Adaptive Information Seeking for Open-Domain Question Answering

Yunchang Zhu[†§], **Liang Pang**[†*], **Yanyan Lan**[◇*], **Huawei Shen**[†§], **Xueqi Cheng**[‡§]

[†]Data Intelligence System Research Center
and [‡]CAS Key Lab of Network Data Science and Technology,
Institute of Computing Technology, Chinese Academy of Sciences
[§]University of Chinese Academy of Sciences
[◇]Institute for AI Industry Research, Tsinghua University
{zhuyunchang17s, pangliang, shenhuawei, cxq}@ict.ac.cn
lanyanyan@tsinghua.edu.cn

Abstract

Information seeking is an essential step for open-domain question answering to efficiently gather evidence from a large corpus. Recently, iterative approaches have been proven to be effective for complex questions, by recursively retrieving new evidence at each step. However, almost all existing iterative approaches use predefined strategies, either applying the same retrieval function multiple times or fixing the order of different retrieval functions, which cannot fulfill the diverse requirements of various questions. In this paper, we propose a novel adaptive information-seeking strategy for open-domain question answering, namely AISO. Specifically, the whole retrieval and answer process is modeled as a partially observed Markov decision process, where three types of retrieval operations (e.g., BM25, DPR, and hyperlink) and one answer operation are defined as actions. According to the learned policy, AISO could adaptively select a proper retrieval action to seek the missing evidence at each step, based on the collected evidence and the reformulated query, or directly output the answer when the evidence set is sufficient for the question. Experiments on SQuAD Open and HotpotQA fullwiki, which serve as single-hop and multi-hop open-domain QA benchmarks, show that AISO outperforms all baseline methods with predefined strategies in terms of both retrieval and answer evaluations.

1 Introduction

Open-domain question answering (QA) (Voorhees et al., 1999) is a task of answering questions using a large collection of texts (e.g., Wikipedia). It relies on a powerful information-seeking method to efficiently retrieve evidence from the given large corpus.

Traditional open-domain QA approaches mainly follow the two-stage retriever-reader pipeline (Chen et al., 2017; Yang et al., 2018; Karpukhin

Figure 1: An example derived from HotpotQA development set. P1, P2 and P3 are the most relevant passages, of which P2 and P3 are supporting passages, which are essential to answer the question. Except for the adaptive strategy in the last row, fixed strategy methods such as using BM25 or dense retrieval multiple times and first using BM25 and then entity linking have failed, due to the rank of the remaining supporting passages larger than 1k. The number between two arrows indicates the highest rank of the remaining supporting passages in the retrieval list, unless ranked first.

et al., 2020), in which the retriever uses a determinate sparse or dense retrieval function to retrieve evidence, independently from the reading stage. But these approaches have limitations in answering complex questions, which need multi-hop or logical reasoning (Xiong et al., 2021).

To tackle this issue, iterative approaches have been proposed to recurrently retrieve passages and reformulate the query based on the original question and the previously collected passages. Nevertheless, all of these approaches adopt fixed information-seeking strategies in the iterative process. For example, some works employ a single retrieval function multiple times (Das et al., 2019a; Qi et al., 2019; Xiong et al., 2021), and the other works use a pre-defined sequence of retrieval functions (Asai et al., 2020; Dhingra et al., 2020).

However, the fixed information-seeking strategies cannot meet the diversified requirements of

---

*Corresponding Author

various problems. Taking Figure 1 as an example, the answer to the question is 'Catwoman' in P3. Due to the lack of essential supporting passages, simply applying BM25/dense retrieval (DR) multiple times (strategy 1 (Qi et al., 2019) or 2 (Xiong et al., 2021)), or using the mixed but fixed strategy (strategy 3 (Asai et al., 2020)) cannot answer the question. Specifically, it is hard for Qi et al. (2019) to generate the ideal query 'Catwoman game' by considering P1 or P2, thus BM25 (Robertson and Zaragoza, 2009) suffers from the mismatch problem and fails to find the next supporting passage P3. The representation learning of salient but rare phrases (e.g. 'Pitof') still remains a challenging problem (Karpukhin et al., 2020), which may affect the effectiveness of dense retrieval, i.e., the supporting passage P3 is ranked 65, while P1 and P2 do not appear in the top-1000 list at the first step. Furthermore, link retrieval functions fail when the current passage, e.g., P2, has no valid entity links.

Motivated by the above observations, we propose an **A**daptive **I**nformation-**S**eeking approach for **O**pen-domain QA, namely AISO. Firstly, the task of open-domain QA is formulated as a partially observed Markov decision process (POMDP) to reflect the interactive characteristics between the QA model (i.e., agent) and the intractable large-scale corpus (i.e., environment). The agent is asked to perform an action according to its state (belief module) and the policy it learned (policy module). Specifically, the belief module of the agent maintains a set of evidence to form its state. Moreover, there are two groups of actions for the policy module to choose, 1) retrieval action that consists of the type of retrieval function and the reformulated query for requesting evidence, and 2) answer action that returns a piece of text to answer the question, then completes the process. Thus, in each step, the agent emits an action to the environment, which returns a passage as the observation back to the agent. The agent updates the evidence set and generates the next action, step by step, until the evidence set is sufficient to trigger the answer action to answer the question. To learn such a strategy, we train the policy in imitation learning by cloning the behavior of an oracle online, which avoids the hassle of designing reward functions and solves the POMDP in the fashion of supervised learning.

Our experimental results show that our approach achieves better retrieval and answering performance than the state-of-the-art approaches on SQuAD Open and HotpotQA fullwiki, which are the representative single-hop and multi-hop datasets for open-domain QA. Furthermore, AISO significantly reduces the number of reading steps in the inference stage.

