Hierarchical answer selection framework for Multi-passage Machine Reading Comprehension

Zhaohui Li, Jun Xu*, YanYan Lan, Jiafeng Guo, Yue Feng, and Xueqi Cheng

CAS Key Lab of Network Data Science and Technology
Institute of Computing Technology, Chinese Academy of Sciences, China
{lizhaohui,fengyue}@software.ict.ac.cn,{junxu,lanyanyan,guojiafeng,cxq}@ict.ac.cn

Abstract. Machine reading comprehension (MRC) on real web data, which means finding answers from a set of candidate passages for a question, is a quite arduous task in natural language processing. Most state-of-the-art approaches select answers from all passages or from only one single golden paragraph, which may cause the overlapping information and the lack of key information. To address these problems, this paper proposes a hierarchical answer selection framework that can select **main content** from a set of passages based on the question, and predict final answer within this main content. Specifically, three main parts are employed in this pipeline: First, the passage selection model uses a classification mechanism to select passages by passages content and title information which is not fully used in other models; Second, a key sentences sequence selection mechanism is modeled by Markov-Decision-Process (MDP) in order to gain as much as effectual answer information as possible; Finally, a match-LSTM model is employed to extract the final answer from the selected main content. These three modules that shared the same attention-based semantic network and we conduct experimental on DuReader search dataset. The results show that our framework outperforms the baseline by a large margin.

Keywords: Machine reading comprehension · Markov decision process· Reinforcement learning· Natural language process

1 Introduction

In NLP community, Machine reading comprehension (MRC) task which aims to endow the machine with the ability of answering questions after reading a passage or a set of passages has been popular in recent years. At the outset, MRC task dataset was cloze task [2, 3], and then was multiple-choice exam [4] which collect multiple-choice questions from exams. Finally, it evolved into answering more complex questions based on single or multiple documents [8, 5, 1]

Recently, several significant successes in answer extraction on single passage [13, 9, 6] help MRC research barging its way into a high level, especially some methods have outperformed human annotators on the SQuAD dataset [8].

* Corresponding author: Jun Xu

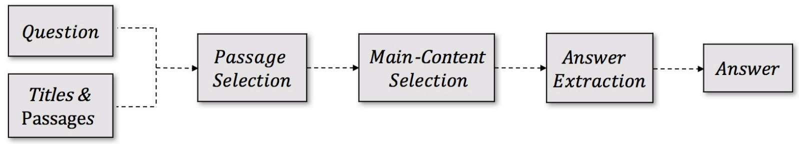

Fig. 1. Overview pipeline of question based hierarchical answer selection framework.

These methods formulate MRC task as predicting the start and end positions of the answer in the passage. However, this milestone is not strong enough when considering the real web data. Therefore, some realistic MRC dataset [5, 1] are released based on search engines. For every question, the MRC model need to consider all passages related to the question so as to get the result. To handle these complex MRC tasks, most existing methods follow the single passage extraction based approach in the SQuAD dataset, and several novel methods such as v-net [16] and s-net [12] have achieved high performance in MS-MARCO [5] leaderboard.

However, when it comes to DuReader task [1], these methods may not work well for they are ether doing verification between candidate answers or synthesizing candidate answers which generated in a single passage extraction. Compared with MS-MARCO, DuReader is much larger and complex, and each question has multiple candidate passages. DuReader example[1] illustrates that DuReader has more unrelated and overlapping information, some are even incorrect. Moreover, the simple span-prediction in a continuous passage text is unlikely to work well because the components of the answers are relatively far from source passage (not in a single continuous text span). Thus, it is unreasonable to get continuous answer spans from all source passages.

In order to solve this intrinsic challenge for multi-passage MRC, inspired by human MC process, this paper proposes a hierarchical answer selection framework which is designed for MRC on real web data. It can gets **Main Content**, that is the crucial information from the passages according the question' s content, and then the final answer will be selected within this main content by the state-of-the-art single passage MRC method.

