The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI-20)

Improving Question Generation with Sentence-Level Semantic Matching and Answer Position Inferring

Xiyao Ma,[1] Qile Zhu,[1] Yanlin Zhou,[1] Xiaolin Li[2]
[1]NSF Center for Big Learning, University of Florida
[2]AI Institute, Tongdun Technology
{maxiy, valder, zhou.y}@ufl.edu, xiaolin.li@tongdun.net

Abstract

Taking an answer and its context as input, sequence-to-sequence models have made considerable progress on question generation. However, we observe that these approaches often generate wrong question words or keywords and copy answer-irrelevant words from the input. We believe that lacking global question semantics and exploiting answer position-awareness not well are the key root causes. In this paper, we propose a neural question generation model with two general modules: sentence-level semantic matching and answer position inferring. Further, we enhance the initial state of the decoder by leveraging the answer-aware gated fusion mechanism. Experimental results demonstrate that our model outperforms the state-of-the-art (SOTA) models on SQuAD and MARCO datasets. Owing to its generality, our work also improves the existing models significantly.

Introduction

Question Generation (QG), an inverse problem of Question Answering (QA), aims to generate a semantically relevant question given a context and a corresponding answer. It has huge potential in education scenario (Du, Shao, and Cardie 2017), dialogue system, and question answering (Du and Cardie 2018). A bunch of models using sequence-to-sequence (seq-to-seq) models (Sutskever, Vinyals, and Le 2014) with the attention mechanism (Bahdanau, Cho, and Bengio 2014) have been proposed for neural question generation (Zhou et al. 2017; Du, Shao, and Cardie 2017).

Enriched linguistic features with Part-Of-Speech (POS) tags, relative position information, and paragraph context are incorporated in the embedding layers (Zhou et al. 2017; Kim et al. 2018; Zhao et al. 2018). Copy mechanism (Gulcehre et al. 2016) is exploited to enhance the output quality of decoders (Zhao et al. 2018; Sun et al. 2018).

However, checking the questions generated by the strong baseline models NQG++ (Zhou et al. 2017) and Pointer-generator (See, Liu, and Manning 2017) originally solving text summarization, the modern question generation models face the two main issues as follows: (1) *Wrong keywords and question words:* The model may ask questions with wrong keywords and wrong question words, as shown in the examples in Table 1. (2) *Poor copy mechanism:* The model copies the context words semantically irrelevant to the answer (Sun et al. 2018), as illustrated in the examples in Table 2.

Generally, the decoder with parameters θ_d in seq-to-seq models (Zhou et al. 2017; Sun et al. 2018; Zhao et al. 2018) is trained by maximizing the generation probability $p(y_t|y_{<t}, z; \theta_d)$ of the reference question word y_t, given the previous generated words conditioned on the encoded context z. However, the decoder may focus on local word semantics while ignoring the global question semantics during generation, resulting in above-mentioned issues. Meanwhile, the answer position-aware features are not exploited well by the copy mechanism, resulting in copying answer-irrelevant context words from input.

To alleviate these issues, we claim that learning the sentence-level semantics and answer position-awareness in a Multi-Task Learning (MTL) fashion results in a better performance as shown in Table 1 and 2. To do so, we first propose sentence-level semantic matching module for learning global semantics from both the encoder and decoder simultaneously. Then, answer position inferring module is introduced to enforce the model with the copy mechanism (See, Liu, and Manning 2017) to emphasize the relevant context words with the answer position-awareness. Furthermore, we propose answer-aware gated fusion mechanism for improved answer-aware sentence vector for decoder.

We further conduct extensive experiments on SQuAD (Rajpurkar et al. 2016) and MS MARCO (Nguyen et al. 2016) dataset to show the superiority of our proposed model. The experimental results show that our model not only outperforms the SOTA models on main metrics, auxiliary metrics, and human judgments, but also improves different models due to its generality. Our contributions are three-fold:

- We analyze the questions generated by strong baselines and find two issues: wrong keywords and wrong question words and copying answer-irrelevant context words. We identify that lacking whole question semantics and expoiting answer position-awareness not weel are the key root causes.
- To address the issues, we propose neural question generation model with sentence-level semantic matching, an-

Sentence: starting in 1965, donald davies at the national physical laboratory, uk, independently developed the same message routing methodology as developed by baran.
Reference: what did donald davies develop?
NQG++: what is *the national physical laboratory*?
Pointer-generator: what did *baran* develop?
Our model: what did donald devies develop at the national physical laboratory?