In summary, our contributions include:

- To the best of our knowledge, we are the first to introduce the adaptive information-seeking strategy to the open-domain QA task;

- Modeling adaptive information-seeking as a POMDP, we propose AISO, which learns the policy via imitation learning and has great potential for expansion.

- The proposed AISO achieves state-of-the-art performance on two public dataset and wins the first place on the HotpotQA fullwiki leaderboard. Our code is available at https://github.com/zycdev/AISO.

2 Related Work

Traditional approaches of open-domain QA mainly follow the two-stage retriever-reader pipeline (Chen et al., 2017): a retriever first gathers relevant passages as evidence candidates, then a reader reads the retrieved candidates to form an answer. In the retrieval stage, most approaches employ a determinate retrieval function and treat each passage independently (Wang et al., 2018; Lin et al., 2018; Lee et al., 2018; Yang et al., 2018; Pang et al., 2019; Lee et al., 2019; Guu et al., 2020; Karpukhin et al., 2020; Izacard and Grave, 2021). As an extension, some approaches further consider the relations between passages through hyperlinks or entity links and extend evidence with the linked neighbor passages (Nie et al., 2019; Das et al., 2019b; Zhao et al., 2020). However, pipeline approaches retrieve evidence independently from reader, leading to 1) introduce less-relevant evidence to the question, and 2) hard to model the complex question which has high-order relationship between question and evidence.

Instead, recent iterative approaches sequentially retrieve new passages by updating the query inputted to a specific retrieval function at each step, conditioned on the information already gathered. At each step, Das et al. (2019a); Feldman and El-Yaniv (2019); Xiong et al. (2021) reformulate the dense query vector in a latent space, while Ding et al. (2019); Qi et al. (2019); Zhang et al. (2020);
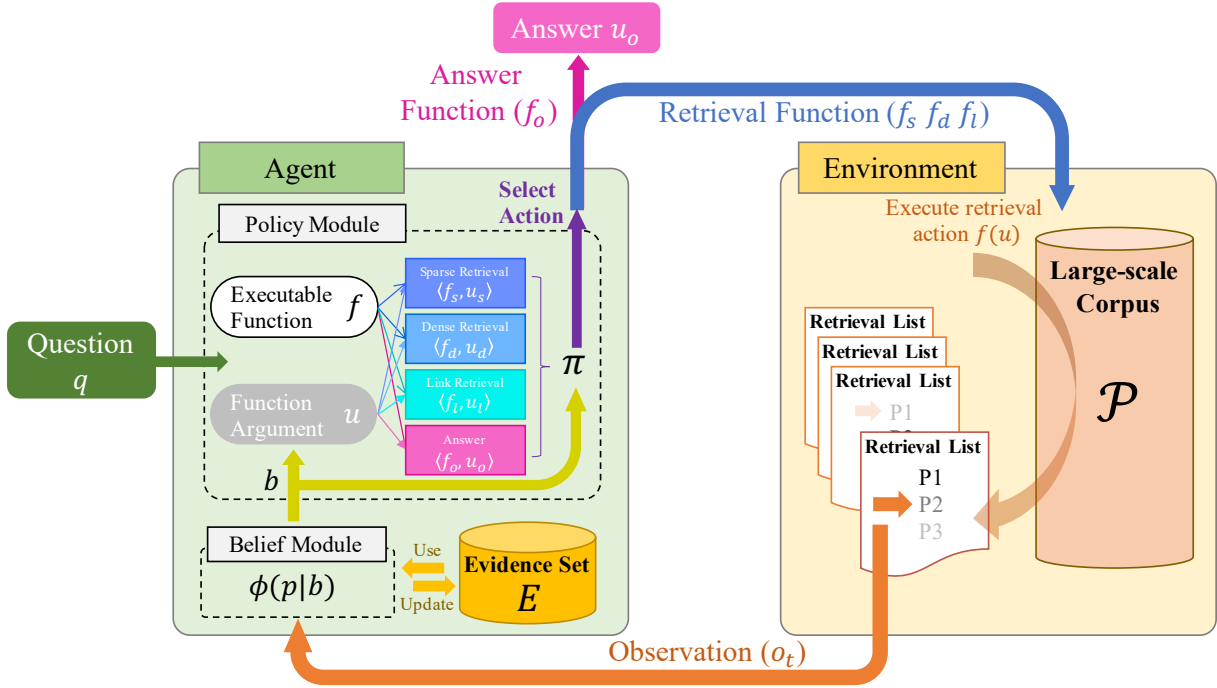
Figure 2: The overview of the AISO.

Qi et al. (2020) update the natural language query. After the first step retrieval using TF-IDF, Asai et al. (2020) and Li et al. (2021) recursively select subsequent supporting passages on top of a hyperlinked passage graph. Nevertheless, all of these approaches adopt fixed information-seeking strategies, employing the same retrieval function multiple times (Das et al., 2019a; Feldman and El-Yaniv, 2019; Xiong et al., 2021; Ding et al., 2019; Qi et al., 2019; Zhang et al., 2020; Qi et al., 2020) or pre-designated sequence of applying retrieval functions (Asai et al., 2020; Li et al., 2021). Due to the diversity of questions, these fixed strategies established in advance may not be optimal for all questions, or even fail to collect evidence.

3 Method

In this section, we first formulate the open-domain QA task as a partially observed Markov decision process (POMDP) and introduce the dynamics of the environment. Then, we elaborate on how the agent interacts with the environment to seek evidence and answer a question. Finally, to solve the POMDP, we describe how to train the agent via imitation learning.

3.1 Open-Domain QA as a POMDP

Given a question q and a large corpus $\mathcal{P}$ composed of passages, the task of open-domain QA is to col-

lect a set of evidence $E \subset \mathcal{P}$ and answer the question based on the gathered evidence.

