Paragraph-level Neural Question Generation with Maxout Pointer and Gated Self-attention Networks

Yao Zhao Xiaochuan Ni Yuanyuan Ding Qifa Ke
Microsoft AI & Research
Sunnyvale, California
{yaozhao, xiaon, yuand, qke}@microsoft.com

Abstract

Question generation, the task of automatically creating questions that can be answered by a certain span of text within a given passage, is important for question-answering and conversational systems in digital assistants such as Alexa, Cortana, Google Assistant and Siri. Recent sequence to sequence neural models have outperformed previous rule-based systems. Existing models mainly focused on using one or two sentences as the input. Long text has posed challenges for sequence to sequence neural models in question generation – worse performances were reported if using the whole paragraph (with multiple sentences) as the input. In reality, however, it often requires the whole paragraph as context in order to generate high quality questions. In this paper, we propose a maxout pointer mechanism with gated self-attention encoder to address the challenges of processing long text inputs for question generation. With sentence-level inputs, our model outperforms previous approaches with either sentence-level or paragraph-level inputs. Furthermore, our model can effectively utilize paragraphs as inputs, pushing the state-of-the-art result from 13.9 to 16.3 (BLEU_4).

1 Introduction

Question generation (QG), aiming at creating questions from natural language text, e.g. a sentence or paragraph, is an important area in natural language processing (NLP). It is receiving increasing interests in recent years from both industrial and academic communities, due to the booming of Question-and-Answer (QnA) and conversation systems, including Alexa, Cortana, Google Assistant and Siri, the advancement of QnA or machine comprehension technologies together with the releases of datasets like SQuAD (Rajpurkar et al., 2016) and MS MARCO (Nguyen et al., 2016), and the success of language generation technologies for tasks like machine translation (Wu et al., 2016) and text summarization (See et al., 2017) in NLP. A conversational system can be proactive by asking the user questions (Shum et al., 2018), while a QnA system can benefit from a large scale question-answering corpus which can be created by an automated QG system (Duan et al., 2017). Education is another key application where QG can help with reading comprehension (Heilman and Smith, 2010).

In NLP, QG has been mainly tackled by two approaches: 1) rule-based approach, e.g. (Heilman and Smith, 2010; Mazidi and Nielsen, 2014; Labutov et al., 2015) 2) neural QG approach: end-to-end training a neural network using the sequence to sequence (also called encoder-decoder) framework, e.g. (Du et al., 2017; Yuan et al., 2017; Song et al., 2017; Zhou et al., 2017). In this paper, we adopt the second approach.

More specifically, we focus on an answer-aware QG problem, which takes a passage and an answer as inputs, and generates a question that targets the given answer. It is also assumed the answer is comprised of certain spans of the text from the given passage. This is the exact setting of SQuAD, and similar problems have been addressed in, e.g. (Zhou et al., 2017; Yuan et al., 2017; Wang et al., 2017a).

A paragraph often contains much richer context than a sentence, as shown in Figure 1. (Du et al., 2017) pointed out that about 20% questions in SQuAD require paragraph-level information to be asked and using the whole paragraph can improve QG performance on those questions. However, a paragraph can contain irrelevant information w.r.t. the answer for generating the question. The challenge is thus how to effectively utilize relevant information at paragraph-level for QG. Existing neural QG works conducted on SQuAD use

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 3901–3910
Brussels, Belgium, October 31 - November 4, 2018.

1 **Paragraph**: carolina suffered a major setback when thomas davis , an 11-year veteran who had already overcome three acl tears in his career , went down with a broken arm in the nfc championship game . *despite this , he insisted he would still find a way to play in the super bowl* . his prediction turned out to be accurate
Human generated: what game did thomas davis say he would play in , despite breaking a bone earlier on ?
Sentence-level QG: what sports game did spielberg decide to play in ?
Paragraph-level QG: what competition did thomas davis think he would play in ?

2 **Paragraph**: walt disney and his brother roy contacted goldenson at the end of 1953 for abc to agree to finance part of the disneyland project in exchange for producing a television program for the network . *walt wanted abc to invest $ 500,000 and accrued a guarantee of $ 4.5 million in additional loans , a third of the budget intended for the park* . around 1954 , abc agreed to finance disneyland in exchange for the right to broadcast a new sunday night program , disneyland , which debuted on the network on october 27 , 1954 as the first of many anthology television programs that disney would broadcast over the course of the next 50 years
Human generated: how much did walt disney want abc to invest in disneyland ?
Sentence-level QG: how much money did walt wanted to invest ?
Paragraph-level QG: how much money did walt wanted to invest in 1953 ?

