Computers, Materials & Continua
DOI:10.32604/cmc.2021.016517
Article

Tech Science Press

Tibetan Question Generation Based on Sequence to Sequence Model

Yuan Sun[1,2,*], Chaofan Chen[1,2], Andong Chen[3] and Xiaobing Zhao[1,2]

[1]School of Information Engineering, Minzu University of China, Beijing, 100081, China
[2]Minority Languages Branch, National Language Resource and Monitoring Research Center
[3]Queen Mary University of London, London, E1 4NS, UK
[*]Corresponding Author: Yuan Sun. Email: tracy.yuan.sun@gmail.com
Received: 04 January 2021; Accepted: 06 March 2021

Abstract: As the dual task of question answering, question generation (QG) is a significant and challenging task that aims to generate valid and fluent questions from a given paragraph. The QG task is of great significance to question answering systems, conversational systems, and machine reading comprehension systems. Recent sequence to sequence neural models have achieved outstanding performance in English and Chinese QG tasks. However, the task of Tibetan QG is rarely mentioned. The key factor impeding its development is the lack of a public Tibetan QG dataset. Faced with this challenge, the present paper first collects 425 articles from the Tibetan Wikipedia website and constructs 7,234 question–answer pairs through crowdsourcing. Next, we propose a Tibetan QG model based on the sequence to sequence framework to generate Tibetan questions from given paragraphs. Secondly, in order to generate answer-aware questions, we introduce an attention mechanism that can capture the key semantic information related to the answer. Meanwhile, we adopt a copy mechanism to copy some words in the paragraph to avoid generating unknown or rare words in the question. Finally, experiments show that our model achieves higher performance than baseline models. We also further explore the attention and copy mechanisms, and prove their effectiveness through experiments.

Keywords: Tibetan; question generation; copy mechanism; attention

1 Introduction

Aiming at generating specific answer-related questions from a given text, question generation (QG) is a basic and important task in the natural language generation community. With a wide range of applications, QG has attracted much attention from industrial and academic communities. The QG task can not only extend the corpus of question answering systems, but also be applied in many fields such as dialogue systems, expert systems, and machine reading comprehension [1]. Based on the form of the input text, QG can be categorized into two types: sentence-level QG and paragraph-level QG. The inputting information of the former is a sentence, while that of the latter is a paragraph. Compared with sentence-level QG, paragraph-level QG contains much richer text information.

Early works on QG tasks were mainly based on templates and rules [2,3]. These works transformed the input text into a question by complex question templates and linguistic rules crafted by humans. However, such methods heavily rely on manual rules and professional linguistic knowledge, and thus cannot be directly adapted to other domains. Recently, with the advancement of deep learning, researchers have begun to adopt neural network approaches to solve this problem [4–6]. Different from the template-based or rule-based methods, the deep learning method does not require complex question templates or rules. Therefore, researchers no longer pay attention to design question rules and templates, but concentrate on training an end-to-end neural network. At the same time, the models based on the deep learning method have a good generalizability, and can be easily applied to different domains. However, most QG works have been applied to English or Chinese tasks. For low-resource languages such as Tibetan, few relevant experiments have been conducted. Two main factors hinder the development of Tibetan QG: (1) there is no large-scale public Tibetan QG dataset; and (2) with complex grammar, most QG models have difficulty understanding the Tibetan sentence. Thus, it is difficult to achieve satisfactory results.

To address these issues, this study proposes a Tibetan QG model to generate valid and fluent question sentences. Our model takes a paragraph and a target answer as input. Different from other text generation tasks, our model aims to (1) generate questions that can be answered from a paragraph, and (2) generate fluent questions. Specifically, to achieve the first goal, our model needs to incorporate the target answer's position information and capture the key information in the paragraph. To achieve the second goal, we need to avoid generating rare or unknown words. Considering both goals, we adopt an attention mechanism and copy mechanism. Our contributions to the literature are three-fold:

(1) To solve the problem of Tibetan QG corpus shortage, we construct a high-quality Tibetan QG dataset.
(2) To generate answer-aware questions, we adopt an attention mechanism to help the model encode the answer-aware paragraph.
(3) To avoid generating unknown/rare words, we adopt a copy mechanism to copy some words from the paragraph.

