A Multi-Agent Communication Framework for Question-Worthy Phrase Extraction and Question Generation

Siyuan Wang,[1] **Zhongyu Wei,**[1*] **Zhihao Fan,**[1] **Yang Liu,**[2] **Xuanjing Huang**[3]
[1]School of Data Science, Fudan University, China
[2]LAIX Inc., China
[3]School of Computer Science, Fudan University, China
{18110980001,zywei,18210980005}@fudan.edu.cn, yang.liu@liulishuo.com, xjhuang@fudan.edu.cn

Abstract

Question generation aims to produce questions automatically given a piece of text as input. Existing research follows a sequence-to-sequence fashion that constructs a single question based on the input. Considering each question usually focuses on a specific fragment of the input, especially in the scenario of reading comprehension, it is reasonable to identify the corresponding focus before constructing the question. In this paper, we propose to identify *question-worthy phrases* first and generate questions with the assistance of these phrases. We introduce a multi-agent communication framework, taking phrase extraction and question generation as two agents, and learn these two tasks simultaneously via message passing mechanism. The results of experiments show the effectiveness of our framework: we can extract question-worthy phrases, which are able to improve the performance of question generation. Besides, our system is able to extract more than one question worthy phrases and generate multiple questions accordingly.

Sentence:
CBS provided digital streams of the game via *CBSSports.com*, and the CBS Sports apps on tablets, Windows *10*, *Xbox One* and other digital media players (such as Chromecast and Roku).

Questions:
- What CBS website provided a stream?
- What version of Windows supported the CBS sports app?
- On what game console was the CBS Sports app available?

Figure 1: An example sentence from SQuAD with multiple questions. *Phrases of light italic font and underlined* are corresponding focuses (answers in the dataset).

Introduction

Question generation has attracted attention from the research area of Natural Language Processing in recent years. Given a piece of text, automatic question generation aims to produce questions according to the input content. It acts as an essential component in some interactive systems, such as dialogue system (Piwek et al. 2007). It also has various applications in educational area, e.g., generate questions for reading comprehension. Besides, it is helpful for constructing question sets for the task of question answering (QA) to further improve the performance of QA systems.

Most previous studies on question generation are rule-based relying on hand-crafted features. However, these features are usually hard to design and are difficult to be transferred among different domains. Inspired by the success of the encoder-decoder structure in machine translation (Bahdanau, Cho, and Bengio 2015) and image captioning (Xu et al. 2015), Du et al. (2017) propose to apply sequence-to-sequence model with attention mechanism for question generation without human designed rules, which is called neural question generation (NQG), and has become the state of the art.

As shown in the example in Figure 1 from the dataset of Stanford Question Answering Dataset (SQuAD) (Rajpurkar et al. 2016) for reading comprehension, a question generated for the input sentence usually focuses on a specific fragment of the input that is deemed to contain significant information. Besides, there are several focuses in an input sentence. Based on these observations, we argue that a question generation system should be able to generate questions according to various focuses in the input content. Zhou et al. (2017) make a move by encoding the position of the ground-truth answer as auxiliary feature for question generation. Although the performance has been improved, it requires pre-identified answers which are not available in the real-environment for question generation. To tackle this problem, we propose to identify focuses from the input content first and generate questions using such information as assistance.

In the input text, not all but some content pieces are significant. Therefore, we propose a concept: *question-worthy phrases*, i.e. phrases which are worthwhile to be asked about. By extracting multiple phrases, our system is designed to generate various questions with different focuses.

*Corresponding author

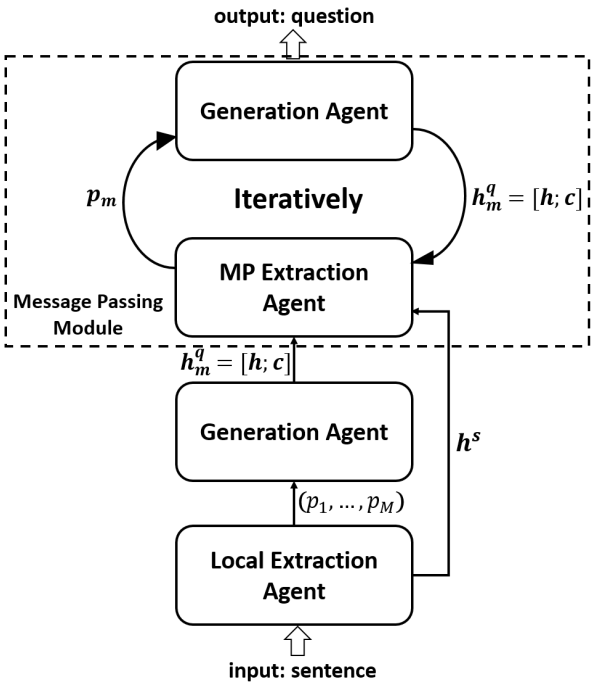

Figure 2: A brief overview of our proposed framework. From bottom to up, a local extraction agent, a generation agent and a iteratively MP module are presented. The generation agent after the local extraction agent is the same as the one in the message passing module. The arrows mean that messages are communicated between different agents.