3 **Paragraph**: following the peterloo massacre of 1819 , poet percy shelley wrote the political poem the mask of anarchy later that year , that begins with the images of what he thought to be the unjust forms of authority of his timeand then imagines the stirrings of a new form of social action . *it is perhaps the first modern [vague] statement of the principle of nonviolent protest* . a version was taken up by the author henry david thoreau in his essay civil disobedience , and later by gandhi in his doctrine of satyagraha . gandhi 's satyagraha was partially influenced and inspired by shelley 's nonviolence in protest and political action . in particular , it is known that gandhi would often quote shelley 's masque of anarchy to vast audiences during the campaign for a free india
Human generated: his poem is considered the first kind of what type of protest ?
Sentence-level QG: what is the principle of the protest ?
Paragraph-level QG: what type of protest did percy shelley write ?

4 **Paragraph**: the victoria and albert museum (often abbreviated as the v & a) , london , is the world 's largest museum of decorative arts and design , housing a permanent collection of over 4.5 million objects . *it was founded in 1852 and named after queen victoria and prince albert.* the v & a is located in the brompton district of the royal borough of kensington and chelsea , in an area that has become known as " albertopolis " because of its association with prince albert , the albert memorial and the major cultural institutions with which he was associated . these include the natural history museum , the science museum and the royal albert hall . the museum is a non-departmental public body sponsored by the department for culture , media and sport . like other national british museums , entrance to the museum has been free since 2001
Human generated: when was the victoria and albert museum founded ?
Sentence-level QG: when was prince albert and prince albert founded ?
Paragraph-level QG: when was the victoria and albert museum founded ?

Figure 1: Examples where paragraph-level information is required to ask right questions. *Sentences* contain *answers* are in *italic* font, while *answers* are underscored. QG results are generated by model *s2s-a-ct-mp-gsa*.

only a sentence as context, e.g. (Du et al., 2017; Song et al., 2017; Zhou et al., 2017); when applied to paragraph-level context (Yuan et al., 2017) we observed large gaps compared to state-of-the-art results achieved by using sentence-level context.

In this paper, we extend previous sequence to sequence attention model with a maxout pointer mechanism and a gated self-attention encoder which outperforms existing neural QG approaches with either sentence or paragraph as inputs. Furthermore, with paragraph-level inputs, it outperforms the results of previous approaches with sentence-level inputs, improving state-of-the-art result from 13.9 to 16.3 (BLEU_4). This is the first model that demonstrates large improvement with paragraph as input over sentence as input. In addition, our model is more concise compared to most of existing ones, e.g. (Yuan et al., 2017; Song et al., 2017). Techniques like incorporating rich features (Zhou et al., 2017) and policy gradient (Song et al., 2017; Yuan et al., 2017) are orthogonal to ours and can be leveraged for further performance improvement in the future.

2 Our Model

2.1 Problem Definition

We use P and A to represent input passage and answer respectively, and use Q to represent the generated question. "Passage" in this section can represent either a sentence or a paragraph. The task is to find $\bar{Q}$ that:

$$\bar{Q} = \underset{Q}{\operatorname{argmax}}\, Prob(Q|P, A)$$

where passage is comprised of sequence of words:$P = \{x_t\}_{t=1}^{M}$, answer A must be sub spans of the passage. Words generated in $Q = \{y_t\}_{t=1}^{K}$ are either from the input passage, $\{x_t\}_{t=1}^{M}$, or from a vocabulary V.

Figure 2 illustrates the end-to-end structure of our model proposed in this paper.

2.2 Passage and Answer Encoding

Different types of encoders are designed for various domains (Chung et al., 2014; Hochreiter and Schmidhuber, 1997). We are agnostic to the form of the encoder and simply use recurrent neural network (RNN) to present the encoding process:

$$\mathbf{u}_t = RNN^E(\mathbf{u}_{t-1}, [\mathbf{e}_t, \mathbf{m}_t]) \quad (1)$$

Answer Tagging. In Eq. 1, $\mathbf{u}_t$ represents the RNN hidden state at time step t, $\mathbf{e}_t$ is the word embedding representation of word x_t in passage P. $\mathbf{m}_t$ is the meta-word representation of whether word x_t is in or outside the answer. $[\mathbf{a}, \mathbf{b}]$ represents the concatenation of vector $\mathbf{a}$ and $\mathbf{b}$. We call this approach answer tagging which is similar to the techniques in (Zhou et al., 2017; Yuan et al., 2017). For applications, it is essential to be able to generate question that is coherent to an answer.

