CQG: A Simple and Effective Controlled Generation Framework for Multi-hop Question Generation

Zichu Fei[1,2], **Qi Zhang**[1,2]*, **Tao Gui**[3]*, **Di Liang**[4], **Sirui Wang**[4], **Wei Wu**[4], **Xuanjing Huang**[1,2,3]
School of Computer Science, Fudan Univeristy[1]
Shanghai Key Laboratory of Intelligent Information Processing, Shanghai, China[2]
Institute of Modern Languages and Linguistics, Fudan University, Shanghai, China[3]
Meituan Inc., Beijing, China[4]
`{zcfei19, qz, tgui,xjhuang}@fudan.edu.cn`
`{liangdi04, wangsirui}@meituan.com, wuwei19850318@gmail.com`

Abstract

Multi-hop question generation focuses on generating complex questions that require reasoning over multiple pieces of information of the input passage. Current models with state-of-the-art performance have been able to generate the correct questions corresponding to the answers. However, most models can not ensure the complexity of generated questions, so they may generate shallow questions that can be answered without multi-hop reasoning. To address this challenge, we propose the CQG, which is a simple and effective controlled framework. CQG employs a simple method to generate the multi-hop questions that contain key entities in multi-hop reasoning chains, which ensure the complexity and quality of the questions. In addition, we introduce a novel controlled Transformer-based decoder to guarantee that key entities appear in the questions. Experiment results show that our model greatly improves performance, which also outperforms the state-of-the-art model about **25%** by **5** BLEU points on HotpotQA [1].

Paragraph A: Celtic nations
The Celtic nations are territories in Western Europe where Celtic languages or cultural traits have survived. The term nation is used in its original sense to mean a people who share a common identity and culture and are identified with a traditional territory.

Paragraph B: Pan Celtic Festival
The Pan Celtic Festival is a Celtic-language music festival held annually in the week following Easter, since its inauguration in 1971. Its aim is to promote the modern Celtic languages and cultures and artists from all six Celtic nations: Brittany, Cornwall, Ireland, Isle of Man, Scotland and Wales.

Answer: Brittany, Cornwall, Ireland, Isle of Man, Scotland and Wales.

Shallow Question: Which six Celtic nations are modern Celtic languages from?

Deep Question: In which six Western European territories have Celtic languages or cultural traits survived?

Figure 1: An example that the uncontrolled question generation model may generate the correct but shallow questions. In this example, the model ignores the important entity Western European in paragraph A and then generate a shallow question without multi-hop reasoning chains.

1 Introduction

Question generation (QG) aims to endow machines with the ability to ask relevant and to-the-point questions about a document. QG plays a vital role in question answering (QA), dialogue systems, and automated tutoring applications: – by enriching the training QA corpora (Tang et al., 2017; Yuan et al., 2017), helping chatbots start conversations with intriguing questions (Mostafazadeh et al., 2016), and automatically generating assessment questions (Heilman and Smith, 2010), respectively.

Most prior research on QG has focused on shallow *factoid-based* questions where answering the question simply by extracting the span of the text from a single input document (Zhou et al., 2018; Zhao et al., 2018; Kim et al., 2019; Fei et al., 2021). Recently, motivated by building the NLP systems that are capable of understanding and reasoning (Kaushik and Lipton, 2018; Sinha et al., 2019), there is an increasing interest in developing systems that are capable of more complex multi-hop question generation, where answering the questions requires reasoning over multiple documents (Pan et al., 2020; Sachan et al., 2020; Xie et al., 2020; Yu et al., 2020; Su et al., 2020).

Compared with shallow QG, there are two

* Corresponding authors.
[1]Our code and models are publicly available at https://github.com/sion-zcfei/CQG

challenges for multi-hop QG (MQG). At first, generating multi-hop questions requires the model to understand the relationship between disjointed pieces of information in multiple context documents (Sachan et al., 2020). Secondly, multi-hop questions must have complex chains of connecting the mentioned entities, which ensure the complexity of multi-hop questions, as such, multi-hop questions are also called deep questions (Pan et al., 2020).