The fashion of iterative evidence gathering, proven effective by previous works (Das et al., 2019a; Asai et al., 2020; Xiong et al., 2021), is essentially a sequential decision-making process. Besides, since the corpus is large, ranging from millions (e.g., Wikipedia) to billions (e.g., the Web), and the input length of a QA model is limited, the QA model can only observe a part of the corpus. Owing to the above two reasons, we model open-domain QA as a partially observed Markov decision process.

In the POMDP we designed, as shown in Figure 2, the agent is the QA model that needs to issue actions to seek evidence from the large-scale corpus hidden in the environment and finally respond to the question. By executing the received action, the environment can return a retrieved passage to the agent as an observation of the corpus. Formally, the POMDP is defined by $(\mathcal{S}, \mathcal{A}, \mathcal{O}, \Omega, Z, R)$, where R is the reward function.

Actions: At timestep $t = 0, 1, \cdots, T$, the action a_t in the action space $\mathcal{A} = \mathcal{F} \times \mathcal{U}$ is a request for an executable function $f \in \mathcal{F}$, expressed as $\langle f, u \rangle$, where $u \in \mathcal{U}$ is the text argument that gets passed to f. The space of executable functions $\mathcal{F}$ includes two groups of functions, 1) retrieval function that takes the query u and corpus $\mathcal{P}$ as

input and ranks a retrieval list of passages as $\mathcal{P}_{f(u)}$, 2) answer function that replies to the question q with the answer u and ends the process. The action a_t is performed following the policy Π described in Subsection 3.2.2.

States: The environment state s_t in the state space $\mathcal{S}$ contains revealing states of retrieval lists of all history retrieval actions. When the agent issues an action $a_t = \langle f, u \rangle$, s_t will transfer to s_{t+1} governed by a deterministic transition dynamics $\Omega(s_t, a_t)$. Specifically, Ω will mark the topmost unrevealed passage in the retrieval list $\mathcal{P}_{f(u)}$ as revealed. If the environment has never executed a_t before, it will first search and cache $\mathcal{P}_{f(u)}$ for possible repeated retrieval actions in the future.

Observations: On reaching the new environment state s_{t+1}, the environment will return an observation o_{t+1} from the observation space $\mathcal{O} = \{q\} \cup \mathcal{P}$, governed by the deterministic observation dynamics Z. At the initial timestep, the question q will returned as o_0. In other cases, Z is designed to return only the last passage marked as revealed in $\mathcal{P}_{f(u)}$ at a time. For example, if the action $\langle f, u \rangle$ is received for the kth time, the kth passage in $\mathcal{P}_{f(u)}$ will be returned.

3.2 Agent

The agent interacts with the environment to collect evidence for answering the question. Without access to the environment state s_t, the agent can only perform sub-optimal actions based on current observations. It needs to build its belief b_t in the state that the environment may be in, based on its experience $h_t = (o_0, a_0, o_1, \cdots, a_{t-1}, o_t)$. Therefore, the agent consists of two modules: **belief module** Φ that generates the belief state $b_t = \Phi(h_t)$ from the experience h_t, and **policy module** Π that prescribes the action $a_t = \Pi(b_t)$ to take for current belief state b_t.

Both belief and policy modules are constructed based on pretrained Transformer encoders (Clark et al., 2020), respectively denoted as Ψ^{belief} and Ψ^{policy}, which encode each inputted token into a d-dimensional contextual representation. The input of both encoders is a belief state, formatted as "`[CLS] [YES] [NO] [NONE]` question `[SEP]` title$_o$ `[SOP]` content$_o$ `[SEP]` title$_1$ `[SOP]` $\cdots$ content$_{|E|}$ `[SEP]`", where the subscript o denotes the observation passage, and the others passages come from the collected evidence set E, `[SOP]` is a special token to separate the title and con-

tent of a passage, `[YES]` and `[NO]` are used to indicate yes/no answer, and `[NONE]` is generally used to indicate that there is no desired answer/query/evidence. In this way, the self-attention mechanism across the concatenated sequence allows each passage in the input to interact with others, which has been shown crucial for multi-hop reasoning (Wang et al., 2019a).

3.2.1 Belief Module

The belief module Φ transforms the agent's experience h_t into a belief state b_t by maintaining a set of evidence E_{t-1}. At the end of the process, the evidence set E is expected to contain sufficient evidence necessary to answer the question and no irrelevant passage. In the iterative process, the agent believes that all the passages in E may help answer the question. In other words, those passages that were observed but excluded from the evidence set, i.e., $o_{1:t-1} \setminus E_{t-1}$, are believed to be irrelevant to the question.

For simplicity, assuming that the negative passages $o_{1:t-1} \setminus E_{t-1}$ and action history $a_{<t}$ are not helpful for subsequent decision-making, the experience h_t is equivalent to $\{q, o_t\} \cup E_{t-1}$. Thus, let $C_t = E_{t-1} \cup \{o_t\}$ be the current candidate evidence set, then the original question and current evidence candidates can form the belief state b_t as

$$b_t = \Pi(h_t) = \langle q, C_t \rangle = \langle q, E_{t-1} \cup \{o_t\} \rangle. \quad (1)$$

At the beginning, the belief state b_0 is initialized to $\langle q, \varnothing \rangle$, and the evidence set E_0 is initialized to $\varnothing$.

To maintain the essential evidence set E_t, we use a trainable scoring function $\phi(p|b_t)$ to identify each evidence candidate $p \in C_t$. Specifically, each passage is represented as the contextual representation of the special token `[SOP]` in it, which is encoded by Ψ^{belief}. Then, the representation of each candidate is projected into a score through a linear layer. Besides, we use a pseudo passage p_0, represented as `[None]`, to indicate the dynamic threshold of the evidence set. In this way, after step t, the evidence set is updated as

$$E_t = \{p_i | \phi(p_i|b_t) > \phi(p_0|b_t), p_i \in C_t\}. \quad (2)$$

It is worth noting that these evidence candidates are scored jointly since encoded together in the same input, different from conventional rerankers that score separately.