2 Related Work

Question generation requires the system to generate fluent questions when given a paragraph and a target answer. Early works have focused on template-based methods, and mostly adopt linguistic rules. Under this scenario, researchers have spent a considerable amount of time designing the question rules and templates, and then transforming the input text information into a question. Heilman et al. [7] proposed a method to generate a large number of candidate questions. They matched the text with complex question templates and rules designed by humans. Finally, they sorted out the generated questions and selected the top questions. However, designing the question rules and templates were time-consuming. To solve this problem, Mitkov et al. [8] proposed the semi-automatic method: they began by using some simple natural language processing tools, such as the shallow semantic analysis tool, named entity recognition tool, and part-of-speech tagging tool, to better understand the input text. Their work shortened the development time, but the natural language processing tools introduced some errors. Therefore, the quality of questions generated by their system could not be guaranteed. To generate high-quality questions, Mostow et al. [9] expanded the question templates and rules by summarizing many question structures. At the same time, Labutov et al. [10] regarded the QG task as an "ontology crowd-relevance"

workflow. They firstly represented the original text as a low-dimensional ontological space, and then generated questions by aligning question templates and rules. The aforementioned works all converted paragraphs into questions through question templates and rules. Moreover, these works all followed three steps: sentence processing, template and rule matching, and question sorting. Thus, the performance of these systems heavily depends on the quality of the rules and templates. Additionally, these rule-based methods are difficult to apply in other domains.

Recently, deep learning methods have achieved remarkable performance on QG tasks. Deep learning methods do not require researchers to design complex question templates and rules. However, they do require a large amount of labeled data to train the model. Existing English and Chinese QG models are based on the sequence to sequence framework [11–14]. Serban et al. [15] and Du et al. [16] used two different recurrent neural networks (RNN) to encode and decode the input text. However, they did not consider that some sentences would not generate valuable questions. Thus, Du et al. [17] proposed a high-level neural network to identify which sentences in the paragraph were worth asking questions. Finally, they used these sentences to generate more questions, and the F1 value reached 79.8%. However, the aforementioned works did not have any restriction on the generated questions, and the questions could not be answered after reading the paragraph. To generate answer-aware questions, Zhou et al. [18] proposed a method to integrate answer position information. Their model not only took the original text as input but also considered the position information of the answer in the text. Finally, the experimental results proved the effectiveness of their model: they achieved 13.29% BLEU-4. Kim et al. [19] found that the generated questions would contain the answers. They inferred that the sequence to sequence model would overlearn some paragraph information, and thus proposed the answer masking method, which replaced the answer with a special label in the original text. They obtained 43.96% ROUGE-L on the Stanford Question Answering corpus (SQuAD) [20]. However, their method required a large-scale training corpus. To solve this problem, Song et al. [21] proposed a matching strategy to enhance the performance of the model. They utilized three different strategies (fully matching, attention matching and maximum matching) on the SQuAD dataset, and found that the full matching attention mechanism reached 42.72% ROUGE-L, and demonstrated higher performance than other strategies. Subsequently, Zhao et al. [22] adopted gated attention and a maxout pointer network in the encoder stage, and achieved 44.48% ROUGE-L. Note that the above works were all based on the English QG task.

As a minority language in China, Tibetan has a complicated grammatical structure. Tibetan is a Pinyin language, and the smallest unit of a Tibetan word is a syllable. Some words also have some changes in syllables, which means that the QG model must have a deep understanding of Tibetan text. Due to the particularity of minority languages, there are few studies on the Tibetan automatic question generation. Ban et al. [23] analyzed the Tibetan interrogative questions from a linguistic view. Similarly, Sun [24] analyzed the difference between questions and paragraphs. While these works were not related to automatic question generation tasks, they did provide the correct forms of Tibetan questions. In an attempt at Tibetan sentence-level question generation, Xia et al. [25] proposed a semi-automated method to generate questions; however, their work required human participation. To achieve automatically generated questions, Sun et al. [26] combined generative adversarial networks (GANs) [27] with reinforcement learning to automatically generate Tibetan questions. They achieved 6.7% higher than baseline methods on BLEU-2, but it took a long time to train the QG model.