In this paper, we formulate the task of question generation with an auxiliary task of question-worthy phrase extraction. We expect question-worthy phrases extracted can help to generate better questions, and learning to ask questions can benefit extracting question-worthy phrases, therefore we develop a multi-agent communication framework (Sukhbaatar, Fergus, and others 2016; Xu et al. 2017; Celikyilmaz et al. 2018), which constructs two different agents to implement phrase extraction and question generation respectively and learn these two tasks simultaneously via message passing mechanism. We conduct experiments on SQuAD (Rajpurkar et al. 2016) and evaluate our framework from two aspects: phrase extraction and question generation. The results of experiments show the effectiveness of our framework: our phrase extractor with message passing outperforms the state-of-the-art extraction model by a large margin and the performance of generation agent can be improved significantly with the assistance of the extraction agent.

Framework

Given an input sentence $x = (x_1, ..., x_n)$, where x_i is a token, our framework is designed to generate one or more questions $(y_1, ..., y_M), M \geqslant 1$. The overall framework can be seen in Figure 2. It can be split into three components from bottom to up: a local extraction agent, a generation agent and an iterative Message Passing (MP) module. The work flow starts from the local extraction agent. It extracts M phrases from the input sentence and conveys them to the generation agent to help generate M questions. Then the question messages and sentence representations for extraction are fed into the iteratively MP module which consists of an extraction agent and a generation agent. These two agents work iteratively and collaborate via passing messages to each other. Note that the structure of the generation agent in the MP module is the same as the one after the local extraction agent.

Local Extraction Agent

The local extraction agent extracts question-worthy phrases given the input sentence. Different from previous rule-based phrase extraction systems, which relies on part-of-speech information to extract phrases, we apply a neural model called Pointer Network (Vinyals, Fortunato, and Jaitly 2015).

Pointer Network (P-N) is proposed to use attention mechanism and a softmax probability distribution as a pointer to pick the index of tokens from the input sequence as output. Given the number of phrases (M) to be extracted in a sentence, P-N detects M pairs of start and end indexes, and tokens between each pair of start and end index are target phrases.

Local extraction agent, denoted as $\mathcal{E}_{Local}$, takes the sentence x as input and uses a bi-directional LSTM (Huang, Xu, and Yu 2015) to encode x.

$$\begin{aligned} \overrightarrow{h_t^s} &= \overrightarrow{LSTM}(x_t, \overrightarrow{h_{t-1}^s}) \\ \overleftarrow{h_t^s} &= \overleftarrow{LSTM}(x_t, \overleftarrow{h_{t+1}^s}) \end{aligned} \tag{1}$$

The $\overrightarrow{h_t^s}$ is the forward hidden state at step t while the $\overleftarrow{h_t^s}$ is the backward one. Then we take the concatenation of the forward and backward hidden states as the output of the encoder.

$$h^s = [\overrightarrow{h^s}; \overleftarrow{h^s}] \tag{2}$$

Then another LSTM is employed as the decoder. We only take two steps in the decoder, the first step is to point out the start index while the second is to identify the end of the phrase. The hidden states for the decoder are represented as (h_1, h_2). The attention vector (Bahdanau, Cho, and Bengio 2015) at each output time t is computed as follows:

$$\begin{aligned} u_{i,t} &= v^T \, tanh\left(W_1 h_i^s + W_2 h_t\right) \\ a_{i,t} &= softmax(u_{i,t}) \\ d_t &= \sum_{i=1}^{n} a_{i,t} h_i^s \end{aligned} \tag{3}$$

$a_{i,t}$ is the probability of i_{th} token to be pointed out at time t, and the attention vector d_t will then be fed into next time step as input. Then after two steps of decoding, we obtain a_s and a_e as start and end probabilities over all input tokens. The loss function is shown as Equation 4, where $\bar{a}_1$ and $\bar{a}_2$ are the start and end probabilities of ground truth positions, respectively.

$$\mathcal{L} = -(\bar{a}_1 \, log \, P(a_1) + \bar{a}_2 \, log \, P(a_2)) \tag{4}$$

We pick top M start indexes and end indexes respectively according to their probability a_s and a_e and map them into M (start,end) pairs as our final extraction results $p = (p_1, ..., p_M)$. The mapping is performed to pair each start index with its closest end index.

$$p = map[\underset{i}{topM}\ a_{i,1}; \underset{i}{topM}\ a_{i,2}] \tag{5}$$

Generation Agent

After extracting question-worthy phrases in the local extraction agent, the question generation agent is responsible for generating questions taking extracted phrases as assistance.