3.2.2 Policy Module

The policy module Π decides the next action a_t to be taken based on the current belief state b_t. In this paper, we equipped the agent with three retrieval functions and one answer function, which means that the action space $\mathcal{A}$ consists of three types of retrieval actions and one type of answer actions. However, unlike the finite space of executable functions $\mathcal{F}$, the space of function arguments $\mathcal{U}$ includes all possible natural-language queries and answers. To narrow the search space, for each executable function, we employ a suggester to propose a plausible query or answer as the argument passed to the function. Finally, we apply an action scoring function in the narrowed action space and select the action with the highest score.

Equipped Functions Formally, the space of executable functions is defined as $\mathcal{F} = \{f_s, f_d, f_l, f_o\}$.

Among them, except f_o is the answer function used to reply to the question, the rest are three distinct off-the-shelf retrieval functions (RF) used to explore the corpus. f_s is a sparse RF, implemented as BM25 (Robertson and Zaragoza, 2009). It performs well when the query is concise and contains highly selective keywords but often fails to capture the semantics of the query. f_d is a dense RF, implemented as MDR (Xiong et al., 2021) for multi-hop questions, and DPR (Karpukhin et al., 2020) for single-hop questions. Dense RFs can capture lexical variations and semantic relationships, but they struggle when encountering out-of-vocabulary words. f_l is a link RF, implemented as hyperlink. When hyperlink markups are available in a source passage, it can readily map a query (i.e., anchor text) to the target passage.

Argument Generation The space of function arguments $\mathcal{U}$, composed of textual queries and answers, is too large to perform an exhaustive search due to the complexity of natural language. To reduce the search complexity, inspired by Yao et al. (2020), we employ four argument generators to generate the most plausible query/answer for the equipped functions.

g_o is a trainable reading comprehension model for f_o. It is a span extractor built upon the contextual representations outputted by the encoder Ψ^{policy}. Like conventional extractive reading comprehension models (Yang et al., 2018; Clark et al., 2020), g_o uses the contextual representations to

calculate the start and end positions of the most plausible answer u_o. If the current context C_t is insufficient to answer the question, the special token `[NONE]` will be extreacted.

g_s is a query reformulation model for f_s. In this work, we directly employ the well-trained query reformulator from Qi et al. (2019) for multi-hop questions, which takes the belief state b_t as input and outputs a span of the input sequence as the sparse query u_s. As for single-hop questions, since there exists no off-the-shelf multi-step query reformulator, we leave g_s as an identity function that returns the original question directly. In this case, requesting the same RF multiple times is equivalent to traverse the retrieval list of original question.

g_d is a query reformulator for f_d. For multi-hop questions, g_d concatenates the question q and the passage with the highest score in evidence set E_t as the dense query u_d, the same as the input of MDR (Xiong et al., 2021). If E_t is empty, u_d is equal to the question q. Similar to g_s, g_d for single-hop questions also leaves original questions unchanged.

g_l is a trainable multi-class classifier for f_l. It selects the most promising anchor text from the belief state b_t. To enable rejecting all anchors, `[NONE]` is also treated as a candidate anchor. g_l shares the encoder Ψ^{policy}, where each anchor is represented by the average of contextual representations of its tokens. Upon Ψ^{policy}, we use a linear layer to project the hidden representations of candidate anchors to real values and select the anchor with the highest value as the link query u_l.

In this way, the action space is narrowed down to $\check{A} = \{\langle f_s, u_s \rangle, \langle f_d, u_d \rangle, \langle f_l, u_l \rangle, \langle f_o, u_o \rangle\}$.

Action Selection The action scoring function π is also built upon the output of Ψ^{policy}. To score an action $\langle f, u \rangle$ for current belief state b_t, an additional two-layer ($3d \times 4d \times 1$) MLP, with a ReLU activation in between, projects the concatenated representation of b_t, executable function f, and function argument u, i.e., $\boldsymbol{v}_{[\text{CLS}]}$, $\boldsymbol{w}_f$, and $\boldsymbol{v}_u$, into a real value. $\boldsymbol{w}_f \in \mathbb{R}^d$ is a trainable embedding for each executable function, the same dimension as the token embedding. $\boldsymbol{v}_u$ is specific for each function. Since u_s, u_l and u_o have explicit text span in the b_t, thus their $\boldsymbol{v}_u$ are the averages of their token representations. As for u_d, if g_d does not expand the original question, $\boldsymbol{v}_{u_d}$ is the contextual representation of `[NONE]`. Otherwise, $\boldsymbol{v}_{u_d}$ is the `[SOP]` of the passage concatenated to the question.

In short, the next action is selected from the narrowed action space $\check{A}$ by the scoring function π,

$$a_t = \Pi(b_t) = \arg\max_{a \in \check{A}} \pi(a|b_t). \qquad (3)$$

3.3 Training

In the agent, in addition to the encoders Ψ^{belief} and Ψ^{policy}, we need to train the evidence scoring function ϕ, link classifier g_l, answer extractor g_o, and action scoring function π, whose losses are L_ϕ, L_l, L_o, and L_π. Since the policy module is dependent on the belief module, we train the agent jointly using the following loss function,

$$L = L_\phi + L_l + L_o + L_\pi. \qquad (4)$$

Unlike ϕ, g_l and g_o that can be trained in supervised learning through human annotations in QA datasets, the supervision signal for π is hard to be derived directly from QA datasets. Even though policies are usually trained via reinforcement learning, reinforcement learning algorithms (Sutton et al., 2000; Mnih et al., 2015) are often sensitive to the quality of reward functions. For a complex task, the reward function R is often hard to specify and exhaustive to tune. Inspired by Choudhury et al. (2017), we explore the use of imitation learning (IL) by querying a model-based oracle online and imitating the action $a^\star$ chose by the oracle, which avoids the hassle of designing R and solves the POMDP in the fashion of supervised learning. Thus, the loss of π is defined as the cross entropy,

$$L_\pi = -\log \frac{e^{\pi(a^\star|b)}}{\sum_{a \in \check{A}} e^{\pi(a|b)}}, \qquad (5)$$

where b is the belief state of the agent.

