Joint Training of Candidate Extraction and Answer Selection for Reading Comprehension

Zhen Wang   Jiachen Liu   Xinyan Xiao   Yajuan Lyu   Tian Wu

Baidu Inc., Beijing, China

{wangzhen24, liujiachen, xiaoxinyan, lvyajuan, wutian}@baidu.com

Abstract

While sophisticated neural-based techniques have been developed in reading comprehension, most approaches model the answer in an independent manner, ignoring its relations with other answer candidates. This problem can be even worse in open-domain scenarios, where candidates from multiple passages should be combined to answer a single question. In this paper, we formulate reading comprehension as an extract-then-select two-stage procedure. We first extract answer candidates from passages, then select the final answer by combining information from all the candidates. Furthermore, we regard candidate extraction as a latent variable and train the two-stage process jointly with reinforcement learning. As a result, our approach has improved the state-of-the-art performance significantly on two challenging open-domain reading comprehension datasets. Further analysis demonstrates the effectiveness of our model components, especially the information fusion of all the candidates and the joint training of the extract-then-select procedure.

1 Introduction

Teaching machines to read and comprehend human languages is a long-standing objective in natural language processing. In order to evaluate this ability, reading comprehension (RC) is designed to answer questions through reading relevant passages. In recent years, RC has attracted intense interest. Various advanced neural models have been proposed along with newly released datasets (Hermann et al., 2015; Rajpurkar et al., 2016; Dunn et al., 2017; Dhingra et al., 2017b; He et al., 2017).

Q	Cocktails: *Rum*, *lime*, and *cola* drink make a ------------.
A	**Cuba Libre**
P₁	**Daiquiri**, the custom of mixing *lime* with *rum* for a cooling drink on a hot Cuban day, has been around a long time.
P₂	Cocktail recipe for a **Daiquiri**, a classic *rum* and *lime* drink that every bartender should know.
P₃	Hemingway Special **Daiquiri**: Daiquiris are a family of cocktails whose main ingredients are *rum* and *lime* juice.
P₄	A homemade Cuba Libre Preparation To make a **Cuba Libre** properly, fill a highball glass with ice and half fill with *cola*.
P₅	The difference between the **Cuba Libre** and *Rum* is a *lime* wedge at the end.

Table 1: The answer candidates are in a **bold** font. The key information is marked in *italic*, which should be combined from different text pieces to select the correct answer "Cuba Libre".

Most existing approaches mainly focus on modeling the interactions between questions and passages (Dhingra et al., 2017a; Seo et al., 2017; Wang et al., 2017), paying less attention to information concerning answer candidates. However, when human solve this problem, we often first read each piece of text, collect some answer candidates, then focus on these candidates and combine their information to select the final answer. This collect-then-select process can be more significant in open-domain scenarios, which require the combination of candidates from multiple passages to answer one single question. This phenomenon is illustrated by the example in Table 1.

With this motivation, we formulate an extract-then-select two-stage architecture to simulate the above procedure. The architecture contains two components: (1) an extraction model, which generates answer candidates, (2) a selection model, which combines all these candidates and finds out

the final answer. However, answer candidates to be focused on are often unobservable, as most RC datasets only provide golden answers. Therefore, we treat candidate extraction as a latent variable and train these two stages jointly with reinforcement learning (RL).

In conclusion, our work makes the following contributions:

1. We formulate open-domain reading comprehension as a two-stage procedure, which first extracts answer candidates and then selects the final answer. With joint training, we optimize these two correlated stages as a whole.

2. We propose a novel answer selection model, which combines the information from all the extracted candidates using an attention-based correlation matrix. As shown in experiments, the information fusion is greatly helpful for answer selection.

3. With the two-stage framework and the joint training strategy, our method significantly surpasses the state-of-the-art performance on two challenging public RC datasets Quasar-T (Dhingra et al., 2017b) and SearchQA (Dunn et al., 2017).

2 Related Work

In recent years, reading comprehension has made remarkable progress in methodology and dataset construction. Most existing approaches mainly focus on modeling sophisticated interactions between questions and passages, then use the pointer networks (Vinyals et al., 2015) to directly model the answers (Dhingra et al., 2017a; Wang and Jiang, 2017; Seo et al., 2017; Wang et al., 2017). These methods prove to be effective in existing close-domain datasets (Hermann et al., 2015; Hill et al., 2015; Rajpurkar et al., 2016).