3 Model Structure

3.1 Task Definition

For each sample data in our dataset, our goal is to generate answer-aware questions $\widetilde{Q}$ from a paragraph. Given a paragraph and a target answer, we first embed the paragraph and answer into two sequences $P = \{p_1, p_2, p_3, \ldots, p_n\}$ and $A = \{a_1, a_2, a_3, \ldots, a_m\}$, respectively. Next, the QG task can be defined as in Eq. (1).

$$\widetilde{Q} = \arg\max_{Q} prob\,(Q \mid P, A) \tag{1}$$

Therefore, our goal is to find the best $\widetilde{Q}$. Different from the traditional sequence to sequence model, the word in the generated question is not only from the dictionary but also from the paragraph sequence P.

3.2 The Overall Framework

In this section, we briefly introduce our model for the Tibetan QG. The overall framework, which can be roughly divided into two parts, is shown in Fig. 1.

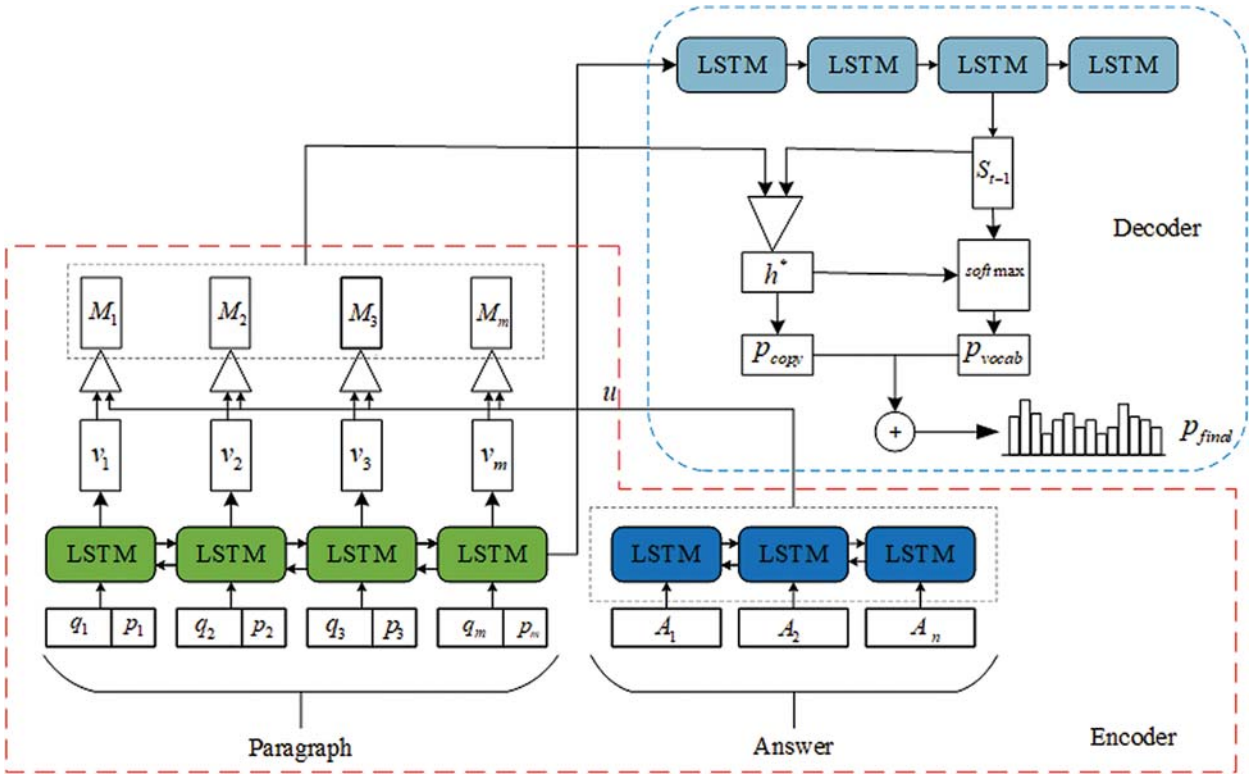

Figure 1: Framework of proposed model

(1) **Encoding stage**. In this stage, the Tibetan paragraph and the target answer are segmented into word sequences through word segment tools [28], and these sequences are sent to the neural network to be encoded. To ensure that the generated question is related to the given

answer, the attention mechanism is used to aggregate the key information related to the target answer. Finally, we can obtain the answer-aware paragraph representation.

