LEARNING TO ATTEND ON ESSENTIAL TERMS: AN ENHANCED RETRIEVER-READER MODEL FOR OPEN-DOMAIN QUESTION ANSWERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Open-domain question answering remains a challenging task as it requires models that are capable of understanding questions and answers, collecting useful information, and reasoning over evidence. Previous work typically formulates this task as a reading comprehension or entailment problem given evidence retrieved from search engines. However, existing techniques struggle to retrieve indirectly related evidence when no directly related evidence is provided, especially for complex questions where it is hard to parse precisely what the question asks. In this paper we propose a retriever-reader model that learns to attend on essential terms during the question answering process. We build (1) an essential term selector which first identifies the most important words in a question, then reformulates the query and searches for related evidence; and (2) an enhanced reader that distinguishes between essential terms and distracting words to predict the answer. We evaluate our model on multiple open-domain QA datasets where it outperforms the existing state-of-the-art, notably leading to a relative improvement of 8.1% on the AI2 Reasoning Challenge (ARC) dataset.

1 INTRODUCTION

Open-domain question answering (QA) has been extensively studied in recent years. Many existing works have followed the 'search-and-answer' strategy and achieved strong performance (Chen et al., 2017; Kwon et al., 2018; Wang et al., 2018) spanning multiple QA datasets such as TriviaQA (Joshi et al., 2017), SQuAD (Rajpurkar et al., 2016), MS-Macro (Nguyen et al., 2016), among others.

However, open-domain QA tasks become inherently more difficult when (1) dealing with questions with little available evidence; (2) solving questions where the answer type is free-form text (e.g. multiple-choice) rather than a span among existing passages (i.e., 'answer span'); or when (3) the need arises to understand long and complex questions and reason over multiple passages, rather than simple text matching. As a result, it is essential to incorporate commonsense knowledge or to improve retrieval capability to better capture partially related evidence (Chen et al., 2017).

As shown in Table 1, the TriviaQA, SQuAD, and MS-Macro datasets all provide passages within which the correct answer is guaranteed to exist. However, this assumption ignores the difficulty of retrieving question-related evidence from a large volume of open-domain resources, especially when considering complex questions which require reasoning or commonsense knowledge. On the other hand, ARC does not provide passages known to contain the correct answer. Instead, the task of identifying relevant passages is left to the solver. However, questions in ARC have multiple answer choices that provide indirect information that can help solve the question. As such an effective model needs to account for relations among passages, questions, and answer choices.

Figure 1 shows an example of a question in the ARC dataset and demonstrates the difficulties in retrieval and reading comprehension. As shown for Choice 1 (C1), a simple concatenation of the question and the answer choice is not a reliable query and is of little help when trying to find supporting evidence to answer the question (e.g. we might retrieve sentences similar to the question or the answer choice, but would struggle to find evidence explaining *why* the answer choice is correct). On the other hand, a reformulated query consisting of essential terms in the question and

Table 1: Differences among popular QA datasets.

Dataset	Open-domain	Multiple choice	Passage retrieval	No ranking supervision[1]
ARC (Clark et al., 2018)	✓	✓	✓	✓
SQuAD (Rajpurkar et al., 2016)	✓			
TriviaQA (Joshi et al., 2017)	✓			
MS-Macro (Nguyen et al., 2016)	✓			