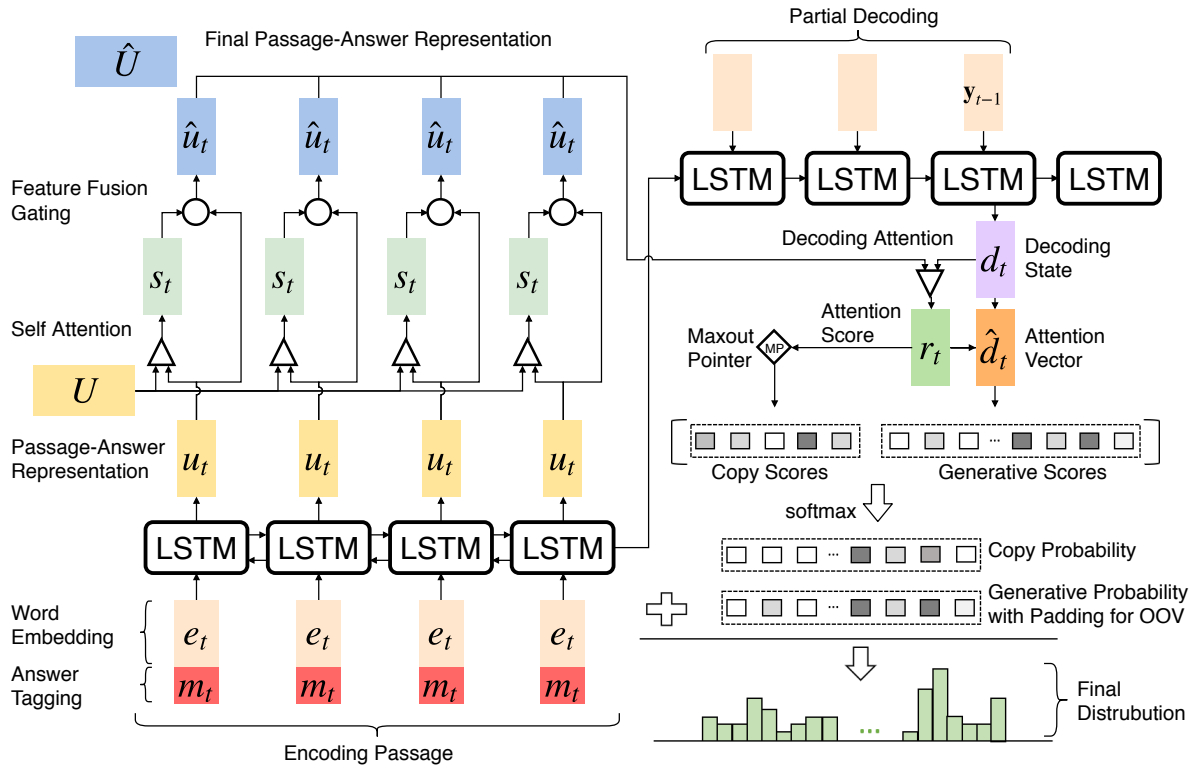

Figure 2: End-to-end diagram for the model with answer tagging, gated self-attention and maxout pointer mechanism.

If RNN^E is bi-directional, $\mathbf{u}$ is the concatenated representation of the forward and backward passes: $\mathbf{U} = \{[\overrightarrow{\mathbf{u}_t}, \overleftarrow{\mathbf{u}_t}]\}_{t=1}^{M}$.

Gated Self-attention. Our gated self-attention mechanism is designed to aggregate information from the whole passage and embed intra-passage dependency to refine the encoded passage-answer representation at every time step. It has two steps: 1) taking encoded passage-answer representation $\mathbf{u}$ as input and conducting matching against itself to compute self matching representation; (Wang et al., 2017b) 2) combining the input with self matching representation using a feature fusion gate (Gong and Bowman, 2017).

$$\mathbf{a^s}_t = softmax(\mathbf{U}^\mathsf{T}\mathbf{W}^s\mathbf{u}_t) \quad (2)$$

$$\mathbf{s}_t = \mathbf{U} \cdot \mathbf{a^s}_t \quad (3)$$

Step 1. In Eq. 2, $\mathbf{W}^s$ is a trainable weight matrix. In Eq. 3, $\mathbf{s}_t$ is the weighted sum of all words' encoded representation in passage based on their corresponding matching strength to current word at t. $\mathbf{s} = \{\mathbf{s}_t\}_{t=1}^{M}$ is the final self matching representation.

$$\mathbf{f}_t = tanh(\mathbf{W}^f[\mathbf{u}_t, \mathbf{s}_t]) \quad (4)$$

$$\mathbf{g}_t = sigmoid(\mathbf{W}^g[\mathbf{u}_t, \mathbf{s}_t]) \quad (5)$$

$$\hat{\mathbf{u}}_t = \mathbf{g}_t \odot \mathbf{f}_t + (1 - \mathbf{g}_t) \odot \mathbf{u}_t \quad (6)$$

Step 2. The self matching representation $\mathbf{s}_t$ is combined with original passage-answer representation $\mathbf{u}_t$ as the new self matching enhanced representation $\mathbf{f}_t$, Eq. 4. A learnable gate vector $\mathbf{g}_t$, Eq. 5, chooses the information between the original passage-answer representation and the new self matching enhanced representation to form the final encoded passage-answer representation $\hat{\mathbf{u}}_t$, Eq. 6, where $\odot$ is the element-wise multiplication.