Sentence: in 1979 , the soviet union deployed its 40th army into afghanistan , attempting to suppress an islamic rebellion against an allied marxist regime in the afghan civil war.
Reference: who deployed its army into afghanistan in 1979?
NQG++: *in what year* did the soviet union invade afghanistan?
Pointer-generator: *what* deployed their army into afghanistan?
Our model: who deployed their army into afghanistan?

Table 1: Bad cases of the baselines: the models ask questions with wrong question words and wrong keywords. The answers are shown with underline. The italicized text indicates the poor performance of existing models, while the gray highlighted text shows the improved performance with our proposed model.

Sentence:as of 2012 , quality private schools in the united states charged substantial tuition , close to $ 40,000 annually for day schools in new york city , and nearly $ 50,000 for boarding schools.
Reference: what would a parent have to pay to send their child to a boarding school in 2012?
NQG++: how much money did *quality private schools* in the us have in 2012?
Pointer-generator: how much money is *charged substantial tuition* for boarding school?
Our model: how much money for boarding schools in new york city in 2012?

Sentence: during his second year of study at graz , tesla developed a passion for and became very proficient at billiards , chess and card-playing , sometimes spending more than 48 hours in a stretch at a gaming table.
Reference: how long would tesla spend gambling sometimes?
NQG++: how long did *the billiards of tesla get* in a stretch?
Pointer-generator: how long did tesla *become very proficient* in a stretch at a gaming table ?
Our model: how many hours did tesla spend in a stretch at a gaming table ?

Table 2: Bad cases of the baselines: the models copy the answer-irrelevant context words from sentences.

swer position inferring, and gated fusion.

- We conduct extensive experiments to demonstrate the superiority of our proposed model for improving question generation performance in terms of the main metrics, auxiliary machine comprehension metrics, and human judgments. Besides, our work can improve current models significantly due to its generality.

Proposed Model

In this section, we describe the details of our proposed models, starting with an overview of question generation problem. Then, we illustrate our backbone seq-to-seq model with gated fusion for improved answer-aware sentence vector for generation. Finally, we illustrate sentence-level semantic matching and answer position inferring to alleviate the issues we discussed in the previous section.

Problem Formulation

In a question generation problem, a sentence $X = \{x_t\}_{t=1}^{M}$ containing an answer A, a contiguous span of the sentence, is given to generate a question $Y = \{y_t\}_{t=1}^{N}$ matching with the sentence X and the answer A semantically.

Seq-to-seq model with Answer-aware Gated Fusion

Encoder: Following the baseline model (Zhou et al. 2017), we use an attention-based seq-to-seq model with the same enriched semantic and lexical features (i.e., NER features (Sang and De Meulder 2003), POS tag (Brill 1992), case, and answer position features) as input $x_i \in \mathbb{R}^{(d_w+d_n+d_p+d_c+d_ap)}$ in the embedding layer.

With a bi-directional LSTM (Hochreiter and Schmidhuber 1997) as the encoder, the sentence representation, a sequence of D-dim hidden state $H = [h_1, h_2, ..., h_m] \in \mathbb{R}^{M*D}$, is produced by concatenating a forward hidden state and a backward hidden state given the input sentence $X = [x_1, x_2, ..., x_m]$:

$$h_i = [\overrightarrow{h_i}, \overleftarrow{h_i}], \tag{1}$$

$$\overrightarrow{h_i} = LSTM_{Enc}(x_i, \overrightarrow{h_{i-1}}), \tag{2}$$

$$\overleftarrow{h_i} = LSTM_{Enc}(x_i, \overleftarrow{h_{i+1}}) \tag{3}$$

Answer-aware Gated Fusion: Instead of passing the last hidden state h_m of the encoder to the decoder as the initial hidden state, we propose gated fusion to provide an improved answer-aware sentence vector z for the decoder.

