposed evidence aggregation for answer re-ranking. Recently, Hu *et al.* [38] presented an end-to-end open-domain textual QA architecture to jointly perform context retrieval, reading comprehension, and answer re-ranking.

To summarize these works, we propose a general technical architecture of open-domain textual QA system in Fig. 1. The architecture mainly consists of three modules including *paragraph index&ranking*, *candidate answer extraction*, and *final answer selection*. Specifically, the paragraph index&ranking module first retrieves top-k paragraphs related to questions. Then these paragraphs are sent into the answer extraction module to locates multiple candidate answers. Finally, the answer selection module predicts the final answer. Moreover, in order to improve the efficiency of QA systems, some acceleration techniques, such as jump reading [54] and skim reading [55], can be applied in the system.

III. MODELS AND HOT TOPICS

In this section, we illustrate the individual component of the generalized open-domain textual QA system described in Fig. 1. Specifically, we introduce: (i) the paragraph index&ranking module in subsection III-A, (ii) the candidate answer extraction module in subsection III-B, (iii) the final answer selection module in subsection III-C, and (iv) the acceleration techniques in subsection III-D. Finally, we give a brief introduction of recent open-domain textual QA datasets in subsection III-E, as well as experimental evaluation and model performance in subsection III-F.

A. PARAGRAPH INDEX AND RANKING

The first step of open-domain textual QA is to retrieve several top-ranked paragraphs that are relevant to the question. There are two sub-stages here: retrieving documents through indexing, and ranking the context fragments (paragraphs) in these documents. The paragraph-index module builds the light-weight index for the original documents. During processing, the index dictionary is loaded into memory, while the original documents are stored in file-systems. This method can effectively reduce memory overhead, as well as accelerates the retrieval process. The paragraph-ranking module analyzes the relevance between query and paragraphs and selects top-ranked paragraphs to feed into the reading comprehension module. In recent years, along with the development of information retrieval and NLP, a large number of new technologies regarding to index and ranking have been proposed. Here we mainly focus on the deep learning-based approaches.

1) PARAGRAPH INDEX

Paragraph index can be classified into *query-dependent index* and *query-independent index*. The query-dependent index mainly includes dependence model and pseudo relevance feedback(PRF) [56], [57], which considers approximation between query and document terms. However, due to the index dependence on queries, the corresponding ranking models are difficult to scale and generalize. The query-independent index mainly includes TF-IDF, BM25, and language modeling [56], [57], which contains a relatively simple index feature and with low computational complexity on matching. IBM Watson adopted a search method to combine the query-dependent similarity score with the query-independent score to determine the overall search score for each passage [58]. Although those index features are relatively efficient and scalable on processing, they are mainly based on the terms without the contextual semantic information.

Recently, several deep learning-based methods have been proposed. These approaches usually embed the terms or phrases into dense vectors and use them as indices. Kato *et al.* [59] constructed a demo to compare the efficiency and effectiveness of LSTM and BM25. Seo *et al.* proposed *Phrase-indexed Question Answering* (PIQA) [60], which employed bi-directional LSTMs and self-attention mechanism to obtain the representation vectors for both query and paragraph. Lee *et al.* leveraged BERT encoder [61] to pre-train the retrieval module [62], unlike previous works that retrieve candidate paragraphs, the evidence passage retrieved from Wikipedia was seen as a latent variable.

2) PARAGRAPH RANKING

The traditional ranking technologies are based on manually-designed feature [63], but in recent years, learning to rank (L2R) approaches have become a hot-spot. L2R refers to ranking methods based on supervised learning, it can be classified into *Pointwise*, *Pairwise*, and *Listwise* [64]. Pointwise (e.g., McRank [65], Prank [66]) converts the document into feature vectors, then gives out the relevance scores according to the classification or regression function learned from the training data, from which to indicate the ranking results. Pointwise focuses on the relevance between the query and documents, ignoring the information interaction inside the documents. Hence Pairwise (e.g., RankNet [67], FRank [68]) estimates whether the order of document pairs is reasonable. However, the number of relevant documents varies greatly from different queries. Thus the generalization ability of Pairwise is difficult to estimate. Unlike the above two methods, Listwise (e.g., LambdaRank [69], SoftRank [70]) trains the optimization scoring function with a list of all search results for each query as a training sample. Since the aim of paragraph ranking is to filter out irrelevant paragraphs, Pointwise seems to be adequate in most cases. However, the scores between queries and paragraphs also can be helpful for predictions on the final answer, as we discuss in subsection III-C. Consequently, Listwise ranking methods are also important to the open-domain textual QA task.

Moreover, the paragraph ranking model trained with deep neural networks mainly includes four categories [56]: (i) learning the ranking model through manual features, and only using the neural network to match the query and document; (ii) estimating relevance based on the query-document exactly matching pattern; (iii) learning the embedded representations of queries and documents, and evaluating them by a simple function, such as cosine similarity or dot-product; (iv) conducting query expansion with neural network embeddings, and calculating the query expectation.

Similar to (ii), Wang *et al.* [47] proposed *Reinforced Ranker-Reader* (R^3) model, which is also a kind of Pointwise method. It consisted of: (1) a *Ranker* to select a paragraph most relevant to the query, and (2) a *Reader* to extract the answer from the paragraph selected by *Ranker*. The deep learning-based *Ranker* model was trained using reinforcement learning, where the accuracy of the answer extracted by *Reader* determined the reward. Both the *Ranker* and *Reader* leveraged Match-LSTM [71] model to match the query and passages. Similar to (iii), Tan *et al.* [72] studied several representation learning models and found that attentive LSTM can be very effective on the Pairwise mode training. And PIQA [60] employed similarity clustering to retrieve the nearest indexed phrase vector to the query vector by asymmetric locality-sensitive hashing (aLSH) [73] or Fassi [74].

There are also combinations of the above categories, Htut *et al.* [75] combined (i) and (iii), which took the embedded representations to train the ranking model, and proposed two kinds of ranking models: *InferSent ranker* and *Relation-Networks ranker*. The rankers leveraged the Listwise ranking method, which were trained by minimizing the margin ranking loss, so as to obtain the optimal score.

$$\sum_{i=1}^{k} max(0, 1 - f(q, p_{pos}) + f(q, p^{i}_{neg})) \tag{1}$$

Here f is the scoring function, p_{pos} is a paragraph that contains the ground-truth answer, p_{neg} is a negative paragraph that does not contain the ground-truth answer, and k is the number of negative samples. *InferSent ranker* leveraged sentence-embedded representations [76] and evaluated the semantic similarity in ranking for QA, which employed a feed-forward neural network as the scoring function:

$$\boldsymbol{x}_{classifier} = \begin{bmatrix} q \\ p \\ \text{q-p} \\ q \odot p \end{bmatrix} \tag{2}$$

$$\boldsymbol{z} = \boldsymbol{W}^{(1)}\boldsymbol{x}_{classifier} + \boldsymbol{b}^{(1)} \tag{3}$$

$$score = \boldsymbol{W}^{(2)}ReLU(\boldsymbol{z}) + \boldsymbol{b}^{(2)} \tag{4}$$

Relation-Networks ranker focused on measuring the relevance between words in the question and words in

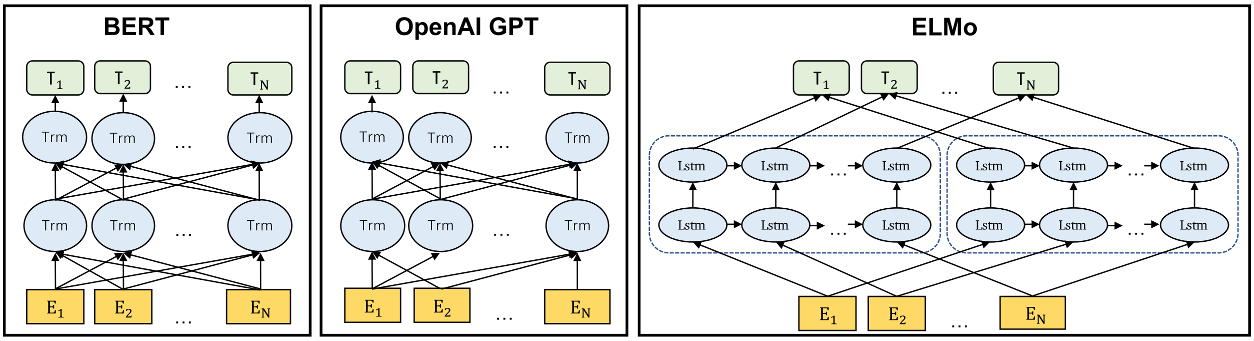


FIGURE 2. Differences between BERT, GPT, and ELMo. BERT uses a bi-directional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTM to generate features for downstream tasks.(Figure source: Devlin *et al.* [61])

the paragraph, where the word pairs were the inputs of Relation-Networks which is formulated as follows.

$$RN(q, p) = f_\phi\Big(\sum_{i,j} g_\theta([E(q_i); E(p_j)])\Big) \tag{5}$$

Here $E(\cdot)$ is a 300 dimensional GloVe embedding [77], f_ϕ and g_θ are 3 layer feed-forward neural networks with ReLU activation function. The experimental results showed that the performance of QA part [75] even exceed reinforcing feedback ranking model [47].