The general framework of this model is demonstrated in Figure 1, which consists of one encoding module and other three function modules. Specifically, the encoding module employs the Bi-Directional Attention mechanism [9] to obtain a question-aware word representation. And then three other modules are implemented. First, we presume that the *is_selected* labeled passage contain all of the answer information, which will be testified in the latter discussion experiment. And the whole passages content is represented by the combination of the title of passage and the sentence sequence in passage. Therefore, a candidate passage selection model can be trained according to *is_selected* label. Second, a key-sentence selection model is developed using *Markov-Decision-Process*(MDP) [11,

[1] See examples at https://ai.baidu.com/broad/introduction?dataset=dureader.

7] and *Policy Gradient* [11] to predict a sequence of key sentences, which is called **Main Content** in our approach. Finally, a state-of-the-art model [13] is employed to obtain a answer from the Main Content.

We conduct statistics and experiments on the DuReader [1] datasets. The results show that our hierarchical answer MRC model outperforms the baseline models. According these results, several contributions can be summarized: 1) We formulate the candidate answer selection as a MDP model, which performs well in large search engine dataset. 2) We first use the title of passage to filter the passages, which is a complement for passage representation. 3) We propose a real world MRC task pipeline based on these modules, which can tackle the redundant information problems.

2 Related Work

In resent years, machine reading comprehension has gained more and more attention, and existing main-steam works are building data-drive, end-to-end neural network models.

At first, some datasets for studying machine comprehension were created in Cloze style, and the task is to predict the missing word [2, 3]. Then instead of Cloze style, a significant dataset, the SQuAD dataset [8] was also created by human annotators, people have to predict answers from given passages. Mainstream methods are all boundary models [9, 13, 17], that is treating MRC as extracting answer span from the given passage, which is usually achieved by predicting the start and end position of the answer.

Recently, two multi-passage real world web MRC dataset released: DuReader [1] and MS-MARCO [5]. Some studies [10, 15] concatenate those passages and employ the same models designed for single-passage MRC. On the other hand, more and more latest studies start to design special methods for multiple passages. For example, Wang(2017) [14] implemented a pipeline method that ranking the passages first and then extract answer from the selected passages. Tan [12] also use the similar method, which treats the passage ranking as an auxiliary task that can be trained jointly with the reading comprehension model. In comparison with these passage selection model, our approach has a unique hierarchical framework, which use the title information to select the candidate passages and then a sentence-level MDP filter is employed to obtain the main content. In addition, there are also some joint training end-to-end models, such as answer verification method v-net [16], that is, extract answers from passages and then do answer verification process. Actually, answer verification method and our model have a same motivation, that is reduce the overlapping information. However, we implemented our framework as three separate step, while they trained their model jointly.

3 Hierarchical answer selection framework

Figure 1 shows the architecture of our hierarchical MRC model which is composed of an attention encoding layer and three selection modules including Passage Selection module, Main Content Selection module and Answer Span Prediction module. The Attention Encoding module uses Bi-Directional Attention mechanism [9] to obtain a question-aware representation for each passage and title. And then is the answer selection model: First, the passage selection module is determined by titles and passages; Second, the main content selection module is modeled by MDP method. Finally, the Answer Span Prediction module which is a state-of-the-art model [13] is employed to obtain a continuous answer span from the Main Content.

3.1 Attention Encoding Module

For every word in data set, its embedding is assigned at first. Then following Seo [9], we calculate the word-level, sentence level and passage level by attention mechanism. Given a question $\mathbf{Q}$, a set of passages $\mathbf{P}$ and the passages' title $\mathbf{T}$, we first map these words with their word-level embeddings. And then three Bi-directional LSTM are employed to get the new contextual representation: $\mathbf{u}_t^Q$,$\mathbf{u}_t^{P_i}$,and $\mathbf{u}_t^{T_i}$, which represent vectors of the $\mathbf{t}^{th}$ word in $\mathbf{Q}$, $\mathbf{P}_i$, $\mathbf{T}_i$ respectively.