Given the sentence and extracted question-worthy phrases as input, the generation agent processes a phrase at a time. In particular, taking sentence x and one phrase p_m as input, the agent generates a question y_m. And our target can be formulated as Equation 6:

$$y_m = \underset{y}{argmax}\, P(y|x, p_m) \tag{6}$$

With $M > 1$ extracted phrases, we repeat such process M times, and finally generate multiple questions $y = (y_1, ..., y_M)$.

In this process, we first use the BIO tagging (Zhou et al. 2017) of sentence $b = (b_1, ..., b_n)$ to represent the phrase, where b_i indicates whether the i_{th} token of sentence x_i is in question-worthy phrase p_m or not, denoted as Equation 7:

$$b_i = \begin{cases} B & x_i\ is\ the\ start\ of\ p_m \\ I & x_i\ is\ inside\ or\ the\ end\ of\ \ p_m \\ O & x_i\ is\ outside\ of\ p_m \end{cases} \tag{7}$$

Then we concatenate the representation of x and b as the input of the generation process. The generation process follows an encode-decoder structure. We first employ bi-directional LSTM to encode the input and get encoder hidden states $h^s = (h_1^s, ..., h_n^s)$. For the decoder, a LSTM is applied with the attention mechanism. At each time step t in the decoder, a new state h_t is generated:

$$h_t = LSTM(y_{t-1}, h_{t-1}) \tag{8}$$

Furthermore, we take an attention-weighted average (Luong, Pham, and Manning 2015) over hidden states obtained in the encoder.

$$a_i^t = \frac{exp\big(h_t^T W h_i^s\big)}{\sum_j exp\big(h_t^T W h_j^s\big)}$$
$$c_t = \sum_i a_i^t h_i^s \tag{9}$$

Then h_t and c_t are concatenated to calculate the softmax probability distribution over vocabulary $P\left(w_t|x, w_{<t}, p_m\right)$, where w_t is the prediction at time t, and $y_m = (w_1, ..., w_t, ...)$. The loss function which needs to be minimized is expressed as Equation 10:

$$\mathcal{L}_g = -\sum_{t=1}^{|y_m|} log\, P\left(w_t|x, w_{<t}, p_m\right) \tag{10}$$

$h_t^q = [h_t; c_t]$ is also treated as the representation of the question, and will be passed to MP extraction agent.

Message Passing Module

Here it comes to the MP module which consists of a phrase extraction agent and a generation agent. The MP phrase extractor is an upgrade version of local phrase extractor by taking information from question generator. Note that the generation agent in the MP module is the same as the one introduced in the previous section, so we will skip it here and focus on the MP extraction agent.

MP Extraction Agent Different from local extraction agent, $\mathcal{E}_{MP}$ has an auxiliary input $h^q = (h_1^q, ..., h_M^q)$, which is the question representations passed from generation agent. It doesn't take the original sentence x as input, but use encoder hidden states h^s from local extraction agent $\mathcal{E}_{Local}$ to save the computation.

At each time given the sentence representation h^s and one question representation $h_m^q = (h_{m,1}^q, ..., h_{m,l}^q)$ from h^q, we extract one corresponding phrase.

We add a Match-LSTM module to help Pointer Network (Wang and Jiang 2015; 2017) to incorporate the information passed by question generator. Match-LSTM usually has two inputs, one is a premise and the other is a hypothesis. It uses the attention mechanism and sequentially combines the attention-weighted premise to each token of the hypothesis and uses the aggregated matching result to do the prediction.

$\mathcal{E}_{MP}$ takes question representation h_m^q as a premise while takes sentence representation h^s as a hypothesis. At each time step t in the encoder, we compute attention-weighted question representation d_t as below:

$$u_{m,j,t} = v^T\, tanh\left(W_1 h_{m,j}^q + W_2 h_t^s\right)$$
$$a_{m,j,t} = softmax(u_{m,j,t})$$
$$d_{m,t} = \sum_{j=1}^{l} a_{m,j,t} h_{m,j}^q \tag{11}$$

Then we concatenate h_t^s and $d_{m,t}$ as our final input $i_{m,t}$:

$$i_{m,t} = [h_t^s; d_{m,t}] \tag{12}$$

For the phrase extraction, we first feed $i_m = (i_{m,1}, ..., i_{m,n})$ into a Bi-LSTM encoder, and then in decoder, compute start and end probability over all tokens in the sentence and compute the loss function. We finally pick the start and end indexes with maximum likelihood:

$$p_m = [\underset{j}{argmax}\, a_{m,j,1}; \underset{j}{argmax}\, a_{m,j,2}] \tag{13}$$

When $M > 1$, we have multiple question representation $h^q = (h_1^q, ..., h_M^q)$, MP extraction agent will run M times to extract multiple phrases $p = (p_1, ..., p_M)$.

Actually this is an improvement for $\mathcal{E}_{Local}$ because we can verify whether generated questions in the generation agent $\mathcal{G}$ are mapped to the former extracted phrases, then discard mismatching phrases and adjust phrases that are not exact-matched.