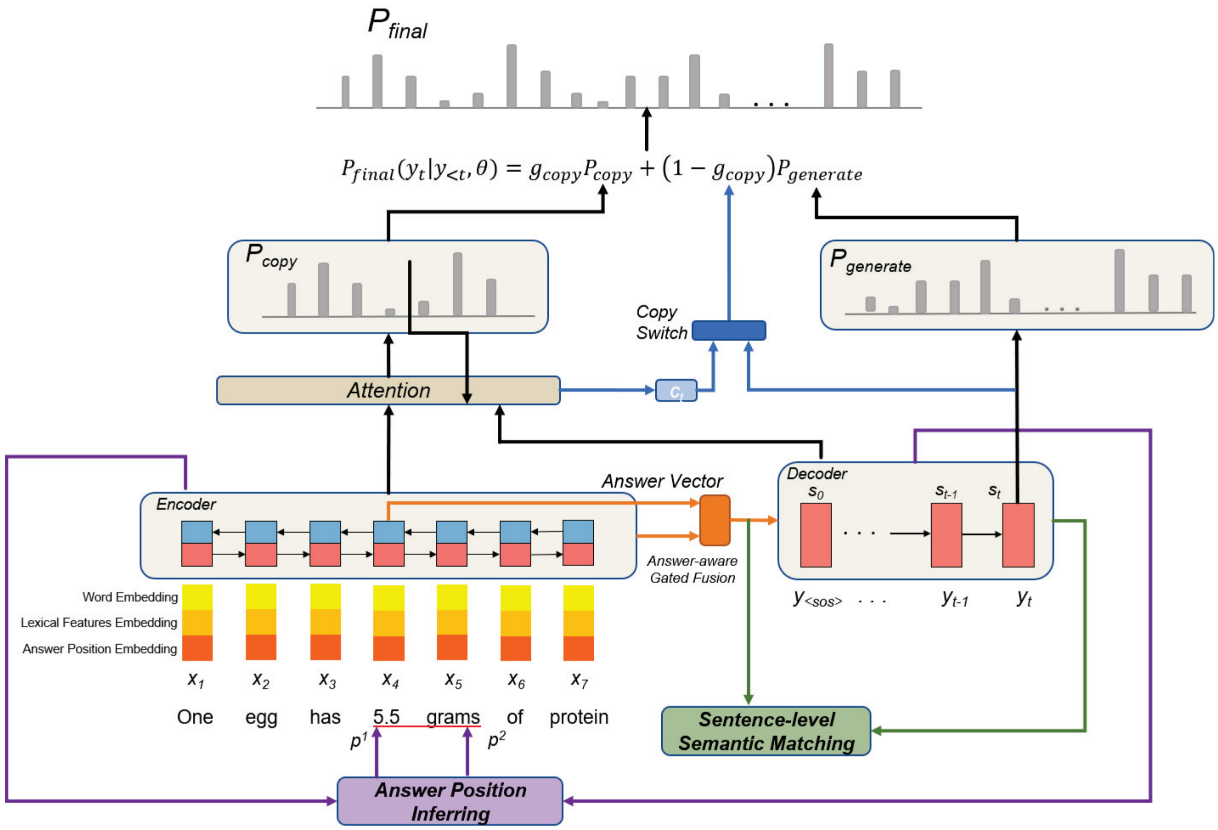

Figure 1: Diagram for neural question generation model with sentence-level semantic matching, answer position inferring, and gated fusion.

Similar to the gates in LSTM, we use two information flow gates computed by $Sigmoid$ functions to control the information flow of sentence vector and answer vector:

$$g_m = \sigma(W_m^T * [h_m, h_a] + b_m), \tag{4}$$
$$g_a = \sigma(W_a^T * [h_m, h_a] + b_a), \tag{5}$$
$$z = g_m \cdot h_m + g_a \cdot h_a \tag{6}$$

where W_m, W_a, b_m, and b_a are trainable weights and biases. We take the hidden state at the answer starting position as the answer vector $h_a \in \mathbb{R}^D$ since it encodes the whole answer semantics with the bi-directional LSTM.

Decoder: Taking the encoder hidden states $H = [h_1, h_2, ..., h_n] \in \mathbb{R}^{N*D}$ as the context and the improved answer-aware sentence vector z as the initial hidden state s_1, an one layer uni-directional LSTM updates its current hidden state s_t with the previous decoded word as the input w_t:

$$s_t = LSTM_{Dec}([w_t; c_{t-1}], s_{t-1}) \tag{7}$$

Meanwhile, the attention mechanism (Bahdanau, Cho, and Bengio 2014) is exploited by attending the current decoder state s_t to the encoder context $H = [h_1, h_2, ..., h_n]$. The context vector c_t is computed with normalized attention vector α_t by weighted-sum:

$$e_t = H^T W_e s_t, \tag{8}$$
$$\alpha_t = Softmax(e_t), \tag{9}$$
$$c_t = H^T \alpha_t \tag{10}$$

Question word y_t is generated from vocabulary V with $Softmax$ function:

$$p_{generate}(y_t) = Softmax(f(s_t, c_t)) \tag{11}$$

where f is realized by a two-layer feed-forward network.