(2) **Decoding stage**. The decoder is used to decode the answer-aware paragraph representation and generate questions. To avoid generating unknown or rare words in the question, we adopt a copy mechanism to copy some words from the paragraph.

4 Model Details

4.1 Paragraph and Answer Encoding

Different from the traditional sequence to sequence model, we use two different neural networks to separately encode the paragraph and answer. In order to obtain context information, this study adopts a bi-directional long short-term memory network (BiLSTM) [29] to encode questions and answers. Assuming that there is a paragraph sequence $q = \{q_1, q_2, q_3, \ldots, q_m\}$ with m words, the paragraph embedding can be calculated as follows:

$$v_t = LSTM\,(v_{t-1}, [q_t, p_t]) \tag{2}$$

where v_t presents the RNN hidden state at time step t, q_t is the word embedding of the tth word in a paragraph, and v_{t-1} is the previous RNN hidden state. Because it is important to generate an answer-aware question, we need to incorporate the answer's position information. p_t is the answer tagging sequence of whether the word q_t is inside or outside the answer. $[q_t, p_t]$ represents the concatenation of word embedding and answer tagging embedding. Finally, each word in the paragraph can be represented by $v = \left\{\left[\overrightarrow{v_t}, \overleftarrow{v_t}\right]\right\}^M$.

To avoid the influence of irrelevant information when encoding answer sequences, we use another BiLSTM to encode the target answer sequence:

$$u_t = LSTM\,(u_{t-1}, a_t) \tag{3}$$

where u_{t-1} is the RNN hidden state at time step $t-1$, and a_t is the word embedding in the answer A. Finally, the final answer embedding is represented by $u = \left\{\left[\overrightarrow{u_t}, \overleftarrow{u_t}\right]\right\}$. In this study, we use GloVe to embed the original text [30].

4.2 Interaction of Questions and Answers

Attention mechanisms are widely used in natural language processing tasks [31,32]. They allow the model to dynamically assign weights to different information. The model will pay more attention to words with greater weights. For the QG task, paragraphs contain a lot of information, but there may be some noise information as well. Therefore, the model must filter out the noise information. We introduce the attention mechanism to determine which word is important, and the answer-aware paragraph encoding can be described as follows:

$$S_u = V^T \tan\left(W^1 v_t + W^2 u_t\right) \tag{4}$$

where V^T, W^1, and W^2 are weight matrices, and the matrix S_u is the weight of each word in the paragraph. Next, we normalize S_u by using the softmax function as shown in Eq. (5).

$$a_u \propto \exp\,(S_u) \tag{5}$$

Next, we calculate the weight of every word in the paragraph as follows:

$$M_t = v_t \cdot a_u \tag{6}$$

where a_u represents the weight factor, and M_t is the final paragraph context embedding.

4.3 Decoder

After the encoding stage, we obtain a rich semantic paragraph representation. Next, similar to other sequence to sequence models, we use another LSTM network to decode the hidden state. Additionally, to avoid generating unknown or rare words, we allow the decoder to copy some words from the paragraph. We use an LSTM network to decode the hidden vector as shown in Eq. (7):

$$S_t = LSTM\left(W_{t-1}, S_{t-1}\right) \tag{7}$$

Here, S_t presents the current hidden state at time t, S_{t-1} is the previous hidden state, and W_{t-1} is the previously generated word. Next, we employ the attention mechanism in the paragraph. We need to calculate the weight distribution matrix between the paragraph and the hidden state. This step is illustrated in Eqs. (8) and (9).