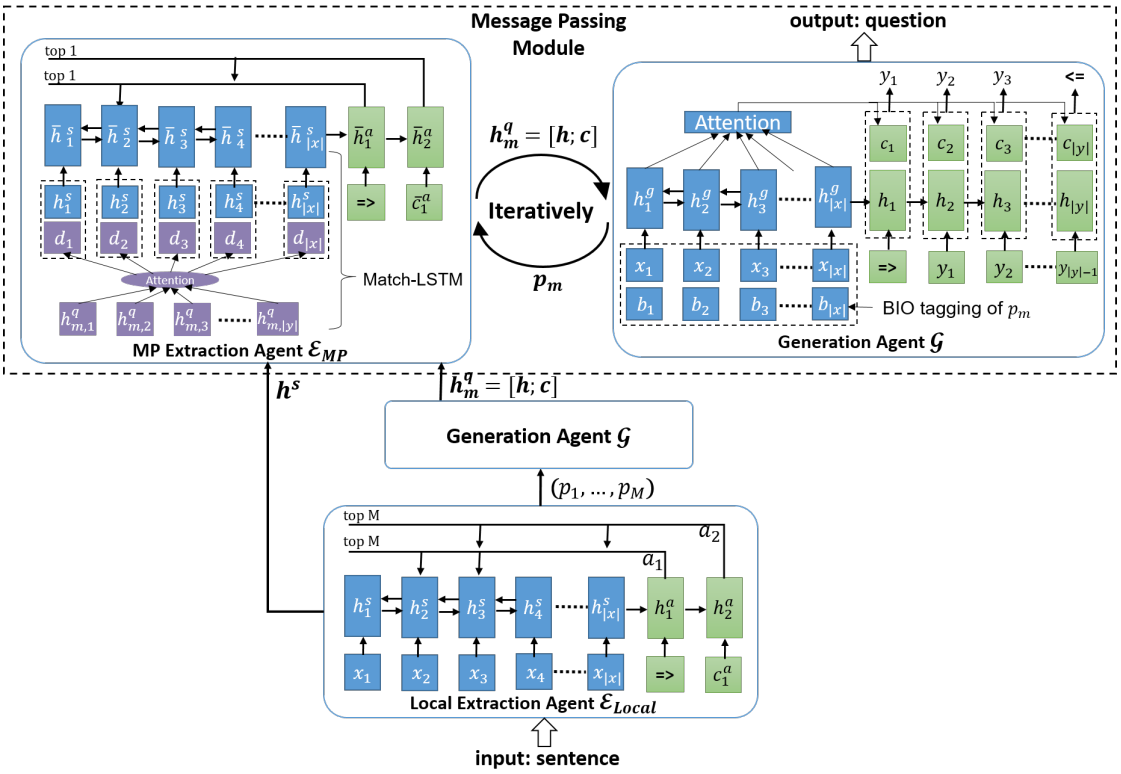

Figure 3: A detailed overview of Multi-agent Communication Framework.

Multi-agent Communication Mechanism

After introducing the local extraction agent, the generation agent and the extractor employed in the MP module, we now describe the communication mechanism and the training detail in the following part, referring to Figure 3.

Only the input sentence x is provided to $\mathcal{E}_{Local}$ and we extract the first batch of question worthy phrases. A loss function $\mathcal{L}_E^L$ is utilized for this stage. We then pass these extracted phrases $(p_1^L, ..., p_M^L)$ to the generation agent $\mathcal{G}$ to generate M questions $(y_1^L, ..., y_M^L)$. Loss $\mathcal{L}_G^L$ is computed accordingly. In the MP module, the sentence encoding message h^s from the local extraction agent and the question messages from $\mathcal{G}$ are passed to the MP extraction agent $\mathcal{E}_{MP}$, aiming to identify better phrases with the assistance of learning to ask questions. A loss $\mathcal{L}_{E,1}^{MP}$ is computed. And phrases extracted from $\mathcal{E}_{MP}$ will be passed to the generation agent $\mathcal{G}$ to continue the process of question generation with a loss $\mathcal{L}_{G,1}^{MP}$. Following the same pattern of operations, MP extraction agent and generation agent work iteratively in the MP module via passing messages to each other. We collect losses $\mathcal{L}_{E,2}^{MP}$, $\mathcal{L}_{G,2}^{MP}$, $\mathcal{L}_{E,3}^{MP}$, $\mathcal{L}_{G,3}^{MP}$,..., continually until the given iteration number.

