Addressing Candidate Divergence with Answer Matching Model for Open-Domain Question Answering

AAAI Press
Association for the Advancement of Artificial Intelligence
2275 East Bayshore Road, Suite 160
Palo Alto, California 94303

Abstract

In this paper, we focus on open-domain question answering task which aims to answer the question given an external knowledge base and multiple candidate options. The AI2 Reasoning Challenge (ARC), a new question answering dataset which is designed for adopting external knowledge for better solving the problems that needs reasoning. Previous works mainly focus on retrieving relevant knowledge and matching between the relevant knowledge and question or matching between the relevant knowledge and the candidate answers, while ignoring the importance of comparing the candidate answers. In this paper, we propose a novel model called Answer Matching Model (AMM) that focuses on the matching between candidate answers and extract relevant information for question answering. We also use the external knowledge to enrich the representation of the question for better understanding its underlying intention. Experimental results on both ARC-Challenge and ARC-Easy subsets demonstrate that the proposed model significantly outperform various state-of-the-art systems by a large margin.

Introduction

As we all known, it's always our goal to enable the machine to correctly answer questions expressed in natural language in the field of Question Answer (QA). This is also one of the biggest challenges in the field of artificial intelligence (Norvig 1978). So we need the machine to really understand the problem. In order to help the machine better unserstand the question and choose the correct answer, many researcheds are working on the open-domain question answering (QA) field (David Ferrucci 2010; Jonathan Berant and Liang. 2013; Danqi Chen 2017). In the open-domain QA field, we first need to use information retrieval (IR) methods to retrieve relevant knowledge from external knowledge bases (KBs). This knowlegde can be some articles or some sentences. Then we use the knowledge gained to choose the correct option for the question. In this paper, we take this knowledge as a document and then we build a neural network to choose the correct answer.

People have achieved ,high performance on some dataset, like SQuAD (Pranav Rajpurkar and Liang. 2016) and so on. But there is still a problem whether our work depends more on retrieving or on reasoning. If we only rely on the means of retrieval to achieve good performance on some datasets, although the experiment looks good, it didn't help much in improving the reading comprehension ability of the machine. We should pay more attention to the reasoning part.

Recently, a data set called ARC (AI2 Reasoning Challenge) (Peter Clark 2018) is proposed, which is mainly to provide a new challenge to the QA field. The biggest difference between ARC dataset and other QA datasets is that in addition to the questions and options, it also includes a knowledge base named ARC Corpus that contains the knowledge needed to answer questions. This ARC Corpus contains 14 million sentences. It's different from the large-scale QA system, IBM's DeepQA (David Ferrucci 2010), with multiple sources, like Wikipedia, KBs, dictionaries, etc. There are only one knowledge base –ARC Corpus and their official analysis sugests that 95% of the questions can be answered using the ARC Corpus. So we can focus on how to fully use the knowledge gained from the retrieval to answer questions.

In this paper , we firstly use a search engine (we use Elasticsearch as (Peter Clark 2018) have used) to retrieve relevant sentences that can answer question q from ARC Corpus. To do this, for each answer option a, we stitch them together $q + a$, and then we retrieve $q + a$ as input to the search engine. It returns the relevant sentence and the search engine's score. Then we have two ways to choose correct answer. One is to use the traditional machine learcing (ML) method to choose the correct answer, and the other is to use the deep learning method to choose the correct answer – we propose a novel Answer Matching Model (AMM) to solve the multi-choice question. Through the comparison of these two methods, we can really understand what our AMM can solve more than traditional machine learcing methods.

For the AMM, we mainly calculate the difference between the current option and other options. We also merge the question q and the option a into one input $q + a$, we call it QA Pair (QAP). Another input to the model is a passage consisting of the retrieved relevant sentences pa. Unlike the previous works, that only compute the attention-weighted vectors of the current QAP and each position in the passage, our model also performs two matching operations, one is the match inside the candidate answer and the other is the match between the answers.

In brief, our main contributions are summarized below:

- We solve multiple choice problems in two directions, one is the traditional machine learning method, the other is the deep learning method. This helps us to truly understand the advantages of AMM.
- We propose a new model AMM that is dedicated to the match between candidate answers.
- Our experimental results on ARC data outperforms various state-of-the-art model.Our model has achieved the state-of-the-art result.