$$e_{t,i} = M_t \cdot W^3 \cdot S_{t-1} \tag{8}$$

$$\alpha_t = soft\max\left(e_t\right) \tag{9}$$

Here, W^3 represents the weight matrix and can be trained from the network, and α_t is the attention distribution of paragraph words. According to the attention distribution, we can obtain the contextual representation of the paragraph by

$$h_t^* = \sum_{i=1}^{n} \alpha_t^i M_i \tag{10}$$

where a_t^i represents the attention score between the ith word in the paragraph and answer at time t. M_i is ith word representation in the paragraph, and is the output of the encoder. Next, the probability of the word generated by the decoder can be calculated as follows:

$$P_{vocab} = soft\max\left(W^4\left[h_t^*, S_t\right]\right) \tag{11}$$

where W^4 is the trainable weight matrix. We find that the question contains some words in the paragraph. Motivated by the above observation, we adopt a copy mechanism to copy some words in the paragraph. Next, we need to calculate the probability of copying a specific word. The copying probability can be calculated as follows:

$$\lambda_t^{copy} = \sigma\left(W^5 h_t^* + W^6 S_t\right) \tag{12}$$

where W^5 and W^6 are the trainable weight matrices, h^* is the context vector, S_t is the previously generated hidden state vector, and σ is the Sigmoid function that transforms the real number into

a probability value. Then, the probability distribution of words in the paragraph can be calculated by using the sum of all weights of the related words, as shown in Eq. (13).

$$P_{copy}(w) = \sum_{i=1}^{n} \alpha_t^i * I\{w == w_i\} \tag{13}$$

Finally, the probability of the generated word is a weighted sum of two ways:

$$P_{final} = \lambda_t^{copy} p_{copy}(w) + \left(1 - \lambda_t^{copy}\right) p_{vocab}(w) \tag{14}$$

5 Experiments and Analysis

5.1 Dataset

Considering the lack of a public Tibetan QG corpus, we collected a large number of Tibetan knowledge Wikipedia articles. These articles cover numerous domains such as science, literature, biology, and education. Next, we invited Tibetan native speakers to write down questions and corresponding answers when they read a paragraph. Finally, we obtained 7,234 Tibetan question-and-answer pairs. The specific format of Tibetan data is shown in Tab. 1. The UUID of each question-and-answer pair is unique.

Table 1: Sample from the Tibetan QG corpus

UUID	01d5c760-a4a3-13e5-e9b8-4a03057ce3d2
Passage	ཐུ་སྐྱེད་ཟས་སྨོང་ཅང་ཆེ་བ་ནི་བསྒྲུར་སྐྱེས་ཀྱི་སྦོས་འགྱིག་གི་རྒྱུ་ཆ་ཡིན་པ་དང་ཉེ་ཡི་ནང་དུ་རྒྱུ་ཆ་གསར་བ་ཞུང་ཤས་འདུས་ཡོད་ཅིང་།ཡང་ཀ་ཤིག་ཕྱེ་མ་ཚད་དེས་ཅན་ཞིག་བསྐེབས་ནས།རྒྱུན་པར་མི་རྣམས་ཀྱིས་བགོ་ལ་སྤྱོད་ཕྱོད་བཞིན་པའི་ཐུ་སྐྱེད་ཟས་སྨོང་དེ་ཐུབ་པ་རེད།…… The materials of most foam plastics are recycled plastics. It usually contains a small number of new materials and talc powder. Foam plastics are widely used as tableware by humans …
Answer	བསྐྱར་སྐྱེས་ཀྱི་སྦོས་འགྱིག Recycled plastics
Question	ཐུ་སྐྱེད་ཟས་སྨོང་ཅང་ཆེ་བ་ནི་གང་གི་རྒྱུ་ཆ་ཡིན། What is the material of the foam plastic?