To train this framework, we compute a combined loss function for optimization and hyper-parameters λ and $\beta = (\beta_0, \beta_1, \beta_2, ...)$ are employed to balance these losses:

$$\mathcal{L} = \beta_0 \mathcal{L}^L + \beta_1 \mathcal{L}_1^{MP} + \beta_2 \mathcal{L}_2^{MP} + ... \quad (14)$$

where

$$\mathcal{L}^L = \lambda \mathcal{L}_E^L + (1 - \lambda) \mathcal{L}_G^L$$
$$\mathcal{L}_i^{MP} = \lambda \mathcal{L}_{E,i}^{MP} + (1 - \lambda) \mathcal{L}_{G,i}^{MP} \quad (15)$$

Compared to (Sukhbaatar, Fergus, and others 2016; Celikyilmaz et al. 2018) which constructs multiple agents of the same function over different contexts, we define two kinds of agents to extract question-worthy phrases and generate questions respectively. And most existing models enable the message passing between different layers, while our communication mechanism is more flexible. Messages are passed from local extraction agent to the generation agent, from these two agents to the MP module and between the MP module iteratively. The proposed communication mechanism is easy to be used for other similar scenarios where agents are designed for different tasks.

Experiments

Datasets

Our experiments are conducted on dataset SQuAD (Rajpurkar et al. 2016), constructed for Reading Comprehension. Each sample is annotated with several questions together with their answers. All given answers are original tokens from the input text, thus we take these answers as ground truth question-worthy phrases. We split each piece of input text into sentences and perform experiments on sentence-level. We have 61,623 sentences in total and there

are multiple question-phrase pairs corresponding to a single sentence. The detailed statistics can be seen in table 1 in terms of the distribution of number of questions to sentences. More than 30% sentences have multiple questions. The average number of questions per sentence is nearly 1.5. We take a sentence-question-phrase tuple as an instance and it results in 90,682 ones in total. We further split them into 68,704, 10,313 and 11,665 ones as training set, validation set and test set respectively.

# of questions	# of sentences	percentage
1	41,356	67.11%
2	14,499	23.53%
3	3,921	6.36%
4	1,198	1.95%
$\geqslant 5$	649	1.05%
in total	61,623	100%

Table 1: Distribution of number of questions per sentence in our dataset.

Implementation Details

We construct different vocabulary for sentences and questions respectively (40,000 and 30,000) by keeping the most frequent words, while other words are replaced by the symbol *UNK*. We use Glove (Pennington, Socher, and Manning 2014) to initialize word embeddings with dimension of 300. The size of hidden units in LSTM cell in encoder is 300 while that for the decoder is 600. The size of embedding for BIO tag is 100.

We initialize parameters randomly using an uniform distribution with Xavier scheme. We use Adam to optimize our models with learning rate of 0.001, and decay at rate 0.96 per epoch, up to 30 epochs. We use dropout with probability of 0.7 and we take mini-batch of size 32 for training. We also clip the gradient once it exceeds 5. We maintain one MP iterations in our framework, and the hyper-parameters in loss optimization are set as $\beta_0 = 0.2, \beta_1 = 0.8, \lambda = 0$ in epoch 0-20 and $\lambda = 0.2$ after tuning.

We evaluate our model on validation set at each epoch and get the best model. During decoding process of testing, we use beam search with 3 as the beam size.

Models for Comparison and Results

Our framework involves question-worthy phrase extraction and question generation. In order to show the effectiveness of our framework, we make comparison for both extraction models and generation models.

Extraction Models We first compare several models of phrase extraction. For all of these models, the number of phrases to be extracted is provided in advance.

- $\mathbf{E_{NER}}$ conducts name entity recognition on the input text and takes extracted entities as question-worthy phrases. We directly use Stanford NER package[1] for the implementation.

[1]https://nlp.stanford.edu/software/CRF-NER.shtml

- $\mathbf{E_{Local}}$ is the local extraction agent of our Multi-Agent Communicating Framework, which just uses pointer network to point out one or more start-end pairs of question-worthy phrases.
- $\mathbf{E_{MP}}$ is our proposed model. It is the extraction agent in MP module of our framework, which matches question information passed from the generation agent to sentence for the extraction of question-worthy phrases.

Model	EM	F1	avg. num
$\mathbf{E_{NER}}$	13.12%	17.33	0.86
$\mathbf{E_{Local}}$	24.27%	38.63	1.43
$\mathbf{E_{MP}}$	**35.77**%	**46.71**	1.38

Table 2: Evaluation results of different phrase extraction models. (underline: diff. with both comparison models ($\mathbf{E_{NER}}$, $\mathbf{E_{Local}}$) $p < 0.01$; **Bold**: the best performance in the column)

Results of Phrase Extraction To evaluate question-worthy phrase extraction, we use ExactMatch score and F1 score, which are widely used for the evaluation of question answering. We also report the average number of extracted phrases to confirm the ability of our framework for multiple phrases extraction. Table 2 shows the overall results for phrase extraction. We have several findings:

- $\mathbf{E_{NER}}$ is to recognize name entities and take them as output. However, the tagger can only recognize limited types of name entities. This leads to a low average number of phrases extracted. Besides, it also gets low EM and F1 scores because the focuses of some sentences are not entities, such as "15-1" in the example shown in Table 4.
- $\mathbf{E_{MP}}$ and $\mathbf{E_{Local}}$ use P-N to identify question-worthy phrases and produce a significantly better performance compared to NER based extractor, indicating that they are able to learn the features of important fragments of the input sentence from training dataset.
- $\mathbf{E_{MP}}$ generates better EM and F1 scores than $\mathbf{E_{local}}$, indicating that messages passed from generation agent is useful for extracting better question worthy phrases. The average number of extracted phrases of $\mathbf{E_{MP}}$ is lower than $\mathbf{E_{local}}$. By looking deep into the results, we find that some mis-extracted phrases are discarded with the assistance of the generation agent.