B. CANDIDATE ANSWER EXTRACTION

With the candidate paragraphs filtered from the index& ranking module, QA systems can locate candidate answers (the start and end positions of answer spans in the document or paragraph) through the reading comprehension model. With the releasing of datasets and test standards [13]–[15], [30], many works have been proposed in the past three years, attracting great attention from the academia and industrial. In this subsection, we illustrate the reading extraction model from three hierarchies: (i) word embeddings and pre-training models for feature encoding in subsection III-B1, (ii) interaction of questions and paragraphs using attention mechanism in subsection III-B2, and (iii) feature aggregation for predicting the candidate answers in subsection III-B3.

1) FEATURE ENCODING LAYER

In this layer, the original text tokens are transformed into vectors that can be computed by the deep neural networks through word embeddings or manual features. Word embeddings can be obtained through dictionary or fine-tuning on pre-trained language models, while manual textual features are usually implemented by part-of-speech tagging (POS) and named entity recognition (NER). Manual features can be constructed by tools such as CoreNLP [78], AllenNLP [79], and NLTK [80]. Generally, the features mentioned above will be fused with embedding vectors.

Embedding vectors can be constructed by pre-trained language models. Glove [77] transferred word-level information to word vectors through the co-occurrence matrix, but cannot distinguish the polysemous words. ELMo [81] leveraged a deep bi-directional language model to yield word embeddings that can vary from different context sentences, which was concatenated by two unidirectional language models. OpenAI GPT [82] used the left-to-right transformer decoder [83], whereas BERT [61] used the bi-directional transformer encoder [83] to pre-train, then both of them adjust the downstream tasks through fine-tuning methods. Fig. 2 shows the difference between ELMo, GPT, and BERT. Specifically, The pre-trained BERT model has been proven as a powerful context-dependent representation and made significant improvements on the open domain textual QA tasks, some works based on BERT, such as RE^{3}QA [38], ORQA [62], and DFGN [84], have achieved state-of-the-art results.

2) INTERACTIVE ATTENTION LAYER

The interactive attention layer constructs representations on the original features of question or paragraph by using attention mechanisms. It can be mainly divided into two types:

(i) Interactive alignment between the question and paragraph, namely *co-attention*, which allows the model to focus on the most relevant question features with respect to paragraph words, and breaks through the limited coding extraction ability of a single model. Wang and Jiang [71] leveraged a textual entailment model Match-LSTM [85] to construct the attention processing. Xiong *et al.* [86] used a co-attention encoder to co-dependent representations of the question and the document, and a dynamic pointer decoder to predict the answer span. Seo *et al.* proposed a six layers model BiDAF [87] along with a memory-less attention mechanism to yield the representations of the context paragraph at character-level, word-level and contextual-level. Gong and Bowman [88] added a multi-hop attention mechanism to BiDAF to solve the problem that the single-pass model cannot reflect on.

(ii) Self alignment inside the paragraph to generate self-aware features, namely *self-attention*, which allows non-adjacent words in the same paragraph to attend to each other, thus alleviating the long-term dependency problem.

For example, Wang *et al.* [89] proposed a self-attention mechanism to refine the question-aware passage representation by matching the passage against itself.

We can find two trends in recent works: (1) the combination of co-attention and self-attention. e.g., DCN+ [90] improved DCN by extending the deep residual co-attention encoder with self-attention. Yu *et al.* leveraged the combination of convolutions and self-attention in the embedding and modeling encoders, and a context-query attention layer after the embedding encoder layer [91]. (2) fusion features at different levels, e.g., Huang *et al.* adopted a three-layers fully-aware-attention mechanism to further enhance the feature representation ability of the models [92]. Wang *et al.* combined co-attention and self-attention mechanism, as well as applied a fusion function to incorporated different levels of features [93]. Hu *et al.* proposed a re-attending mechanism inside a multi-layer attention architecture, where prior co-attention and self-attention were both considered to fine-tune current attention [94].

3) AGGREGATION PREDICTION LAYER

In this layer, aggregation vectors are generated to predict candidate answers, we mainly focus on the following parts.

- **Aggregation strategies.** Aggregation strategies vary from different network frameworks. BiDAF [87] and Multi-Perspective Matching [95] leveraged Bi-LSTM for semantic information aggregation. FastQAExt [96] adopted two feed-forward neural networks to generate the probability distribution of start and end position of the answers, then used beam-search to determine the range of the answers.
- **Iteration prediction strategies.** DCN [86] consisted of a co-attentive encoder and a dynamic pointing decoder, which adopted a multi-round iteration mechanism. In each round of iteration, the decoder estimated the start and end of the answer span. Based on the prediction of the previous iteration, LSTM and Highway Maxout Network are used to update the prediction of the answer span in the next iteration. ReasoNet [97] and Mnemonic Reader [94] used the memory network framework to do iterative prediction. DCN+ [90] and Reinforced Mnemonic Reader [94] iteratively predicted the start and end position by reinforcement learning.
- **Interference discarding strategies**. Discarding interference items dynamically during the prediction process can improve the accuracy performance and generalization of models, such as DSDR [98] and SAN [99].
- **Loss Function.** Based on the extracted answer span, the loss function is generally defined as the sum of the probability distributions of the start and end positions of gold answers [49], which can be formulated as follows.

$$\mathcal{L} = -log\Big(\frac{e^{s_a}}{\sum_{i=1}^{n} e^{s_i}}\Big) - log\Big(\frac{e^{g_b}}{\sum_{j=1}^{n} e^{g_j}}\Big) \qquad (6)$$

Here s_j and g_j are the scores for the start and end bounds produced by the model for token j, a and b are the start and end tokens. In the multi-paragraph reading comprehension tasks, reading comprehension model is employed on both negative paragraphs and positive paragraphs, thus need to add the no-answer prediction term in the loss function as [49], [100]:

$$\mathcal{L} = -log\Big(\frac{(1-\delta)e^{z} + \delta e^{s_a g_b}}{e^{z} + \sum_{i=1}^{n}\sum_{j=1}^{n} e^{s_i g_j}}\Big) \qquad (7)$$

Here δ is 1 if an answer exists and 0 otherwise, and z presents the weight given to a ''no-answer'' possibility.

C. FINAL ANSWER SELECTION

Final answer selection mainly selects the final answer from multiple candidate answers using feature aggregation, aggregation methods can be divided into the following types.

- **Evidence Aggregation**. Wang *et al.* proposed a method of candidate answer re-ranking, mainly based on two types of evidence [53]: (i) replicated evidence: the candidate answer which appears more times in different passages may have a higher probability to be the correct answer. (ii) complementary evidence: aggregating multiple passages can entail multiple aspects of the question, so as to ensure the completeness of the answer. In the inference part, Wang *et al.* leveraged a classical textual entailment model Match-LSTM [71] to infer the relevance of the answer spans [53]. Moreover, Lin *et al.* and Zhong *et al.* adopted the coarse-to-fine strategy to select related paragraphs and aggregated evidence from them to predict the final answer [48], [101].
- **Multi-stages Aggregation.** Wang *et al.* divided the open-domain textual QA task into two stages [47]: candidate paragraph ranking and answer extraction, and jointly optimized the expected losses of the two-stages through reinforcement learning. Wang *et al.* divided reading comprehension into candidate extraction and answer selection, and jointly trained the two-stages process in the end-to-end model and made improvements on the final prediction [102]. Pang *et al.* and Wang *et al.* divided the open-domain textual QA task into reading extraction and answer selection. They leveraged a beam search strategy to find the final answer with maximum probability considering both stages [103], [104]. Hu *et al.* proposed an end-to-end open-domain textual QA model, which contains retrieving, reading, and reranking modules [38].
- **Fusion of Knowledge Bases and Text.** Recently several works attempt to incorporate external knowledge to improve performance on a variety of tasks, such as [105] for natural language inference, [106] for cloze-style QA task, and [107] for Multi-Hop QA task. Sun *et al.* proposed a method to fuse multi-source information in early stage to improve overall QA task [108]. Weissenborn *et al.* proposed an architecture to dynamically integrate explicit background knowledge in Natural Language Understanding models [109].