More recently, open-domain RC has attracted increasing attention (Nguyen et al., 2016; Dunn et al., 2017; Dhingra et al., 2017b; He et al., 2017) and raised new challenges for question answering techniques. In these scenarios, a question is paired with multiple passages, which are often collected by exploiting unstructured documents or web data. Aforementioned approaches often rely on recurrent neural networks and sophisticated attentions, which are prohibitively time-consuming if passages are concatenated altogether. Therefore, some work tried to alleviate this problem in a coarse-to-fine schema. Wang et al. (2018a) combined a ranker for selecting the relevant passage

and a reader for producing the answer from it. However, this approach only depended on one passage when producing the answer, hence put great demands on the precisions of both components. Worse still, this framework cannot handle the situation where multiple passages are needed to answer correctly. In consideration of evidence aggregation, Wang et al. (2018b) proposed a re-ranking method to resolve the above issue. However, their re-ranking stage was totally isolated from the candidate extraction procedure. Being different from the re-ranking perspective, we propose a novel selection model to combine the information from all the extracted candidates. Moreover, with reinforcement learning, our candidate extraction and answer selection models can be learned in a joint manner. Trischler et al. (2016) also proposed a two-step extractor-reasoner model, which first extracted K most probable single-token answer candidates and then compared the hypotheses with all the sentences in the passage. However, in their work, each candidate was considered isolatedly, and their objective only took into account the ground truths compared with our RL treatment.

The training strategy employed in our paper is reinforcement learning, which is inspired by recent work exploiting it into question answering problem. The above mentioned coarse-to-fine framework (Choi et al., 2017; Wang et al., 2018a) treated sentence selection as a latent variable and jointly trained the sentence selection module with the answer generation module via RL. Shen et al. (2017) modeled the multi-hop reasoning procedure with a termination state to decide when it is adequate to produce an answer. RL is suitable to capture this stochastic behavior. Hu et al. (2018) merely modeled the extraction process, using F1 as rewards in addition to maximum likelihood estimation. RL was utilized in their training process, as the F1 measure is not differentiable.

3 Two-stage RC Framework

In this work, we mainly consider the open-domain extractive reading comprehension. In this scenario, a given question Q is paired with multiple passages $\mathbb{P} = \{P_1, P_2, ..., P_N\}$, based on which we aim to find out the answer A. Moreover, the golden answers are almost subspans shown in some passages in $\mathbb{P}$. Our main framework consists of two parts, which are: (1) extracting answer candidates $\mathbb{C} = \{C_1, C_2, ..., C_M\}$ from passages
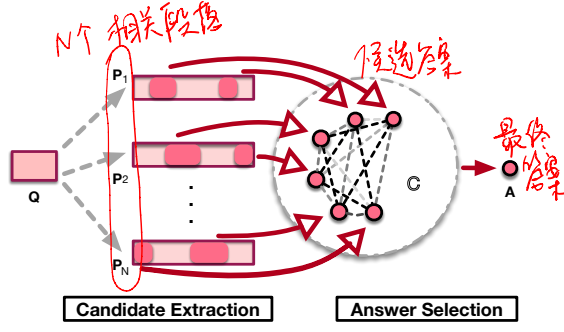
Figure 1: Two-stage RC Framework. The first part extracts candidates (denoted with circles) from all the passages. The second part establishes interactions among all these candidates to select the final answer. The different gray scales of dashed lines between candidates represent different intensities of interactions.

$\mathbb{P}$ and (2) selecting the final answer A from candidates $\mathbb{C}$. This process is illustrated in Figure 1. We design different models for each part and optimize them as a whole with joint reinforcement learning.

3.1 Candidate Extraction

We build candidate set $\mathbb{C}$ by independently extracting K candidates from each passage P_i according to the following distribution:

$$p(\mathbb{C}|Q,\mathbb{P}) = \prod_i^N p(\{C_{ij}\}_{j=1}^K | Q, P_i)$$
$$\mathbb{C} = \bigcup_{i=1}^N \{C_{ij}\}_{j=1}^K \tag{1}$$

where C_{ij} denotes the jth candidate extracted from the ith passage. K is set as a constant number in our formulation. Taking K as 2 for an example, we denote each probability shown on the right side of Equation 1 through sampling without replacement:

$$p(\{C_{i1}, C_{i2}\}) = p(C_{i1})p(C_{i2})/(1 - p(C_{i1}))$$
$$+ p(C_{i1})p(C_{i2})/(1 - p(C_{i2})) \tag{2}$$

where we neglect Q, P_i to abbreviate the conditional distributions in Equation 1.

Consequently, the basic block of our candidate extraction stage turns out to be the distribution of each candidate $P(C_{ij}|Q, P_i)$. In the rest of this subsection, we will elaborate on the model architecture concerning candidate extraction, which is displayed in Figure 2.
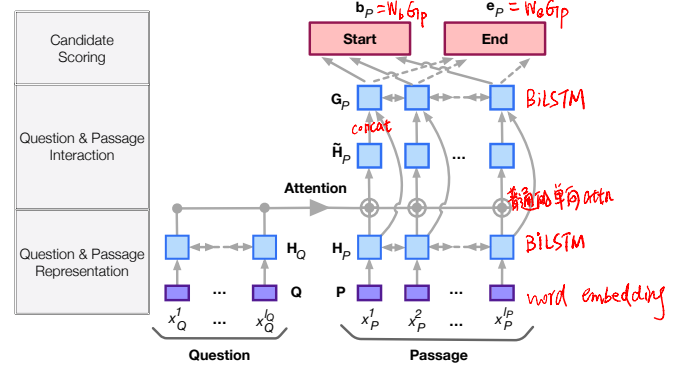

Figure 2: Candidate Extraction Model Architecture.