Following similar decoding structure for phrase extraction, both $\mathbf{E_{MP}}$ and $\mathbf{E_{Local}}$ suffer from the problem of span overlapping among extracted phrases for a given sentence. The ideal situation is that phrases extracted in the same sentence are independent with each other. In order to show how question information passed from generation agent can help extract multiple independent phrases, we calculate the percentage of sentences with overlapped phrases for the two extraction agents. Results show that 32.37% of sentences have overlapped phrases using $\mathbf{E_{Local}}$ while that of $\mathbf{E_{MP}}$ is 22.05%. This indicates that question information can help to identify multiple independent phrases.

Model	BLEU 1	BLEU 2	BLEU 3	BLEU 4	METEOR	ROUGE$_L$
$\mathbf{NQG_{Rule}}$	38.15	21.03	14.15	9.98	13.38	29.00
$\mathbf{NQG_{Base}}$	43.83	23.80	14.46	9.05	14.63	36.50
$\mathbf{NQG_{NER}}$	44.00	23.79	14.52	9.22	14.89	36.32
$\mathbf{NQG_{Local}}$	44.36	24.58	15.23	9.76	15.15	37.00
$\mathbf{NQG_{MP}}$	$\underline{\mathbf{45.70}}^*$	$\underline{\mathbf{25.87}}^*$	$\underline{\mathbf{16.33}}^*$	$\mathbf{10.56}^*$	$\underline{\mathbf{15.76}}^*$	$\underline{\mathbf{38.09}}^*$
$\mathbf{NQG_{G-t}}$	47.49	27.81	17.9	11.81	16.84	40.23

Table 3: Evaluation results of different question generation models in terms of BLEU 1-4, METEOR and ROUGE$_L$. (underline: diff. with all the comparison models ($\mathbf{NQG_{Rule}}$, $\mathbf{NQG_{Base}}$, $\mathbf{NQG_{NER}}$, $\mathbf{NQG_{Local}}$) $p < 0.01$; *: $p < 0.05$; **Bold**: the best performance for each column)

Generation Models For question generation, we compare several models.

- $\mathbf{NQG_{Rule}}$ (Heilman and Smith 2010) is a rule-based model. It uses manually written rules to perform a sequence of general purpose syntactic transformations (e.g., subject-auxiliary inversion) to turn declarative sentences into questions, which are then ranked and selected as the final generated questions.
- $\mathbf{NQG_{Base}}$ (Du, Shao, and Cardie 2017) is the state-of-the-art neural-based model for question generation. It takes sentence as input and uses encoder-decoder model with attention mechanism to generate question without modeling focuses in the input.
- $\mathbf{NQG_{NER}}$ takes phrases extracted by $\mathbf{E_{NER}}$ as additional information for question generation, and uses the same generation model introduced in our paper.
- $\mathbf{NQG_{Local}}$ is our generation agent that takes phrases extracted by $\mathbf{E_{Local}}$ as additional information for question generation.
- $\mathbf{NQG_{MP}}$ is our proposed generation model in the MP module that takes phrases extracted by $\mathbf{E_{MP}}$ as additional information for question generation and is trained with $\mathbf{E_{MP}}$ simultaneously.
- $\mathbf{NQG_{G-t}}$ (Zhou et al. 2017) takes the ground-truth answers as additional information for question generation and uses the same generation model introduced in our paper. This is the upper bound of our generation model.

Results of Question Generation For the evaluation of question generation, we use several metrics including BLEU 1-4 (Papineni et al. 2002), METEOR (Banerjee and Lavie 2005) and ROUGE$_L$ (Lin 2004). Without ground-truth answers, we cannot pair the generated questions with questions in reference. For comparison, the most similar question in reference is selected for a generated question to compute evaluation metrics. Table 3 shows the evaluation results of all the generation models. We have several findings:

At first, we note that $\mathbf{NQG_{Rule}}$ performs poorly as the sentences in our dataset are quite long and complex, which are not suitable to be analyzed using rule-based methods, for example, sentence parsing is hard to be implemented while the sentence is complicated.