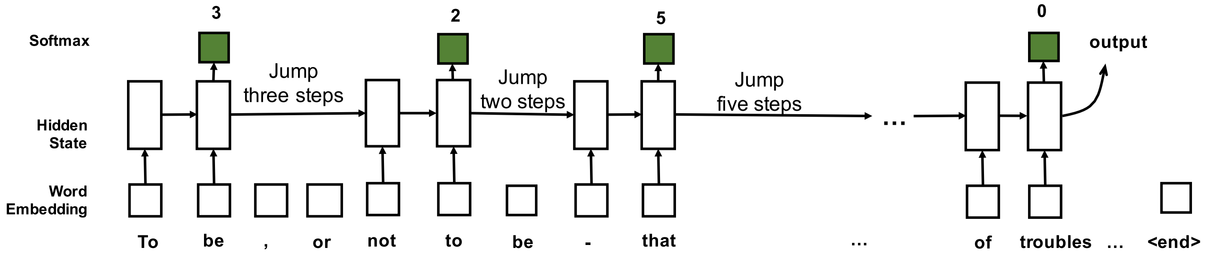


FIGURE 3. A synthetic example of LSTM-Jump model. In this example, the maximum size of jump is 5, the number of tokens read before a jump is 2 and the number of jumps allowed is 10. The green softmax are for jumping predictions. (Figure source: Yu *et al.* [54])

D. ACCELERATION METHODS

Despite that current open-domain textual QA systems have achieved significant advancements, these models become slow and cumbersome [110] with multi-layers [111], multi-stages [53], [102] architectures along with various features [81], [87], [137]. Moreover, ensemble models are employed to further improve performance, which requires a large number of computation resources. Open-domain textual QA systems, however, are required to be fast in paragraph index&ranking as well as accurate in answer extraction. Therefore, we would like to discuss some hot topics regarding acceleration methods in this section.

1) MODEL ACCELERATION

Due to the complex and computationally expensive deep learning models, automated machine learning (AutoML) technologies have aroused widespread concern on hyper-parameter optimization and neural architecture search methods [112]–[114]. However, there is little research about AutoML acceleration for the open-domain textual QA system. In order to reduce complexity under the guarantee of quality, there are many models proposed to accelerate reading processing, namely *model acceleration*. Hu *et al.* [115] proposed a knowledge distillation method, which transferred knowledge from an ensemble model to a single model with little loss in performance. In addition, it is known that LSTMs, which are widely used in the open domain textual QA systems [110], are difficult to parallelize and scale due to their sequential nature. Consequently some researchers replace the recurrent structures [110] or attention layer [96] with more efficient works, such as Transformer [83] and SRU [111], and limit the range of co-attention [116].

2) ACTION ACCELERATIONS

There are some works boosting the sequence reading speed while maintaining the performance, namely *action acceleration*. These approaches can dynamically employ some actions to speed up during reading, such as jumping, skipping, skimming, and early-stopping. We illustrate the details from the following perspectives.

- **Jump reading** determines from the current word how many words should be skipped before next reading. For example, Yu and Liu [54] proposed LSTM-Jump, which was build upon the basics of LSTM network and reinforcement learning, to determine the number of tokens or sentences to jump. As shown in Fig. 3, the softmax gave out a distribution over the jumping steps between 1 and the max jump size. This method can greatly improve reading efficiency, but the decision action can only jump forward, which may be ineffective in complex reasoning tasks. Therefore Yu *et al.* [117] proposed an approach to decide whether to skip tokens, re-read the current sentence or stop reading the feedback answer, and LSTM-shuttle [118] proposed a method to either read forward or read back to increase accuracy during speed reading.
- **Skim reading** determines whether to skim one token before reading the sentence according to the current word or not. Unlike previous methods using reinforcement learning to make action decisions, skip-rnn [119] adjusted the RNN module to determine whether each step input was skipped or directly copied the state of the previous hidden layer. However, previous methods are mainly for sequence reading and classification tasks, and the experiments are mainly for the cloze-style QA task [31]. Then Skim-rnn [55] conducted comparative experiments on the reading comprehension tasks. Specifically, skim-RNN was responsible for updating the first few dimensions of the hidden state through the small RNN, and weighted between the computation amount and the discard rate. Moreover, Hansen *et al.* [120] proposed the first speed reading model including both jump and skip actions.
- **Other speed reading applications**: JUMPER [36] provided fast reading feedback for legal texts, Johansen and Socher [121] focused on sentiment classification tasks. Choi *et al.* [122] tackle long document-oriented QA tasks for sentence selection and reading based on CNN. Hu *et al.* [38] proposed an early-stopping mechanism to efficiently terminate the encoding process of unrelated paragraphs.

E. DATASETS

In this subsection, we introduce several datasets relative to open-domain textual QA. Owing to the release of these datasets, the development of open-domain textual

TABLE 2. Data statistics of datasets.

dataset	query form	answer form	question source	context source	granularity
SQuAD-open [37]	full question	span	crowdsourced	Wikipedia	document level
SearchQA [15]	key word/ phrase	span	Jeopardy!	Google search results	paragraph level
TriviaQA [14]	full question	span	Trivia websites	Wikipedia & Bing search results	document level
Quasar-T [30]	full question	span	Free Database	CluWeb09	paragraph level

TABLE 3. Performance of some models on open-domain textual QA datasets.

model	SQuAD-open		SearchQA		TriviaQA-web		TriviaQA-Wikipedia		TriviaQA-unfiltered		Quasar-T	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
DrQA [37]	28.4	-	51.4	58.2	51.5	57.9	52.6	58.2	48.0	52.1	36.9	45.5
R^3 [47]	29.1	37.5	49.0	55.3	-	-	-	-	47.3	53.7	35.3	41.7
Re-Ranker [53]	-	-	57.0	63.2	63.0	68.5	50.2	55.5	50.6	-	42.3	49.6
Multi-Step [125]	31.9	39.2	55.0	61.6	-	-	-	-	55.9	61.7	40.0	46.7
DS-QA [48]	28.7	36.6	58.5	64.5	-	-	-	-	48.7	56.3	37.3	43.6
HAS-QA [103]	-	-	62.7	68.7	-	-	-	-	63.6	68.9	43.2	48.9
DocumentQA [49]	-	-	-	-	66.4	71.3	64.0	68.9	61.3	67.2	61.3	67.23
MemoReader [126]	-	-	-	-	**68.2**	**73.3**	64.4	69.6	-	-	**69.1**	**71.2**
DynSAN [127]	-	-	**64.2**	**70.3**	-	-	-	-	-	-	48.0	54.8
RE^3**QA** [38]	**41.9**	**50.2**	-	-	-	-	**71.0**	**75.2**	**65.5**	**71.2**	-	-
Human Performance	-	-	43.9	-	75.4	-	79.7	-	-	-	51.5	60.6

QA systems has made great progress in recent years. Table. 2 shows some statistics of the following datasets.

- **SQuAD-open** [37] is an open-domain textual question answering dataset based on SQuAD [13]. In SQuAD-Open, only question-answer pairs are given, while the evidence documents come from the whole Wikipedia articles.
- **SearchQA** [15] contains 140k question-answer pairs crawled from J! Archive. It uses Google search engine to collect the top 50 web page snippets as context fragments for each question.
- **TriviaQA** [14] consists of 650K context-query-answer triples, which contains three settings: web domain, Wikipedia domain, and unfiltered domain. The questions come from 14 trivia and quiz-league websites and needs cross-sentence reasoning to obtain the ground-truth answer. The evidence documents of TriviaQA-web and TriviaQA-Wikipedia are retrospectively crawled from Wikipedia or Web search. TriviaQA-unfiltered is the open domain setting of TriviaQA, which includes 110,495 QA pairs and 740K documents.
- **Quasar-T** [30] mainly consists of 43k open-domain trivia questions and their answers obtained from various Internet sources. For each question-answer pair, 100 paragraphs have been collected to process. There are two sub-sets according to the length of candidate paragraphs, where the short sub-set makes up of paragraphs with less than 10 sentences, and the long one makes up of paragraphs with an average of 20 sentences. This dataset is constructed by two processes: retrieving top-100 documents and adding top-N unique documents to the context document.

F. EVALUATION

For extractive textual QA tasks, in order to evaluate the predicted answer, we usually adopt two evaluation metrics [13], which measure exact match and partially overlapped scores respectively.