2.3 Decoding with Attention and Maxout Pointer

In the decoding stage, the decoder is another RNN that generates words sequentially conditioned on the encoded input representation and the previously decoded words.

$$\mathbf{d}_t = RNN^D(\mathbf{d}_{t-1}, \mathbf{y}_{t-1}) \quad (7)$$

$$p(y_t|\{y_{<t}\}) = softmax(\mathbf{W}^V\mathbf{d}_t) \quad (8)$$

In Eq. 7, $\mathbf{d}_t$ represents the hidden state of the RNN at time t where $\mathbf{d}_0$ is passed from the final hidden state of the encoder. y_t stands for the word generated at time t. The bold font $\mathbf{y}_t$ is used to represent y_t's corresponding word embedding representation. In Eq. 8, first an affine layer projects $\mathbf{d}_t$ to a space with vocabulary-size dimensions, then a $softmax$ layer computes a probability distribution over all words in a fixed vocabulary V . $\mathbf{W}^V$ is a trainable weight matrix.

Attention. Attention mechanism (Bahdanau et al., 2014; Luong et al., 2015) has been used to improve sequence to sequence models' performance and has became a default setting for many applications.

We use Luong attention mechanism to compute raw attention scores $\mathbf{r}_t$, Eq. 9. An attention layer, Eq. 12 is applied above the concatenation of decoder state $\mathbf{d}_t$ and the attention context vector $\mathbf{c}_t$ and its output is used as the new decoder state.

$$\mathbf{r}_t = \hat{\mathbf{U}}^\mathsf{T} \mathbf{W}^a \mathbf{d_t} \tag{9}$$

$$\mathbf{a^d}_t = softmax(\mathbf{r}_t) \tag{10}$$

$$\mathbf{c}_t = \hat{\mathbf{U}} \cdot \mathbf{a^d}_t \tag{11}$$

$$\hat{\mathbf{d}}_t = tanh(\mathbf{W}^b[\mathbf{d}_t, \mathbf{c}_t]) \tag{12}$$

Copy/Pointer. Copy mechanism (Gu et al., 2016) or pointer network (See et al., 2017; Vinyals et al., 2015) was introduced to allow both copying words from input via pointing, and generating words from a predefined vocabulary during decoding.

Similar to (Gu et al., 2016), our pointer mechanism directly leverages raw attention scores $\mathbf{r}_t = \{r_{t,k}\}_{k=1}^M$ over the input sequence which has a vocabulary of χ. Words at every time step (a pointer) are treated as unique copy targets and the final score on one word is calculated as the sum of all scores pointing to the same word, Eq. 13, where x_k and y_t stand for word vocabulary indices of the kth word in input and the tth word in decoded sequence respectively. The scores of the nonoccurence words are set to negative infinity which will be masked out by the downstream $softmax$ function.

$$\mathbf{sc}^{copy}(y_t) = \begin{cases} \sum\limits_{k,where\ x_k=y_t} r_{t,k}, & y_t \in \chi \\ -inf, & otherwise \end{cases} \tag{13}$$

We then concatenate $\mathbf{sc}_t^{copy}$ with the generative scores (from Eq. 8 before $softmax$), $[\mathbf{sc}_t^{gen}, \mathbf{sc}_t^{copy}]$, which has dimension: $|V| + |\chi|$. Then we perform $softmax$ on the concatenated vectors and sum up the probabilities pointing to same words. Taking $softmax$ on the concatenated score vector enforces copy and generative modes to compete with each other due to the shared normalization denominator. Another popular solution is to do $softmax$ independently to the scores from each mode, and then combine their output probabilities with a dynamic weight which is generated by a trainable network (See et al., 2017). We have tested both and didn't find significant difference in terms of accuracy on our QG task. We choose the former copy approach mainly because it doesn't add extra trainable parameters.

Maxout Pointer. Despite the outstanding performance of existing copy/pointer mechanisms, we observed that repeated occurrence of words in the input sequence tends to cause repetitions in output sequence, especially when the input sequence is long, e.g. a paragraph. This issue exacerbates the repetition problem which has already been commonly observed in sequence to sequence models (Tu et al., 2016; See et al., 2017). In this paper, we propose a new maxout pointer mechanism to address this issue and improve the metrics for QG task. Related works (Goodfellow et al., 2013) have explored MLP maxout with dropout.

Instead of combining all the scores to calculate the probability, We limit the magnitude of scores of repeated words to their maximum value, as shown in Eq. 14. The rest remains the same as in the previous copy mechanism.

$$\mathbf{sc}^{copy}(y_t) = \begin{cases} \max\limits_{k,where\ x_k=y_t} r_{t,k}, & y_t \in \chi \\ -inf, & otherwise \end{cases} \tag{14}$$

3 Experiments

In our experiments, we study the proposed model on the QG task on SQuAD (Rajpurkar et al., 2016) and MS MARCO (Nguyen et al., 2016) dataset, demonstrate the performance of proposed components on both sentence and paragraph inputs, and compare the model with existing approaches.