Related Works

As we all known, the definition of open-domain QA is for a question, we use external knowledge bases to choose the answer. People usually use the Wikipedia as the KB. Danqi Chen built a system named DrQA which consists of two components, one is the Document Retriever module for finding relevant articles and the other is a machine comprehension model, Document Reader, for extracting answers from a single document or a small collection of documents.

Unlike previous works, they use the method of reading comprehension to solve QA problems. And they have a good experimental result on some data sets, SQuAD, CuratedTREC (based on TREC Voorhees and Tice). Shuohang Wang and Jiang call these methods Search-and-Reading QA (SR-QA). These methods divide the pipeline into IR and RC two part. The difference between SR-QA and standard RC is that the passage used for training in SR-QA are not given, it need us to retrieval use IR method.

Their system is simple and powerful for open-domain QA. This gives us a lot of inspiration. In this paper, we also use the methods of RC to solve the QA problems. Next, we will introduce some recent work in the field of RC.

The field of RC has been developing rapidly in recent years. These are all due to the release of relevant RC data sets like CNN/Daily Mail(Hermann et al. 2015) built from news articles, SQuAD (Pranav Rajpurkar and Liang. 2016) built from the Wikipedia, RACE(Guokun Lai and Hovy. 2017) extract from middle and high school English examinations in China. Many researchers have proposed many effective deep learning model structures on these data sets.

Hermann et al. have released the CNN/daily Mails dataset. This dataset's answer is a word of the passage, so we can call it the cloze-style's RC. They also proposed an attention-based neural network to handle this task. Experimental results show that the effect of this neural network model far exceeds the traditional baseline.

Pranav Rajpurkar and Liang. have released the SQuAD dataset which answer is a continuous span in the passage, so we need to get the start point and the end point in the passage. M.J.Seo and Hajishirzi. have proposed the Bi-Directional Attention Flow (BIDAF) network, a hierarchical multi-stage architecture for modeling the representations of the context paragraph at different levels. They also use bi-directional attention flow mechanism to obtain a query-aware context represention. Their experimental results show that their model have achieved the state-of-the-art results in SQuAD.

Guokun Lai and Hovy. have relased the RACE dataset that was extracted from middle and high school English examinations in China. The RACE is a new multi-choice machine comprehension dataset. And Shuohang Wang have proposed a Co-Matching model (CMM) for this dataset. The propose of CMM is to match the question-answer pair to a given passage. And for each position in the passage, they compute two attention-weighted vectors, where one is the question and the other from the candidate answer.

Our system is different from the previous work, which we solve this problem through two different ways, one is the traditional machine learning method, and the other is the end-to-end neural network. In this way, we analyze the advantages of the end-to-end neural network. To the end-to-end neural network, we perform two matching operation, one is the answer internal match and the other is the match between answers.

Our System

In the following, we will introduce our question answering system. we will divide it into three parts to introduce this whole system. 1) We use search engines to retrieve relevant sentences in ARC Corpus. 2) We use traditional machine learning methods to solve QA problems. We use three features constructed from three levels of sentence, statictics, and semantics as inputs to the logistic regression (LR) model, we named it SSS model. 3) We use an end-to-end neural network–AMM model, to solve QA problems.

Processing

First, we use a search engine (Elasticsearch) to retrieve relevant sentences that can answer question q from ARC Corpus. As we mentioned above, for each answer option a, we stitch them together $q + a$, and then we retrieve $q + a$ as input to the search engine. At the same time, it also return the search engine's score for each sentence. We sort the sentences of all the options under one question by the search engine's score, $S_es(pa, a, q)$. Similar to SemanticILP (T. Khot and Clark. 2018), we pick the top 8 sentences across the answer choices for each question. That way each option might have 0 to 8 sentences, but the number of sentences for all options, or the number of sentences for a question, must add up to 8. Finally, we will get m(m=0..8) sentences pa for each option a, m can be any value between 0 and 8.

SSS model

Elasticsearch (ES) not only return relevant sentences, but also their confidence scores, S_{es}, which measure their relevance to the entered query $q+a$. So we take S_es as the first feature.

For a given question q and an answer option a, we can get the question and the answer option pair, qa, through appending the answer option behind the question stem. At the same time, we get the relevant sentences pa. For each sentences $pa^t, t = (1...m)$, we also score the following two scores.