- **Exact Match.** EM measures whether the predicted answer exactly matches the ground-truth answers. If the exact matching occurs, then assigns 1.0, otherwise assigns 0.0.
- **F1 Score.** F1 score computes the average word overlap between predicted and ground-truth answers, which can ensure both of precision and recall rate are optimized at the same time, F1 score is calculated as:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{8}$$

We summarize the performance of current state-of-the-art models on different open-domain textual QA datasets, as shown in Table. 3. As we can see, MemoReader [124] has achieved promising results on the TriviaQA-web dataset, while DynSAN [125] is the top-tier model for SearchQA. On the other hand, RE^{3}QA [38] has achieved SOTA results on the remaining three datasets, likely due to the use of pre-trained language models such as BERT [61].

IV. DISCUSSION

In this paper, we introduce some recent approaches in open-domain textual QA. Although these works have established a solid foundation for deep-learning based open-domain textual QA research, there remains ample room for further improvement. In this section, we first summarize the structure of some typical models, then present the challenges and limitations of recent approaches, finally outline

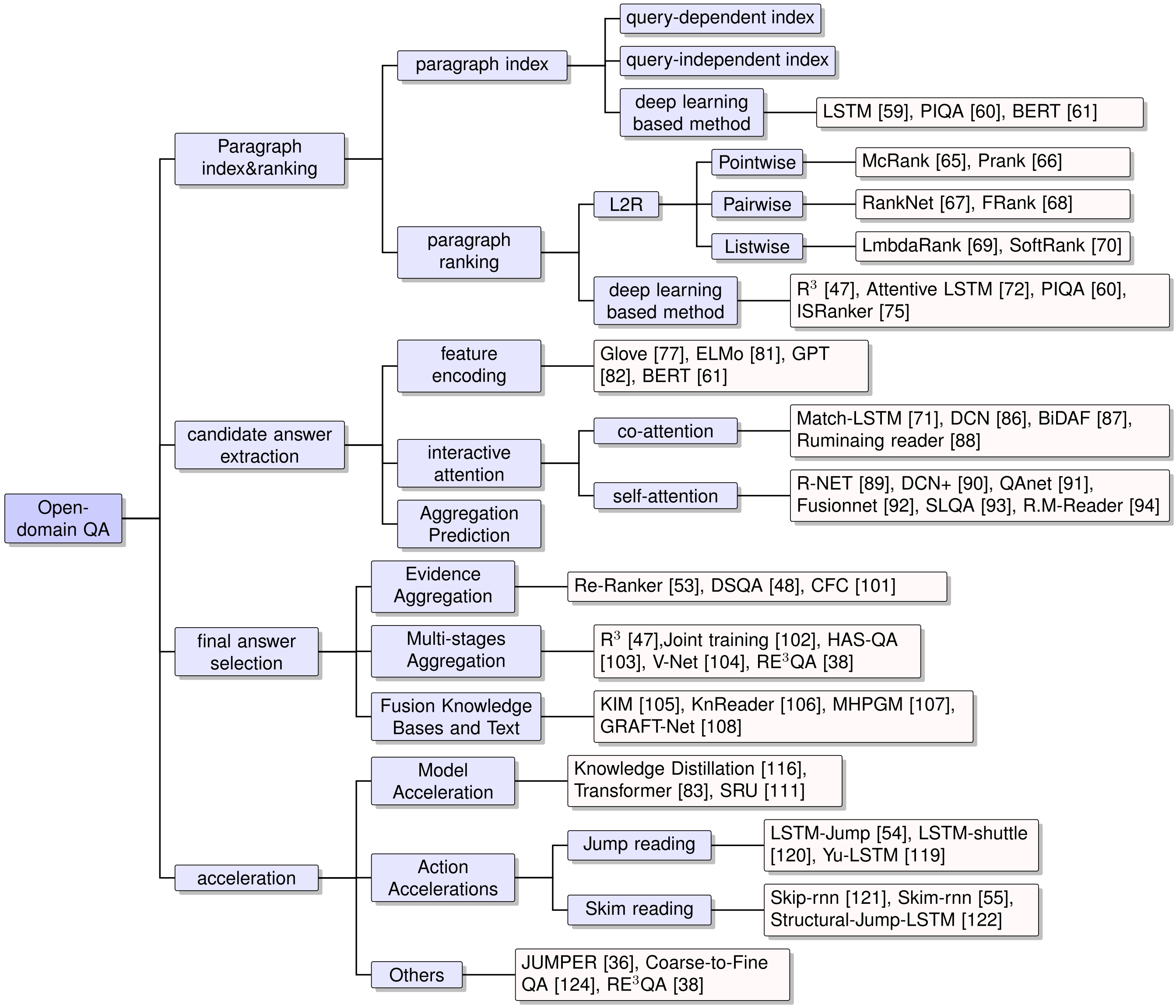

Open-domain QA
Paragraph index&ranking
paragraph index
query-dependent index
query-independent index
deep learning based method
LSTM [59], PIQA [60], BERT [61]
paragraph ranking
L2R
Pointwise
McRank [65], Prank [66]
Pairwise
RankNet [67], FRank [68]
Listwise
LmbdaRank [69], SoftRank [70]
deep learning based method
R^3 [47], Attentive LSTM [72], PIQA [60], ISRanker [75]
candidate answer extraction
feature encoding
Glove [77], ELMo [81], GPT [82], BERT [61]
interactive attention
co-attention
Match-LSTM [71], DCN [86], BiDAF [87], Ruminaing reader [88]
self-attention
R-NET [89], DCN+ [90], QAnet [91], Fusionnet [92], SLQA [93], R.M-Reader [94]
Aggregation Prediction
final answer selection
Evidence Aggregation
Re-Ranker [53], DSQA [48], CFC [101]
Multi-stages Aggregation
R^3 [47],Joint training [102], HAS-QA [103], V-Net [104], RE^{3}QA [38]
Fusion Knowledge Bases and Text
KIM [105], KnReader [106], MHPGM [107], GRAFT-Net [108]
acceleration
Model Acceleration
Knowledge Distillation [116], Transformer [83], SRU [111]
Action Accelerations
Jump reading
LSTM-Jump [54], LSTM-shuttle [120], Yu-LSTM [119]
Skim reading
Skip-rnn [121], Skim-rnn [55], Structural-Jump-LSTM [122]
Others
JUMPER [36], Coarse-to-Fine QA [124], RE^{3}QA [38]

FIGURE 4. Hot topics of open-domain QA and representative examples.

several promising prospective research directions, which we believe are critical to the present state of the field.

A. SUMMARY OF MODELS

We summarize current hot topics in Figure. 4 and categorize structure of some models in Table. 4 according to the technologies illustrated in Section III. There are some works that are designed in single-document QA settings, such as BiDAF [87], QAnet [91], and SLQA [93], where the ranking stage is not needed. On the other hand, some bunch of works need to search and filter the paragraphs from multiple documents in open-domain textual QA settings. So we divide Table. 4 into two parts, the upper for MRC models and the lower for open-domain textual QA models.

As can be seen from Table. 4, most works use IR methods such as TF-IDF and BM25 in the ranking stage. Recently, some works such as ORQA [62] and DFGN [84] adopt BERT [61] to select paragraphs. In the extractive reading stage, most works utilize Glove embeddings [77], while recent models tend to use pre-trained language models such as ELMo [81] or BERT [61] for text feature encoding. As for the attention mechanism, most works adopt either co-attention or self-attention, or combine both of them to better exchange information between questions and documents. For the aggregation prediction, most works adopt RNN-based approaches (LSTM or GRU), while some recent works leverage BERT [61]. In the final answer selection, the multi-stage aggregation is the main solution while few works adopt the evidence aggregation strategy.

B. CHALLENGES AND LIMITATIONS

We first present the challenges and limitations of open-domain textual QA systems due to the use of deep learning techniques. There are several common limitations

TABLE 4. The structure of some models. The top half of the table are the MRC models, and the bottom half are the open-domain textual QA models which contain the paragraph ranking stage.

model	ranking		extractive reading		answer selection
		encoding	attention	aggregation prediction	aggregation type
BiDAF [87]	-	Glove	co-attention	Bi-LSTM	-
DCN+ [90]	-	Glove	co-attention + self attention	Bi-LSTM	-
QAnet [91]	-	Glove	co-attention + self attention	BiDAF	-
FusionNet [92]	-	Glove	co-attention + self attention	GRU	-
Memoreader [126]	-	Glove	co-attention + self attention	GRU	-
FastQAExt [96]	-	Glove	co-attention + self attention	BiLSTM	-
SLQA [93]	-	Glove + ELMo	multi-granularity attention	BiLSTM	-
PIQA [60]	-	charCNN+Glove+ELMo	self-attention	BiLSTM	-
HAS-QA [103]	-	Glove	co-attention + self attention	BiGRU + BiDAF	-
DrQA [37]	TF-IDF	Glove	co-attention	Bi-LSTM	multi-stages
R^3 [47]	BM25 + TF-IDF	Glove	co-attention	Match-LSTM	multi-stages
DocumentQA [49]	TF-IDF	Glove	co-attention+self-attention	BiGRU	multi-stages
Re-Ranker [53]	BM25+TF-IDF	Glove	co-attention	Match-LSTM	evidence aggregation
DSQA [48]	RNN+MLP	Glove	self-attention	BiLSTM	multi-stages
RE^{3}QA [38]	TF-IDF	BERT	self-attention	BERT	multi-stages
ORQA [62]	BERT	BERT	self-attention	BERT	multi-stages
DFGN [84]	BERT	BERT	co-attention + self attention	LSTM	multi-stages

of deep learning techniques [126], which also affect deep learning based open-domain textual QA systems.