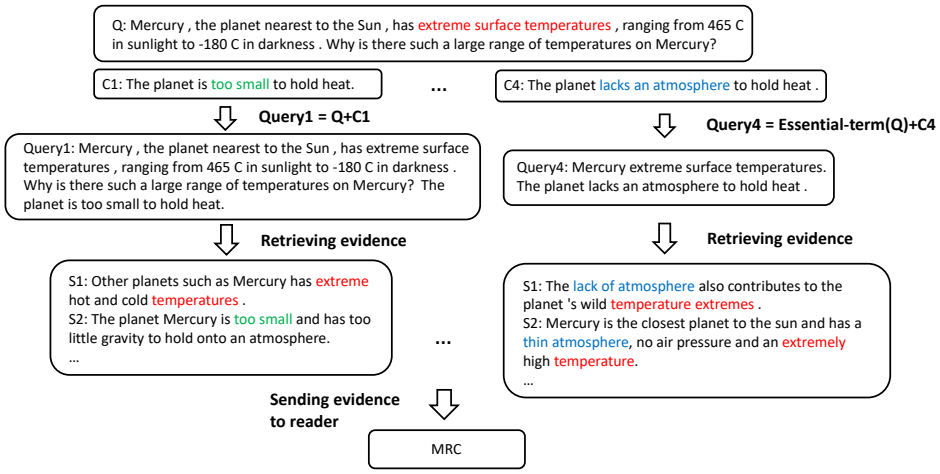

Figure 1: Example of the retrieve-and-read process to solve open-domain questions. Words related with the question, C1, and C4 are marked in red, green, and blue, respectively.

Choice 4 can help retrieve evidence explaining why Choice 4 is a correct answer. To achieve this, the model needs to (1) ensure that the retrieved evidence supports the fact mentioned in both the question and the answer choices and (2) capture this information and predict the correct answer.

To address these difficulties, we propose an essential-term-aware Retriever-Reader (ET-RR) model that learns to attend on essential terms during retrieval and reading. Specifically, we develop a two-stage method with an essential term selector followed by an attention-enhanced reader.

Essential term selector. ET-Net is a recurrent neural network that seeks to understand the question and select essential terms, i.e., key words, from the question. We frame this problem as a classification task for each word in the question. These essential terms are then concatenated with each answer choice and fed into a retrieval engine to obtain related evidence.

Attention-Enhanced Reader. Our neural reader takes the triples (question, answer choice, retrieved passage) as input. The reader consists of a sequence of language understanding layers: an input layer, attention layer, sequence modeling layer, fusion layer, and an output layer. The attention and fusion layers help the model to obtain a refined representation of one text sequence based on the understanding of another, e.g. a passage representation based on an understanding of the question. We further add a choice-interaction module to handle the semantic relations and differences between answer choices. Experiments show that this can further improve the model's accuracy.

We evaluate our model on the ARC dataset, where our model achieves an accuracy of 36.61% on the test set, thus ranking first on the official leaderboard. We also adapt two datasets to the open-domain setting, RACE-Open and MCScript-Open, where we outperform baseline models by a large margin. Ablation studies show that each of our model's components contributes to its accuracy.

[1]For SQuAD and TriviaQA, since their questions are paired with span-type answers, it is convenient to obtain ranking supervision where retrieved passages are relevant via distant supervision; however free-form questions in ARC result in a lack of supervision which makes the problem more difficult. For MS-Macro, the dataset is designed to annotate relevant passages though it has free-form answers.

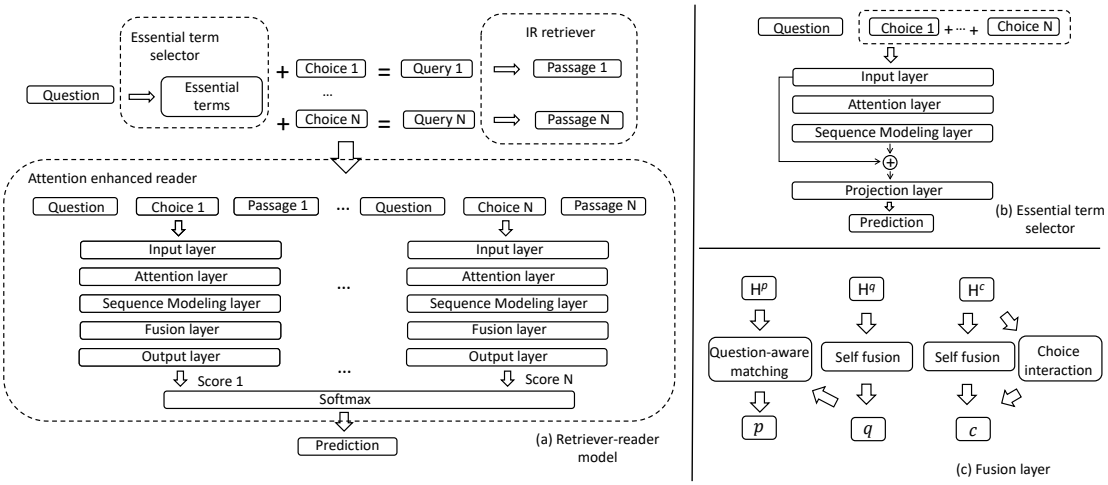

Figure 2: Model structure for our essential-term-aware retriever-reader model.