1) Exact match score: whether the non-stopword in qa appear in pa^t. Then we calculate the match score S^t_{em} using:

$$S^t_{em} = \frac{N}{M} \tag{1}$$

where N is the number of non-stopword in qa that appear in pa^t and M is the total number of non-stopword contained in qa.

2) Aligned embedding score: For nouns and verbs in qa, we first calculate their term frequency-inverse document frequency(TF-IDF).

$$TF_j = \frac{w_j}{W} \tag{2}$$

$$IDF_j = \log \frac{c}{m+1} \tag{3}$$

$$\text{TF-IDF}_j = TF_j * IDF_j \tag{4}$$

where w_j is the number of jth noun or verb appeared in the qa, and W represents the total number of nouns and verbs in the qa , c represents the sentence num that the sentence contain the jth word, m represents the total number of sentences in pa.

Next, we use a 300-dim Glove word embeddings to represent the word in qa, we can named the jth word embedding vector as we_j. Then we multiply TF-IDF$_j$ as a weight to the word vector.

$$E_{qa} = \sum_{j=1}^{M} \text{TF-IDF}_j * we_j \tag{5}$$

In the same way, we can get the E^t_{sen} that represent the tth sentence in word embedding vector. Because both E_{qa} and E^t_{sen} are a 300-dim vector. So we can calculate their cosine similarity as follows:

$$S^t_{emb} = cos(E_{qa}, E^t_{sen}) \tag{6}$$

Finally, we use these three scores $S^t_{es}, S^t_{em}, S^t_{emb}$ as three features to input into the LR model.

Answering Matching Model

In this part, we introduce how we use the end-to-end neural network to solve QA. As we mentioned before, we also use the result of ES. So now we not only have the question and answer options, and the sentences associated with them. The data format can be described as: $q, (a_0, pa_0), (a_1, pa_1), ...(a_n, pa_n)$. So we can unserstand this task as we have a question and a few candidate answers and their related sentences, and we need to choose the right answer from the candidate answers.

For the model, we enter: $(q, ca, pa, (oa^1, oa^2, ...oa^n)$, where q represent the question, ca represents one of the candidate answers, pa represents the relevant sentences of ca, oa represents other candidate answers. The overall architecture of the Answering Matching Model is displayed in Figure 1. Next, we will introduce the details of our model.

Embedding Layer We first transform every word in the question q, current candidate answer ca , the corresponding relevant sentences pa, and other candidate answers $oa^j, j = (1, ...n)$ into one-hot representations. Then we convert them into continuous representations with a shared embedding matrix W_e. We take $E(x)$ to denote the embedding representation. After embedding we concatenate the q embedding result $E_q(x)$ with the ca embedding result of $E_{ca}(x)$, and we also concatenate the $E_q(x)$ with each other candidate answer $oa^j, j = (1, ...n)$ embedding result $E^j_{oa}(x), j = (1, ...n)$. We can name the aggregated representation of the question and the current candidate answer qca and name the aggregated representation of the question and the other candidate answer qoa^j. The reason we do this is that only a combination of questions and candidate answers can express a complete meaning. For each sentecne $pa^t, t = (1, ...m)$ in pa, we can also get the corresponding embedding result $E^t_{pa}, t = (1, ...m)$.

$$E(x) = W_e \cdot x, x \in q, ca, pa^t, oa^j, \tag{7}$$

$$E_{qca_i} = [E_q(x), E_{ca}(x)] \tag{8}$$

$$E^j_{qoa} = [E_q(x), E^j_{oa}(x)] \tag{9}$$

After concatenating, we use the bi-directional LSTMs which only return the result of the last moment to get the contextual representations of E_{qca}, E^j_{qoa} and E^t_{pa}.

$$\overrightarrow{H_s(x)} = \overrightarrow{LSTM}(E(x)) \tag{10}$$

$$\overleftarrow{H_s(x)} = \overleftarrow{LSTM}(E(x)) \tag{11}$$

$$H_s(x) = [\overrightarrow{H_s(x)}; \overleftarrow{H_s(x)}] \tag{12}$$

We take $H_{qca} \in \mathbb{R}^{2d}$, $H^j_{qoa} \in \mathbb{R}^{2d}$, and $H^t_{pa} \in \mathbb{R}^{2d}$ to denote the contextual embedding results of E_{qca}, E^j_{qoa} and E^t_{pa}, where d is the dimension of LSTM(one-way).