Question & Passage Representation Firstly, we embed the question $Q = \{x_Q^k\}_{k=1}^{l_Q}$ and its relevant passage $P = \{x_P^t\}_{t=1}^{l_P} \in \mathbb{P}$ with word vectors to form $\mathbf{Q} \in \mathbb{R}^{d_w \times l_Q}$ and $\mathbf{P} \in \mathbb{R}^{d_w \times l_P}$ respectively, where d_w is the dimension of word embeddings, l_Q and l_P are the length of Q and P.

We then feed $\mathbf{Q}$ and $\mathbf{P}$ to a bidirectional LSTM to form their contextual representations $\mathbf{H}_Q \in \mathbb{R}^{d_h \times l_Q}$ and $\mathbf{H}_P \in \mathbb{R}^{d_h \times l_P}$:

$$\mathbf{H}_Q = \text{BiLSTM}(\mathbf{Q})$$
$$\mathbf{H}_P = \text{BiLSTM}(\mathbf{P}) \tag{3}$$

Question & Passage Interaction Modeling the interactions between questions and passages is a critical step in reading comprehension. Here, we adopt the attention mechanism similar to (Lee et al., 2016) to generate question-dependent passage representation $\widetilde{\mathbf{H}}_P$. Assume $\mathbf{H}_Q = \{\mathbf{h}_Q^k\}_{k=1}^{l_Q}$, $\mathbf{H}_P = \{\mathbf{h}_P^t\}_{t=1}^{l_P}$, we have:

$$\alpha_{tk} = \frac{e^{\mathbf{h}_Q^k \cdot \mathbf{h}_P^t}}{\sum_{k=1}^{l_Q} e^{\mathbf{h}_Q^k \cdot \mathbf{h}_P^t}} \quad 1 \le k \le l_Q, 1 \le t \le l_P$$
$$\widetilde{\mathbf{h}}_P^t = \sum_{k=1}^{l_Q} \alpha_{tk} \mathbf{h}_Q^k \quad 1 \le t \le l_P$$
$$\widetilde{\mathbf{H}}_P = \{\widetilde{\mathbf{h}}_P^t\}_{t=1}^{l_P} \tag{4}$$

After concatenating two kinds of passage representations $\mathbf{H}_P$ and $\widetilde{\mathbf{H}}_P$, we use another bidirectional LSTM to get the final representation of every position in passage P as $\mathbf{G}_P \in \mathbb{R}^{d_g \times l_P}$:

$$\mathbf{G}_P = \text{BiLSTM}([\mathbf{H}_P; \widetilde{\mathbf{H}}_P]) \tag{5}$$

Candidate Scoring Then we use two linear transformations $\mathbf{w}_b \in \mathbb{R}^{1 \times d_g}$ and $\mathbf{w}_e \in \mathbb{R}^{1 \times d_g}$ to

calculate the begin and the end scores for each position:

$$\{b_P^t\}_{t=1}^{l_Q} = \mathbf{b}_P = \mathbf{w}_b \mathbf{G}_P$$
$$\{e_P^t\}_{t=1}^{l_Q} = \mathbf{e}_P = \mathbf{w}_e \mathbf{G}_P \tag{6}$$

At last, we model the probability of every sub-span in passage P as a candidate $C = \{x_P^t\}_{t=C_b}^{C_e}$ according to its begin and end position:

$$p(C|Q, P) = \frac{exp(b_P^{C_b} + e_P^{C_e})}{\sum_{k=1}^{l_P} \sum_{t=k}^{l_P} exp(b_P^k + e_P^t)} \tag{7}$$

In this definition, the probabilities of all the valid answer candidates are already normalized.

3.2 Answer Selection

As the second part of our framework, the answer selection model finds out the most probable answer by calculating $p(C|Q, \mathbb{P}, \mathbb{C})$ for each candidate $C \in \mathbb{C}$. The model architecture is illustrated in Figure 3.

Notably, selection model receives candidate set $\mathbb{C}$ as additional information. This more focused information allows the model to combine evidences from all the candidates, which would be useful for selecting the best answer.

For ease of understanding, we briefly describe the selection stage as follows. After being extracted from a single passage, a candidate borrows information from other candidates across different passages. With this global information, the passage is reread to confirm the correctness of the candidate further. The following are details about the selection model.