After getting the base representation of each word, one essential step is to endow these passages and titles with their question's information. We conduct these $\mathbf{P}_i-\mathbf{Q}$ and $\mathbf{T}_i-\mathbf{Q}$ Matching with Attention Flow Layer (Seo et al., 2016), so the $\mathbf{T}_i/\mathbf{P}_i$**-to-**$\mathbf{Q}$ attention can be easily obtained. Then strictly following Seo et al., 2016, we calculate the question-aware passage and title word representations. And $\{\mathbf{P_i}\}$ and $\{\mathbf{T_i}\}$, and $\{\mathbf{Q}\}$ representation and their sentences representations can be easily calculate by $BiLSTM$ mechanism with the word representations.

3.2 Passage Selection Module

In this part, we formulate the passage representation as the combination of passage title and passage context. So the real representation of each passage is:

$$\tilde{P}_i = [T_i; P_i] \tag{1}$$

where [;] is vector concatenation across row, T_i is the representation of passage title and P_i is the representation of passage content. After that, with these results $\tilde{P}_i$ and Q, we use a simplified classification model to calculate the probability p_i of whether a passage is *is_selected* labeled:

$$p_i = \sigma(Q^T W \tilde{P}_i + b) \tag{2}$$

where $\mathbf{W} \in \mathbb{R}^{d \times d}$ and $b \in \mathbb{R}$, and we use the question-aware passage representation $\tilde{P}_i$ and Q. Given the set of N training samples, each question contains $|P|$

passages with ground truth of *is_selected* label: $(y_1, y_2, ..., y_N)$, it can be trained by minimizing negative as the averaged cross entropy loss:

$$\mathcal{L}_s = -\frac{1}{\mathbf{N}}\frac{1}{|\mathbf{P}|}\sum_{j=1}^{\mathbf{N}}\sum_{i=1}^{|\mathbf{P}|}\left[y_i \log p_i + (1-y_i)(1-\log p_i)\right] \quad (3)$$

3.3 Main Content Selection Module

After getting the candidate passages for each question, we employ Markov Decision Process(MDP) to model answer candidate sentence selection. The difficulties lie in how to formalize MRC under the MDP framework. In addition, how to convert the reference answer to the supervision information that can be utilized by MDP model is also a tough question.

MDP formulation of sentence selection In the encoding module, we have got the representation of each passage $\mathbf{P}$ and its sub-sentence list $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_M\} \subseteq \mathcal{X}$, and $\mathcal{X}$ is the set of all sentences. The goal of sentence selection is to construct a model which can give a set of candidate sentences set so that the following model can find the best answer from this sentence set.The training of a sentence selection model, thus, can be consider as the learning parameters in a MDP model, in which each step corresponds to a selected candidate answer sentence. The states, actions, rewards, transitions, and policy of MDP are set as:

States $\mathcal{S}$: State are designed at step t as a triple $s_t = [\mathbf{Q}, \mathcal{Z}_t, X_t]$, where $\mathbf{Q}$ is the preliminary representation of the question; $\mathcal{Z}_t = \{\mathbf{x}_{(n)}\}_{n=1}^{t}$ is the sequence of t preceding sentences, where $\mathbf{x}_{(n)}$ is the $\mathbf{t}^{th}$ sentence in the main content sequence; X_t is the set of candidate sentences. At the beginning $(t = 0)$, the state is initialized as $s_0 = [\mathbf{q}, \emptyset, X]$, where $\emptyset$ is the empty sequence and X contains all of the M sentences in all the candidate passages. Note that we require that each sentence set ends with a special end-of-content symbol $\langle EOS \rangle$, which enables the model to define a distribution over sequences of all possible lengths.