$\mathbf{NQG_{Base}}$ and $\mathbf{NQG_{G-t}}$ construct the baseline and upper bound, respectively. Because the former generates questions without any extracted phrase information while the latter has the ground truth answers. Between the baseline and the upper bound, the order of different models in terms of all the evaluation metrics are $\mathbf{NQG_{NER}} < \mathbf{NQG_{Local}} < \mathbf{NQG_{MP}}$. This indicates that by taking extracted phrases with higher quality, the generation model is able to produce better questions. This also shows the effectiveness of our framework of utilizing extracted phrases for better question generation.

In order to show the ability of our framework to generate multiple questions, we calculate the average number of different questions generated by $\mathbf{NQG_{MP}}$, which is 1.35. (We identify two questions are different if they have more than 3 different words.) This number is slightly smaller than the average number of extracted phrases from $\mathbf{E_{MP}}$, because sometimes different phrases may correspond to the same question.

Significance Test We also conduct significance test (independent two-sample *t*-test) to compare our model to other comparison models for both phrase extraction and question generation. The results are shown in Table 2 and Table 3 respectively. We can see that our extraction model $\mathbf{E_{MP}}$ outperforms $\mathbf{E_{NER}}$ and $\mathbf{E_{Local}}$ with a large margin with p-value < 0.01. Compared to $\mathbf{NQG_{Rule}}$, $\mathbf{NQG_{Base}}$, $\mathbf{NQG_{NER}}$, and $\mathbf{NQG_{Local}}$, our generation model $\mathbf{NQG_{MP}}$ generates a significant improvement.

Case Study

In order to show the effectiveness of our framework more intuitively, we present two examples of generated questions in Table 4. In both cases, our model can first extract phrases close to ground-truth answers while $\mathbf{E_{NER}}$ can only extract name entities which sometimes are not the question-worthy phrases. And we can also generate good questions with question-worthy focuses, such as the first question we generated in example 1, asking the score of the competition. However, generation based on $\mathbf{E_{NER}}$ is not able to produce questions with high quality.

Related Work

There are two lines of research topics that are related to our work, namely, phrase extraction and question generation.

Sample 1
Input: the panthers finished the regular season with a 15 – 1 record , and quarterback cam newton was named the nfl most valuable player (mvp) .
Phrases
Ground-truth: 15 – 1, quarterback cam newton.
NER: panthers, $\langle blank \rangle$.
$\mathbf{E_{MP}}$**:** 15, quarterback cam newton.
Questions
Ground-truth: what was the ratio in 2015 for the carolina panthers during their regular season ? which carolina panthers player was named most valuable player ?
$\mathbf{NQG_{NER}}$**:** who wons the regular season ? what was the regular session in the afl ?
$\mathbf{NQG_{MP}}$**:** how many wins did the panthers win during the regular season ? who was named the nlf most valuable player?
Sample 2
Input: next to the main building is the basilica of the sacred heart .
Phrases
Answers: main building.
NER: sacred heart
$\mathbf{E_{MP}}$**:** next to the main building.
Questions
Ground Truth: the basilica of the sacred heart at notre dame is beside to which structure ?
$\mathbf{NQG_{NER}}$**:** what is next to main building ?
$\mathbf{NQG_{MP}}$**:** where is the basilica of prayer ?

Table 4: Cases of sentences, phrases extracted and questions generated by different models.

Phrases Extraction

Phrase Extraction is to identify which fragments of sentence are interesting to be asked about, and should be extracted. For this kind of information extraction task, many previous work are rule-based utilizing lexical features such as part-of-speech (POS) and NER tags (Subramanian et al. 2017), then rank these phrases to get final results.

Some other work tries to apply recurrent neural network to predict whether a token in sentence should be extracted (Wang and Jiang 2015). However, this method can only take simple relationship between input tokens into account. Then conditional random field (CRF) is applied after RNN layer to consider the sentence-level tag information (Huang, Xu, and Yu 2015; Lample et al. 2016). Pointer Networks (Vinyals, Fortunato, and Jaitly 2015) also have been employed to this task. Ptr-Net used in machine reading comprehension task (Wang and Jiang 2017; Liu et al. 2018) is to predict answer given sentence according to question which also can be seen as an phrase extraction task.

Question Generation

Rule-based methods are previously applied to the the task of question generation, which usually extract key aspects from the sentences and then insert them into hand-crafted templates to generate interrogative sentences (Heilman and Smith 2010; Heilman 2011; Duan et al. 2017).

Recently, attention-based seq-to-seq model is also developed for question generation (Du, Shao, and Cardie 2017), and performs well. (Zhou et al. 2017) takes the same model and uses the answers as extra information to generate better questions.

(Du and Cardie 2017) proposed that not all sentences in a paragraph are interesting and important and put forward a concept of *question-worthy* sentence. This study also inspires us to come up a concept: question-worthy phrases. There are also two studies (Subramanian et al. 2017; Kumar et al. 2018) proposing the similar concepts: key phrase and pivotal answer in the sentence to be extracted to aid question question. However, both of them just use a two-stage method which first implement selection and then generate question.