Answer Internal Match Now, we get the aggregated contextual representations of the question and the current candidate answer–H_{qca}, the aggregated contextual representations of the question and the other candidate answers–H^j_{qoa}, and the contextual representation of their relevant sentences H^t_{pa}. In order to analyze the correlation between H^t_{pa} and H^j_{qoa} or H_{qca}, we first calculate the cosine similarity between them.

$$W^t_{qca} = cos(H_{qca}, H^t_{pa}) \tag{13}$$

$$W^t_{qoa} = cos(H^j_{qoa}, H^t_{pa}) \tag{14}$$

$W_{qca} = [W^1_{qca}, W^2_{qca}, ...W^m_{qca}] \in \mathbb{R}^m$, $W^j_{qoa} = [W^{j1}_{qoa}, W^{j2}_{qoa}, ...W^{jm}_{qoa}] \in \mathbb{R}^m$. Each value in W_{qca} or W^j_{qoa} is the cosine score of each sentence in H^t_{pa} with H_{qca} or H^j_{qoa} which indicates the relevance weight over H_{qca} or H^j_{qoa} in the view of H^t_{pa}. Then we combine the view of each vector of H_{pa} by multiplying the cosine score with their corresponding H^t_{pa} and then adding them together to get the final attention vector G:

$$G_{qca} = \sum_{t=1}^{m} W^t_{qca} \cdot H^t_{pa} \tag{15}$$

$$G^j_{qoa} = \sum_{t=1}^{m} W^{jt}_{qoa} \cdot H^t_{pa} \tag{16}$$

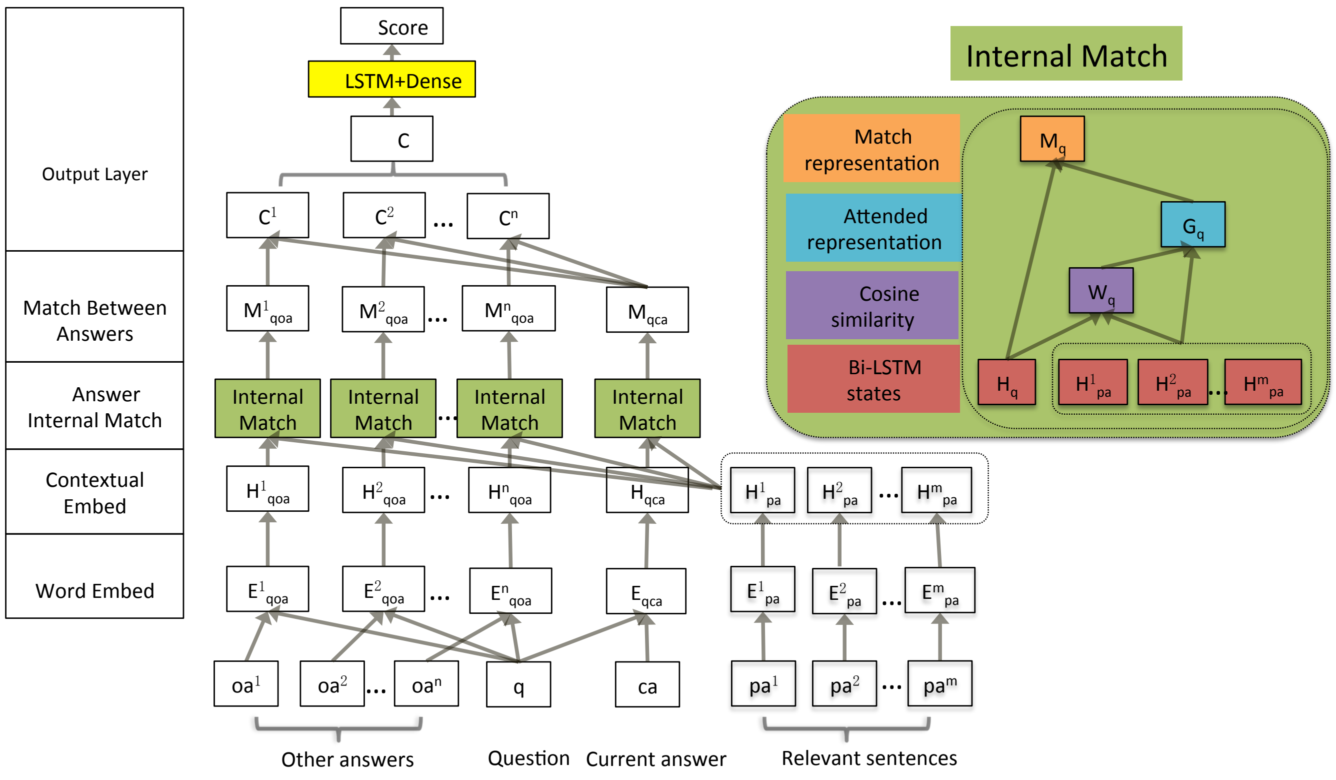

Figure 1: The neural architecture of Answering Matching Model.