Actions $\mathcal{A}$: At each time step t, the $\mathcal{A}(s_t)$ is the set of actions the agent can choose, each corresponds to a sentence from X_t. That is, the action $a_t \in \mathcal{A}(s_t)$ at the time step t selects a sentence $\mathbf{x}_{m(a_t)} \in X_t$ for the main content sequence, where $m(a_t)$ is the index of the sentence selected by a_t. **Transition T:** The transition function $T : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is defined as follows:

$$\begin{aligned}\mathbf{s}_{t+1} = T(s_t, a_t) &= T([\mathbf{Q}, \mathcal{Z}_t, X_t], a_t) \\ &= \left[\mathbf{Q}, \mathcal{Z}_t \oplus \{\mathbf{x}_{m(a_t)}\}, X_t \setminus \{\mathbf{x}_n\}_{\mathbf{o}}^{a_t}\right],\end{aligned} \quad (4)$$

where $\oplus$ appends $\mathbf{x}_{m(a_t)}$ to $\mathcal{Z}_t$ and $\setminus$ removes $\mathbf{x}_n\}_{\mathbf{o}}^{a_t}$ from X_t, o is the first sentence number in X_t. At each time step t, based on state s_t the system chooses an action a_t. Then, the system moves to step $t+1$ and the system transits to a new state s_{t+1}: The selected sentence is appended to the end of $\mathcal{Z}_t$, generating

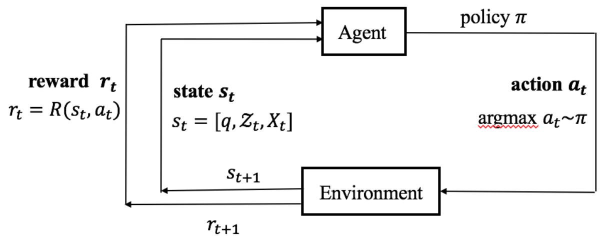

Fig. 2. The Agent-Environment model in MDP.

a new sentence sequence, and the sentences in the precede place of the selected sentence at step t are removed from the candidate set: $X_{t+1} = X_t \setminus \{\mathbf{x}_n\}_{\mathbf{o}}^{a_t}$.

Reward R**:** The reward can be considered as the evaluation of the information quality of the main content sequence, for we aimed at maximize the information related to the answer in this main content selection module. Since we need to maximize the information, we define the reward function on the combination of F1 and Rouge-L:

$$\mathbf{r}\,(t) = \frac{1}{2} * [\text{F1}\,(t) + \text{Rouge-L}\,(t)] \tag{5}$$

where t is the t^{th} sentence in the main content, and the position 0 is defined as zero. Then the reward function caused by choosing the action a_t is:

$$\mathbf{R}\,(s_t, a_t) = \mathbf{r}\,(\mathbf{t+1}) - \mathbf{r}\,(\mathbf{t}) \tag{6}$$

Policy function p: The policy $\mathbf{p}(s)$ defines a function that takes the state as input and output a distribution over all of the possible actions $a \in \mathcal{A}(s)$. Specifically, each probability in the distribution is a normalized function whose input is the bilinear product of the LSTM function and the selected sentence:

$$p(a|s) = \frac{\exp\left\{\mathbf{x}_{m(a)}^T \mathbf{U}_p \text{ LSTM}(s)\right\}}{\sum_{a' \in \mathcal{A}(s)} \exp\left\{\mathbf{x}_{m(a')}^T \mathbf{U}_p \text{ LSTM}(s)\right\}} \tag{7}$$

where The deep neural network model $LSTM : \mathcal{S} \to \mathbb{R}^L$ maps a state to a real vector where L is the number of dimensions. Given $s = [\mathbf{Q}, \mathcal{Z} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_t\}, X_t]$, where $\mathbf{x}_k (k = 1, \cdots, t)$ is the sentence at k-th position and represented with its embedding. Thus, the policy function $\mathbf{p}(s)$ is:

$$\mathbf{p}(s) = \langle p(a_1|s), \cdots, p(a_{|\mathcal{A}(s)|}|s)\rangle. \tag{8}$$