Copy Mechanism / Pointer-generator: Copy Mechanism (Gulcehre et al. 2016) and Pointer-generator network (See, Liu, and Manning 2017) are introduced to enable the model to generate words from the vocabulary $\mathbb{V}$ with size $|\mathbb{V}|$ or copy words from the input sentence $\mathbb{X}$ with size $|\mathbb{X}|$ by taking the i^{th} word with the highest attention weight $\alpha_{t,i}$ computed in Equation 9.

Generally, when generating the question word y_t, a copy switch g_{copy} is computed to decide whether the generated word is generated from vocab or copied from source sentence, given the current decoder hidden state s_t and context vector c_t:

$$g_{copy} = \sigma(W^c s_t + U^c c_t + b^c) \tag{12}$$

where W^c, U^c, and b^c are learnable weights and biases.

The final word distribution is obtained by combining the probability of generate mode and the probability of copy mode:

$$\begin{aligned} p_{\text{final}}\left(y_t | y_{<t}; \theta_{s2s}\right) = g_{copy} p_{\text{copy}}(y_t, \theta_1) \\ + \left(1 - g_{\text{copy}}\right) p_{\text{generate}}(y_t, \theta_2) \end{aligned} \tag{13}$$

where θ, θ_1, and θ_2 are the parameters of neural network.

We use the negative log likelihood for the seq-to-seq loss:

$$L\left(\theta_{s2s}\right) = -\frac{1}{N} \sum_{i}^{N} \log p_{\text{final}}\left(y_t | y_{<t}; \theta_{s2s}\right) \tag{14}$$

where θ_{s2s} is the parameters of the seq-to-seq model, and N is the number of data in the train dataset.

Sentence-level Semantic Matching

Existing models, especially the decoders, generate question words given the generated and partial question words without considering the global whole question semantic, prone to wrong question words or keywords. Meanwhile, we found that there exist different reference questions targeting the different answers in the same sentence in SQuAD and MARCO datasets. For example, we have $< sentence, answer1, question1 >$ and $< sentence, answer2, question2 >$. However, the baseline model is prone to generating generic questions in this case. To overcome this problem, we propose the sentence-level semantic matching module to learn the sentence-level semantics from both the encoder and decoder sides in a Multi-Task Learning way.

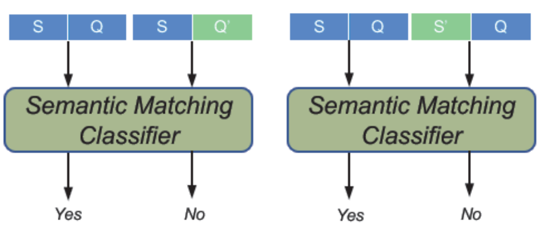

Figure 2: Pipeline of Sentence-level Semantic Matching

Generally, we have the improved answer-aware sentence vector z obtained by our gated fusion. Regarding the decoder, a uni-directional LSTM, as an encoder for question, we take the last hidden state s_n as the question vector.

Then, as illustrated in Figure 2, we train two classifiers to distinguish the not semantic-matching pairs $[z, s'_n]$ (S, Q') and $[z', s_n]$ (S', Q) from the semantic-matching sq pair $[z, s_n]$ (S, Q), respectively, where z' and s'_n are the vector of randomly sampled mismatched sentence and question from the same passage:

$$p = Softmax(W_c[z, s_n] + b_c) \tag{15}$$

where $[z, s_n]$ is the concatenation of the sentence vector z and the question vector s_n.

We take the sum of the binary cross entropy of the two classifiers as the sentence-level semantic matching loss:

$$L(\theta_{sm}) = -\frac{1}{K} \sum_{i}^{K} L_{BCE}(p_1, y_1) + L_{BCE}(p_2, y_2) \tag{16}$$

$$L_{BCE}(p, y) = y \log(p) + (1 - y) \log(1 - p) \tag{17}$$

where θ_{sm} is the parameters of the two classifiers. p_1 and p_2 are the prediction probabilities of the two classifiers, and y_1 and y_2 refer to labels indicating the S, Q pair is matched or not. K is the number of S, Q pairs.

Answer Position Inferring

Another issue of the baseline model is that it copies the answer-irrelevant words from the input sentence. One potential reason is that the model does not learn the answer position features well, and the attention matrix is not signified by the context words relevant to the answer. To address the issue, we leverage answer position inferring module to enforce the model with answer position-awareness, still in a Multi-Task Learning fashion. We borrow the bidirectional Attention Flow network and output layer from BiDAF model (Seo et al. 2016) to infer the answer position as shown in Figure 3, taking the sentence representation $H \in \mathbb{R}^{M*D}$ and question representation $S \in \mathbb{R}^{N*D}$ from the encoder and the decoder as inputs.