- **Interpretability.** It is well-known that the process of deep learning likes a black-box. Due to the activation function and backward derivation, it is hard to model the neural network function, which makes the final the answer unpredictable in theoretical.
- **Data Hungry.** As mentioned in subsection II-B, deep learning is data-driven, which also bring some challenges [126]. We can also find the fact in subsection III-E, where the total samples of each dataset are larger than 10k. It is very expensive to build large-scale datasets on open-domain textual QA even though annotation tools are provided. Specifically, the public dataset released by Google [127] consists of 307,373 training examples with single annotations; 7,830 examples with 5-way annotations for development data; and a further 7,842 examples 5-way annotated sequestered as test data.
- **Computing Resource Reliance.** In addition to large-scale data, large-scale neural network models are generally employed for the complex processing by deep learning based open-domain textual QA as mentioned in III-D. However, it is very consumption to train such complex models, while real time feedback is often required by user on QA systems. In such case, large-scale computing resource is the basic configuration for training or inference.

We then present several problems from the following three parts corresponding to Section III.

- **Index & Ranking.** Recent works usually adopt interactive attention mechanisms to improve the accuracy of ranking. However, it is not beneficial in both efficiency and scalability since each passage needs to be encoded along with individual questions. Although using BERT [61] or other self-attention pre-training model [82] to extract text features can improve the scalability, running these models over hundreds of paragraphs is computationally costly since these models usually have large size and consist of numerous layers. Moreover, Using indexable query-agnostic phrase representation can reduce the computationally cost while ensuring accuracy in reading comprehension, whereas the accuracy is still low in open-domain textual QA [128].
- **Machine Reading Comprehension.** Existing extractive reading technology has made great progress. Several reading comprehension models even surpass human performance. However, these MRC models are complex and lack of interpretation, which makes it difficult to evaluate the performance and analyze the generalization ability of each neuron module. With the improvement of performance along with the increase of model size, it is also a problem that running these models consumes a lot of energy [129]. Moreover, existing models are vulnerable to adversarial attacks [130], making it difficult to deploy them in real-world QA applications.
- **Aggregation Prediction.** Existing predictive reasoning usually supposes that the answer span only appears in the single paragraph, or the answer text is short [37]. However, in the real world, the answer span usually appears in several paragraphs or even requires multi-hop inference. How to aggregate evidence across multiple mentioned text snippets to find the answer remains to be a great challenge.

C. RECENT TRENDS

We summarize several recent trends regarding to open-domain textual QA, which are listed as follows.

1) **Complex Reasoning.** As the datasets get larger, reasoning becomes more complex, open-domain textual

QA task come up with a great deal of challenging subtasks. For example, multi-hop QA tasks, which include multiple evidence inference across documents [104], [107], [131], symbolic reasoning like numeric calculation, count and sort [132], and extraction-based generation [33], [34]. Combining complex reasoning modules such as graph-based reasoning [84], [133], [134], numerical reasoning [135] and logical reasoning [84], [131] with existing paragraph ranking and extractive reading comprehension models is a new trend in open-domain textual QA.

2) **Complexity Improvement.** Making accurate QA requires a deep understanding of documents and queries. As a result, most of recently proposed models become extremely complex and large [124], [136], resulting in low efficiency. It is nontrivial to speed up the whole computation, especially for the RNN-based models [83], [111]. Since AutoML [112], [113] technologies can automatically search optimal parameters or network structures, applying them in open-domain textual QA may be a good approach to find a light-weighted network structure for improving the efficiency.

3) **Technology Integration.** Technology integration refers to the combination of multiple technologies from different fields, which is a typical trend in the recent deep learning works. For example, the semantic paragraph ranking approaches [60], [75] may use the technologies from the fields of information retrieval and natural language processing. As for the answer selection module, knowledge base QA and natural language processing technologies are combined to improve the overall QA performance [106], [109]. Moreover, we can find that many machine learning technologies, such as transfer learning [61], reinforcement learning [47], and meta-learning [51], are integrated into open-domain textual QA systems to improve the performance.

V. CONCLUSION

In this paper, we provided an extensive review of the notable works on deep learning-based open-domain textual QA. We first explicitly gave the task scope of open-domain textual QA and then briefly overviewed the deep learning based open-domain textual QA systems, which consist of history of the task, reason of why deep learning are chosen and technical architecture of open-domain textual QA systems. Later we gave a detailed introduction on individual components inside the technical architecture, including paragraph index and ranking, candidate answer extraction, and final answer selection. Moreover, several acceleration methods, open-domain textual QA datasets and evaluation metrics are also discussed. Finally, we summarized current models, limitations and challenges, gave some of the recent trends and shed light on promising future research directions.

REFERENCES

[1] D. Ahn, V. Jijkoun, G. Mishne, K. Müller, M. de Rijke, and S. Schlobach, ''Using Wikipedia at the TREC QA Track,'' in *Proc. Text Retr. Conf. (TREC)*, 2004, pp. 1–11.

[2] P. Baudiš, ''YodaQA: A modular question answering system pipeline,'' in *Proc. Int. Student Conf. Electr. Eng. POSTER*, 2015, p. 8.

[3] E. Brill, S. Dumais, and M. Banko, ''An analysis of the AskMSR question-answering system,'' in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2002, pp. 257–264.

[4] A. M. Turing, ''Computing machinery and intelligence-AM Turing,'' *Mind*, vol. 59, pp. 433–460, Oct. 1950.

[5] X. Yao, ''Feature-driven question answering with natural language alignment,'' Ph.D. dissertation, Dept. Comput. Sci., Johns Hopkins Univ., Baltimore, MD, USA, 2014.

[6] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2009.

[7] B. Magnini, S. Romagnoli, A. Vallin, J. Herrera, A. Peñas, V. Peinado, F. Verdejo, and M. D. Rijke, ''The multiple language question answering track at CLEF 2003,'' in *Comparative Evaluation of Multilingual Information Access Systems* (Lecture Notes in Computer Science), vol. 3237. Cham, Switzerland: Springer, 2003, pp. 471–486.

[8] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, ''Freebase: A collaboratively created graph database for structuring human knowledge,'' in *Proc. ACM SIGMOD Int. Conf. Manage. Data* Aug. 2008, pp. 1247–1250.

[9] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, ''Dbpedia: A nucleus for a Web of open data,'' in *Proc. 6th Int. The Semantic Web 2nd Asian Conf. Asian Semantic Web Conf.*, Berlin, Germany, 2007, pp. 722–735.

[10] J. Berant, A. Chou, R. Frostig, and P. S. Liang, ''Semantic parsing on freebase from question-answer pairs,'' in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2013, pp. 1533–1544.

[11] A. Bordes, N. Usunier, S. Chopra, and J. Weston, ''Large-scale simple question answering with memory networks,'' 2015, *arXiv:1506.02075*. [Online]. Available: http://arxiv.org/abs/1506.02075

[12] 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 Mag.*, vol. 31, no. 3, p. 59, 2010.

[13] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, ''SQuAD: 100,000+ Questions for machine comprehension of text,'' in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 2383–2392.

[14] M. Joshi, E. Choi, D. Weld, and L. Zettlemoyer, ''TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension,'' in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2017, pp. 1601–1611.

[15] M. Dunn, L. Sagun, M. Higgins, V. Ugur Guney, V. Cirik, and K. Cho, ''SearchQA: A new Q&amp;A dataset augmented with context from a search engine,'' 2017, *arXiv:1704.05179*. [Online]. Available: http://arxiv.org/abs/1704.05179

[16] P. Gupta and V. Gupta, ''A survey of text question answering techniques,'' *Int. J. Comput. Appl.*, vol. 53, no. 4, pp. 1–8, 2012.

[17] 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, Dec. 2011.