To address the first challenge, existing research on MQG relies on the Graph-to-Sequence (G2S) architecture (Pan et al., 2020; Su et al., 2020; Yu et al., 2020). These methods construct a semantic-level graph or entity-graph to capture the information among multiple context documents that employ a graph neural network(GNN) and then feed it to the decoder. However, these models can not handle the second challenge because they can not ensure the complexity of generated questions; thus, they may generate shallow questions that can be answered without multi-hop reasoning chains. We show an example in Figure 1, where the uncontrolled model generates a shallow question that can be answered by a single sentence but ignores the other sentences and entities.

To solve this issue, we propose the CQG, a simple and effective controlled framework. (De Cao et al., 2018; Qiu et al., 2019) claim that the reasoning chains can be captured by propagating information along the edges in an entity graph using a GNN. Motivated by this, we construct the entity graph from the input documents first and then employ the Graph Attention Network (GAT) to extract the key entities that appear in multi-hop reasoning chains. Intuitively, all these key entities should appear in the generated questions to ensure the generated questions have complex and complete reasoning chains. We introduce the flag tag (Wang et al., 2021), a lexical constraint for generation at each decoding step, which will assist the controlled generation. In detail, in decoding progressing, each input token is provided a flag tag that indicates whether the constraint of this token has been satisfied. Three possible types of flag tags exist for each token, *is not a constrain*, *does not appear in question* and *appear in question*. As shown in Figure 2, the flag tag of *six* updates to *appear in question* at the fourth step because six is generated at this step. We represent the three flag tags by training the embedding and injecting them into the Transformer generator. The flag tag can explicitly inform the generator to satisfy as many as possible constraints. In the training stage, when the generation is stopped, the flag tags for each token are either not a constrain or satisfied. It is a strong signal for the model to try to satisfy all constrains.

We conduct experiments on HotpotQA (Yang et al., 2018): a challenging dataset in which the questions are generated by reasoning over text from separate Wikipedia pages. Results show that our model greatly improves performance; it outperforms the state-of-the-art about **25%** by **5** BLEU (Papineni et al., 2002) points.

Our main contributions are summarized as follows:

- We propose a simple and effective controlled generation framework for MQG; we are also the first one to provide a method to ensure the complexity of generated questions and the first one to introduce the controlled generation methods to MQG.

- Experiment results show that our model greatly improves the performance; it also outperforms the state-of-the-art about **25%** by **5** BLEU points.

2 Related Work

2.1 Question Generation

Early works on QG (Mostow and Chen, 2009; Heilman and Smith, 2010) focus on the rule-based approaches that rely on heuristic rules or hand-crafted templates, with low generalizability and scalability. Recent works adopt the attention-based sequence-to-sequence neural model for QG tasks, taking sentences with the answer as input and outputting the question (Du et al., 2017), which proved to work better than the rule-based methods. (Zhou et al., 2018) proposes the feature-enriched encoder to encode the input sentence. To generate a question for a given answer, (Sun et al., 2018; Kim et al., 2019; Song et al., 2018) apply various techniques to encode answer location information into an annotation vector corresponding to the word positions, thus allowing for better quality answer focused questions. (Chen et al., 2020) presents a syntactic feature-based method to represent words in a document and to decide what words to focus on while generating the question. Furthermore, recent

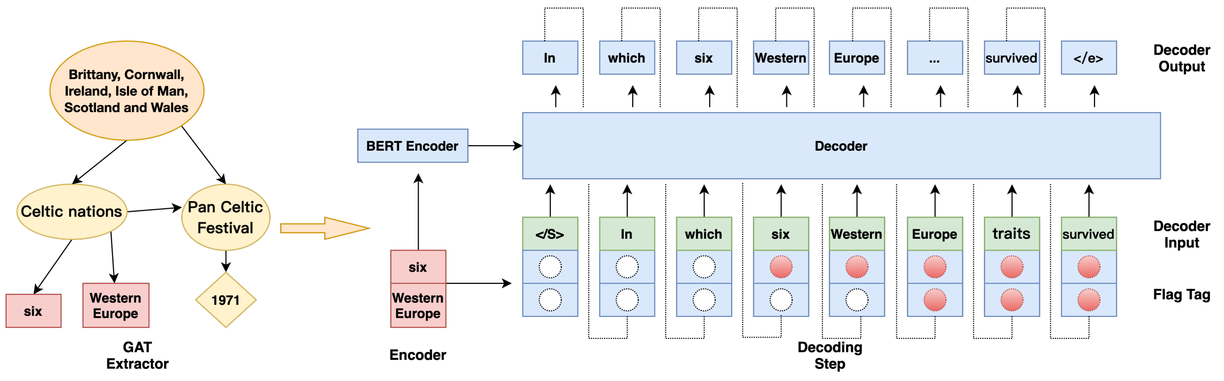


Figure 2: Overview architecture of the CQG model.

concurrent works apply the large-scale language model pre-training strategy for QG to achieve a new state-of-the-art performance (Chan and Fan, 2020).