Learning with policy gradient The model has some parameters to learn, we donate as $\mathbf{\Theta}$. In this training phase, suppose there are N training questions

Algorithm 1 MDP-MCS

Input: Training set $D = \{(\mathbf{Q}^{(n)}, X^{(n)}, A^{(n)})\}_{n=1}^{N}$, learning rate η, dropout keep rate d, and value function R

Output: $\boldsymbol{\Theta}$

```
Initialize Θ ← random values in [−1, 1]
repeat
  for all (Q, X, A) ∈ D do
    (s_0, a_0, r_1, ··· , s_{M−1}, a_{M−1}, r_M) ← SampleEpisode(Θ, Q, X, A, R){Algorithm(2)}
    for t = 0 to M − 1 do
      G_t ← Σ_{k=0}^{M−1−t} r_{t+k+1}{Equation(17)}
      Θ ← Θ − ηG_t ∇_Θ log a_t|s_t; Θ{According to Equation(18)}
    end for
  end for
until converge
return Θ
```

$\{(Q^{(n)}, X^{(n)}, A^{(n)})\}_{n=1}^{N}$, where $A^{(n)}$ denotes the reference answers of the question. Inspired by the RL algorithm policy gradient, we devised a novel algorithm which can learn the parameters toward the Main Content selection Model. It is referred as MDP-MCS and shown in Algorithm 1. The Algorithm 2 shows the procedure of sampling a sentence episode for Algorithm 1. The definition of long-term return G_t is crucial important, for it equals the ground truth in this task. So we define the discounted sum of rewards from position t as G_t:

$$\mathbf{G}_t = \sum_{k=0}^{M-1-t} \lambda \mathbf{r}_{t+k+1} \tag{9}$$

where M is the length of the selected sentence episode and λ is the discount rate of policy gradient. Note that in our model, $\lambda = 1$. And using the long-term return G_t, we can calculate the loss of each iteration, an sentence episode(consisting a sequence of states, actions, and rewards) is sampled according to current policy.

$$\mathcal{L}(E) = -\sum_{t=1}^{|E|} \left(\sum_{a \in \mathcal{A}(s_t)} G_t(a|s_t) \log \frac{1}{p(a|s_t)} \right). \tag{10}$$

Testing method After the training phase, we can get an agent that can select main content sentences from passages according to its policy function. Specifically, given a question $\mathbf{Q}$, a set of M candidate sentence X, the system state is initialized as $s_0 = [\mathbf{Q}, \mathcal{Z}_0 = \emptyset, X_0 = X]$. Then, at each of the steps $t = 0, \cdots, M-1$, the agent receives the state $s_t = [\mathbf{Q}, \mathcal{Z}_t, X_t]$ and searches the policy $\boldsymbol{\pi}$, on the basis of policy function $\mathbf{p}$. Then, it chooses an action a according to $\boldsymbol{\pi}$. Moving to the next step $t+1$, the state becomes $s_{t+1} = [\mathbf{q}, \mathcal{Z}_{t+1}, X_{t+1}]$. The process is repeated until the candidate set becomes empty.

Algorithm 2 Sample Episode

Input: Parameters $\mathbf{\Theta}$, question Q, candidate passage sentences X, reference answers A, and value function R