[18] L. Kang and Y. Feng, *Deep Learning in Question Answering*. Singapore: Springer, 2018, Ch. 7, doi: 10.1007/978-981-10-5209-5.

[19] A. A. Shah, S. D. Ravana, S. Hamid, and M. A. Ismail, ''Accuracy evaluation of methods and techniques in Web-based question answering systems: A survey,'' *Knowl. Inf. Syst.*, vol. 58, no. 3, pp. 611–650, Mar. 2019.

[20] S. Liu, X. Zhang, S. Zhang, H. Wang, and W. Zhang, ''Neural machine reading comprehension: Methods and trends,'' 2019, *arXiv:1907.01118*. [Online]. Available: http://arxiv.org/abs/1907.01118

[21] X. Zhang, A. Yang, S. Li, and Y. Wang, ''Machine reading comprehension: A literature review,'' 2019, *arXiv:1907.01686*. [Online]. Available: http://arxiv.org/abs/1907.01686

[22] B. Qiu, X. Chen, J. Xu, and Y. Sun, ''A survey on neural machine reading comprehension,'' 2019, *arXiv:1906.03824*. [Online]. Available: http://arxiv.org/abs/1906.03824

[23] T. Young, D. Hazarika, S. Poria, and E. Cambria, ''Recent trends in deep learning based natural language processing [Review Article],'' *IEEE Comput. Intell. Mag.*, vol. 13, no. 3, pp. 55–75, Aug. 2018.

[24] D. Vrandečić and M. Krötzsch, ''Wikidata: A free collaborative knowledgebase,'' *Commun. ACM*, vol. 57, no. 10, pp. 78–85, Sep. 2014.

[25] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: A core of semantic knowledge," in *Proc. 16th Int. Conf. World Wide Web* New York, NY, USA, 2007, pp. 697–706

[26] S. Sarawagi and S. Chakrabarti, "Open-domain quantity queries on Web tables: Annotation, response, and consensus models," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 711–720.

[27] D. Downey, S. Dumais, D. Liebling, and E. Horvitz, "Understanding the relationship between searchers' queries and information goals," in *Proc. 17th ACM Conf. Inf. Knowl. Mining (CIKM)*, 2008, pp. 449–458.

[28] P. Pasupat and P. Liang, "Compositional semantic parsing on semi-structured tables," in *Proc. Int. Conf. World Wide Web*, Aug. 2014, pp. 1–11.

[29] J. Welbl, P. Stenetorp, and S. Riedel, "Constructing datasets for multi-hop reading comprehension across documents," *Trans. Assoc. Comput. Linguistics*, vol. 6, pp. 287–302, Dec. 2018.

[30] B. Dhingra, K. Mazaitis, and W. W. Cohen, "Quasar: Datasets for question answering by search and reading," 2017, *arXiv:1707.03904*. [Online]. Available: http://arxiv.org/abs/1707.03904

[31] K. M. Hermann, T. Koisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *Proc. Adv. Neural Inf. Process. Syst.*, Montreal, QC, Canada, Jan. 2015, pp. 1693–1701.

[32] F. Hill, A. Bordes, S. Chopra, and J. Weston, "The goldilocks principle: Reading children's books with explicit memory representations," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2016, pp. 1–13.

[33] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, and L. Deng, "MS MARCO: A human generated machine reading comprehension dataset," in *Proc. Workshop Cognit. Comput., Integrating Neural Symbolic Approaches Co-Located 30th Annu. Conf. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 1–10.

[34] T. Kočiský, J. Schwarz, P. Blunsom, C. Dyer, K. M. Hermann, G. Melis, and E. Grefenstette, "The NarrativeQA reading comprehension challenge," *Trans. Assoc. Comput. Linguistics*, vol. 6, pp. 317–328, Dec. 2018.

[35] G. Balikas, A. Krithara, I. Partalas, and G. Paliouras, "BioASQ: A challenge on large-scale biomedical semantic indexing and question answering," in *Multimodal Retrieval in the Medical Domain*. Cham, Switzerland: Springer, 2015, pp. 26–39.

[36] X. Liu, L. Mou, H. Cui, Z. Lu, and S. Song, "JUMPER: Learning when to make classification decisions in reading," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Stockholm, Sweden, Jul. 2018, pp. 4237–4243.

[37] D. Chen, A. Fisch, J. Weston, and A. Bordes, "Reading wikipedia to answer open-domain questions," in *Proc. Annu. Meeting Assoc. Comput. Linguistics (ACL)*, vol. 1. Vancouver, BC, Canada, 2017, pp. 1870–1879.

[38] M. Hu, Y. Peng, Z. Huang, and D. Li, "Retrieve, read, rerank: Towards End-to-End multi-document reading comprehension," in *Proc. 57th Annu. Meeting Assoc. for Comput. Linguistics*, Florence, Italy, 2019, pp. 2285–2295.

[39] B. Hu, Z. Lu, H. Li, and Q. Chen, "Convolutional neural network architectures for matching natural language sentences," in *Proc. Adv. Neural Inf. Process. Syst.*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Cambridge, MA, USA: Curran Associates, 2014, pp. 2042–2050.

[40] T. Kenter, A. Borisov, C. Van Gysel, M. Dehghani, M. de Rijke, and B. Mitra, "Neural networks for information retrieval," in *Proc. 11th ACM Int. Conf. Web Search Data Mining*, New York, NY, USA, 2018, pp. 779–780.

[41] L. Yunjuan, Z. Lijun, M. Lijuan, and M. Qinglin, "Research and application of information retrieval techniques in intelligent question answering system," in *Proc. 3rd Int. Conf. Comput. Res. Develop.*, Mar. 2011, pp. 188–190.

[42] Y. Hao, Y. Zhang, K. Liu, S. He, Z. Liu, H. Wu, and J. Zhao, "An End-to-End model for question answering over knowledge base with cross-attention combining global knowledge," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1. Vancouver, BC, Canada, 2017, pp. 221–231.

[43] B. Katz, S. Felshin, J. J. Lin, and G. Marton, "Viewing the Web as a virtual database for question answering," in *New Directions in Question Answering*. Palo Alto, CA, USA: AAAI Press, 2004, ch. 17, pp. 215–226.

[44] E. M. Voorhees, "The TREC-8 question answering track report," in *Proc. Text Retr. Conf. (TREC)*, 1999, pp. 77–82.

[45] M. Petrochuk and L. Zettlemoyer, "SimpleQuestions nearly solved: A new upperbound and baseline approach," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 554–558.

[46] J. Cheng, L. Dong, and M. Lapata, "Long short-term memory-networks for machine reading," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 551–561.

[47] S. Wang, M. Yu, X. Guo, Z. Wang, T. Klinger, W. Zhang, S. Chang, G. Tesauro, B. Zhou, and J. Jiang, "R 3: Reinforced ranker-reader for open-domain question answering," in *Proc. 32nd AAAI Conf. Artif. Intell.*, New Orleans, LA, USA, 2018, pp. 5981–5988.

[48] Y. Lin, H. Ji, Z. Liu, and M. Sun, "Denoising distantly supervised open-domain question answering," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2018, pp. 1736–1745.

[49] C. Clark and M. Gardner, "Simple and effective multi-paragraph reading comprehension," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2018, pp. 845–855.

[50] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Comput. Surv.*, vol. 52, no. 1, pp. 1–38, Feb. 2019.

[51] J. Chen, X. Qiu, P. Liu, and X. Huang, "Meta multi-task learning for sequence modeling," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 5070–5077.

[52] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 843–852.

[53] 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 *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–14.

[54] A. W. Yu, H. Lee, and Q. Le, "Learning to skim text," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2017, pp. 1880–1890.

[55] M. Seo, S. Min, A. Farhadi, and H. Hajishirzi, "Neural speed reading via skim-RNN," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–14.

[56] B. Mitra and N. Craswell, "An introduction to neural information retrieval," *Found. Trends Inf. Retr.*, vol. 13, no. 1, pp. 1–126, Dec. 2018.

[57] K. D. Onal, Y. Zhang, I. S. Altingovde, M. M. Rahman, P. Karagoz, A. Braylan, B. Dang, H.-L. Chang, H. Kim, Q. McNamara, A. Angert, E. Banner, V. Khetan, T. McDonnell, A. T. Nguyen, D. Xu, B. C. Wallace, M. de Rijke, and M. Lease, "Neural information retrieval: At the end of the early years," *Inf. Retr. J.*, vol. 21, nos. 2–3, pp. 111–182, Jun. 2018.