Question Representation Questions are fundamental for finding out the correct answer. As did for the extraction model, we embed the question Q with word vectors to form $\mathbf{Q} \in \mathbb{R}^{d_w \times l_Q}$. Then we use a bidirectional LSTM to establish its contextual representation:

$$\mathbf{S}_q = \text{BiLSTM}(\mathbf{Q}) \tag{8}$$

A max-pooling operation across all the positions is followed to get the condensed vector representation:

$$\mathbf{r}_q = \text{MaxPooling}(\mathbf{S}_q) \tag{9}$$

Passage Representation Assume the candidate C is extracted from the passage $P \in \mathbb{P}$. To be informed of C, we first build the representation of P. For every word in P, three kinds of features are utilized:
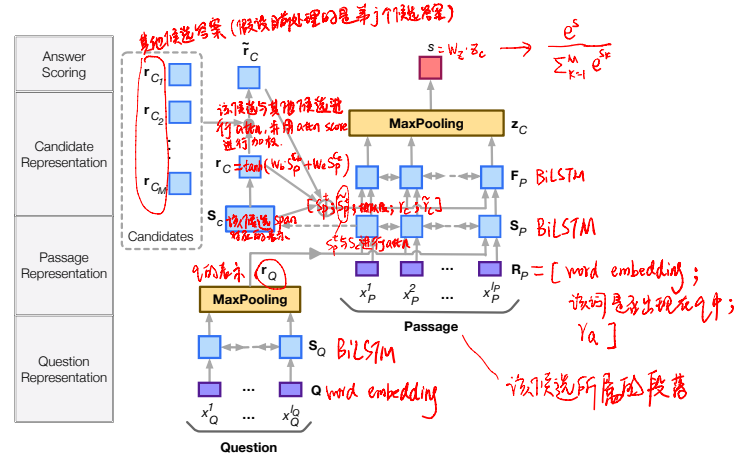

Figure 3: Answer Selection Model Architecture.

- Word embedding: each word expresses its basic feature with the word vector.

- Common word: the feature has value 1 when the word occurs in the question, otherwise 0.

- Question independent representation: the condensed representation $\mathbf{r}_q$.

With these features, information not only in Q but also in P is considered. By concatenating them, we get $\mathbf{r}_P^t$ corresponding to every position t in passage P. Then with another bidirectional LSTM, we fuse these features to form the contextual representation of P as $\mathbf{S}_P \in \mathbb{R}^{d_s \times l_P}$:

$$\mathbf{R}_P = \{\mathbf{r}_P^t\}_{t=1}^{l_P}$$
$$\mathbf{S}_P = \text{BiLSTM}(\mathbf{R}_P) \tag{10}$$

Candidate Representation Candidates provide more focused information for answer selection. Therefore, for each candidate, we first build its independent representation according to its position in the passage, then construct candidates fused representation through combination of other correlated candidates.

Given the candidate $C = \{x_P^t\}_{t=C_b}^{C_e}$ in the passage P, we extract its corresponding span from $\mathbf{S}_P = \{\mathbf{s}_P^t\}_{t=1}^{l_P}$ to form $\mathbf{S}_C = \{\mathbf{s}_P^t\}_{t=C_b}^{C_e}$ as its contextual encoding. Moreover, we calculate its condensed vector representation through its begin and end positions:

$$\mathbf{r}_C = \tanh(\mathbf{W}_b \mathbf{s}_P^{C_b} + \mathbf{W}_e \mathbf{s}_P^{C_e}) \tag{11}$$

where $\mathbf{W}_b \in \mathbb{R}^{d_c \times d_s}$, $\mathbf{W}_e \in \mathbb{R}^{d_c \times d_s}$.

To model the interactions among all the answer candidates, we calculate the correlations of the

candidate C, which is assumed to be indexed by j in $\mathbb{C}$, with others $\{C_m\}_{m=1,m\neq j}^M$ via attention mechanism:

$$V_{jm} = \mathbf{w}_v \tanh(\mathbf{W}_c \mathbf{r}_C + \mathbf{W}_o \mathbf{r}_{C_m}) \qquad (12)$$

where $\mathbf{W}_c \in \mathbb{R}^{d_c \times d_c}$, $\mathbf{W}_o \in \mathbb{R}^{d_c \times d_c}$ and $\mathbf{w}_v \in \mathbb{R}^{1 \times d_c}$ are linear transformations to capture the intensity of each interaction.