2 Related Work

There has recently been growing interest in building better retrievers for open-domain QA. Wang et al. (2018) proposed a Reinforced Ranker-Reader model that ranks retrieved evidence and assigns different weights to evidence prior to processing by the reader. Min et al. (2018) demonstrated that for several popular MRC datasets (e.g. SQuAD, TriviaQA) most questions can be answered using only a few sentences rather than the entire document. Motivated by this observation, they built a sentence selector to gather this potential evidence for use by the reader model. Nishida et al. (2018) developed a multi-task learning (MTL) method for a retriever and reader in order to obtain a strong retriever that considers certain passages including the answer text as positive samples during training. The proposed MTL framework is still limited to the scenario when it is feasible to discover whether the passages contain the answer span. Although these works have achieved progress on open-domain QA by improving the ranking or selection of given evidence, few have focused on the scenario where the model needs to start by searching for the evidence itself.

Scientific Question Answering (SQA) is a representative open-domain task that requires capability in both retrieval and reading comprehension. In this paper, we study question answering on the AI2 Reasoning Challenge (ARC) scientific QA dataset (Clark et al., 2018). This dataset contains elementary-level multiple-choice scientific questions from standardized tests and a large corpus of relevant information gathered from search engines. The dataset is partitioned into "Challenge" and "Easy" sets. The challenge set consists of questions that cannot be answered correctly by any of the solvers based on Pointwise Mutual Information (PMI) or Information Retrieval (IR). Existing models tend to achieve only slightly better and sometimes even worse performance than random guessing, which demonstrates that existing models are not well suited to this kind of QA task.

Khashabi et al. (2017) worked on the problem of finding essential terms in a question for solving SQA problems. They handcrafted over 100 features and used an SVM classifier to uncover essential terms within a question. They also published a dataset containing over 2,200 science questions annotated with essential terms. We leverage this dataset to build an essential term selector.

More recently, Boratko et al. (2018) developed a labeling interface to obtain high quality labels for the ARC dataset. One interesting finding is that human annotators tend to retrieve better evidence after they reformulate the search queries which are originally constructed by a simple concatenation of question and answer choice. By feeding the evidence obtained by human-reformulated queries into a pre-trained MRC model (i.e. DrQA (Chen et al., 2017)) they achieved an accuracy increase of 42% on a subset of 47 questions. This shows the potential for a "human-like" retriever to boost performance on this task. Inspired by this work, we focus on selecting essential terms to reformulate more efficient queries, similar to those that a human would construct.

3 APPROACH

In this section, we introduce the essential-term-aware retriever-reader model (ET-RR). As shown in Figure 2, we build a term selector to discover which terms are essential in a question. The selected terms are then used to formulate a more efficient query enabling the retriever to obtain related evidence. The retrieved evidence is then fed to the reader to predict the final answer.

For a question with q words $\mathbf{Q} = \{w_t^Q\}_{t=1}^q$ along with its N answer choices $\mathbf{C} = \{\mathbf{C}_n\}_{n=1}^N$ where $\mathbf{C}_n = \{w_t^C\}_{t=1}^c$, the essential-term selector chooses a subset of essential terms $\mathbf{E} \subset \mathbf{Q}$, which are then concatenated with each $\mathbf{C}_n$ to formulate a query. The query for each answer choice, $\mathbf{E} + \mathbf{C}_n$, is sent to the retriever (e.g. Elastic Search[2]), and the top K retrieved sentences based on the scores returned by the retriever are then concatenated into the evidence passage $\mathbf{P}_n = \{w_t^P\}_{t=1}^p$.

Next, given these text sequences $\mathbf{Q}$, $\mathbf{C}$, and $\mathbf{P} = \{\mathbf{P}_n\}_{n=1}^N$, the reader will determine a matching score for each triple $\{\mathbf{Q}, \mathbf{C}_n, \mathbf{P}_n\}$. The answer choice $\mathbf{C}_{n^*}$ with the highest score is selected.

We first introduce the reader model in Section 3.1 and then the essential term selector in Section 3.2.