3.1 Dataset

SQuAD

The SQuAD dataset contains 536 Wikipedia articles and more than 100K questions posed about the articles by crowd-workers. Answers are also

	BLEU_1		BLEU_2		BLEU_3		BLEU_4		METEOR		ROUGE-L	
Model	**Sen.**	**Par.**	**Sen.**	**Par.**	**Sen.**	**Par.**	**Sen.**	**Par.**	**Sen.**	**Par.**	**Sen.**	**Par.**
s2s	30.41	28.49	12.68	10.43	6.33	4.70	3.44	2.38	11.98	10.69	29.93	27.32
s2s-a	34.46	31.26	18.07	14.37	11.20	8.02	7.42	4.80	14.95	12.52	34.69	30.11
s2s-a-at	40.57	40.56	24.30	24.23	16.40	16.33	11.54	11.46	18.35	18.42	40.76	40.40
s2s-a-at-cp	42.15	41.66	26.28	25.52	18.35	17.48	13.37	12.43	18.02	17.76	41.97	41.30
s2s-a-at-mcp	**43.65**	44.22	**28.23**	28.56	20.33	20.57	15.23	15.43	19.19	19.55	43.60	43.65
s2s-a-at-mcp-gsa	43.47	**45.07**	**28.23**	**29.58**	**20.40**	**21.60**	**15.32**	**16.38**	**19.29**	**20.25**	**43.91**	**44.48**

Table 1: Performance of our models on **Split1** with both sentence-level input and paragraph-level input. **Sen.** means sentence, while **Par.** means paragraph.

provided to the questions, which are spans of tokens in the articles.

Following (Du et al., 2017; Zhou et al., 2017; Song et al., 2017), our experiments are conducted using the accessible part of SQuAD: train and development (dev*) sets. To be able to directly compare with their works, we adopt two types of data split: 1) **Split1**: similar to (Du et al., 2017), we use dev* set as test set, and split train set into train and dev sets randomly with ratio 90%-10%. The split is done at article level. However, we keep all samples instead of only keeping the sentence-question pairs that have at least one non-stop-word in common (with 6.7% pairs dropped) as in (Du et al., 2017). This makes our dataset harder for training and evaluation. 2) **Split2**: similar to (Zhou et al., 2017), we split dev* set into dev and test sets randomly with ratio 50%-50%. The split is done at sentence level.

MS MARCO

MS MARCO datasets contains 100,000 queries with corresponding answers and passages. All questions are sampled from real anonymized user queries and context passages are extracted from real web documents. We picked a subset of MS MARCO data where answers are sub-spans within the passages, and use dev set as test set (7k), and split train set with ratio 90%-10% into train (51k) and dev (6k) sets.

3.2 Implementation Details

We used 2 layers LSTM as the RNN cell for both encoding and decoding. For encoding, bi-directional LSTM was used. The cell hidden size was 600. Dropout with probability 0.3 was applied between vertical LSTM stacks. For word embedding, we used pre-trained GloVe word vectors with 300 dimensions (Pennington et al., 2014), and froze them during training. Dimension of answer tagging meta-word embedding was 3. Both encoder and decoder shared the same vocabulary of the most frequent 45k GloVe words. For optimization, we used SGD with momentum (Qian, 1999; Nesterov, 1983). Learning rate was initially set to 0.1 and halved since epoch 8 at every 2 epochs afterwards. Models were totally trained with 20 epochs. The mini-batch size for parameter update was 64. After training, we looked at the 4 models with lowest perplexities and selected the one which used the most number of epochs as final model. During decoding for prediction, we used beam search with the beam size of 10, and stopped decoding when every beam in the stack generates the EOS token.

3.3 Evaluation

We conduct automatic evaluation with metrics: BLEU_1, BLEU_2, BLEU_3, BLEU_4 (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2014) and ROUGE-L (Lin, 2004), and use evaluation package released by (Sharma et al., 2017) to compute them.

4 Results and Analysis

4.1 Comparison of Techniques

Table 1 shows evaluation results for different models on SQuAD Split1. Results with both sentence-level and paragraph-level inputs are included. Similar results also have been observed on SQuAD Split2. The definitions of the models under comparison are:

s2s: basic sequence to sequence model
s2s-a: *s2s* + attention mechanism
s2s-a-at: *s2s-a* + answer tagging
s2s-a-at-cp: *s2s-a-at* + copy mechanism

s2s-a-at-mp: *s2s-a-at* + maxout pointer mechanism
s2s-a-at-mp-gsa: *s2s-a-at-mp* + gated self-attention

Attention Mechanism

s2s-a vs. *s2s*: we can see attention brings in large improvement on both sentence and paragraph inputs. The lower performance on paragraph indicates the challenge of encoding paragraph-level information.

Answer Tagging

s2s-a-at vs. *s2s-a*: answer tagging dramatically boosts the performance, which confirms the importance of answer-aware QG: to generate good question, we need to control/learn which part of the context the generated question is asking about. More importantly, answer tagging clearly reduces the gap between sentence and paragraph inputs, which could be explained with: by providing guidance on answer words, we can make the model learn to neglect noise when processing a long context.