In this way, we form a *correlation matrix* $\mathbf{V} \in \mathbb{R}^{M \times M}$, where M is the total number of candidates. With the correlation matrix, for the candidate C, we normalize its interactions via a $softmax$ operation, which emphasizes the influence of stronger interactions:

$$\alpha_m = \frac{e^{V_{jm}}}{\sum_{m=1,m\neq j}^M e^{V_{jm}}} \qquad (13)$$

To take into account different influences of all the other candidates, it is sensible to generate a candidates fused representation according to the above normalized interactions:

$$\widetilde{\mathbf{r}}_C = \sum_{m=1,m\neq j}^M \alpha_m \mathbf{r}_{C_m} \qquad (14)$$

In this formulation, all the other candidates contribute their influences to the fused representation by their interactions with C, thus information from different passages is gathered altogether. In our experiments, this kind of information fusion is the key point for performance improvements.

Passage Advanced Representation As more focused information of the candidate C is available, we are provided with a better way to confirm its correctness by rereading its corresponding passage P. Specifically, we equip each position t in P with following advanced features:

- Passage contextual representation: the former passage representation $\mathbf{s}_P^t$.

- Candidate-dependent passage representation: replace $\mathbf{H}_Q$ with $\mathbf{S}_C$ and $\mathbf{H}_P$ with $\mathbf{S}_P$ in Equation 4 to model the interactions between candidates and passages to form $\widetilde{\mathbf{s}}_P^t$.

- Candidate related distance feature: the relative distance to the candidate C can be a reference of the importance of each position.

- Candidate independent representation: use $\mathbf{r}_C$ to consider the concerned candidate C.

- Candidates fused representation: use $\widetilde{\mathbf{r}}_C$ to consider all the other candidates interacting with the concerned candidate C.

With these features, we capture the information from the question, the passages and all the candidates. By concatenating them, we get $\mathbf{u}_P^t$ in every position in the passage P. Combining these features with a bidirectional LSTM, we get:

$$\begin{aligned} \mathbf{U}_P &= \{\mathbf{u}_P^t\}_{t=1}^{l_P} \\ \mathbf{F}_P &= \text{BiLSTM}(\mathbf{U}_P) \end{aligned} \qquad (15)$$

Answer Scoring At last, the max pooling of each dimension of $\mathbf{F}_P$ is performed, resulting in a condensed vector representation, which contains all the concerned information in a candidate:

$$\mathbf{z}_C = \text{MaxPooling}(\mathbf{F}_P) \qquad (16)$$

The final score of this candidate as the answer is calculated via a linear transformation, which is then normalized across all the candidates:

$$\begin{aligned} s &= \mathbf{w}_z \mathbf{z}_C \\ p(C|Q, \mathbb{P}, \mathbb{C}) &= \frac{e^s}{\sum_{k=1}^M e^{s_k}} \end{aligned} \qquad (17)$$

3.3 Joint Training with RL

In our formulation, the answer candidate set influences the result of answer selection to a large extent. However, with only golden answers provided in the training data, it is not apparent which candidates should be considered further.

To alleviate the above problem, we treat candidate extraction as a latent variable, jointly train the extraction model and the selection model with reinforcement learning. Formally, in the extraction and selection stages, two kinds of actions are modeled. The action space for the extraction model is to select from different candidate sets, which is formulated by Equation 1. The action space for the selection model is to select from all extracted candidates, which is formulated by Equation 17. Our goal is to select the final answer that leads to a high reward. Inspired by Wang et al. (2018a), we define the reward of a candidate to reflect its accordance with the golden answer:

$$r(C, A) = \begin{cases} 2 & \text{if } C == A \\ f1(C, A) \in [0,1] & \text{else if } C \cap A \neq \varnothing \\ -1 & \text{else} \end{cases} \qquad (18)$$

	#q(train)	#q(dev)	#q(test)	#p
Quasar-T	28,496	3,000	3,000	100
SearchQA	99,811	13,893	27,247	50

Table 2: The statistics of our experimental datasets. #q represents the number of questions for each split of the datasets. #p is the number of passages for each question.

where $f1(.,.) \in [0,1]$ is the function to measure word-level F1 score between two sequences. Incorporating this reward can alleviate the overstrict requirements set by traditional maximum likelihood estimation as well as keep consistent with our evaluation methods in experiments.

The learning objective becomes to maximize the expected reward modeled by our framework, where θ stands for all the parameters involved:

$$L(\theta) = -E_{\mathbb{C} \sim P(\mathbb{C}|Q,\mathbb{P})}[E_{C \sim P(C|Q,\mathbb{P},\mathbb{C})} r(C,A)]$$
$$= -E_{\mathbb{C} \sim P(\mathbb{C}|Q,\mathbb{P})}[\sum_C P(C|Q,\mathbb{P},\mathbb{C}) r(C,A)]$$

(19)

Following REINFORCE algorithm, we approximate the gradient of the above objective with a sampled candidate set, $\mathbb{C} \sim P(\mathbb{C}|Q,\mathbb{P})$, resulting in the following form:

$$\nabla L(\theta) \approx - \sum_C \nabla P(C|Q,\mathbb{P},\mathbb{C}) r(C,A)$$
$$-\nabla \log P(\mathbb{C}|Q,\mathbb{P})[\sum_C P(C|Q,\mathbb{P},\mathbb{C}) r(C,A)]$$

(20)

4 Experiments

4.1 Datasets

We evaluate our models on two publicly available open-domain RC datasets, which are commonly adopted in related work.