[58] J. Chu-Carroll, J. Fan, B. K. Boguraev, D. Carmel, D. Sheinwald, and C. Welty, "Finding needles in the haystack: Search and candidate generation," *IBM J. Res. Develop.*, vol. 56, nos. 3–4, p. 6, May 2012.

[59] S. Kato, R. Togashi, H. Maeda, S. Fujita, and T. Sakai, "LSTM vs. BM25 for open-domain QA: A hands-on comparison of effectiveness and efficiency," in *40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, 2017, pp. 1309–1312.

[60] M. Seo, T. Kwiatkowski, A. Parikh, A. Farhadi, and H. Hajishirzi, "Phrase-indexed question answering: A new challenge for scalable document comprehension," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 559–564.

[61] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. ACL Conf. NAACL HLT*, Minneapolis, MN, USA, Jun. 2019, pp. 4171–4186.

[62] K. Lee, M.-W. Chang, and K. Toutanova, "Latent retrieval for weakly supervised open domain question answering," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, Florence, Italy, 2019, pp. 6086–6096.

[63] E. H. Hovy, L. Gerber, U. Hermjakob, M. Junk, and C. Lin, "Question answering in webclopedia," in *Proc. 9th Text Retr. Conf. (TREC)*, 2000, pp. 1–10.

[64] T.-Y. Liu, "Learning to rank for information retrieval," *Found. Trends Inf. Retr.*, vol. 3, no. 3, pp. 225–331, 2007.

[65] P. Li, C. J. C. Burges, and Q. Wu, "McRank: Learning to rank using multiple classification and gradient boosting," in *Proc. 20th Int. Conf. Neural Inf. Process. Syst.*, Jul. 2007, pp. 897–904.

[66] K. Crammer and Y. Singer, "Pranking with ranking," in *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 2001, pp. 641–647.

[67] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," in *Proc. 22nd Int. Conf. Mach. Learn. (ICML)*, 2005, pp. 89–96.

[68] M.-F. Tsai, T.-Y. Liu, T. Qin, H.-H. Chen, and W.-Y. Ma, "FRank: A ranking method with fidelity loss," in *Proc. 30th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2007, pp. 383–390.

[69] C. J. C. Burges, R. Ragno, and Q. V. Le, "Learning to rank with nonsmooth cost functions," in *Proc. 19th Int. Conf. Neural Inf. Process. Syst.*, Jun. 2006, pp. 193–200.

[70] M. Taylor, J. Guiver, S. Robertson, and T. Minka, "SoftRank: Optimizing non-smooth rank metrics," in *Proc. Int. Conf. Web Search Data Mining*, Aug. 2008, pp. 77–86.

[71] S. Wang and J. Jiang, "Machine comprehension using match-LSTM and answer pointer," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–3.

[72] M. Tan, C. dos Santos, B. Xiang, and B. Zhou, "Improved representation learning for question answer matching," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2016.

[73] A. Shrivastava and P. Li, "Improved asymmetric locality sensitive hashing (ALSH) for maximum inner product search (MIPS)," in *Proc. 31st Conf. Uncertainty Artif. Intell.*, Arlington, VA, USA, 2015, pp. 812–821.

[74] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with GPUs," 2017, *arXiv:1702.08734*. [Online]. Available: http://arxiv.org/abs/1702.08734

[75] P. M. Htut, S. Bowman, and K. Cho, "Training a ranking function for open-domain question answering," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Student Res. Workshop*, 2018, pp. 120–127.

[76] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 670–680.

[77] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.

[78] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, "The stanford CoreNLP natural language processing toolkit," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics: Syst. Demonstrations*, 2014, pp. 55–60.

[79] M. Gardner, J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N. F. Liu, M. Peters, M. Schmitz, and L. Zettlemoyer, "AllenNLP: A deep semantic natural language processing platform," in *Proc. Workshop NLP Open Source Softw. (NLP-OSS)*, 2018, pp. 1–6.

[80] S. Bird and E. Loper, "NLTK: The natural language toolkit," in *Proc. ACL Interact. Poster Demonstration Sessions*, 2004, pp. 1–5.

[81] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, vol. 1, 2018, pp. 2227–2237.

[82] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding with unsupervised learning," Openai, San Francisco, CA, USA, Tech. Rep., 2018. [Online]. Available: https://openai.com/blog/language-unsupervised/

[83] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., 2017, pp. 5998–6008.

[84] L. Qiu, Y. Xiao, Y. Qu, H. Zhou, L. Li, W. Zhang, and Y. Yu, "Dynamically fused graph network for multi-hop reasoning," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 6140–6150.

[85] S. Wang and J. Jiang, "Learning natural language inference with LSTM," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2016, pp. 1442–1451.

[86] C. Xiong, V. Zhong, and R. Socher, "Dynamic coattention networks for question answering," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–11.

[87] M. J. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Toulon, France, 2017, pp. 1–14.

[88] Y. Gong and S. Bowman, "Ruminating reader: Reasoning with gated multi-hop attention," in *Proc. Workshop Mach. Reading Question Answering*, 2018, pp. 1–11.

[89] W. Wang, N. Yang, F. Wei, B. Chang, and M. Zhou, "Gated self-matching networks for reading comprehension and question answering," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2017, pp. 189–198.

[90] C. Xiong, V. Zhong, and R. Socher, "DCN+: Mixed objective and deep residual coattention for question answering," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–10.

[91] A. W. Yu, D. Dohan, Q. Le, T. Luong, R. Zhao, and K. Chen, "QANet: Combining local convolution with global self-attention for reading comprehension," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–16.

[92] H.-Y. Huang, C. Zhu, Y. Shen, and W. Chen, "Fusionnet: Fusing via fully-aware attention with application to machine comprehension," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–20.

[93] W. Wang, M. Yan, and C. Wu, "Multi-granularity hierarchical attention fusion networks for reading comprehension and question answering," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2018, pp. 1705–1714.

[94] M. Hu, Y. Peng, Z. Huang, X. Qiu, F. Wei, and M. Zhou, "Reinforced mnemonic reader for machine reading comprehension," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 4099–4106.

[95] Z. Wang, H. Mi, W. Hamza, and R. Florian, "Multi-perspective context matching for machine comprehension," 2016, *arXiv:1612.04211*. [Online]. Available: http://arxiv.org/abs/1612.04211

[96] D. Weissenborn, G. Wiese, and L. Seiffe, "Making neural QA as simple as possible but not simpler," in *Proc. 21st Conf. Comput. Natural Lang. Learn. (CoNLL)*, 2017, pp. 271–280.

[97] Y. Shen, P.-S. Huang, J. Gao, and W. Chen, "ReasoNet: Learning to stop reading in machine comprehension," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, 2017, pp. 1047–1055.

[98] X. Wang, Z. Huang, Y. Zhang, L. Tan, and Y. Liu, "DSDR: Dynamic semantic discard reader for open-domain question answering," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–7.

[99] X. Liu, Y. Shen, K. Duh, and J. Gao, "Stochastic answer networks for machine reading comprehension," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2018, pp. 1694–1704.

[100] M. Hu, F. Wei, Y. Peng, Z. Huang, N. Yang, and D. Li, "Read + verify: Machine reading comprehension with unanswerable questions," *Proc. AAAI Conf. Artif. Intell.*, vol. 33, pp. 6529–6537, Jul. 2019.

[101] V. Zhong, C. Xiong, N. Keskar, and R. Socher, "Coarse-grain fine-grain coattention network for multi-evidence question answering," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–27.

[102] Z. Wang, J. Liu, X. Xiao, Y. Lyu, and T. Wu, "Joint training of candidate extraction and answer selection for reading comprehension," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2018, pp. 1715–1724.

[103] L. Pang, Y. Lan, J. Guo, J. Xu, L. Su, and X. Cheng, "HAS-QA: Hierarchical answer spans model for open-domain question answering," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019., pp. 6875–6882

[104] Y. Wang, K. Liu, J. Liu, W. He, Y. Lyu, H. Wu, S. Li, and H. Wang, "Multi-passage machine reading comprehension with cross-passage answer verification," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, 2018, pp. 1918–1927.

[105] Q. Chen, X. Zhu, Z.-H. Ling, D. Inkpen, and S. Wei, "Neural natural language inference models enhanced with external knowledge," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1. Melbourne, NSW, Australia, 2018, pp. 2406–2417.

[106] T. Mihaylov and A. Frank, "Knowledgeable reader: Enhancing cloze-style reading comprehension with external commonsense knowledge," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*,vol. 1. Melbourne, NSW, Australia, 2018, pp. 821–832.

[107] L. Bauer, Y. Wang, and M. Bansal, "Commonsense for generative multi-hop question answering tasks," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Brussels, Belgium, 2018, pp. 1–32.