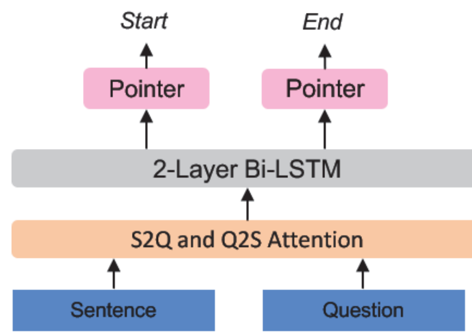

Figure 3: Framework of Answer Position Inferring

Formally, we take Sentence-to-Question (S2Q) attention and Question-to-Sentence (Q2S) attention to emphasize the mutual semantic relevance between each sentence word and each question word, and we obtain the question-aware sentence representation $\tilde{H}$ and the sentence-aware question representation $\tilde{S}$ by using similar attention mechanism to Equation 9:

$$\tilde{H} = attn(H, S), \tag{18}$$

$$\tilde{S} = attn(S, H) \tag{19}$$

Then, two two-layer bidirectional LSTMs are used to capture the interactions among the sentence words conditioned on the question (Seo et al. 2016). The answer starting index and end index are predicted by the output layer with

Table 3: Comparison of models performances in terms of the main metrics on SQuAD and MARCO dataset

	SQuAD						MARCO					
Models	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L
NQG++ (Zhou et al. 2017)	42.13	25.98	18.24	13.29	17.59	40.75	46.62	32.67	22.98	16.13	20.22	46.35
Pointer-generator (See, Liu, and Manning 2017)	42.43	26.75	18.99	14.33	18.77	43.19	47.10	34.26	24.87	17.95	22.34	47.69
Answer-focused (Sun et al. 2018)	43.02	28.14	20.51	15.64	-	-	-	-	-	-	-	-
Gated Self-attention (Zhao et al. 2018)	44.51	29.07	21.06	15.82	19.67	44.24	-	-	-	-	-	-
Model with Sentence-level Semantic Matching	43.67	28.53	20.59	15.66	19.23	43.86	48.97	35.84	26.31	19.79	23.83	48.93
Model with Answer Position Inferring	43.88	28.55	28.87	15.77	19.55	43.98	49.73	36.77	26.46	20.14	24.22	49.33
Combined Model	**44.71**	**29.89**	**21.77**	**16.32**	**20.84**	**44.79**	**50.33**	**37.10**	**27.23**	**20.46**	**24.69**	**49.89**

$Softmax$ function:

$$M_1 = LSTM(f(H, \tilde{H}, \tilde{S})), \tag{20}$$

$$M_2 = LSTM(f(H, \tilde{H}, \tilde{S})), \tag{21}$$

$$p^1 = \text{Softmax}\left(W_{(p^1)}^{\top}[\tilde{H}, M_1]\right), \tag{22}$$

$$p^2 = \text{Softmax}\left(W_{(p^2)}^{\top}[\tilde{H}, M_2]\right) \tag{23}$$

where W_{p^1} and W_{p^2} are trainable weights, and f function is a trainable multi-layer perception (MLP) network.

We compute the loss with the negative log likelihood of the ground truth answer starting index y_i^1 and ending index y_i^2 with the predicted distribution:

$$L(\theta_{ap}) = -\frac{1}{N}\sum_{i}^{N} \log\left(p^1_{y_i^1}\right) + \log\left(p^2_{y_i^2}\right) \tag{24}$$

where θ_{ap} is the parameters to be updated of the answer position inferring module.

To joint train the generation model with the proposed modules in a Multi-Task Learning approach, we minimize the total loss during the training:

$$L(\theta) = L(\theta_{s2s}) + \alpha * L(\theta_{sm}) + \beta * L(\theta_{ap}) \tag{25}$$

where α and β control the magnitude of the sentence-level semantic matching loss and the answer position inferring loss. By minimizing the above loss function, our model is expected to discover the sentence-level and answer position-aware semantics of the question and sentence.

Experiments and Results

In this section, we conduct extensive experiments on the SQuAD and MS MARCO dataset, demonstrating the superiority of our proposed model compared with existing approaches.

Experiment Settings

Dataset SQuAD V1.1 dataset contains 536 Wikipedia articles and more than 100K questions posed about the articles (Rajpurkar et al. 2016). The answer is also given with corresponding questions as the sub-span of the sentence. Following the baseline (Zhou et al. 2017), we use the training dataset (86635) to train our model, and we split the dev dataset into dev (8965) and test dataset (8964) with a ratio of 50%-50% for evaluation.