Quasar-T (Dhingra et al., 2017b) consists of 43,000 open-domain trivia questions and corresponding answers obtained from various internet sources. Each question is paired with 100 sentence-level passages retrieved from ClueWeb09 (Callan et al., 2009) based on Lucene. **SearchQA** (Dunn et al., 2017) starts from existing question-answer pairs, which are crawled from J!Archive, and is augmented with text snippets retrieved by Google, resulting in more than 140,000 question-answer pairs with each pair having 49.6 snippets on average.

The detailed statistics of these two datasets is shown in Table 2.

4.2 Model Settings

We initialize word embeddings with the 300-dimensional Glove vectors[1]. All the bidirectional LSTMs hold 1 layer and 100 hidden units. All the linear transformations take the size of 100 as output dimension. The common word feature and the candidate related distance feature are embedded with vectors of dimension 4 and 50 respectively. By default, we set K as 2 in Equation 1, which means each passage generates two candidates based on the extraction model.

For ease of training, we first initialize our models by maximum likelihood estimation and fine-tune them with RL. The similar training strategy is commonly employed when RL process is involved (Ranzato et al., 2015; Li et al., 2016a; Hu et al., 2018). To pre-train the extraction model, we only use passages containing ground truths as training data. The log likelihood of Equation 7 is taken as the training objective for each question and passage pair. After pre-training the extraction model, we use it to generate two top-scoring candidates from each passage, forming the training data to pre-train our selection model, and maximize the log likelihood of the Equation 17 as our second objective. In pre-training, we use the batch size of 30 for the extraction model, 20 for the selection model and RMSProp (Tieleman and Hinton, 2012) with an initial learning rate of 2e-3. In fine-tuning with RL, we use the batch size of 5 and RMSProp with an initial learning rate of 1e-4. Also, we use a dropout rate of 0.1 in each training procedure.

4.3 Experimental Results

In addition to results of previous work, we add two baselines to demonstrate the effectiveness of our framework. The first baseline only applies the extraction model to score the answers, which is aimed at explaining the importance of the selection model. The second one only uses the pre-trained extraction model and selection model to illustrate the benefits from our joint training schema.

The often used evaluation metrics for extractive RC are exact match (EM) and F1 (Rajpurkar et al., 2016). The experimental results on Quasar-T and SearchQA are shown in Table 3.

	Quasar-T		SearchQA	
	EM	F1	EM	F1
GA (Dhingra et al., 2017a)	26.4	26.4	-	-
BIDAF (Seo et al., 2017)	25.9	28.5	28.6	34.6
AQA (Buck et al., 2018)	-	-	38.7	45.6
R^3 (Wang et al., 2018a)	35.3	41.7	49.0	55.3
Re-Ranker (Wang et al., 2018b)				
Strength-Based Re-Ranker (Probability)	36.1	42.4	50.4	56.5
Strength-Based Re-Ranker (Counting)	37.1	46.7	54.2	61.6
Coverage-Based Re-Raner	40.6	49.1	53.6	60.6
Full Re-Ranker	42.3	49.6	57.0	63.2
Our Methods				
Extraction Model	35.4	41.6	44.7	51.2
Extraction + Selection (Isolated Training)	41.6	49.5	49.7	56.6
Extraction + Selection (Joint Training)	**45.9**	**53.9**	**58.3**	**64.2**

Table 3: Experimental results on the test set of Quasar-T and SearchQA. Full re-ranker is the ensemble of three different re-rankers in (Wang et al., 2018b).

As seen from the results on Quasar-T, our quite simple extraction model alone almost reaches the state-of-the-art result compared with other methods without re-rankers. The combination of the extraction and selection models exceeds our extraction baseline by a great margin, and also results in performance surpassing the best single re-ranker in (Wang et al., 2018b). This result illustrates the necessity of introducing the selection model, which incorporates information from all the candidates. In the end, by joint training with RL, our method produces better performance even compared with the ensemble of three different re-rankers.

On SearchQA, we find that our extraction model alone performs not that well compared with the state-of-the-art model without re-rankers. However, the improvement brought by our selection model isolatedly or jointly trained still demonstrates the importance of our two-stage framework. Not surprisingly, comparing the results, our isolated training strategy still lags behind the single re-ranker proposed in (Wang et al., 2018b), partly because of the deficiency with our extraction model. However, uniting our extraction and selection models with RL makes up the disparity, and the performance surpasses the ensemble of three different re-rankers, let alone the result of any single re-ranker.