5.2 Evaluation

To better evaluate the performance of our model, this paper uses two metrics: BLEU-2 and ROUGE-L. BLEU-2 adopts the 2-gram language model to match the gold question. The 2-gram model can be calculated as follows:

$$CP_n = \frac{\sum_i \sum_k \min\left(h_k(c_j), \max_{j \in m} h_k(s_{ij})\right)}{\sum_i \sum_k h_k(c_i)} \tag{15}$$

where s_{ij} represents the gold question, c_j represents the question generated by the model, $h_k(c_j)$ is the occurrence frequency of the word W_k in the generated question, and $h_k(S_{ij})$ is the occurrence

frequency of word W_k in the gold question S_{ij}. However, the 2-gram model may encourage the generation of short sentences. In order to solve this problem, the researchers introduce a length penalty factor:

$$b(C, S) = \begin{cases} 1 & l_c < l_s \\ e^{1-\frac{l_s}{l_c}} & l_c \geq l_s \end{cases} \tag{16}$$

where l_c represents the length of the generated question, and l_s is the length of the effective words. Finally, the calculation of BLUE-2 is as expressed in Eq. (17).

$$BLUE = b(C, S) \exp\left(\sum_{n=1}^{2} \omega_n \log Cp_n(C, S)\right) \tag{17}$$

ROUGE-L is a method based on the longest common subsequence. It can be calculated by Eqs. (18)–(20).

$$R_{lcs} = \frac{LCS(X, Y)}{len(Y)} \tag{18}$$

$$P_{lcs} = \frac{LCS(\mathrm{X}, \mathrm{Y})}{len(X)} \tag{19}$$

$$F_{lcs} = \frac{(\beta^2 + 1) R_{lcs} P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}} \tag{20}$$

Here, X is the question sentence generated by the model, and Y is the gold question. $LCS(X, Y)$ is the length of the common subsequence between the gold question sentence and the generated question sentence. β is a fixed parameter. In this paper, we set β to 1.2.

5.3 Results Analysis

To prove the effectiveness of our model, we compare some algorithms based on the sequence to sequence model.

(1) **Seq2Seq**: This is the baseline model. To generate the question sentences, this model directly uses two LSTM networks to encode and decode original input texts.
(2) **Du et al. [17] model:** This is an attention-based sequence learning model that generates questions from a paragraph. However, the authors of this model did not consider whether the generated question was related to the answer. Therefore, in the present paper we concatenate the target answer embedding with the paragraph embedding to generate answer-aware questions.
(3) **NQG [18]**: This work is based on the sequence to sequence model, and provides feature-rich encoding. In encoding stages, NQG adopts rich lexical features to encode the paragraph. In the decoding stages, it adopts a copy mechanism to generate questions.

From Tab. 2, it can be found that our model exhibits better performance on Tibetan QG. Our model achieves 25.34 BLEU-2 and 36.47 ROUGE-L. The baseline model (Seq2Seq) reaches 16.42 BLEU-2 and 27.13 ROUGE-L, respectively. Du et al. model achieves 31.26 ROUGE-L, and the ROUGE-L of NQG reaches 21.72 BLEU-2 and 32.71 ROUGE-L. Compared with the

basic Seq2Seq, our model demonstrates improvements of 8.92 BLEU-2 and 9.34 ROUGE-L. Compared with Du et al. [17], our model demonstrates improvements of 4.73 BLEU-2 and 5.21 ROUGE-L. Lastly, compared with NQG, our model shows improvements of 3.62 BLEU-2 and 3.76 ROUGE-L. To further verify our model, we also conduct some ablation experiments, and the results are shown in Tab. 3.

Table 2: Experimental results of different models

Model	BLEU-2	Increase	ROUGE-L	Increase
Seq2Seq	16.42	–	27.13	–
Du et al. [17]	20.61	+4.19	31.26	+4.13
NQG	21.72	+5.30	32.71	+5.58
Our model	**25.34**	**+8.92**	**36.47**	**+9.34**

Table 3: Model ablation results on Tibetan QG task

Model	BLUE-2	Increase	ROUGE-L	Increase
Seq2Seq	16.42	–	27.13	–
S2S + ATT	21.72	+5.30	32.24	+5.11
S2S + ATT + CP (our model)	**25.34**	**+8.92**	**36.47**	**+9.34**

Tab. 3 shows significant improvements in performance when adding the attention and copy mechanism. BLUE-2 improves by +5.3 when adding the attention mechanism. The performance of the sequence to sequence model also greatly increased when adding a copy mechanism. These results show that the attention mechanism and copy mechanism can help improve model performance.