MS MARCO contains more than one million queries along with answers either generated by human or selected from passages (Nguyen et al. 2016). We select a subset of MS MARCO, where the answers are sub-spans of the passages. We split them into train set (86039), dev set (9480), and test set (7921) for model training and evaluation purpose.

We report automatic evaluation with BLEU-1, BLEU-2, BLEU-3, BLEU-4 (Papineni et al. 2002), METEOR (Denkowski and Lavie 2014), and ROUGE-L (Lin 2004) as the main metrics.

Baselines In the experiments, we have several baselines for comparisons:

- NQG++ (Zhou et al. 2017): It is a baseline for Neural Question Generation task. It uses enriched semantic and lexical features in the encoder embedding layer of the seq-to-seq model. Attention mechanism and copy mechanismare also used.
- Feature-enriched Pointer-generator (See, Liu, and Manning 2017): It is a seq-to-seq model with attention mechanism and copy mechanism. The copy mechanism is realized differently from NQG++. We add enriched features used in NQG++ in the embedding layer.
- Answer-focused (Sun et al. 2018): It is a SOTA model on QG that uses an additional vocabulary for question word generation with relative answer position information instead of BIO used in NQG++.
- Gated Self-attention (Zhao et al. 2018). It is also a SOTA model on QG that leverages paragraph as input with gated self-attention above RNN in the encoder. Meanwhile, an improved maxout pointer is introduced.

Results and Analysis

Main Metrics We report the main metrics of different models on SQuAD and MS MARCO dataset in Table 3.

Answer-focused model (Sun et al. 2018) improves the performance by using separate vocabulary for question word generation along with answer relative position. The Gated Self-Attention model (Zhao et al. 2018) emphasizes the intra-attention among the sentence with improved maxout pointer.

Different from the models above, our work aims to improve the model by learning the sentence-level semantic-matching features on both the encoder and decoder sides. The result shows that our model outperforms the two SOTA models on the main metrics.

Auxiliary Metrics Although the main metrics can reflect the similarity between the generated question and the refer-

Table 4: Machine Comprehension Performance in terms of Exact Match (EM) and F1on SQUAD dataset

Questions	EM (%)	F1 (%)
Reference Questions	49.68	65.97
NQG++ (Zhou et al. 2017)	35.26	50.88
Pointer-generator (See, Liu, and Manning 2017)	38.89	54.06
Our model	**42.70**	**57.68**

ences, it has its limits on reflecting the semantics of generated question (Xu et al. 2018).

Alternatively, considering that machine comprehension takes the article and the corresponding question as the input to find the answer in the passages, we adopt the machine comprehension metrics (Rajpurkar et al. 2016) to evaluate the quality of the questions generated by different models (Wang et al. 2017).

We show the performances of BiDAF (Seo et al. 2016) pre-trained by AllenNLP (Gardner et al. 2017) in terms of Exact Match (EM) and F1 metrics on reference questions, questions generated by baseline, and questions generated by our model in Table 4.

Our model outperforms NQG++ and Pointer-generator on EM and F1 significantly, since our model generates more answer-relevant questions by discovering sentence-level semantics and answer position features.

Sentence-level Semantic Matching Analysis To analyze the quality of our model on generating the right question words and keywords, we randomly sample 200 questions generated by NQG++, Pointer-generator, and our model, respectively. Generally, the generated question is claimed to have the right question words if it has the same question words to the reference question. For example, we have a generated question "what place ..." and a reference question "where ...", and we claim that the model generate a question with the right question words. In addition, we choose the words with most semantics importance as the keywords, which indicate the sentence topic and content. We report the number of the questions with right question words and keywords by different models in Table 5.

Table 5: Question words and keywords generation performance by different models on SQuAD dataset

Models	# right question words	# right keywords
NQG++ (Zhou et al. 2017)	134	143
Pointer-generator (See, Liu, and Manning 2017)	140	148
Model with Sentence-level Semantic Matching	**150**	**156**

The main reason that our model outperforms the existing model is that learning the sentence-level semantics helps to capture the key semantics and results in better performance on generating the semantic-matching keywords.