4.4 Further Analysis

Effect of Features in Selection Model As the incorporation of the selection model improves the overall performance significantly, we conduct ab-

Quasar-T	EM	F1
Extraction + Selection (Joint Training)	**45.9**	**53.9**
-question representation	42.5	50.5
-question and passage common words	41.0	48.7
-candidate independent representation r_c	44.5	53.3
-candidate related distance feature	44.7	53.0
-candidate dependent passage representation $\hat{r}_c$	44.4	52.3
-candidates fused representation $\tilde{r}_c$	39.2	45.8

Table 4: Ablation results concerning the selection model on the test set of Quasar-T. Obviously, candidates fused representation is the most evident feature when modeling the answer selection procedure.

lation analysis on the Quasar-T to prove the effectiveness of its major components. As shown in Table 4, all these components modeling the selection procedure play important roles in our final architecture.

Specifically, introducing the independent representation of the question and its common words with the passage seems an efficient way to consider the information of questions, which is consistent with previous work (Li et al., 2016b; Chen et al., 2017).

As for features related to candidates, the incorporation of the candidate independent information contributes to the final result more or less. These features include candidate-dependent passage representation, candidate independent representation and candidate related distance feature.

Most importantly, the candidates fused representation, which combines the information from

Q	Cocktails : Rum , lime , and cola drink make a _____ .
A	**Cuba Libre**
P_1	In Nicaragua , when it is mixed using Flor de Ca a -LRB- the national brand of rum -RRB- and cola , it is called a **Nica Libre** .
P_2	The drink ... **Daiquiri** The custom of mixing lime with rum for a cooling drink on a hot Cuban day has been around a long time .
P_3	If you only learn to make two cocktails , the **Manhattan** should be one of them .
P_4	**Daiquiri** Cocktail recipe for a Daiquiri , a classic rum and lime drink that every bartender should know .
P_5	Hemingway Special **Daiquiri** : Daiquiris are a family of cocktails whose main ingredients are rum and lime juice .
P_6	In the Netherlands the drink is commonly called **Baco** , from the two ingredients of Bacardi rum and cola .
P_7	A homemade Cuba Libre Preparation To make a **Cuba Libre** properly , fill a highball glass with ice and half fill with cola .
P_8	**Bacardi** Cocktail Cocktail recipe for a Bacardi Cocktail , a classic cocktail of Bacardi rum , lemon or lime juice and grenadine Roy Rogers -LRB- non-alcoholic -RRB- Cocktail recipe for a Roy Rogers ,
P_9	**Margarita** Cocktail recipe for a Margarita , a popular refreshing tequila and lime drink for summer .
P_{10}	The difference between the **Cuba Libre** and Rum is a lime wedge at the end .

Table 5: An example from Quasar-T to illustrate the necessity of fused information. Candidates extracted from passages are in a **bold** font. To correctly answer the question, information in P_7 and P_{10} should be combined.

all the candidates, demonstrates its indispensable role in candidate modeling, with a performance drop of nearly 8% when discarded. This phenomenon also verifies the necessity of our extract-then-select procedure, showing the importance of combining information scattered in different text pieces when picking out the final answer.

Example for Candidates Fused Representation We conduct a case study to demonstrate the importance of candidates fused information further. In Table 5, each candidate only partly matches the description of the question in its independent context. To correctly answer the question, information in P_7 and P_{10} should be combined. In experiments, our selection model provides the correct answer, while the wrong candidate "Daiquiri", a different kind of cocktail, is selected if candidates fused representation is discarded. The attention map established when modeling the fusion of candidates (corresponding to Equation 13) in this example is illustrated in Figure 4, in which we can see the interactions among all the candidates from different passages. In this figure, it is obvious that the interaction of "Cuba Libre" in P_7 and P_{10} is the key point to answer the question correctly.

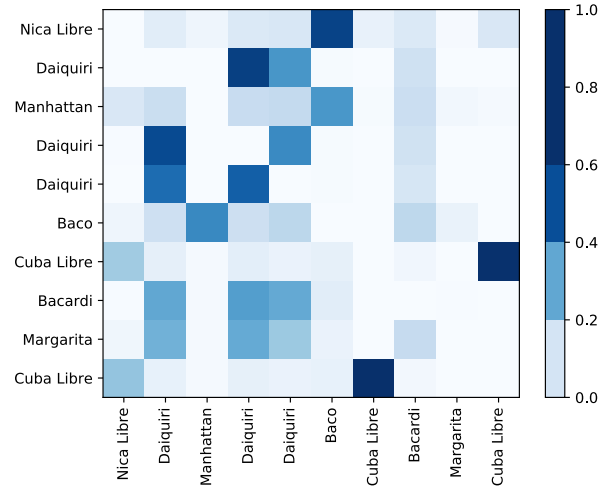**Effect of Candidate Number** The candidate extraction stage takes an important role to decide

Figure 4: The attention map generated when modeling candidates fused representations for the example in Table 5.