The link classifier g_l and the answer extractor g_o are also optimized with multi-class cross-entropy losses. For g_l, denoting its loss as L_l, the classification label is set to the anchor text that links to a gold supporting passage, if there is no such anchor, then the pseudo hyperlink [NONE] is labeled. g_o is trained as a classifier of start and end position following previous work (Clark et al., 2020), denoting its loss as L_o. Considering the belief state $b = \langle q, \{p_1, p_2, \cdots, p_{|C|}\} \rangle$, the ListMLE (Xia et al., 2008) ranking loss of the evidence scoring function ϕ is defined as the negative log likelihood of the ground truth permutation,

$$L_\phi(\boldsymbol{y}, b) = -\log P(\tau_{\boldsymbol{y}} | \{\phi(p_i|b)\}_{i=0}^{|C|}), \qquad (6)$$

where $\boldsymbol{y}$ is the relevance label of $\{p_0, p_1, \cdots, p_{|C|}\}$ and $\tau_{\boldsymbol{y}}$ is their ground truth permutation. To learn the dynamic threshold $\phi(p_0|b)$, we set the relevance label of the pseudo passage p_0 to $\boldsymbol{y}_0 = 0.5$. And passages in C are labeled as 1/0 according to whether they are gold supporting passages.

Model-based Oracle The model-based oracle has full access to the environment and can foresee the gold evidence and answer of every question, which means that the oracle can infer the rank of a supporting passage in the retrieval list of any retrieval action. Thus, given a state, the oracle can easily select a near-optimal one from candidate actions according to a greedy policy $\pi^\star$. Specifically, if all gold evidence is collected and the argument of an answer action is a correct answer, the oracle will select the answer action. Otherwise, the oracle will use a greedy algorithm to select the retrieval action that helps to gather a missing passage of evidence in the fewest steps.

Belief States Sampling We train the agent on sampled belief states instead of long trajectories. In every epoch, one belief state is sampled for each question. To sample a belief state $\langle q, C \rangle$, we first uniformly sample a subset from q's gold evidence as C, which could be an empty set. However, at testing time, it is impossible for the candidate evidence set C to contain only gold evidence. To alleviate the mismatch of the state distribution between training and testing, we inject a few negative passages into C and shuffle them. We treat the first passage in the candidate set as the observation, and the others as evidence collected before.

The distribution of injected negative passages can affect the test performance. In this work, to make it simple, we sample 0~2 passages from all top-ranked negative passages in retrieval lists of f_s, f_d, and f_l.

4 Experiments

We evaluate AISO and baselines on two Wikipedia-sourced benchmarks. We first introduce the experimental setups, then describe the experimental results on evidence gathering and question answering. Furthermore, detailed analyses are discussed.

4.1 Experimental Setup

Data HotpotQA (Yang et al., 2018), a multi-hop QA benchmark. We focus on its fullwiki (open-

3620

domain) setting[1]. It requires gathering two supporting passages (paragraphs) to answering a question, given the introductory (first) paragraphs of 5M Wikipedia articles dumped on October 1, 2017.

SQuAD Open (Chen et al., 2017), a single-hop QA benchmark, whose questions are from the SQuAD dataset (Rajpurkar et al., 2016) and can be answered based on a single passage. We preprocess the Wikipedia dump on December 21, 2016 and extract hyperlinks using WikiExtractor[2]. Following Karpukhin et al. (2020), we split articles into some disjoint passages, resulting in 20M passages in total. We add two extra hyperlinks to each passage, one linking to its previous passage in the article, the other to the next passage.

Metrics To test whether the top-2 passages in the evidence set exactly cover both gold supporting passages, we use Supporting Passage Exact Match (P EM) as the evaluation metric following (Asai et al., 2020). To test the performance of answer extraction, we use EM and F1 as our metrics following (Yang et al., 2018).

Implementation Details For sparse retrieval, we index all passages in the corpus with Elasticsearch and implement BM25 following Qi et al. (2019)[3]. For dense retrieval, we leverage the trained passage encoder and query encoder from Karpukhin et al. (2020)[4] and Xiong et al. (2021)[5] and index all passage vectors using FAISS (Johnson et al., 2019) offline. During training, we use the HNSW-based index for efficient low-latency retrieval; in test time, we use the exact inner product search index for better retrieval results. For link retrieval, the filtered hyperlinks are used, whose targets have to be another article from this dump.

Based on Huggingface Transformers (Wolf et al., 2020), we use ELECTRA (Clark et al., 2020) ($d = 768/1024$ for base/large)[6] as the initializations for our encoders Ψ^{belief} and Ψ^{policy}. The maximum number of passages inputted into the encoders is set to 3 and the length of input tokens is limited to

---

[1]https://hotpotqa.github.io/wiki-readme.html

[2]https://github.com/attardi/wikiextractor. We do not use the processed data provided by Chen et al. (2017) because it removed the hyperlinks required by our link RF.

[3]https://github.com/qipeng/golden-retriever

[4]https://github.com/facebookresearch/DPR, the multi-set version is used

[5]https://github.com/facebookresearch/multihop_dense_retrieval

[6]Many recent approaches are based on ELECTRA, so we use ELECTRA for fair comparison.