the attention vector $G \in \mathbb{R}^{2d}$ can be regarded as qca or qoa representation that fuses the relevant sentence information. Then we can calculate the matching matrix between H_{qca} and G_{qca} to reveal how much information is used in the relevant sentence and the same operation is done for H^j_{qoa} and G^j_{qoa}.

$$M_{qca} = [H_{qca} \ominus G_{qca}; H_{qca} \otimes G_{qca}] \tag{17}$$

$$M^j_{qoa} = [H^j_{qoa} \ominus G^j_{qoa}; H^j_{qoa} \otimes G^j_{qoa}] \tag{18}$$

and $M_{qca} \in \mathbb{R}^{4d}$, $M^j_{qoa} \in \mathbb{R}^{4d}$.

Match Between Answers Now, we have M_{qca}, and $M^j_{qoa}, j = 1, ...n$ which is obtained by qca or qoa^j itself through the matching before and after the fusion of relevant sentence information. Below we are going to make a match between qca and qoa^j. This will help us understand the difference between current options and other candidate options. More specifically, we get a new match representation $C^j \in \mathbb{R}^{8d}$ by the subtraction and multiplication of the alignment between M_{qca} and M^j_{qoa}.

$$C^j = [M^j_{qoa} \ominus M_{qca}; M^j_{qoa} \otimes M_{qca}] \tag{19}$$

Final Prediction Therefore, we now get the match representation of the current options and all other candidate options, $C^j, j = 1, ...n$. Then we follow the columns to concante them together $C \in \mathbb{R}^{n*8d}$.

$$C = \begin{bmatrix} C^1 \\ C^2 \\ \vdots \\ C^n \end{bmatrix} \tag{20}$$

Additionally, we further apply a bi-directional LSTM which returns the hidden layer results at all times to model the candidate answers, connecting the relationship between different candidate answers. The number of hidden layers of this bi-directional LSTM is the same as the previous one.

$$V = Bi - LSTM(C) \tag{21}$$

So in the end, we encode the candidate answer information from other candidate answers into a single feature vector $V \in \mathbb{R}^{n*2d}$.

For final prediction, we enter the vector V into a fully-connected layer with sigmoid activation.

$$score = sigmoid(W_v^T \cdot V + b_v) \qquad (22)$$

where W_v, b_v are trainable weights. Our goal is to predict whether the current option can use the sentences from the search to answer questions. We use the model-predicted score to characterize the credibility of the candidate answer to answer the question, which we call it as the credibility score. We can think of it as a binary classification task. To train our model, we minimize the cross entropy loss between the prediction and ground truth.

In practice, after the model gives the credibility score, we need to choose an optimal answer from multiple options(usually four options). we rank the candidate answer by the credibility score and select the top one as the final answer for the question.

Experiments

In this section we first introduces the two experimental datasets, and then presents the evaluation of our SSS model and AMM model, and finally their experimental results on the two datasets.

Data

We evaluate our model on ARC dataset (Peter Clark 2018). The ARC dataset totally contains 7787 questions. The ARC Dataset is divided into two datasets. One is the ARC Easy Dataset, the other is the ARC Challenge Dataset. Next we will introduce these two datasets.

ARC Challenge dataset: To encourage people to focus on the reasoning part of the data, Peter Clark divides some data that cannot be solved by both a retrieval-based algorithm and a word co-occurrence algorithm into the Challenge dataset. The ARC Challenge dataset contains 2590 questions.

ARC Easy dataset: The Easy dataset contains the remaining part of the ARC data. That is to say, in theory, it can be solved by retrieval-based algorithm or word co-occurrence algorithm. The ARC Easy dataset contains 5197 questions.

Finding Relevant Sentences

As we mentioned before, for each answer option a, we conver the question q plus the answer option a into a fusion representation qa, use this as a query to retrieve relevant sentences pa in ARC Corpus by Elasticsearch.