3.1 READER MODEL

3.1.1 INPUT LAYER

To simplify notation, we ignore the subscript n denoting the answer choice until the final output layer. In the input layer, all text inputs—the question, answer choices, and passages, i.e., retrieved evidence—are converted into embedded representations. Similar to Wang (2018), we consider the following components for each word:

Word Embedding. Pre-trained GloVe word embedding with dimensionality $d_w = 300$.

Part-of-Speech Embedding and Named-Entity Embedding. The part-of-speech tags and named entities for each word are mapped to embeddings with dimension 16.

Relation Embedding. A relation between each word in $\mathbf{P}$ and any word in $\mathbf{Q}$ or $\mathbf{C}$ is mapped to an embedding with dimension 10. In the case that multiple relations exist, we select one uniformly at random. The relation is obtained by querying ConceptNet (Speer et al., 2017).

Feature Embeddings. Three handcrafted features are used to enhance the word representations:

1. Word Match. If a word or its lemma of $\mathbf{P}$ exists in $\mathbf{Q}$ or $\mathbf{C}$, then this feature is 1 (0 otherwise).
2. Word Frequency. A logarithmic term frequency is calculated for each word.
3. Essential Term. For the i-th word in $\mathbf{Q}$, this feature, denoted as w_{e_i}, is 1 if the word is an essential term (0 otherwise). Let $\mathbf{w}_e = [w_{e_1}, w_{e_2}, ..., w_{e_q}]$ denote the essential term vector.

For $\mathbf{Q}, \mathbf{C}, \mathbf{P}$, all of these components are concatenated to obtain the final word representations $\mathbf{W}_Q \in \mathbb{R}^{q \times d_Q}, \mathbf{W}_C \in \mathbb{R}^{c \times d_C}, \mathbf{W}_P \in \mathbb{R}^{p \times d_P}$, where d_Q, d_C, d_P are the final word dimensions of $\mathbf{Q}, \mathbf{C}$, and $\mathbf{P}$.

3.1.2 ATTENTION LAYER

As shown in Figure 2, after obtaining word-level embeddings, attention is added to enhance word representations. Given two word embedding sequences $\mathbf{W}_U, \mathbf{W}_V$, word-level attention is calculated as:

$$\mathbf{M}'_{UV} = \mathbf{W}_U\mathbf{U} \cdot (\mathbf{W}_V\mathbf{V})^\top; \quad \mathbf{M}_{UV} = \text{softmax}(\mathbf{M}'_{UV}); \quad \mathbf{W}_U^V = \mathbf{M}_{UV} \cdot (\mathbf{W}_V\mathbf{V}), \quad (1)$$

where $\mathbf{U} \in \mathbb{R}^{d_U \times d_w}$ and $\mathbf{V} \in \mathbb{R}^{d_V \times d_w}$ are two matrices that convert word embedding sequences to dimension d_w, and $\mathbf{M}'_{UV}$ contains dot products between each word in $\mathbf{W}_U$ and $\mathbf{W}_V$, and softmax is applied on $\mathbf{M}'_{UV}$ row-wise.

[2]https://www.elastic.co/products/elasticsearch

Three types of attention are calculated using Equation (1): (1) question-aware passage representation $\mathbf{W}_P^Q \in \mathbb{R}^{p \times d_w}$, (2) question-aware choice representation $\mathbf{W}_C^Q \in \mathbb{R}^{c \times d_w}$, and (3) passage-aware choice representation $\mathbf{W}_C^P \in \mathbb{R}^{c \times d_W}$.

3.1.3 SEQUENCE MODELING LAYER

To model the contextual dependency of each text sequence, we use BiLSTMs to process the word representations obtained from the input layer and attention layer:

$$\mathbf{H}^q = \mathbf{BiLSTM}[\mathbf{W}_Q]; \quad \mathbf{H}^c = \mathbf{BiLSTM}[\mathbf{W}_C; \mathbf{W}_C^P; \mathbf{W}_C^Q]; \quad \mathbf{H}^p = \mathbf{BiLSTM}[\mathbf{W}_P; \mathbf{W}_P^Q], \tag{2}$$

where $\mathbf{H}^q \in \mathbb{R}^{q \times l}, \mathbf{H}^c \in \mathbb{R}^{c \times l}$, and $\mathbf{H}^p \in \mathbb{R}^{p \times l}$ are the hidden states of the BiLSTMs, ';' is feature-wise concatenation, and l is the size of the hidden states.