Besides, some studies (Wang, Yuan, and Trischler 2017; Tang et al. 2017) take question generation as a subtask, and learn it with other tasks simultaneously, such as question answering. However none of the previous work combines question generation with phrase extraction to do multi-task learning, and all of them require the ground truth information for its counter-part task when testing while we just use the input sentence.

Other researchers also work on question generation from images (Fan et al. 2018a; 2018b).

Conclusion and Future Work

In this paper, we propose to extract *question-worthy phrases* and use such information for better question generation. We propose a multi-agent communication framework to combine tasks of phrase extraction and question generation. As a result, we can generate multiple questions given input sentence without any ground-truth answers. Experimental results on SQuAD also show the effectiveness of our proposed framework. In the future, there are two research directions as follow-up. First, the performance of phrase extraction in our framework is still limited. Working on the end of phrase extraction, especially for multiple phrases extraction would be really interesting. Second, existing evaluation metrics for question generation is only able to test the perspective of relevance. We will explore better evaluation schemes for question generation taking other aspects into consideration.

Acknowledgments

This work is partially supported by China National Key R&D Program (No. 2017YFB1002104), National Natural Science Foundation of China (No. 61702106) and Shanghai Science and Technology Commission (No. 17JC1420200, No. 17YF1427600 and No. 16JC1420401).

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate.

In Proceedings of the International Conference on Learning Representations.

Banerjee, S., and Lavie, A. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, 65–72.

Celikyilmaz, A.; Bosselut, A.; He, X.; and Choi, Y. 2018. Deep communicating agents for abstractive summarization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, 1662–1675.

Du, X., and Cardie, C. 2017. Identifying where to focus in reading comprehension for neural question generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2067–2073.

Du, X.; Shao, J.; and Cardie, C. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 1342–1352.

Duan, N.; Tang, D.; Chen, P.; and Zhou, M. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on EMNLP*, 866–874.

Fan, Z.; Wei, Z.; Li, P.; Lan, Y.; and Huang, X. 2018a. A question type driven framework to diversify visual question generation. In *IJCAI*, 4048–4054.

Fan, Z.; Wei, Z.; Wang, S.; Liu, Y.; and Huang, X. 2018b. A reinforcement learning framework for natural question generation using bi-discriminators. In *Proceedings of the 27th International Conference on Computational Linguistics*, 1763–1774.

Heilman, M., and Smith, N. A. 2010. Good question! statistical ranking for question generation. In *The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 609–617.

Heilman, M. 2011. Automatic factual question generation from text. *Language Technologies Institute School of Computer Science Carnegie Mellon University* 195.

Huang, Z.; Xu, W.; and Yu, K. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Kumar, V.; Boorla, K.; Meena, Y.; Ramakrishnan, G.; and Li, Y.-F. 2018. Automating reading comprehension by generating question and answer pairs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*.

Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; and Dyer, C. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the NAACL*, 260–270.

Lin, C.-Y. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop* 74–81.

Liu, X.; Shen, Y.; Duh, K.; and Gao, J. 2018. Stochastic answer networks for machine reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 1694–1704.

Luong, M.-T.; Pham, H.; and Manning, C. D. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 1412–1421.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, 311–318.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on EMNLP*, 1532–1543.

Piwek, P.; Hernault, H.; Prendinger, H.; and Ishizuka, M. 2007. T2d: Generating dialogues between virtual agents automatically from text. In *International Workshop on Intelligent Virtual Agents*, 161–174. Springer.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2383–2392.

Subramanian, S.; Wang, T.; Yuan, X.; Zhang, S.; Bengio, Y.; and Trischler, A. 2017. Neural models for key phrase detection and question generation. In *Proceedings of the Workshop on Machine Reading for Question Answering*, 78–88.

Sukhbaatar, S.; Fergus, R.; et al. 2016. Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems*, 2244–2252.

Tang, D.; Duan, N.; Qin, T.; and Zhou, M. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027*.

Vinyals, O.; Fortunato, M.; and Jaitly, N. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, 2692–2700.

Wang, S., and Jiang, J. 2015. Learning natural language inference with lstm. *Proceedings of the 2016 NAACL-HLT* 1442–1451.

Wang, S., and Jiang, J. 2017. Machine comprehension using match-lstm and answer pointer. *In International Conference on Learning Representations*.

Wang, T.; Yuan, X.; and Trischler, A. 2017. A joint model for question answering and question generation. *arXiv preprint arXiv:1706.01450*.

Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhudinov, R.; Zemel, R.; and Bengio, Y. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2048–2057.

Xu, D.; Zhu, Y.; Choy, C. B.; and Fei-Fei, L. 2017. Scene graph generation by iterative message passing. In *Proceedings of the IEEE Conference on CVPR*, volume 2.

Zhou, Q.; Yang, N.; Wei, F.; Tan, C.; Bao, H.; and Zhou, M. 2017. Neural question generation from text: A preliminary study. In *National CCF NLPCC*, 662–671.