Answer Position Inferring Analysis We also conduct the similar experiment on evaluating the copy mechanisms in different models in terms of precision and recall used in (Sun et al. 2018). Given one generated question G and reference question R, we definite precision and recall as:

$$\text{Precision} = \frac{\sum_i^N \text{\# OOV words in both } G_i \text{ and } R_i}{\sum_i^N \text{\# OOV words in } R_i} \tag{26}$$

$$\text{Recall} = \frac{\sum_i^N \text{\# OOV words in both } G_i \text{ and } R_i}{\sum_i^N \text{\# OOV words in } R_i} \tag{27}$$

Table 6: Copy mechanism performance by different models

Models	Precision	Recall
NQG++ (Zhou et al. 2017)	46.28%	32.13%
Pointer-generator (See, Liu, and Manning 2017)	47.21%	38.38%
Model with Answer Position Inferring	**48.35%**	**40.27%**

As reported in Table 6, the improvement of Precision and Recall indicates that answer position inferring can help copy OOV words from the input sentence.

Model Generality To show the effectiveness and generality of our work, we evaluate the validness of our work by applying it to current representative models without revising the models. As shown in Table 7, our work can improve existing models by more than 2% on QG tasks due to its effectiveness and generality.

Table 7: Performance Improvement on existing models on SQuAD dataset

Models	BLEU-4
NQG++ (Zhou et al. 2017)	13.29
NQG++ (Zhou et al. 2017) + our work	**14.97**
Pointer-generator model (See, Liu, and Manning 2017)	14.33
Pointer-generator model (See, Liu, and Manning 2017) + our work	**16.32**

Human Evaluation

We also conduct human evaluation to examine the quality of the questions generated by the models and reference questions by scoring them on a scale of 1 to 5 in terms of semantics matching, fluency, and syntactically correctness. As reported in Table 8, our model generates questions with higher scores on the three metrics than the two baseline models, indicating the superiority of our proposed model by utilizing the sentence-level semantics and answer position-awareness.

Table 8: Human evaluation on questions generated by the models

Models	Semantic Matching	Fluency	Syntactically Correctness
NQG++	1.88	2.70	3.26
Pointer-generator	2.34	3.2	3.5
Our model	**2.87**	**3.46**	**3.89**

Case Study

In this section, we present some examples of questions generated by our model.

Furthermore, we present a pair of examples, which have the same input sentence in Table 9. Different from that

Sentence: another example was the insignificance of the ministry of war compared with native chinese dynasties , as the real military authority in yuan times resided in the privy council.
Reference: who had no real military power during the yuan?
NQG++: the insignificance of *what war* was compared to native chinese dynasties?
Pointer-generator: what was the insignificance?
Our model: what was insignificance compared with native Chinese dynasties?
Sentence: another example was the insignificance of the ministry of war compared with native chinese dynasties , as the real military authority in yuan times resided in the privy council.
Reference: who had military control during the yuan?
NQG++: what did the chinese dynasties *call the insignificance of the ministry of war*?
Pointer-generator: in where *the insignificance as* the real military authority in yuan times?
Our model: the real military authority in yuan times resided where?

Table 9: Examples of questions with asking about the right keywords generated by our model.

NQG++ generate similar and non-semantic-matching questions, our model can ask different and more semantic-matching questions than baselines, targeting the different answers.

Implementation Details

Followed NQG++ (Zhou et al. 2017), we conduct our experiments on the preprocessed data provided by (Zhou et al. 2017). We use 1 layer LSTM as the RNN cell for both the encoder and the decoder, and a bidirectional LSTM is used for the encoder. The hidden size of the encoder and decoder are 512. We use a 300-dimension pre-trained Glove vector as the word embedding (Pennington, Socher, and Manning 2014). As same as NQG++ (Zhou et al. 2017), the dimensions of lexical features and answer position are 16. We use Adam (Kingma and Ba 2014) Optimizer for model training with an initial learning rate as 0.001, and we halve it when the validation score does not improve. During the training of Sentence-level Semantic Matching module, we sample the negative sentences and questions from nearby data samples in the same batch, due to the preprocessed data (Zhou et al. 2017) lacking of the information about which data samples are from the same passage. We compute our total loss function with α of 1 and β of 2. Models are trained for 20 epochs with mini-batch of size 32. We choose model achieving the best performance on the dev dataset.

Related Work

Question generation tasks can be categorized into two classes: one is the rule-based method, meaning manually design lexical rules or templates to convert context into questions without deep understanding on the context semantic (Mazidi and Nielsen 2014; Labutov, Basu, and Vanderwende 2015). The other one is neural network based methods, which adopt seq-to-seq (Sutskever, Vinyals, and Le 2014) or an encoder-decoder (Cho et al. 2014) framework to generate question words from scratches (Du, Shao, and Cardie 2017; Zhou et al. 2017). Our work focuses on the second category.