Output: A selected sentence Episode

Initialize $\mathbf{\Theta} \leftarrow$ random values in $[-1, 1]$
$s \leftarrow [\mathbf{Q}, \emptyset, X]$
$E = ()$ {empty episode}
while $\mathbf{x}_{m(a)}$is not $< EOS >$ **do**
  $\mathcal{A} \leftarrow \mathcal{A}(s)$ {Possible actions according to X in state s}
  for all $a \in \mathcal{A}$ **do**
    $P(a) \leftarrow \pi(a|s; \mathbf{\Theta}$
  end for
  $\tilde{a} = \arg\max_{a \in \mathcal{A}(s)} \pi(a|s)$ {Sample an action $\tilde{a} \in \mathcal{A}$ according to P}
  $r \leftarrow \mathbf{R}\,(s, \tilde{a})$ {Calculate on the basis of A}
  $E \leftarrow E \oplus \{(s, \tilde{a}, r)\}$
  $s \leftarrow [s, \mathbf{Q}, \mathcal{Z} \oplus \{\mathbf{x}_{m(a)}\}, X \setminus \{\mathbf{x}_n\}_{\mathbf{0}}^{m(a)}]$
end while
return $E = (\mathbf{s_0}, \mathbf{a_0}, \mathbf{r_1}, \cdots, \mathbf{s_{M-1}}, \mathbf{a_{M-1}}, \mathbf{r_M})$

This MDP-MCS Reinforcement learning method can imitate the reading process of human being. It formulates the main content selection process as a sequence selection episode step by step. By this method, answer informations which have not in the same paragraph can be selected and the overlapping information can be filtered. These are the merits of MDP-MCS model.

3.4 Answer Span Prediction Module

In order to extract the final answer from the selected main content, we employed a main-steam boundary model to local the answer span. Pointer Network and Match-LSTM are used to compute the probability of each word:

$$\mathbf{g}_t^k = w_1 \mathbf{tanh}\left(w_2\left[\mathbf{v}_t^C, \mathbf{h}_{k-1}\right]\right) \qquad \mathbf{h}_k = \mathbf{LSTM}\left(h_{k-1}, c_k\right) \tag{11}$$

$$\alpha_t^k = \mathbf{exp}\left(g_t^k\right) \sum_{j=1}^{|C|} \mathbf{exp}\left(g_t^k\right) \qquad \mathbf{c}_k = \sum_{t=1}^{|C|} \alpha_t^k \mathbf{v}_t^C \tag{12}$$

where α_t^1 and α_t^2 is the probability of the t^{th} word in the passage to be the start and the end position of the answer span. C is the main content selected by the last sub-section v_t^C is the new main content word representation calculated by the first encode module. This boundary model can be trained by minimizing the negative log probabilities of the true start and end positions:

$$\mathcal{L}_{boundary} = -\frac{1}{\mathbf{N}} \sum_{i=1}^{N} \left(\log \alpha_{y_i^1}^1 + \log \alpha_{y_i^2}^2\right). \tag{13}$$

where y_i^1 and y_i^2 are the gold start and end positions, N is the scale of dataset.

4 EXPERIMENT

4.1 Dataset and Evaluation Metrics

Considering the large scale of DuReader dataset(The training, development and test sets consist of 181K, 10K and 10K questions, 855K, 45K and 46K documents, 376K, 20K and 21K answers, respectively.) and the time consumption of policy gradient method, we use about half of the train dataset which is classified as 'Baidu Search' dataset by DuReader. And we evaluate the reading comprehension task via BLEU-4 and Rouge-L, which are widely used for evaluating the quality of language generation. And for the main content selection model, $F1$ and *Recall* are also used as evaluation method.

4.2 Implementation Details

First, we pre-train the corpus with Glove[2] as the initial embeddings, and words whose count number is less than 5 will not be involved in the vocabulary. Our models are optimized using Adam algorithm with a initial learning rate as 0.001 and dropout rate 0.6. And in the passage selection phase, we simply treat the question with no selected passage or empty main content as the *No-Answer* question. In the main content selection layer, we use zero vector to represent the end sentence $< EOS >$. Besides, the word embedding size is 300-dimension and all hidden state sizes is 150-dimension. Note that we do not initialize the model parameters every times, the last train parameters are used in the next train time. When training the finally answer extraction module, we choose the text span in the main content with the highest BLEU-4 score as gold span. Moreover, we only use the main content whose ROUGE-L score is higher than 0.7.