Document Retriever Sort

- **Search Engine Solver (SNS):** As we mentioned above, inspired by Peter Clark, for each answer option a, we conver the question q plus the answer option a into a fusion representation qa, use this as a query to retrieve relevant sentences pa in ARC Corpus by Elasticsearch. And then we can get the S_es score for each relevant sentence.
- **Exact Match Solver (EMS):** For each sentences in pa, we can calculate the match score through equation 1. This score we can named it as S_em score.
- **Aligned Embedding Solver (AES):** For each sentences in pa, we can also calculate the allign embedding score through equation 2-6. This score we can named it as S_emb score.

Combination: Now each solver outputs a non-negative confidence score for each of the answer options. Inspired by Peter Clark, we use the LR (logistic regression classifier) model as a combiner to produce a combined confidence score (between 0 and 1), by inputing these three scores into the LR model as three features.

AMM Implementation Details

The general settings of our AMM model are listed as follows.

We initialize word embeddings with 100D pre-trained GloVe embedding (Jeffrey Pennington and Manning. 2014), which are further updated in training phrase. Then we use the bidirectional LSTM with $h = 70$ hidden units for qca, qoa^t, pa encoding. And in the section of getting matching between answers, we also use one bidirectional LSTM with $h = 70$ hidden units. Finally, we adopted ADAM optimizer for weight updating (Kingma and Ba. 2014), with an initial learning rate of 0.001. A mini-batch of 64 samples is used to update the model parameter per step. We keep 25,000 most frequent words in the ARC Corpus as vocabulary and add a special token UNK for out-of-vocabularly words. The maximum length of question is 25. The maximum length of answer is 8. The maximum number of related sentences is 5. The maximum length of each related sentence is 35. We implement our model by Keras (Chollet and others 2015) with Theano backend (Theano Development Team 2016) .

Baseline Models

- **IR Solver:** The IR Solver (Peter Clark 2016) combine the question with an answer option. Then they use the result of combination to search in the ARC Corpus. It will not only return the related sentence, but also a search score. The higher the score, the closer the relationship is. So the option with the highest search score is finally selected.
- **Guess All (Random):** A naive baseline, choosing all the answer options works equally well, scoring $1/k$ for each question and having K answer options. A system that chooses a single answer at random will also converge to this score after enough trials.
- **BIDAF:** The BIDAF model (M.J.Seo and Hajishirzi. 2017) has achieved superior performance on the SQuAD (Pranav Rajpurkar and Liang. 2016) dataset. They concatenate the retrieved sentences to form a passage. And then they use the BIDAF model to select the answer span from the passage. At last, among the multiple options, the highest overlap with the answer span is the correct answer.
- **DecompAttn:** DecompAttn (A.P. Parikh and Uszkoreit. 2016) is a neural entailment model. First, they convert the question q plus an answer option a into a hypothesis sentence h, use this as a search query to retrieve text sentences p from the ARC Corpus. Then they will compute

Method	Challenge	Easy
IR	20.26	62.55
Guess All/Random	25.02	25.02
DecomAttn	26.41	57.45
BiDAF	26.54	58.36
DGEM	27.11	58.97
SSS	25.76	63.00
AMM	32.76	63.91

Table 1: Experimental results on ARC-Challenge and ARC-Easy dataset.

the entailment scores between h and each sentence in p. And this is repeated for all answer options. Finally, the option with the overall highest entailment score selected.

- **DGEM:** DGEM (T. Khot and Clark. 2018) also is a neural entailment model. It built the hypothesis h in the same way as DecompAttn. Then the DGEM model uses a structured representation of the hypothesis h, extracted with a proprietary parser plus Open IE. It also compute the score for each question, option and retrieved text sentence, then choose the option with the highest score as the correct answer.

Overall Results

Our experiments are carried out on public dataset: ARC Challenge dataset and ARC Easy dataset. The experimental results are given in Table 1. Although this task has the difficulty of searching for relevant sentences and using these related sentences to select the correct answer, our AMM model still outperforms state-of-the-art systems by a large margin, where 5.6% and 4.9% absolute improvements over DGEM in ARC Challenge dataset and ARC Easy dataset, which demonstrate the effectiveness of our model. Compared with the traditional machine learning method–SSS model, our AMM model obviously exceeds it in the ARC Challenge dataset, and AMM model is still slightly higher in the ARC Easy dataset. This shows that our AMM model not only has the reasoning ability, but also works well on problems that can be solved by retrieval-based algorithm and word co-occurrence algorithm .