6 Conclusion

In this paper, we propose a Tibetan QG model based on the sequence to sequence model. Additionally, faced with the problem of lacking a Tibetan QG corpus, we construct a medium-sized Tibetan QG dataset to explore the Tibetan question generation task. Furthermore, we propose a QG model that adopts an attention mechanism to gain answer-aware paragraph embedding, and introduce a copy mechanism to copy some words from the paragraph when generating the question. The copy mechanism enables us to avoid generating unknown or rare words. Experimental results demonstrate the effectiveness of our model. However, the performance of our model still needs to be improved. In the future, we will introduce some linguistic knowledge and common sense to the model to improve its performance.

Acknowledgement: We thank LetPub (www.letpub.com) for its linguistic assistance during the preparation of this manuscript.

Funding Statement: This work is supported by the National Nature Science Foundation (No. 61972436).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] L. Pan, W. Lei, S. Chua and Y. Kan, *Recent Advances in Neural Question Generation*, Ithaca, NY, USA: Arxiv press, pp. 1–12, 2019. [Online]. Available: http://arxiv.org/abs/1905.08949.

[2] C. Yllias and A. H. Sadid, "Towards topic to-question generation," *Computation Linguistics*, vol. 41, no. 1, pp. 1–20, 2015.

[3] L. David, P. Fred, N. John and W. Phil, "Generating natural language questions to support learning on-line," in *The 14th Annual Meeting of the Association for Computational Linguistics*, Sofia, Bulgaria, pp. 105–114, 2013.

[4] N. Preksha, A. K. Mohankumar, M. M. Srinivasan, B. V. Srinivasan, B. Ravindran *et al.,* "Let's ask again: Refine network for automatic question generation," in *Conf. on Empirical Methods in Natural Language Processing*, Hong Kong, China, pp. 3314–3323, 2019.

[5] Y. H. Chan and Y. C. Fan, "BERT for question generation," in *Int. Conf. Natural Language Processing*, Tokyo, Japan, pp. 173–177, 2019.

[6] A. T. Lu, J. S. Darsh and B. Regina, "Capturing greater context for question generation," in *The Thirty-Fourth AAAI Conf. on Artificial Intelligence*, New York, USA, pp. 9065–9072, 2020.

[7] M. Heilman and N. A. Smith, "Good question! statistical ranking for question generation," in *The 48th Annual Meeting of the Association for Computational Linguistics, ACL 2010, Proc.: Human Language Technologies*, Uppsala, Sweden, pp. 609–617, 2010.

[8] R. Mitkov and L. A. Ha, "Computer-aided generation of multiple-choice tests," in *Proc. of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing*, Atlanta, USA, pp. 17–22, 2003.

[9] J. Mostow and W. Chen, "Generating instruction automatically for the reading strategy of self-questioning," in *14th Int. Conf. on Artificial Intelligence in Education Workshops Proc.*, Brighton, UK, pp. 465–472, 2009.

[10] I. Labutov, S. Basu and L. Vanderwende, "Deep questions without deep understanding," in *The 53th Annual Meeting of Association for Computational Linguistics*, Beijing, China, pp. 889–898, 2015.

[11] T. Wang, X. Yuan and A. Trischler, "A joint model for question answering and question generation," in *The 34th Int. Conf. on Machine Learning*, Sydney, Australia, pp. 1–7, 2017.

[12] Z. Wang, W. Hanza and R. Florian, "Bilateral multi-perspective matching for natural language sentence," in *Proc. of the Twenty-Sixth Int. Joint Conf. on Artificial Intelligence*, Melbourne, Australia, pp. 4144–4150, 2017.

[13] T. Gu, D. Lu, H. Li and K. Li, "Incorporating copying mechanism in sequence-to-sequence learning," in *Proc. of the 54th Annual Meeting of the Association for Computational Linguistics*, Berlin, Germany, pp. 1631–1640, 2016.

[14] N. Smith, "Linguistic structure prediction," *Synthesis Lectures on Human Language Technologies*, vol. 4, no. 2, pp. 1–274, 2011.