4.3 Experimental Results

We compared our method with several state-of-the-arts baselines in MRC. The results is demonstrated in Table 1: The first part is the single passage selection methods; The second part are the boosting models and the multi-passage MRC method including our model.

GP: Golden Paragraph, a heuristic approach which chooses paragraph has the largest overlap with the answers in a document as answers. In testing phase, choosing paragraph which have the largest overlap with the question as answer. **Match-LSTM** [13]: Using Pointer Network and Match-LSTM to predict the beginning or ending points in the whole passages set. **BiDAF** [9]: a method which employed a bi-directional attention flow mechanism to achieve a question-aware context representations for the passage, then the beginning and ending points were predict based on the representations. **PR + BiDAF**: Using Passage Ranking to select the passages and then using BiDAF model to predict the answer. **V-Net** [16]: a method which extract candidate answers from all passages, and

[2] Pre-trained word vectors(http://nlp.stanford.edu/data/glove.6B.zip)

then do verification among those candidate answers. **S-Net** [12]: a model that consists of evidence extraction part and answer synthesis part.

The results of several baseline systems and our model are shown in Table 1.We can see that the *GP* method can improve the baseline methods significantly, but it cannot beat our main content selection model for there is no reference answer in the test dataset and simply matching the question words and with passage paragraph cannot lead to a better performance. The passage ranking method cannot outperform ours as well, for we get the selected sentences information within the selected passages.

Table 1. Performance of all methods on DuReader search test dataset.

Method	BLEU-4%	ROUGE-L%
GP	27.7	60.2
Match-LSTM	23.1	31.2
BiDAF	23.1	31.1
PR + BiDAF	37.55	41.81
V-Net	40.97	44.18
S-Net	41.12	44.52
GP+Match-LSTM	39.99	44.15
GP+BiDAF	39.83	42.01
Our model	**42.68**	**44.95**

5 Analysis and Discussion

In this section, we conduct experiments to show the reasons why our hierarchical answer selection model outperformed the baselines. Since answers on the test set are not published, we analyze our model on the development set.

5.1 Reasoning of Passage Selection Module

In theory, there are two reasons why the passage selection module and main content selection module are effective on DuReader Search dataset. First, we do not need read all the passages to get our answers when reading especially in test process. Second, some real-world answer passages contain the wrong answers or irrelevant answers which are noise in the answer span predict model. DuReader train and dev datasets are the real-world data collected by Baidu Search Engine, so each passages in them has the *is_selected* label which can provide ground truth for passage selection, we conduct statistics experiments to prove it. From the example in the website[3] we can see that the candidate passages are always about the same theme and some even talk about unrelated things. And by mathematical statistics (Table 2) of train and dev dataset, we found that the *is_selected*

[3] DuReader dataset(https://ai.baidu.com/broad/introduction?dataset=dureader).

Table 2. Statistics of the relation between selected passages and answers of question

Dataset	Answer in selected Passage	Answer not in selected Passage	total
Train	87502	3706	91208
Dev	4883	117	5000

passages almost contain all of the answer information. And Table 3 demonstrated that the passage selection layer do improve the performance of our model, for trash and overlapping information will influence the following sentence selection MDP model and the answer span predict model.

Table 3. Comparison of each module of our model on DuReader search dev dataset.

Method	F1%	Recall%	ROUGE-L%	BLEU-4%
Passage Selection	9.05	90.44	25.56	9.90
MDP-MCS	10.14	60.21	21.30	10.63
PS + MDP-MCS	22.14	86.42	**62.5**	**30.50**

5.2 Necessity of Main Content Selection Module

After getting the selected passages, we use MDP-MCS model to selected sentence-level answer information, for real-world dataset answer information may not in a continuous span of a passage, and answers always have a far edit distance. That is why some methods which work well in Ms-Marco do not performance well in DuReader dataset. And the experiment in Table 3 shows MDP-MCS combined with PS model can significantly improve the F1 and Recall score, which means it can select main content effectively. In addition, the use of MDP-MCS based on passage selection could significantly boosts the overall performance(ROUGE-L and BLEU-4) of answer span predict model. But the single MDP-MCS does not work well because the MDP model cannot handle a long sequence, which means that passages must be filtered before this layer. So it is necessary to implement a passage selection model and a main content selection model in a MRC pipeline.