Model Ablation

To evaluate how different components contribute to the model's performance, we conduct an ablation study of our model architecture on the ARC Challenge dataset and ARC Easy dataset, and results ar illustrated in Table 2. We studied two key factors: (1) the Answer Internal Match module (AIM), (2) the Match Between Answers module (MBA).

We observed a 2.47 percentage performance decrease on the ARC Challenge dataset and 0.32 percentage performance decrease on the ARC Easy dataset by replacing the AIM module with a single state (i.e., only H_{qca_i}, H_{qoa_j}, that means $M_{qca} = H_{qca}$, $M^j_{qoa} = H^j_{qoa}$ in Equation 17, 18). We can clearly see that the decline on the ARC Challenge dataset is much larger than the decline on the ARC Easy dataset. This reveals that our AIM module takes full

	Challenge	Easy
AMM	32.76	63.91
AMM-AIM	30.29	63.59
AMM-MBA	31.39	63.55

Table 2: Ablations on ARC-Challenge dataset.

advantage of the knowledge in relevant sentences(because the AIM module is mainly used to compute the difference between the question answer fusion representation with the knowledge in relevant sentences and the original question answer fusion representation).

We also observe about a 1.37 percentage performance decrease on the ARC Challenge dataset and 0.36 percentage performance decrease on the ARC Easy dataset by directly concatenating each candidate answer in MBA module (that means in Equation 19 $C^j = M^j_{qoa}$), we do not perform the alignment subtration and alignment multiplication operation. This reveal the importance of calculating the relationship between candidate answers.

From the previous two experiments we can see that they all have large fluctuations in the ARC Challenge dataset, and small fluctuations in the ARC Easy dataset, which also reveals that the two pieces of the model are mainly dedicated to solving the reasoning part.

Analysis and Discussion

In this section, we are aimed to couduct an indepth analysis and answer the following question: What are the advantages of our AMM model compared to SSS model the traditional method in ARC Challenge dataset? In order to know the effectiveness of our AMM, we have carefully analyzed the results in the test of ARC Challenge dataset.

Firstly, among the 1172 test samples, our AMM has correctly predicted 384 samples and our SSS model has correctly predicted 291 samples. There are 84 samples that these two methods both have chosen the correct answer. In other words, there are 300 samples, only AMM has chosen the correct answer, and there are 207 samples that only the SSS model has chosen the correct answer. Then we randomly select 50 correctly-predicted samples from their non-oberlapping parts.

We divided these samples correctly-predicted into two categories which depends on whether we need to use reasoning to solve the question. Figure 3 illustrates one sample which qa has the same meaning as the sentence in pa, and the keywords in qa and pa, some meaningful nouns or verbs, are the same, so we don't need any reasoning ability to answer this question. Figure 4 illustrates one sample which the meaning of qa and the sentence in pa is not exactly the same. But we can use the contained knowledge in the sentence pa to infer the conclusion of qa. From Figure 4 we can know that alothough the relevant sentence don't directly tell us that *a star with twice the mass as the Sun would use its fuel source much more quickly*, we can use the knowledge contained in pa that *the larger the mass of the star, the quickly it burns its fuel sources* to infer this conclusion, because the Sun also can be seen as a star.

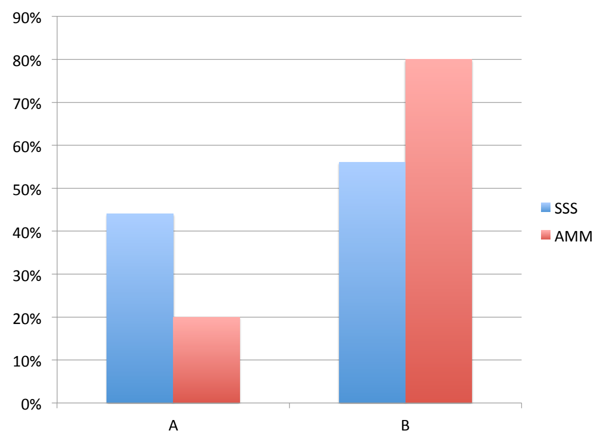

Figure 2: Percentage of the two categories. A represents the category that qa has the same meaning as the sentence in pa. B represents another category, which requires some reasoning ability to answer the question.

question q:What is potential energy ?
answer a:The energy an object has due to its position.
relevant sentences pa:
pa^1:Potention energy is energy an object possesses due to its position or condition.
pa^2:Potential energy is the energy an object has a result of its position.