(Du, Shao, and Cardie 2017) firstly proposes to generate question with a seq-to-seq model given a context automatically. However, the model does not take the answer into consideration. Then (Zhou et al. 2017) proposes to use a feature-enriched encoder to encode the input sentence by concatenating word embedding with lexical features as the encoder input, and answer position are involved in informing the model where the answer is. It is shown that it brings considerable improvements to the model. With the success of reinforcement learning, (Yuan et al. 2017) propose to combine supervised learning and reinforcement learning together for question generation by using policy gradient after training the model in supervised learning way. The reward term in the policy gradient loss function can be perplexity and the BLEU scores (Papineni et al. 2002). To tackle the issue that question words do not match with the answer type, (Sun et al. 2018) introduce a vocabulary only to generate question words. (Zhao et al. 2018) propose to use paragraph as the input for providing more semantic information with an improved maxout pointer for copying words from the input.

Different from existing methods focusing on utilizing more informative features and improving the copy mechanism, we point out that incapability of capturing sentence-level semantics and exploiting answer-aware features are the main reasons, and we alleviate the problem by proposing two modules which can be integrated with any base models named sentence-level semantic matching and answer position inferring in Multi-Task Learning fashion.

Conclusion

In this paper, we observe two issues with the widely used baseline model on question generation. We point out the root cause is that existing models neither consider the whole question semantics nor exploit the answer position-aware features well. To address the issue, we propose the neural question generation model with sentence-level semantic matching, answer position inferring, and gated fusion. Extensive experimental results show that our work improves existing models significantly and outperforms the SOTA models on SQuAD and MARCO datasets.

Acknowledgement

This research was supported by CBL industry and agency members and by the IUCRC Program of the National Science Foundation under Grant No. CNS-1747783.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Brill, E. 1992. A simple rule-based part of speech tagger. In *Proceedings of the third conference on Applied natural language processing*, 152–155. Association for Computational Linguistics.

Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Denkowski, M., and Lavie, A. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, 376–380.

Du, X., and Cardie, C. 2018. Harvesting paragraph-level question-answer pairs from wikipedia. *arXiv preprint arXiv:1805.05942*.

Du, X.; Shao, J.; and Cardie, C. 2017. Learning to ask: Neural question generation for reading comprehension. *arXiv preprint arXiv:1705.00106*.

Gardner, M.; Grus, J.; Neumann, M.; Tafjord, O.; Dasigi, P.; Liu, N. F.; Peters, M.; Schmitz, M.; and Zettlemoyer, L. S. 2017. Allennlp: A deep semantic natural language processing platform.

Gulcehre, C.; Ahn, S.; Nallapati, R.; Zhou, B.; and Bengio, Y. 2016. Pointing the unknown words. *arXiv preprint arXiv:1603.08148*.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Kim, Y.; Lee, H.; Shin, J.; and Jung, K. 2018. Improving neural question generation using answer separation. *arXiv preprint arXiv:1809.02393*.

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

Labutov, I.; Basu, S.; and Vanderwende, L. 2015. Deep questions without deep understanding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, volume 1, 889–898.

Lin, C.-Y. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, 74–81.

Mazidi, K., and Nielsen, R. D. 2014. Linguistic considerations in automatic question generation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, 321–326.

Nguyen, T.; Rosenberg, M.; Song, X.; Gao, J.; Tiwary, S.; Majumder, R.; and Deng, L. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

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. Association for Computational Linguistics.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

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

Sang, E. F., and De Meulder, F. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

See, A.; Liu, P. J.; and Manning, C. D. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Seo, M.; Kembhavi, A.; Farhadi, A.; and Hajishirzi, H. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Sun, X.; Liu, J.; Lyu, Y.; He, W.; Ma, Y.; and Wang, S. 2018. Answer-focused and position-aware neural question generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 3930–3939.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, 3104–3112.

Wang, W.; Yang, N.; Wei, F.; Chang, B.; and Zhou, M. 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: Long Papers)*, 189–198.

Xu, Z.; Jiang, N.; Liu, B.; Rong, W.; Wu, B.; Wang, B.; Wang, Z.; and Wang, X. 2018. Lsdscc: A large scale domain-specific conversational corpus for response generation with diversity oriented evaluation metrics. In *Proceedings of NAACL-HLT*, 2070–2080.

Yuan, X.; Wang, T.; Gulcehre, C.; Sordoni, A.; Bachman, P.; Subramanian, S.; Zhang, S.; and Trischler, A. 2017. Machine comprehension by text-to-text neural question generation. *arXiv preprint arXiv:1705.02012*.

Zhao, Y.; Ni, X.; Ding, Y.; and Ke, Q. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 3901–3910.

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 Conference on Natural Language Processing and Chinese Computing*, 662–671. Springer.