Quasar-T	EM	F1
K=1	43.9	52.4
K=2	45.9	53.9
K=3	45.8	53.9

Table 6: Different number of extracted candidates results in different final performance on the test set of Quasar-T.

what information should be focused on further. Therefore, we also test the influence of different K when extracting candidates from each passage. The results are shown in Table 6. Taking $K = 1$ degrades the performance, which conforms to the expectation, as the correct candidates become less in this stricter situation. However, taking $K = 3$ can not improve the performance further. Although a larger K means a higher possibility to include good answers, it raises more challenges for the selection model to pick out the correct one from candidates with more varieties.

5 Conclusion

In this paper, we formulate the problem of RC as a two-stage process, which first generates candidates with an extraction model, then selects the final answer by combining the information from all the candidates. Furthermore, we treat candidate extraction as a latent variable and jointly train these two stages with RL. Experiments on public open-domain RC datasets Quasar-T and SearchQA show the necessity of introducing the

selection model and the effectiveness of fusing candidates information when modeling. Moreover, our joint training strategy leads to significant improvements in performance.

Acknowledgments

References

Christian Buck, Jannis Bulian, Massimiliano Ciaramita, Andrea Gesmundo, Neil Houlsby, Wojciech Gajewski, and Wei Wang. 2018. Ask the right questions: Active question reformulation with reinforcement learning. In *ICLR*.

Jamie Callan, Mark Hoy, Changkuk Yoo, and Le Zhao. 2009. Clueweb09 data set.

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*. Association for Computational Linguistics, Vancouver, Canada, pages 1870–1879.

Eunsol Choi, Daniel Hewlett, Jakob Uszkoreit, Illia Polosukhin, Alexandre Lacoste, and Jonathan Berant. 2017. Coarse-to-fine question answering for long documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Vancouver, Canada, pages 209–220.

Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William Cohen, and Ruslan Salakhutdinov. 2017a. Gated-attention readers for text comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Vancouver, Canada, pages 1832–1846.

Bhuwan Dhingra, Kathryn Mazaitis, and William W Cohen. 2017b. Quasar: Datasets for question answering by search and reading. *arXiv preprint arXiv:1707.03904* .

Matthew Dunn, Levent Sagun, Mike Higgins, Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179* .

Wei He, Kai Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, Xuan Liu, Tian Wu, and Haifeng Wang. 2017. Dureader: a chinese machine reading comprehension dataset from real-world applications. *arXiv preprint arXiv:1711.05073* .

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. pages 1693–1701.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2015. The goldilocks principle: Reading children's books with explicit memory representations. *arXiv preprint arXiv:1511.02301* .

Minghao Hu, Yuxing Peng, and Xipeng Qiu. 2018. Reinforced mnemonic reader for machine comprehension. In *IJCAI*.

Kenton Lee, Shimi Salant, Tom Kwiatkowski, Ankur Parikh, Dipanjan Das, and Jonathan Berant. 2016. Learning recurrent span representations for extractive question answering. *arXiv preprint arXiv:1611.01436* .

Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. 2016a. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, pages 1192–1202.

Peng Li, Wei Li, Zhengyan He, Xuguang Wang, Ying Cao, Jie Zhou, and Wei Xu. 2016b. Dataset and neural recurrent sequence labeling model for open-domain factoid question answering. *arXiv preprint arXiv:1607.06275* .

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016)*.

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*. Association for Computational Linguistics, Austin, Texas, pages 2383–2392.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732* .

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *ICLR*.

Yelong Shen, Po-Sen Huang, Jianfeng Gao, and Weizhu Chen. 2017. Reasonet: Learning to stop reading in machine comprehension. In *Proceedings*

of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pages 1047–1055.

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning* 4(2):26–31.

Adam Trischler, Zheng Ye, Xingdi Yuan, Philip Bachman, Alessandro Sordoni, and Kaheer Suleman. 2016. Natural language comprehension with the epireader. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, pages 128–137.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. pages 2692–2700.

Shuohang Wang and Jing Jiang. 2017. Machine comprehension using match-lstm and answer pointer. In *ICLR*.

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

Shuohang Wang, Mo Yu, Jing Jiang, Wei Zhang, Xiaoxiao Guo, Shiyu Chang, Zhiguo Wang, Tim Klinger, Gerald Tesauro, and Murray Campbell. 2018b. Evidence aggregation for answer re-ranking in open-domain question answering. In *ICLR*.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. volume 1, pages 189–198.