Strategy	Method	P EM	# read
f_s	BM25	11.11	2
	BM25 + Reranker	29.60	20
f_d	DPR (Karpukhin et al., 2020)	14.18	2
$f_s \circ f_l$	Semantic Retrieval*◇	69.35	39.4
	Entity Centric IR*♡	34.90	-
$f_s \circ f_s$	GoldEn Retriever♣	47.77	10
$f_d \circ f_d$	MDR (Xiong et al., 2021)	64.52	2
	MDR + Reranker†*	81.20	≥200
	Ballen†* (Khattab et al., 2021)	86.70	-
f_s^n	CogQA* (Ding et al., 2019)	57.80	-
	DDRQA†* (Chen et al., 2017)	79.80	-
	IRRR†* (Qi et al., 2020)	84.10	≥150
$f_s \circ f_l^{n-1}$	GRR†* (Asai et al., 2020)	75.70	≥500
	HopRetriever†* (Li et al., 2021)	82.54	≥500
	HopRetriever-plus†*	86.94	>500
	TPRR†* (Xinyu et al., 2021)	86.19	≥500
$(f_s \| f_d)^n$	DrKit* (Dhingra et al., 2020)	38.30	-
$(f_s|f_d|f_l)_\Pi^n$	AISO$_{base}$	85.69	36.7
	AISO$_{large}$	**88.17**	35.7

Table 1: Evidence gathering performance and reading cost on the HotpotQA fullwiki development set. The symbol † denotes the baseline methods use the large version of pretrained language models comparable to our AISO$_{large}$. The results with ∗ are from published papers, otherwise they are our implementations. The symbol ∘ denotes sequential apply RFs, f^n denotes apply the RF f multiple times, $||$ denotes combining the results of different RFs, and $(\cdot|\cdot)_\Pi$ means choosing one of RFs to use according to the policy Π. ◇: (Nie et al., 2019), ♡: (Qi et al., 2019), ♣: (Qi et al., 2019)

512. To avoid the high confidence passages from being truncated, we input the passages of evidence in descending order of their belief scores from the previous step.

To accelerate the model training, for the first 24 epochs, Ψ^{belief} and Ψ^{policy} share parameters, for the next 6 epochs, they are trained separately. The batch size is 32. We use Adam optimization with learning rate 2×10^{-5}. To select the best agent (QA model), we first save several checkpoints that perform well on heuristic single-step metrics, such as action accuracy. Then we choose the one that performs best in the whole process on the development set. In test time, the number of interaction steps is limited to T. We set the maximum number of steps to $T = 1000$ if not specified. Once the agent has exhausted its step budget, it is forced to answer the question.

4.2 Results

Evidence Gathering We first evaluate the performance and reading cost on the evidence gathering, illustrating the effectiveness and efficiency of AISO. In Table 1, we split evidence gathering methods into different groups according to their

Method	Dev						Test					
	Ans		Sup		Joint		Ans		Sup		Joint	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
Semantic Retrieval (Nie et al., 2019)	46.5	58.8	39.9	71.5	26.6	49.2	45.3	57.3	38.7	70.8	25.1	47.6
GoldEn Retriever (Qi et al., 2019)	-	-	-	-	-	-	37.9	49.8	30.7	64.6	18.0	39.1
CogQA (Ding et al., 2019)	37.6	49.4	23.1	58.5	12.2	35.3	37.1	48.9	22.8	57.7	12.4	34.9
DDRQA† (Zhang et al., 2020)	62.9	76.9	51.3	79.1	-	-	62.5	75.9	51.0	78.9	36.0	63.9
IRRR+†* (Qi et al., 2020)	-	-	-	-	-	-	66.3	79.9	57.2	82.6	43.1	69.8
MUPPET (Feldman and El-Yaniv, 2019)	31.1	40.4	17.0	47.7	11.8	27.6	30.6	40.3	16.7	47.3	10.9	27.0
MDR† (Xiong et al., 2021)	62.3	75.1	56.5	79.4	42.1	66.3	62.3	75.3	57.5	80.9	41.8	66.6
GRR† (Asai et al., 2020)	60.5	73.3	49.2	76.1	35.8	61.4	60.0	73.0	49.1	76.4	35.4	61.2
HopRetriever† (Li et al., 2021)	62.2	75.2	52.5	78.9	37.8	64.5	60.8	73.9	53.1	79.3	38.0	63.9
HopRetriever-plus† (Li et al., 2021)	66.6	79.2	56.0	81.8	42.0	69.0	64.8	77.8	56.1	81.8	41.0	67.8
EBS-Large*	-	-	-	-	-	-	66.2	79.3	57.3	84.0	42.0	70.0
TPRR†* (Xinyu et al., 2021)	67.3	80.1	60.2	84.5	45.3	71.4	67.0	79.5	59.4	84.3	44.4	70.8
AISO$_{base}$	63.5	76.5	55.1	81.9	40.2	66.9	-	-	-	-	-	-
AISO$_{large}$	**68.1**	**80.9**	**61.5**	**86.5**	**45.9**	**72.5**	67.5	80.5	61.2	86.0	44.9	72.0

Table 2: Answer extraction and supporting sentence identification performance on HotpotQA fullwiki. The methods with † use the large version of pretrained language models comparable to AISO$_{large}$. The results marked with * are from the official leaderboard otherwise originated from published papers.