6 Conclusion

In this paper, we propose a hierarchical answer selection framework pipeline to tackle the multi-passage MRC task. This framework contains three modules: The first module, passage selection layer first use the passage title and content information to predicted which passage will be selected; Then a main content selection module is modeled by MDP and trained by policy gradient; Finally, using Match-LSTM method, final answer can be generated. This hierarchical answer selection framework has achieved the state-of-the-art performance on a challenging dataset DuReader, which is designed for MRC on real web data.

References

1. He, W., Liu, K., Lyu, Y., Zhao, S., Xiao, X., Liu, Y., Wang, Y., Wu, H., She, Q., Liu, X., et al.: Dureader: a chinese machine reading comprehension dataset from real-world applications. arXiv preprint arXiv:1711.05073 (2017)
2. Hermann, K.M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., Blunsom, P.: Teaching machines to read and comprehend. In: Advances in Neural Information Processing Systems. pp. 1693–1701 (2015)
3. Hill, F., Bordes, A., Chopra, S., Weston, J.: The goldilocks principle: Reading children's books with explicit memory representations. arXiv preprint arXiv:1511.02301 (2015)
4. Lai, G., Xie, Q., Liu, H., Yang, Y., Hovy, E.: Race: Large-scale reading comprehension dataset from examinations. arXiv preprint arXiv:1704.04683 (2017)
5. Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., Deng, L.: Ms marco: A human generated machine reading comprehension dataset. arXiv preprint arXiv:1611.09268 (2016)
6. Pan, B., Li, H., Zhao, Z., Cao, B., Cai, D., He, X.: Memen: Multi-layer embedding with memory networks for machine comprehension. arXiv preprint arXiv:1707.09098 (2017)
7. Puterman, M.L.: Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons (2014)
8. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250 (2016)
9. Seo, M., Kembhavi, A., Farhadi, A., Hajishirzi, H.: Bidirectional attention flow for machine comprehension. arXiv preprint arXiv:1611.01603 (2016)
10. Shen, Y., Huang, P.S., Gao, J., Chen, W.: Reasonet: Learning to stop reading in machine comprehension. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 1047–1055. ACM (2017)
11. Sutton, R.S., Barto, A.G.: Reinforcement learning: An introduction, vol. 1. MIT press Cambridge (1998)
12. Tan, C., Wei, F., Yang, N., Lv, W., Zhou, M.: S-net: From answer extraction to answer generation for machine reading comprehension. arXiv preprint arXiv:1706.04815 (2017)
13. Wang, S., Jiang, J.: Machine comprehension using match-lstm and answer pointer. arXiv preprint arXiv:1608.07905 (2016)
14. Wang, S., Yu, M., Guo, X., Wang, Z., Klinger, T., Zhang, W., Chang, S., Tesauro, G., Zhou, B., Jiang, J.: Reinforced reader-ranker for open-domain question answering. arXiv preprint arXiv:1709.00023 (2017)
15. Wang, S., Yu, M., Jiang, J., Zhang, W., Guo, X., Chang, S., Wang, Z., Klinger, T., Tesauro, G., Campbell, M.: Evidence aggregation for answer re-ranking in open-domain question answering. arXiv preprint arXiv:1711.05116 (2017)
16. Wang, Y., Liu, K., Liu, J., He, W., Lyu, Y., Wu, H., Li, S., Wang, H.: Multi-passage machine reading comprehension with cross-passage answer verification. arXiv preprint arXiv:1805.02220 (2018)
17. Xiong, C., Zhong, V., Socher, R.: Dynamic coattention networks for question answering. arXiv preprint arXiv:1611.01604 (2016)