Copy Mechanism

s2s-a-at-cp vs. *s2s-a-at*: as expected, copy mechanism further improves the performance on the QG task. (Du et al., 2017) pointed out most of the sentence-question pairs in SQuAD have over 50% overlaps in non-stop-words. Our results prove that sequence to sequence models with copy mechanism can very well learn when to generate a word and when to copy one from input on such QG task. More interestingly, the performance is lower when paragraph is given as input than sentence as input. The gap, again, reveals the challenge of leveraging longer context. We found that, when paragraph is given, the model tends to generate more repetitive words, and those words (often entities/concepts) usually appear multiple times in the context, Figure 3. The repetition issue can also be seen for sentence input, but it is more severe for paragraph.

Maxout Pointer

s2s-a-at-mp vs. *s2s-a-at-cp*: Maxout pointer is designed to resolve the repetition issue brought by the basic copy mechanism, for example Figure 3. The maxout pointer mechanism outperforms the basic copy mechanism in all metrics. Moreover, the effectiveness of maxout pointer is more significant when paragraph is given as the model input, as it reverses the performance gap between models trained with sentence and paragraph inputs, Table 1.

Paragraph: a problem is regarded as inherently difficult if its solution requires significant resources , whatever the algorithm used . the theory formalizes this intuition , by introducing mathematical models of computation to study these problems and quantifying the amount of resources needed to solve them , such as time and storage . *other complexity measures are also used , such as the amount of communication (used in communication complexity) , the number of gates in a circuit (used in circuit complexity) and the number of processors (used in parallel computing)* . one of the roles of computational complexity theory is to determine the practical limits on what computers can and can not do
Human generated: what unit is measured to determine circuit complexity ?
Basic copy QG: what is an example of a circuit complexity in complexity complexity ?
Maxout Pointer QG: what is another name for circuit complexity ?

Figure 3: Example for maxout pointer vs. basic copy/pointer.

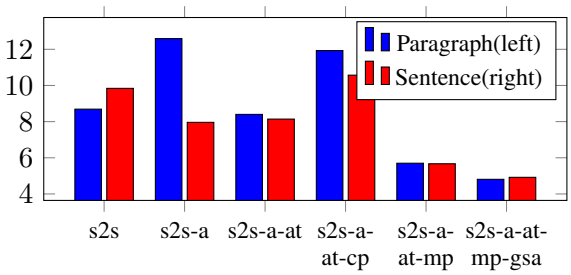

Figure 4: Word duplication rates of QG on Split1, duplication rate on human generated question is **3.53**.

To demonstrate that maxout pointer reduces repetitions in generated questions, we present the word duplication rates in generated questions for various models in Figure 4. Word duplication rate was computed by counting the number of words which appear more than once, and then taking a ratio of them over the total word counts. As shown in Figure 4, both the attention mechanism and the basic copy mechanism introduce more repetitions, although they improve overall accuracy according to Table 1. For models trained with paragraph inputs, where the duplication rates are much higher, maxout pointer reduces the duplication rates to half of their values in the basic copy and to the same level as model trained with sentence inputs.

Such repetition issue was also observed in other sequence to sequence applications, e.g. (See et al., 2017) who proposed a coverage model and coverage loss to resolve this issue. We implemented and tested their approach on our QG task. Even though the duplication ratio dropped as expected, we observed a slight decline in the accuracy when coverage loss was added.

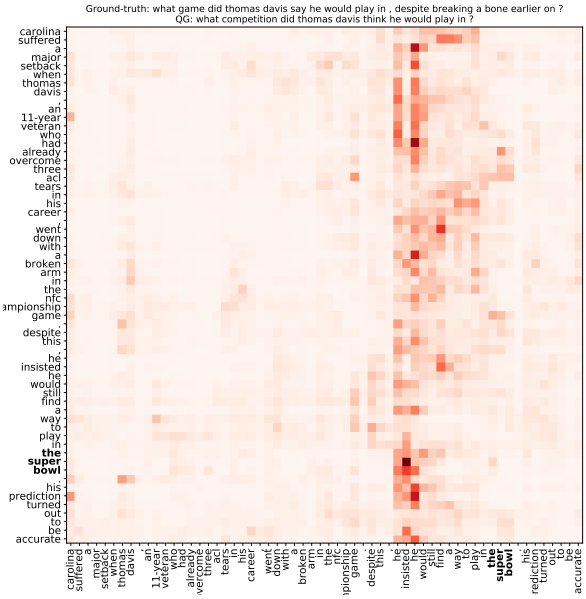

Figure 5: Self-attention alignments map: each row represents an alignment vector of self-attention.

Gated Self-attention

s2s-a-at-mp-gsa vs. *s2s-a-at-mp*: the results demonstrate the effectiveness of gated self-attention, in particular, when working with paragraph inputs. This is the first time, as we know, taking paragraph as input is better than sentence for neural QG tasks. The observation is consistent across all metrics. Gated self-attention helps refine encoded context by fusing important information with the context's self representation properly, especially when the context is long.

To better understand how gated self-attention works, we visualize the self alignment vectors at each time step of the encoded sequence for one example, in Figure 5. This example corresponds to the example 1 in Figure 1. We can see the alignments distribution concentrates near the answer sequence and the most relevant context: "thomas davis" in this example. Such alignments in turn would help to promote the most valuable information for decoding.