3.1.4 FUSION LAYER

We further convert each question and answer choice into a single vector: $\mathbf{q} \in \mathbb{R}^l$ and $\mathbf{c} \in \mathbb{R}^l$:

$$\boldsymbol{\alpha}_q = \mathrm{softmax}([\mathbf{H}^q; \mathbf{w}_e] \cdot \mathbf{w}_{sq}^\top); \quad \mathbf{q} = {\mathbf{H}^q}^\top \boldsymbol{\alpha}_q; \quad \boldsymbol{\alpha}_c = \mathrm{softmax}(\mathbf{H}^c \cdot \mathbf{w}_{sc}^\top); \quad \mathbf{c} = {\mathbf{H}^c}^\top \boldsymbol{\alpha}_c, \tag{3}$$

where the essential-term feature $\mathbf{w}_e$ from Section 3.1.1 is concatenated with $\mathbf{H}^q$, and $\mathbf{w}_{sq}$ and $\mathbf{w}_{sc}$ are learned parameters.

Finally, a bilinear sequence matching is calculated between $\mathbf{H}^p$ and $\mathbf{q}$ to obtain a question-aware passage representation, which is used as the final passage representation:

$$\boldsymbol{\alpha}_p = \mathrm{softmax}(\mathbf{H}^p \cdot \mathbf{q}); \quad \mathbf{p} = {\mathbf{H}^p}^\top \boldsymbol{\alpha}_p. \tag{4}$$

3.1.5 CHOICE INTERACTION

When a QA task provides multiple choices for selection, the relationship between the choices can provide useful information to answer the question. Therefore, we integrate a choice interaction layer to handle the semantic correlation between multiple answer choices. Given the hidden state $\mathbf{H}^{c_n}$ of choice c_n and $\mathbf{H}^{c_i}$ of other choices c_i, $\forall i \neq n$, we calculate the differences between the hidden states and apply max-pooling over the differences:

$$\mathbf{c}_{inter} = \mathrm{Maxpool}(\mathbf{H}^{c_n} - \frac{1}{N-1} \sum_{i \neq n} \mathbf{H}^{c_i}), \tag{5}$$

where N is the total number of answer choices. Here, $\mathbf{c}_{inter}$ characterizes the differences between an answer choice c_n and other answer choices. The final representation of an answer choice is updated by concatenating the self-attentive answer choice vector and inter-choice representation as

$$\mathbf{c}^{final} = [\mathbf{c}; \mathbf{c}_{inter}]. \tag{6}$$

3.1.6 OUTPUT LAYER

For each tuple $\{\mathbf{q}, \mathbf{p}_n, \mathbf{c}_n\}_{n=1}^N$, two scores are calculated by matching (1) the passage and answer choice and (2) question and answer choice. We use the bilinear form for both matchings. Finally, a softmax function is applied over N answer choices to determine the best answer choice:

$$s_n^{pc} = \mathbf{p}_n \mathbf{W}^{pc} \mathbf{c}_n^{final}; \quad s_n^{qc} = \mathbf{q} \mathbf{W}^{qc} \mathbf{c}_n^{final}; \quad \mathbf{s} = \mathrm{softmax}(\mathbf{s}^{pc}) + \mathrm{softmax}(\mathbf{s}^{qc}), \tag{7}$$

where s_n^{pc}, s_n^{qc} are the scores for answer choice $1 \leq n \leq N$; $\mathbf{s}^{pc}, \mathbf{s}^{qc}$ are score vectors for all N choices; and $\mathbf{s}$ contains the final scores for each answer choice. During training, we use a cross-entropy loss.

3.2 ESSENTIAL TERM SELECTOR

Essential terms are key words in a question that are crucial in helping a retriever obtain related evidence. Given a question $\mathbf{Q}$ and N answer choices $\mathbf{C}_1, \ldots, \mathbf{C}_N$, the goal is to predict a binary variable y_i for each word Q_i in the question $\mathbf{Q}$, where $y_i = 1$ if Q_i is an essential term and 0

Table 2: Example of essential term data.