Figure 3: A sample that the fusion representation of question and answerqa has the same meaning as the sentence in the relevant sentences pa.

As we can see in the Figure 2 , our AMM model significantly exceeds the SSS model in terms the percentage of class B which requires the reasoning ability. This also shows that our AMM model is more focused on reasoning.

Conclusion

In this paper, we present a novel neural architecture, called Answer Match Model, to tackle the open-domain question answering task. The proposed AMM model aims to compute the answer internal match and the match between answers, which will make full use of the retrieved relevant sentences and the difference between the candidate answers. Experimental results show that the proposed model achieves state-of-the-art results on both ARC-Challenge and ARC-Easy datasets, which demonstrates the effectiveness and generalization of our model. On the other hand, by comparing with the SSS model, we analyzed that the AIM and MBA modules in our AMM model are dedicated to the reasoning part. In the future, we would apply the proposed answer matching model to other NLP tasks to further testify its extensibility.

question q:A star with twice the mass as the Sun would ?
answer a:use its fuel source much more quickly.
relevant sentences pa:
pa^1: And surprisingly, the larger the mass of the star, the quicker it burns its fuel sources and the shorter its lifespan.

Figure 4: A sample that we can use the knowledge contained in the relevant sentence to answer the question.

References

A.P. Parikh, O. Tackstrom, D. D., and Uszkoreit., J. 2016. A decomposable attention model for natural language inference. *In Empirical Methods in Natural Language Processing(EMNLP).*

Chollet, F., et al. 2015. Keras. `https://keras.io`.

Danqi Chen, Adam Fisch, J. W.-A. B. 2017. Reading wikipedia to answer open-domain questions. *In Association for Computational Linguistics(ACL).*

David Ferrucci, Eric Brown, J. C.-C. J. F. D. G. A. A. K. A. L. J. W. M. E. N. J. P. e. a. 2010. Building watson: An overview of the deepqa project. *AI maganize*.

Guokun Lai, Qizhe Xie, H. L.-Y. Y., and Hovy., E. 2017. Race:large-scale reading comprehension dataset from examinations. *Proceedings of the Conference on Empirical Methods in Natural Language Processing(EMNLP).*

Hermann, K. M.; Kocisky, T.; Grefenstette, E.; Espeholt, L.; Kay, W.; Suleyman, M.; and Blunsom, P. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, 1684–1692.

Jeffrey Pennington, R. S., and Manning., C. 2014. Glove: Global vectors for word representation. *In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 1532–1543.

Jonathan Berant, Andrew Chou, R. F., and Liang., P. 2013. Semantic parsing on freebase from question-answer pairs. *In Empirical Methods in Natural Language Processing(EMNLP).*

Kingma, D., and Ba., J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

M.J.Seo, A. Kembhavi, A. F., and Hajishirzi., H. 2017. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Norvig, P. 1978. A unified theory of inference for text understanding . *Ph.D.thesis*.

Peter Clark, Oren Etzioni, T. K.-A. S. O. T. P. T. 2016. Combining retrieval, statistics, and inference to answer elementary science questions. *In AAAI.*

Peter Clark, Isaac Cowhey, O. E.-T. K. A. S. C. S. O. T. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Pranav Rajpurkar, Jian Zhang, K. L., and Liang., P. 2016. Squad:100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Shuohang Wang, Mo Yu, X. G.-Z. W. T. K. W. Z. S. C. G. T. B. Z., and Jiang, J. 2017. R3: Reinforced ranker-reader for open-domain question answering. *arXiv preprint arXiv:1709.00023*.

Shuohang Wang, Mo Yu, S. C.-J. J. 2018. A co-matching model for multi-choice reading comprehension. *In Association for Computational Linguistics(ACL).*

T. Khot, A. S., and Clark., P. 2018. Scitail: A textual entailment dataset from science question answering. *AAAI*.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints* abs/1605.02688.

Voorhees, E., and Tice, D. 2000. Building a question answering test collection. *In Proc. of 23rd annual Intl. ACM SIGIR Conf.*