Beam Search

Beam search is commonly used for decoding for predictions. Existing neural QG works, e.g. (Du et al., 2017; Zhou et al., 2017), evaluated their models with beam search decoding. However, so far, none of them have reported the comparison between beam search decoding with greedy decoding. In this paper, we give such comparison for our best model: *s2s-a-at-mp-gsa* with both sentence and paragraph inputs in Table 2. We can clearly see beam search decoding boosts all metrics for both sentence and paragraph inputs. The effectiveness of beam search has also been demonstrated for other tasks, like neural machine translation in (Wu et al., 2016).

	Beam_Search		Greedy	
Metric	**Sen.**	**Par.**	**Sen.**	**Par.**
BLEU_1	43.47	45.07	42.25	43.48
BLEU_2	28.23	29.58	26.36	27.67
BLEU_3	20.40	21.60	18.35	19.64
BLEU_4	15.32	16.38	13.28	14.50
METEOR	19.29	20.25	18.25	19.17
ROUGE-L	43.91	44.48	42.58	43.62

Table 2: Comparison of beam search and greedy decodings for model **s2s-a-ct-mp-gsa** on Split1.

4.2 Comparison with Existing Neural Question Generation Works

On SQuAD dataset, we compare the BLEU, METEOR, ROUGE-L scores of our best model, *s2s-a-at-mp-gsa*, with the numbers in the existing works in Table 3. Comparison with (Du et al., 2017) and (Song et al., 2017) is conducted on SQuAD data Split1, while comparison with (Zhou et al., 2017) and (Song et al., 2017) is conducted on data SQuAD split2. Because (Song et al., 2017) had results on both splits, we compare with both of them.

On MS MARCO dataset, we compare the BLEU_4 scores reported by (Duan et al., 2017) in Table 4.

Our model with maxout pointer and gated self-attention achieves the state-of-the-art results in QG. Note that all those existing works in SQuAD encoded only sentence-level information, the results from our model surpass them on the same sentence input while achieving much higher numbers when working with paragraph.

4.3 Case Study

In Figure 1, we present some examples for which paragraph-level information is needed to ask good/correct questions. Generated questions from model *s2s-a-ct-mp-gsa* are also presented for both sentence and paragraph inputs. Those examples demonstrate that generated questions contain richer information when paragraphs are provided

	Model	BLEU_1	BLEU_2	BLEU_3	BLEU_4	METEOR	ROUGE-L
Split1	(Du et al., 2017)	43.09	25.96	17.50	12.28	16.62	39.75
	(Song et al., 2017)	x	x	x	13.98	18.77	42.72
	ours, sentence	43.47	28.23	20.40	15.32	19.29	43.91
	ours, paragraph	**45.07**	**29.58**	**21.60**	**16.38**	**20.25**	**44.48**
Split2	(Zhou et al., 2017)	x	x	x	13.27	x	x
	(Song et al., 2017)	x	x	x	13.91	x	x
	ours, sentence	44.51	29.07	21.06	15.82	19.67	44.24
	ours, paragraph	**45.69**	**30.25**	**22.16**	**16.85**	**20.62**	**44.99**

Table 3: Comparison of results on SQuAD dataset.

	Model	BLEU_4
MSMARCO	(Du et al., 2017)	10.46
	(Duan et al., 2017)	11.46
	ours, sentence	16.02
	ours, paragraph	**17.24**

Table 4: Comparison of results on MSMARCO dataset.

instead of sentences.

In example 1 and 3, name "thomas davis" and "percy shelley" appear in the paragraphs not the sentences contain the answers.

In example 2, paragraph-level QG can generate richer description "in 1953" from the paragraph, although human generated is "disneyland". Both generated questions lack one relative important piece of information "want **abc** to invest".

In example 4, paragraph-level QG correctly identifies "it" is referring to the museum which is out of the sentence.

5 Related Work

QG has been mainly tackled with two types of approaches. One is built on top of heuristic rules that creates questions with manually constructed template and ranks the generated results, e.g. (Heilman and Smith, 2010; Mazidi and Nielsen, 2014; Labutov et al., 2015). Those approaches heavily depend on human effort, which makes them hard to scale up to many domains. The other one, which is becoming increasingly popular, is to train an end-to-end neural network from scratch by using sequence to sequence or encoder-decoder framework, e.g. (Du et al., 2017; Yuan et al., 2017; Song et al., 2017; Zhou et al., 2017). The second one is more related to us, so we will focus on describing those approaches.

(Du et al., 2017) pioneered the work of automatic QG using an end-to-end trainable sequence to sequence neural model. Automatic and human evaluation results showed that their system outperformed the previous rule-based systems (Heilman and Smith, 2010; Rus et al., 2010). However, in their study, there was no control about which part of the passage the generated question was asking about.