Question	If an object is attracted to a magnet, the object is most likely made of (A) wood (B) plastic (C) cardboard (D) metal
# annotators	5
Annotation	If,0; an,0; object,3; is,0; attracted,5; to,0; a,0; magnet,,5; the,0; object,1; is,0; most,0; likely,0; made,2; of,0

Table 3: Precison, recall and F1 scores of different selectors.

Model	Precision	Recall	F1
MaxPMI	0.88	0.65	0.75
SumPMI	0.88	0.65	0.75
PropSurf	0.68	0.64	0.66
PropLem	0.76	0.64	0.69
ET Classifier	0.91	0.71	0.80
ET-Net	0.74	**0.90**	**0.81**

otherwise. To address this problem, we build a neural model, ET-Net, which has the same design as the reader model for the input layer, attention layer, and sequence modeling layer to obtain the hidden state $\mathbf{H}^q$ for question $\mathbf{Q}$.

In detail, we take the question $\mathbf{Q}$ and the concatenation $\mathbf{C}$ of all N answer choices as input to ET-Net. $\mathbf{Q}$ and $\mathbf{C}$ first go through an input layer to convert to the embedded word representation, and then word-level attention is calculated to obtain a choice-aware question representation $\mathbf{W}_Q^C$ as in Equation (1). We concatenate the word representation and word-level attention representation of the question and feed it into the sequence modeling layer:

$$\mathbf{H}^q = \mathbf{BiLSTM}[\mathbf{W}_Q; \mathbf{W}_Q^C]. \tag{8}$$

As shown in Figure 2, the hidden states obtained from the attention layer are then concatenated with the embedded representations of $\mathbf{Q}$ and fed into a projection layer to obtain the prediction vector $\mathbf{y} \in \mathbb{R}^q$ for all words in the question:

$$\mathbf{y} = [\mathbf{H}^q; \mathbf{W}_Q^f] \cdot \mathbf{w}^s, \tag{9}$$

where $\mathbf{w}^s$ contains the learned parameters, and $\mathbf{W}_Q^f$ is the concatenation of the POS embedding, NER embedding, relation embedding, and feature embedding from Section 3.1.1.

After obtaining the prediction for each word, we use a binary cross-entropy loss to train the model. During evaluation, we take words with y_i greater than 0.5 as essential terms.

4 Experiments

In this section, we first discuss the performance of the essential term selector, ET-Net, on a public dataset. We then discuss the performance of the whole retriever-reader pipeline, ET-RR, on the ARC, RACE-Open and MCScript-Open datasets. For both the ET-Net and ET-RR models, we use 96-dimensional hidden states and 1-layer BiLSTMs in the sequence modeling layer. A dropout rate of 0.4 is applied for the embedding layer and the BiLSTMs' output layer. We use adamax (Kingma & Ba, 2014) with a learning rate of 0.02 and batch size of 32. The model is run for 100 epochs.

4.1 Performance on Essential Term Selection

We use the public dataset from Khashabi et al. (2017) which contains 2,223 annotated questions, each accompanied by four answer choices. Table 2 gives an example of an annotated question. As shown, the dataset is annotated for binary classification. For each word in the question, the data measures whether the word is an "essential" term according to 5 annotators. We then split the dataset into training, development, and test sets using an 8:1:1 ratio and select the the model that performs best on the development set.

Table 3 shows the performance of our essential term selector and baseline models from Khashabi et al. (2017). MAXPMI and SUMPMI score the importance of a word w by taking the max or sum of its PMI $p(w, c)$ scores for all answer choices c. PROPSURF and PROPLEM are baselines that consider a word as an essential term if it or its lemmatized word appears at least a certain proportion of times as essential in the dataset. ET Classifier is an SVM-based model from Khashabi et al.

Table 4: Examples of essential term prediction (in questions) by ET-Net. True positives are marked in red while false positives are marked in blue.

Example questions
Which unit of measurement can be used to describe the length of a desk ?
One way animals usually respond to a sudden drop in temperature is by
Organisms require energy to survive. Which of the following processes provides energy to the body ?

Table 5: Statistics on ARC and RACE-Open. Corpus size is the number of sentences.