[15] I. V. Serban, G. D. Alberto, C. Gulcehre, S. Ahn, S. Chander *et al.,* "Generating factoid questions with recurrent neural networks: The 30 m factoid question-answer corpus," in *Proc. of the 54th Annual Meeting of the Association for Computational Linguistics*, Berlin, Germany, pp. 588–598, 2016.

[16] X. Du and C. Cardie, "Identifying where to focus in reading comprehension for neural question generation," in *Conf. on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, pp. 2067–2073, 2017.

[17] X. Du and C. Cardie, "Harvesting paragraph-level question-answer pairs from wikipedia," in *Proc. of the 56th Annual Meeting of the Association for Computational Linguistics*, Melbourne, Australia, pp. 1907–1917, 2018.

[18] Q. Zhou, N. Yang, F. Wei, C. Tan, H. Bao *et al.,* "Neural question generation from text: A preliminary study," in *The Sixth Conf. on Natural Language Processing and Chinese Computing*, Dalian, China, pp. 662–671, 2017.

[19] Y. Kim, H. Lee, J. Shin and K. Jung, "Improving neural question generation using answer separation," in *The Thirty-Third AAAI Conf. on Artificial Intelligence*, Hawaii, USA, pp. 6602–6609, 2019.

[20] P. Rajpurkar, J. Zhang, K. Lopyrev and P. Liang, "SQuAD: 100,000+ question for machine reading comprehension of text," in *Conf. on Empirical Methods in Natural Language Processing*, Austin, TX, USA, pp. 1–10, 2016.

[21] L. Song, Z. Wang, W. Hamza, Y. Zhang, D. Gildea *et al.,* "Leveraging context information for natural question generation," in *The 16th Annual Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, New Orleans, USA, pp. 568–574, 2018.

[22] Y. Zhao, X. Ni, Y. Ding and Q. Ke, "Paragraph-level neural question generation with max-out pointer and gated self-attention networks," in *Empirical Methods in Natural Language Processing*, Brussels, Belgium, pp. 3901–3910, 2018.

[23] M. Ban, Z. Cai and Z. La, "Tibetan interrogative sentences parsing based on PCFG," *Journal Chinese Information Processing*, vol. 33, no. 2, pp. 67–74, 2019.

[24] L. P. Sun, "Classification of Tibetan problem for campus all-knowing," M.S. theses, Northwest Minzu University, China, 2019.

[25] T. Xia, Y. Sun, X. Zhao, W. Song, Y. Guo *et al.,* "Generating questions based on semi-automated and end-to-end neural network," *Computers, Materials & Continua*, vol. 61, no. 2, pp. 617–628, 2019.

[26] Y. Sun, C. Chen, T. Xia and X. Zhao, "QuGAN: Quasi generative adversarial network for Tibetan question answering corpus generation," *IEEE Access*, vol. 7, no. 3, pp. 116247–116255, 2019.

[27] I. Goodfellow, J. Pouget, M. Mehdi, B. Xu, W. F. David *et al.,* "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, Montreal, QC, Canada, pp. 2672–2680, 2014.

[28] J. Long, D. Liu, H. Nuo and J. Wu, "Tibetan pos tagging based on syllable tagging," *Chinese Information Process*, vol. 29, no. 5, pp. 211–216, 2015.

[29] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[30] J. Pennington, R. Socher and C. D. Manning, "Glove: Global vectors for word representation," in *Conf. on Empirical Methods in Natural Language Processing*, Doha, Qatar, pp. 1532–1543, 2014.

[31] B. Zhang, H. W. Wang, L. Q. Jiang, S. H. Yuan, M. Z. Li *et al.,* "A novel bidirectional LSTM and attention mechanism based neural network for answer selection in community question answering," *Computers, Materials & Continua*, vol. 62, no. 3, pp. 1273–1288, 2020.

[32] S. Chaudhari, V. Mithal, G. Polatkan and R. Ramanath, "An attentive survey of attention models," *Journal of the ACM*, vol. 37, no. 4, pp. 1–20, 2020.