Answer-aware sequence to sequence neural QG systems (Zhou et al., 2017; Subramanian et al., 2017; Yuan et al., 2017) encoded answer location information using an annotation vector corresponding to the answer word positions. (Zhou et al., 2017) utilized rich features of the passage including answer positions. (Subramanian et al., 2017) deployed a two-stage neural model that detects key phrases and subsequently generates questions conditioned on them. (Yuan et al., 2017) combined supervised and reinforcement learning in the training of their model using policy gradient techniques to maximize several rewards that measure question quality. Instead of using an annotation vector to tag the answer locations, (Song et al., 2017) proposed a unified framework for QG and question answering by encoding both the answer and the passage with a multi-perspective matching mechanism.

(Tang et al., 2017; Wang et al., 2017a) proposed joint models to address QG and question answering together. (Duan et al., 2017) conducted QG for improving question answering. Due to the mixed objectives including question answering, their approaches' performance on QG were lower than the state-of-the-art results.

6 Conclusion and Future Work

In this paper, we proposed a new sequence to sequence network which contains a gated self-

attention encoder and a maxout pointer decoder to address the answer-aware QG problem for long text input. We demonstrated the model can effectively utilize paragraph-level context, and outperformed the results with sentence-level context. The new model exceeded state-of-the-art approaches with either paragraph or sentence inputs.

We would like to discuss some potential challenges when applying the QG model in practice. 1) Answer spans aren't provided as input. One straight-forward method is to extract entities or noun phrases and use them as potential answer spans. A neural entity selection model can also be leveraged to extract good answer candidates to improve the precision as proposed in (Subramanian et al., 2017). 2) An input passage does not contain any eligible answers. In such case, we do not expect the model to output valid questions. We could remove questions with low generation probability, while a better approach could be running entity selection or quality detection model before question generation step to eliminate ineligible passages. 3) An answer could be shared by different questions. We could output multiple questions using beam search. However, beam search does not guarantee to output diversified candidates. We would need to explicitly model diversity among candidates during generation, for example, leveraging the approach described in (Li and Jurafsky, 2016).

Our future work lands in the following directions: incorporate rich features, such as POS and entity, in input passages; directly optimize sequence-level metrics with policy gradient; relax the constraint on answer to accept abstractive answers; jointly model question generation and question answering; ask multiple questions simultaneously with diverse perspectives.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*. Association for Computational Linguistics.

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

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Yichen Gong and Samuel R Bowman. 2017. Ruminating reader: Reasoning with gated multi-hop attention. *arXiv preprint arXiv:1704.07415*.

Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. 2013. Maxout networks. *arXiv preprint arXiv:1302.4389*.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*.

Michael Heilman and Noah A Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617. Association for Computational Linguistics.

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

Igor Labutov, Sumit Basu, and Lucy Vanderwende. 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*, page 889898. Association for Computational Linguistics.

Jiwei Li and Dan Jurafsky. 2016. Mutual information and diverse decoding improve neural machine translation. *arXiv preprint arXiv:1601.00372*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Stan Szpakowicz Marie-Francine Moens, editor, Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*. Association for Computational Linguistics.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Karen Mazidi and Rodney D. Nielsen. 2014. Linguistic considerations in automatic question generation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, page 321326. Association for Computational Linguistics.

Yurii Nesterov. 1983. A method for unconstrained convex minimization problem with the rate of convergence o (1/kˆ 2). In *Doklady AN USSR*, volume 269, pages 543–547.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. In *Proceedings of 30th Conference on Neural Information Processing Systems (NIPS)*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Ning Qian. 1999. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Vasile Rus, Brendan Wyse, Paul Piwek, Mihai Lintean, Svetlana Stoyanchev, and Cristian Moldovan. 2010. The first question generation shared task evaluation challenge. In *Proceedings of the 6th International Natural Language Generation Conference*, pages 251–257. Association for Computational Linguistics.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. 2017. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *CoRR*, abs/1706.09799.

Heung-Yeung Shum, Xiaodong He, and Di Li. 2018. From eliza to xiaoice: challenges and opportunities with social chatbots. *arXiv preprint arXiv:1801.01957*.

Linfeng Song, Zhiguo Wang, and Wael Hamza. 2017. A unified auery-based generative model for question generation and question answering. *arXiv preprint arXiv:1709.01058*.

Sandeep Subramanian, Tong Wang, Xingdi Yuan, and Adam Trischler. 2017. Neural models for key phrase detection and question generation. *arXiv preprint arXiv:1706.04560*.

Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027*.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Association for computation linguistics*.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Neural Information Processing Systems*.

Tong Wang, Xingdi (Eric) Yuan, and Adam Trischler. 2017a. A joint model for question answering and question generation. In *Learning to generate natural language workshop, ICML 2017*.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017b. 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)*, volume 1, pages 189–198.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, ukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Xingdi Yuan, Tong Wang, Caglar Gulcehre, Alessandro Sordoni, Philip Bachman, Sandeep Subramanian, Saizheng Zhang, and Adam Trischler. 2017. Machine comprehension by text-to-text neural question generation. *arXiv preprint arXiv:1705.02012*.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017. Neural question generation from text: A preliminary study. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 662–671. Springer.