Method	EM	F1	# read
DrQA (Chen et al., 2017)	27.1	-	5
Multi-passage BERT (Wang et al., 2019b)	53.0	60.9	100
DPR (Karpukhin et al., 2020)	29.8	-	100
BM25+DPR (Karpukhin et al., 2020)	36.7	-	100
Multi-step Reasoner (Das et al., 2019a)	31.9	39.2	5
MUPPET (Feldman and El-Yaniv, 2019)	39.3	46.2	45
GRR† (Asai et al., 2020)	56.5	63.8	≥ 500
SPARTA† (Zhao et al., 2021)	59.3	66.5	-
IRRR† (Qi et al., 2020)	56.8	63.2	≥ 150
AISO$_{large}$	**59.5**	**67.6**	24.8

Table 3: Question answering performance on SQuAD Open benchmark. † denotes the methods use the large pretrained language models comparable to AISO$_{large}$.

strategies. Moreover, the first three groups are the traditional pipeline approaches, and the others are iterative approaches.

For effectiveness, we can conclude that 1) almost all the iterative approaches perform better than the pipeline methods, 2) the proposed adaptive information-seeking approach AISO$_{large}$ outperforms all previous methods and achieves the state-of-the-art performance. Moreover, our AISO$_{base}$ model outperforms some baselines that use the large version of pretrained language models, such as HopRetriever, GRR, IRRR, DDRQA, and MDR.

For efficiency, the cost of answering an open-domain question includes the retrieval cost and reading cost. Since the cost of reading a passage along with the question online is much greater than the cost of a search, the total cost is linear in # read, reported in the last column of Table 1. # read means

the total number of passages read along with the question throughout the process, which is equal to the adaptive number of steps. We can find that the number of read passages in AISO model, i.e., the is about 35, which is extremely small than the competitive baselines (P EM > 80) that need to read at least 150 passages. That is to say, our AISO model is efficient in practice.

Question Answering Benefit from high-performance evidence gathering, as shown in Tables 2 and 3, AISO outperforms all existing methods across the evaluation metrics on the HotpotQA fullwiki and SQuAD Open benchmarks. This demonstrates that AISO is applicable to both multi-hop questions and single-hop questions. Notably, on the HotpotQA fullwiki blind test set[7], AISO$_{large}$ significantly outperforms the second place TPRR (Xinyu et al., 2021) by 2.02% in Sup F1 (supporting sentence identification) and 1.69% on Joint F1.

4.3 Analysis

We conduct detailed analysis of AISO$_{base}$ on the HotpotQA fullwiki development set.

The effect of the belief and policy module As shown in the second part of Table 4, we examine the variations of AISO with the oracle evidence scoring function $\phi^\star$ or oracle action scoring function $\pi^\star$, which are key components of the belief

---

[7]https://hotpotqa.github.io. As of September 2021, AISO is still at the top of the fullwiki leaderboard.

Model	P EM	Ans F1	# read
$AISO_{base}$	85.69	76.45	36.64
w. $\phi^\star$	97.52	79.99	40.01
w. $\phi^\star + \pi^\star$	98.88	80.34	8.92
f_s^t	68.51	67.33	58.74
f_d^t	79.80	72.91	68.63
$(f_d|f_l)_\Pi^n$	83.97	74.93	61.41
$(f_s|f_l)_\Pi^n$	82.44	74.44	37.76
$(f_s|f_d)_\Pi^n$	79.66	73.36	42.01

Table 4: Analysis experiments on HotpotQA fullwiki.

Figure 3: Performance and cost of $AISO_{base}$ on the HotpotQA development set with different step limits.

and policy module. When we replace our learned evidence scoring function with $\phi^\star$ that can identify supporting passage perfectly, the performance increase a lot while the reading cost do not change much. This means that the belief module has a more impact on the performance than the cost. If we further replace the learned π with $\pi^\star$, the cost decreases a lot. This shows that a good policy can greatly improve the efficiency.

The impact of retrieval functions As shown in the last part Table 4, the use of a single RF, such as f_s^t and f_d^t, leads to poor performance and low efficiency. Moreover, lack of any RF will degrade performance, which illustrates that all RFs contribute to performance. Specifically, although the link RF f_l cannot be used alone, it contributes the most to performance and efficiency. Besides, the sparse RF f_s may be better at shortening the information-seeking process than the dense RF f_d, since removing f_s from the action space leads to the number of read passages increase from 36.64 to 61.41. We conjecture this is because f_s can rank the evidence that matches the salient query very high.

The impact of the maximum number of steps As shown in Figure 3, with the relaxation of the step limit T, $AISO_{base}$ can filter out negative passages and finally observe low-ranked evidence through more steps, so its performance improves and tends to converge. However, the cost is more paragraphs to read. Besides, once T exceeds 1000, only a few questions (about 1%) can benefit from the subsequent steps.

The ability to recover from mistakes We count three types of mistakes in gathering evidence on the HotpotQA development set. In the process of collecting evidence for 7405 questions, false evidence was added into the evidence set for 1061 questions, true evidence was missed for 449 questions, and
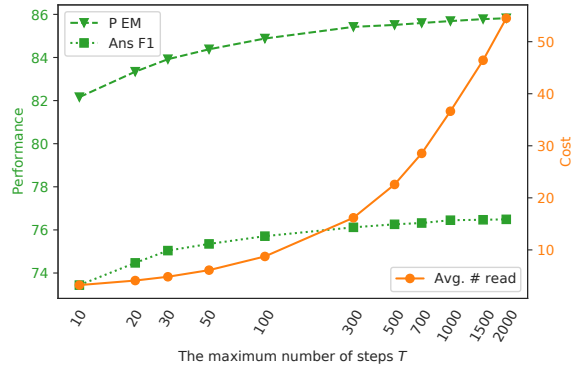
true evidence was deleted from the evidence set for 131 questions. And we find that AISO recovered from 17.7%, 43.9%, and 35.9% of these three types of errors respectively, which implies that even without beam search, $AISO_{base}$ can make up for previous mistakes to some extent. Besides, we can see that false evidence is the most harmful to evidence gathering and the most difficult to remedy.