Dataset	ARC	RACE-Open	MCScript-Open
Train	1,119	9,531	1,036
Dev	299	473	156
Test	1,172	528	319
Corpus	1.46M	0.52M	24.2K

Table 6: Ablation test on attention components of ET-RR on ARC. '–' denotes the ablated feature.

Model	Test
ET-RR	**36.61**
– inter-choice	36.36
– passage-choice	35.41
– question-choice	34.47
– passage-question	34.05

(2017) requiring over 100 handcrafted features. As shown, our ET-Net achieves a comparable result with the ET Classifier in terms of F1 Score.

Table 4 shows example predictions made by ET-Net. As shown, ET-Net is capable of selecting most ground-truth essential terms. It rejects certain words such as "organisms" which have a high TF-IDF in the corpus but are not relevant to answering a particular question. This shows its ability to discover essential terms according to the context of the question.

4.2 Performance on Open-domain Multiple-choice QA

With the trained essential term selector (ET-Net) from the previous experiment, we train and evaluate the reader model on three open-domain multiple-choice QA datasets. All datasets are associated with a sentence-level corpus. In the experiments, ET-RR generates a query for each of the N answer choices. For each query, ET-RR then obtains the top K sentences returned by the retriever and considers these sentences as a passage for the reader. We set $K = 10$ for all experiments and report results for different K in the ablation test. Detailed statistics are shown in Table 5.

- ARC (Clark et al., 2018): We consider the 'Challenge' set in the ARC dataset and use the provided corpus during retrieval.
- RACE-Open: We adapted the RACE dataset (Lai et al., 2017) to the open-domain setting. Originally, each question in RACE comes with a specific passage. To enable passage retrieval, we concatenate all passages into a corpus with sentence deduplication.[3]
- MCScript-Open: The MCScript (Ostermann et al., 2018) dataset is also adapted to the open-domain setting. Again we concatenate all passages to build the corpus.[4]

We compare ET-RR against existing retrieval-reader methods on both datasets. Accuracy is shown in Table 7. Results for ARC are obtained from the official leaderboard.[7] On the ARC dataset, ET-RR outperforms all previous models with a relative 8.1% improvement over the state-of-the-art BiLSTM Max-out method. On the RACE-Open and MCScript-Open datasets, ET-RR achieves a relative improvement of 24.6% and 10.5% on the test set compared with the IR solver respectively.

[3]As short questions are usually passage-specific and retrieval can rarely find any related passage, we only keep questions with more than 15 words.

[4]We keep questions with more than 10 words rather than 15 words to ensure that there is sufficient data.

[5]IR solver sends question plus each answer choice as query to the search engine, then pick the answer choice of which the top retrieved sentence has the highest score as the answer

[6]Different from ET-RR, in the original BiDAF baseline, the sentences returned by each query are mixed together, then the top $N \times K$ sentences are aggregated as a whole passage and passed to the reader.

[7]Snapshot from `http://data.allenai.org/arc/` on September 26, 2018

Table 7: Experimental results on ARC, RACE-Open and MCScript-Open. We report the accuracy of ET-RR on multiple-choice selection.

Model	ARC Test	RACE-Open Test	MCScript-Open Test
IR solver[5] (Clark et al., 2018)	20.26	30.70	60.46
Guess All (random) (Clark et al., 2018)	25.02	25.01	50.02
BiDAF[6] (Clark et al., 2018)	26.54	26.89	50.81
DGEM (Clark et al., 2018)	27.11	/	/
KG^2 (Zhang et al., 2018)	31.70	/	/
TriAN + f(dir)(cs) + f(ind)(cs) (Zhong et al., 2018)	33.39	/	/
BiLSTM Max-out (Mihaylov et al., 2018)	33.87	/	/
ET-RR	**36.61**	**38.26**	**66.78**

Table 8: Comparisons for different query formulation methods and amounts of retrieved evidence (i.e. top K) on ARC dataset, in terms of percentage accuracy.

Model	ET-RR (Concat)		ET-RR (TF-IDF)		ET-RR	
Top K	Dev	Test	Dev	Test	Dev	Test
5	39.26	33.36	39.93	34.73	39.93	35.59
10	38.93	35.33	39.43	35.24	**43.96**	**36.61**
20	41.28	34.56	38.59	33.88	42.28	35.67

4.3 Ablation study

Finally, we investigate how each component contributes to model performance.