[108] 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 *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 4231–4242.

[109] D. Weissenborn, T. Kočiský, and C. Dyer, "Dynamic integration of background knowledge in neural NLU systems," 2017, *arXiv:1706.02596*. [Online]. Available: http://arxiv.org/abs/1706.02596

[110] D. Chen, "Neural reading comprehension and beyond," Ph.D. dissertation, Dept. Comput. Sci., Stanford Univ., Stanford, CA, USA, 2018.

[111] T. Lei, Y. Zhang, S. I. Wang, H. Dai, and Y. Artzi, "Simple recurrent units for highly parallelizable recurrence," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 4470–4481.

[112] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, and F. Hutter, "Efficient and robust automated machine learning," in *Proc. Adv. Neural Inf. Process. Syst. Annu. Conf. Neural Inf. Process. Syst.*, 2015, pp. 2962–2970.

[113] F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., *Efficient and Robust Automated Machine Learning*. Berlin, Germany: Springer, 2018.

[114] S. Estevez-Velarde, Y. Gutiérrez, A. Montoyo, and Y. Almeida-Cruz, "AutoML strategy based on grammatical evolution: A case study about knowledge discovery from text," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, Florence, Italy, 2019, pp. 4356–4365.

[115] M. Hu, Y. Peng, F. Wei, Z. Huang, D. Li, N. Yang, and M. Zhou, "Attention-guided answer distillation for machine reading comprehension," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 2077–2086.

[116] S. Min, V. Zhong, R. Socher, and C. Xiong, "Efficient and robust question answering from minimal context over documents," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2018, pp. 1725–1735.

[117] A. G. S. J. P. Keyi Yu and Y. Liu, "Fast and accurate text classification: Skimming, rereading and early stopping," in *ICLR workshop*, 2018, pp. 1–12.

[118] T.-J. Fu and W.-Y. Ma, "Speed reading: Learning to read ForBackward via shuttle," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 4439–4448.

[119] V. Campos, B. Jou, X. G. I. Nieto, J. Torres, and S.-F. Chang, "Skip RNN: Learning to skip state updates in recurrent neural networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–17.

[120] C. Hansen, C. Hansen, S. Alstrup, J. G. Simonsen, and C. Lioma, "Neural speed reading with structural-JUMP-LSTM," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–10.

[121] A. Johansen and R. Socher, "Learning when to skim and when to read," in *Proc. 2nd Workshop Represent. Learn. (NLP)*, 2017, pp. 257–264.

[122] E. Choi, D. Hewlett, J. Uszkoreit, I. Polosukhin, A. Lacoste, and J. Berant, "Coarse-to-Fine question answering for long documents," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2017, pp. 209–220.

[123] R. Das, S. Dhuliawala, M. Zaheer, and A. McCallum, "Multi-step retriever-reader interaction for scalable open-domain question answering," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–13.

[124] S. Back, S. Yu, S. R. Indurthi, J. Kim, and J. Choo, "MemoReader: Large-scale reading comprehension through neural memory controller," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 2131–2140.

[125] Y. Zhuang and H. Wang, "Token-level dynamic self-attention network for multi-passage reading comprehension," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, Florence, Italy, 2019, pp. 2252–2262.

[126] G. Marcus, "Deep learning: A critical appraisal," 2018, *arXiv:1801.00631*. [Online]. Available: http://arxiv.org/abs/1801.00631

[127] 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," *Trans. Assoc. Comput. Linguistics*, vol. 7, pp. 453–466, Aug. 2019.

[128] 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 *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, Florence, Italy, 2019, pp. 4430–4441.

[129] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in NLP," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, Florence, Italy, 2019, pp. 1–6.

[130] E. Wallace, S. Feng, N. Kandpal, M. Gardner, and S. Singh, "Universal adversarial triggers for attacking and analyzing NLP," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 2153–2162.

[131] 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 *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 1–12.

[132] 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. Conf. North Amer. Chapter Assoc. Comput. Linguistics (NAACL)*, 2019, pp. 1–12.

[133] M. Ding, C. Zhou, Q. Chen, H. Yang, and J. Tang, "Cognitive graph for multi-hop reading comprehension at scale," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, Florence, Italy, 2019, pp. 2694–2703.

[134] M. Tu, G. Wang, J. Huang, Y. Tang, X. He, and B. Zhou, "Multi-hop reading comprehension across multiple documents by reasoning over heterogeneous graphs," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, Florence, Italy, 2019, pp. 2704–2713.

[135] M. Hu, Y. Peng, Z. Huang, and D. Li, "A multi-type multi-span network for reading comprehension that requires discrete reasoning," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 1596–1606.

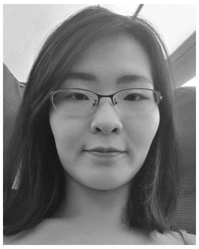
[136] S. Yu, S. R. Indurthi, S. Back, and H. Lee, "A multi-stage memory augmented neural network for machine reading comprehension," in *Proc. Workshop Mach. Reading Question Answering*, 2018, pp. 21–30.

[137] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin, "Advances in pre-training distributed word representations," in *Proc. 11th Int. Conf. Lang. Resour. Eval. (LREC)*. Miyazaki, Japan: European Language Resources Association (ELRA), May 2018. [Online]. Available: https://www.aclweb.org/anthology/L18-1008

ZHEN HUANG was born in Hunan, China, in 1984. He received the B.S. and Ph.D. degrees from the National University of Defense Technology (NUDT), in 2006 and 2012, respectively. He was a Visited Student with Eurecom, in 2009. From 2012 to 2016, he was an Assistant Professor with the Science and Technology on Parallel and Distributed Laboratory (PDL), NUDT, where he is currently an Associate Professor. He is the author of more than 40 articles. His research interests include natural language processing, distributed storage, and artificial intelligence. His Ph.D. Thesis received the Excellence Doctoral Thesis Award of Hunan Province. He also received the Best Paper of ICCCT, in 2011.

SHIYI XU was born in Hubei, China, in 1991. She received the B.E. degree from Minnan Normal University, China, in 2013. She is currently pursuing the master's degree with the Science and Technology on Parallel and Distributed Laboratory (PDL), National University of Defense Technology (NUDT), Changsha, China. Her research interests include natural language processing and artificial intelligence.

MINGHAO HU received the M.S. degree from the National University of Defense Technology (NUDT), in 2016, where he is currently pursuing the Ph.D. degree. He has published articles in top-tier conferences, such as ACL, EMNLP, AAAI, and IJCAI. His research interests include natural language processing and machine reading comprehension.



XINYI WANG was born in China, in 1995. She received the B.E. degree from the National University of Defense Technology (NUDT), where she is currently pursuing the master's degree with the Science and Technology on Parallel and Distributed Laboratory (PDL). Her research interest includes natural language processing.

JINYAN QIU received the M.S. degree in computer science and technology from the National University of Defense Technology (NUDT), in 2008. He is currently an Assistant Engineer with the H.R. Support Center. His research interests include deep learning and big data.

YONGQUAN FU received the M.S. and Ph.D. degrees in computer science and technology from the National University of Defense Technology (NUDT), in 2007 and 2012, respectively. He is currently an Associate Professor with NUDT. His research interests include network machine learning and distributed systems.

YUNCAI ZHAO was born in Hunan, China, in 1975. He received the bachelor's degree from the Naval University of Engineering, in 1994. He is currently a Senior Engineer with the Unit 31011, PLA. His research interests include artificial intelligence, international relation, and international strategy.

YUXING PENG was born in 1963. He received the bachelor's degree in computer from the Beijing University of Aeronautics and Astronautics, and the M.S. and Ph.D. degrees from the National University of Defense Technology (NUDT). He was a Head Coach with the School's ACM Programming Contest. He is currently a Researcher in computer science and a Ph.D. Supervisor with the Science and Technology on Parallel and Distributed Laboratory (PDL), NUDT. He has trained more than 50 gold medal winners and more than 70 silver medal winners in international contests. His research interests include studying distributed computing technology, virtual computing environment, cloud computing, big data, and intelligent computing and other relevant topics. He received the Gold Medal of the Military Academy Talents Cultivation Award, in 2010, the Excellent Doctoral Thesis Mentor of Hunan Province, in 2013, and the ACM ICPC World Final Outstanding Coach Award, in 2015.

CHANGJIAN WANG received the B.S., M.S., and Ph.D. degrees in computer science and technology from the National University of Defense Technology (NUDT), Changsha, China. He is currently an Associate Professor with NUDT. His research interests include database, distributed computing, cloud computing, big data, and machine learning.