5 Conclusion and Future Work

This work presents an adaptive information-seeking approach for open-domain question answering, called AISO. It models the open-domain QA task as a POMDP, where the environment contains a large corpus and the agent is asked to sequentially select retrieval function and reformulate query to collect the evidence. AISO achieves state-of-the-art results on two public datasets, which demonstrates the necessity of different retrieval functions for different questions. In the future, we will explore other adaptive retrieval strategies, like directly optimizing various information-seeking metrics by using reinforcement learning techniques.

Ethical Considerations

We honor and support the ACL code of Ethics. The paper focuses on information seeking and question answering tasks, which aims to answer the question in the open-domain setting. It can be widely used in search engine and QA system, and can help people find the information more accuracy and efficiency. Simultaneously, the datasets we used in this paper are all from previously published works and do not involve privacy or ethical issues.

Acknowledgements

References

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Sanjiban Choudhury, Ashish Kapoor, Gireeja Ranade, Sebastian A. Scherer, and Debadeepta Dey. 2017. Adaptive information gathering via imitation learning. In *Robotics: Science and Systems 2017*, volume 13.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, and Andrew McCallum. 2019a. Multi-step retriever-reader interaction for scalable open-domain question answering. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Rajarshi Das, Ameya Godbole, Dilip Kavarthapu, Zhiyu Gong, Abhishek Singhal, Mo Yu, Xiaoxiao Guo, Tian Gao, Hamed Zamani, Manzil Zaheer, and Andrew McCallum. 2019b. Multi-step entity-centric information retrieval for multi-hop question answering. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 113–118, Hong Kong, China. Association for Computational Linguistics.

Bhuwan Dhingra, Manzil Zaheer, Vidhisha Balachandran, Graham Neubig, Ruslan Salakhutdinov, and William W. Cohen. 2020. Differentiable reasoning over a virtual knowledge base. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. 2019. Cognitive graph for multi-hop reading comprehension at scale. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2694–2703, Florence, Italy. Association for Computational Linguistics.

Yair Feldman and Ran El-Yaniv. 2019. Multi-hop paragraph retrieval for open-domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2296–2309, Florence, Italy. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

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

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Omar Khattab, Christopher Potts, and Matei Zaharia. 2021. Baleen: Robust multi-hop reasoning at scale via condensed retrieval. *arXiv preprint arXiv:2101.00436*.

Jinhyuk Lee, Seongjun Yun, Hyunjae Kim, Miyoung Ko, and Jaewoo Kang. 2018. Ranking paragraphs for improving answer recall in open-domain question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 565–569, Brussels, Belgium. Association for Computational Linguistics.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Shaobo Li, Xiaoguang Li, Lifeng Shang, Xin Jiang, Qun Liu, Chengjie Sun, Zhenzhou Ji, and Bingquan Liu. 2021. Hopretriever: Retrieve hops over wikipedia

to answer complex questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13279–13287.

Yankai Lin, Haozhe Ji, Zhiyuan Liu, and Maosong Sun. 2018. Denoising distantly supervised open-domain question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1736–1745, Melbourne, Australia. Association for Computational Linguistics.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533.

Yixin Nie, Songhe Wang, and Mohit Bansal. 2019. 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)*, pages 2553–2566, Hong Kong, China. Association for Computational Linguistics.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Lixin Su, and Xueqi Cheng. 2019. Has-qa: Hierarchical answer spans model for open-domain question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6875–6882.

Peng Qi, Haejun Lee, Oghenetegiri Sido, Christopher D Manning, et al. 2020. Retrieve, rerank, read, then iterate: Answering open-domain questions of arbitrary complexity from text. *arXiv preprint arXiv:2010.12527*.

Peng Qi, Xiaowen Lin, Leo Mehr, Zijian Wang, and Christopher D. Manning. 2019. 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)*, pages 2590–2602, Hong Kong, China. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063.

Ellen M Voorhees et al. 1999. The trec-8 question answering track report. In *Trec*, volume 99, pages 77–82. Citeseer.

Haoyu Wang, Mo Yu, Xiaoxiao Guo, Rajarshi Das, Wenhan Xiong, and Tian Gao. 2019a. Do multi-hop readers dream of reasoning chains? In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 91–97, Hong Kong, China. Association for Computational Linguistics.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R 3: Reinforced ranker-reader for open-domain question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019b. 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)*, pages 5878–5882, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. 2008. Listwise approach to learning to rank: theory and algorithm. In *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, volume 307 of *ACM International Conference Proceeding Series*, pages 1192–1199. ACM.

Zhang Xinyu, Zhan Ke, Hu Enrui, Fu Chengzhen, Luo Lan, Jiang Hao, Jia Yantao, Yu Fan, Dou Zhicheng, Cao Zhao, and Chen Lei. 2021. Answer complex questions: Path ranker is all you need. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, New York, NY, USA. Association for Computing Machinery.

Wenhan Xiong, Xiang Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Scott Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oguz. 2021. Answering complex open-domain questions with multi-hop dense retrieval. In *International Conference on Learning Representations*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Shunyu Yao, Rohan Rao, Matthew Hausknecht, and Karthik Narasimhan. 2020. Keep CALM and explore: Language models for action generation in text-based games. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8736–8754, Online. Association for Computational Linguistics.

Yuyu Zhang, Ping Nie, Arun Ramamurthy, and Le Song. 2020. Ddrqa: Dynamic document reranking for open-domain multi-hop question answering. *arXiv preprint arXiv:2009.07465*.

Chen Zhao, Chenyan Xiong, Corby Rosset, Xia Song, Paul N. Bennett, and Saurabh Tiwary. 2020. Transformer-xh: Multi-evidence reasoning with extra hop attention. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Tiancheng Zhao, Xiaopeng Lu, and Kyusong Lee. 2021. SPARTA: Efficient open-domain question answering via sparse transformer matching retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 565–575, Online. Association for Computational Linguistics.