Attention components. Table 6 demonstrates how the attention components contribute to the performance of ET-RR. As shown, ET-RR with all attention components performs the best on the ARC test set. The performance of ET-RR without passage-question attention drops the most significantly out of all the components. It is worth noting that the choice interaction layer gives a further 0.24% boost on test accuracy.

Essential term selection. To understand the contribution of our essential-term selector, we introduce two variants of ET-RR:

- ET-RR (Concat). Concatenates the original question and answer choice as the query.
- ET-RR (TF-IDF). We calculate the TF-IDF scores and take top 30% words[8] with the highest scores in the question to concatenate with each answer choice as a query.

Table 8 shows an ablation study comparing different query formulation methods and amount of retrieved evidence K. As shown, with the essential term selector ET-Net, the model consistently outperforms other baselines, given different numbers of retrievals K. The performance of all models works best when $K = 10$. Furthermore, only using TF-IDF to select essential terms in a question is not effective. When $K = 10$, the ET-RR (TF-IDF) method even performs worse than ET-RR (Concat). This illustrates the challenges in understanding what is essential in a question.

5 Conclusion

We presented a new retriever-reader model (ET-RR) for open-domain QA. Our pipeline has the following contributions: (1) we built an essential term selector (ET-Net) which helps the model understand which words are essential in a question leading to more effective search queries when retrieving related evidence; (2) we developed an attention-enhanced reader with attention and fusion among passages, questions, and candidate answers. Experimental results show that ET-RR outperforms existing QA models on the ARC, RACE-Open and MCScipt-Open datasets.

[8] According to the annotated dataset, around 30% of the terms in each question are labelled as essential.

REFERENCES

Michael Boratko, Harshit Padigela, Divyendra Mikkilineni, Pritish Yuvraj, Rajarshi Das, Andrew McCallum, Maria Chang, Achille Fokoue, Pavan Kapanipathi, Nicholas Mattei, Ryan Musa, Kartik Talamadupula, and Michael Witbrock. A systematic classification of knowledge, reasoning, and context within the arc dataset. In *QA@ACL*, 2018.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. In *ACL*, 2017.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *CoRR*, abs/1803.05457, 2018.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke S. Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*, 2017.

Daniel Khashabi, Tushar Khot, Ashish Sabharwal, and Dan Roth. Learning what is essential in questions. In *CoNLL*, 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

Heeyoung Kwon, Harsh Trivedi, Peter Jansen, Mihai Surdeanu, and Niranjan Balasubramanian. Controlling information aggregation for complex question answering. In *ECIR*, 2018.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683*, 2017.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*, 2018.

Sewon Min, Victor Zhong, Richard Socher, and Caiming Xiong. Efficient and robust question answering from minimal context over documents. In *ACL*, 2018.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. Ms marco: A human generated machine reading comprehension dataset. *CoRR*, abs/1611.09268, 2016.

Kyosuke Nishida, Itsumi Saito, Atsushi Otsuka, Hisako Asano, and Junji Tomita. Retrieve-and-read: Multi-task learning of information retrieval and reading comprehension. *CoRR*, abs/1808.10628, 2018.

Simon Ostermann, Michael Roth, Ashutosh Modi, Stefan Thater, and Manfred Pinkal. Semeval-2018 task 11: Machine comprehension using commonsense knowledge. In *SemEval@NAACL-HLT*, 2018.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. In *EMNLP*, 2016.

Robert Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*, 2017.

Liang Wang. Yuanfudao at semeval-2018 task 11: Three-way attention and relational knowledge for commonsense machine comprehension. In *SemEval@NAACL-HLT*, 2018.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerald Tesauro, Bowen Zhou, and Jing Jiang. R3: Reinforced ranker-reader for open-domain question answering. In *AAAI*, 2018.

Yuyu Zhang, Hanjun Dai, Kamil Toraman, and Le Song. KG^2: Learning to reason science exam questions with contextual knowledge graph embeddings. *CoRR*, abs/1805.12393, 2018.

Wanjun Zhong, Duyu Tang, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. Improving question answering by commonsense-based pre-training. *arXiv preprint arXiv:1809.03568*, 2018.