Most prior research on QG has focused on shallow factoid-based questions, where answering the question simply by extracting the span of the text from a single input document.

Recently, there has been an increasing interest in MQG, to capture the complex information among different input documents, (Pan et al., 2020; Su et al., 2020) employ the GNN-based encoder in semantic graph and entity graph respectively, and (Sachan et al., 2020) use the strong transformer-based graph model. However, all these methods can not to ensure the complexity of generated questions where the generated questions may degenerate into shallow questions.

2.2 Controlled Generation

Two different types of control can be applied over generation models: soft control and hard control. Soft control aims at directing the option or the general topic of the generated text. In contrast, hard control aims at ensuring that some explicit constraints are met, e.g., specific words are contained in the text. The soft control can also be achieved via hard control, i.e., text that contains a set of words related to a certain topic should arguably revolve around that topic. Some recent works employ soft control on unconstrained language generation by training or fine-tuning language models (Ziegler et al., 2019; Keskar et al., 2019).

While hard control of constrained generation, such as machine translation, can be attained with grid beam search methods (Hu et al., 2019; Post and Vilar, 2018), which is impractical to use the same approach for hard control of unconstrained generation. Methods such as grid beam search rely on the assumption that there exists a core set of plausible candidates fulfilling the desired criteria, this is not often the case for open-ended generation tasks. Recent work on stochastic search (Sha, 2020) has approached this problem by performing bidirectional search during generation and editing the text until the constraints are fulfilled. Although stochastic search is suitable for bidirectional RNN models, it is not yet clear if it can be applied to forward generation models, e.g., transformer-based models.

3 Methodology

In this section, we formalize the multi-hop question generation (MQG) task and introduce our CQG. In particular, we first describe our Graph Attention Network (GAT) based key entities extractor. Following this, we describe the flag tag and finally we introduce our novel controlled Transformer-based generator with flag tag.

3.1 Problem Formulation

The input to the MQG task is a set of context documents $C = \{d_1, .., d_k\}$ where the k is the number of documents and an answer $A = [a_1, ..., a_m]$ where the m is the length of answer. These documents can be long containing multiple sentences, $d_i = [s_1, ..., s_n]$, where each $s_j = [w_1^j, ..., w_t^j]$ is composed of a sequence of tokens and the n and t are the number of sentences and the length of sentences respectively. The desired goal of MQG is to generate a question $y = [y_1, ..., y_t]$ conditioned on the context and the answer, where answering this question requires reasoning about the content in more than one of the context documents.

3.2 GAT-based Key Entities Extractor

According to existing research in multi-hop QA (De Cao et al., 2018), the reasoning chains can be captured by propagating local contextual information along edges in entity graph using a GNN. Motivated by this, we construct the entity graph from the input documents first and then employ the Graph Attention Network (GAT) (Veličković et al., 2017).

We follow the (Qiu et al., 2019) to construct the entity graph and we use the Stanford corenlp toolkit (Manning et al., 2014) to recognize named entities from the context C. The entity graph is constructed with the entities as nodes and edges built as follows. The edges are added 1. for every pair of entities that appear in the same sentence in C (sentence-level links); 2. for every pair of entities with the same mentioned text in C (context-level links); 3. between a central entity node and other entities within the same paragraph (paragraph-level links). The central entities are extracted from the title sentence for each paragraph. We do not apply co-reference resolution for pronouns because it introduces both additional useful and erroneous links.

We concatenate the answer A with the context C and pass the resulting sequence to a pre-trained BERT model to obtain representations $H = [h_1, h_2, ..., h_M]$ where M is the length of the context and answer. For each entity $e_i = [w_l, w_{l+1}, ..., w_j]$, we obtain its representation by a MaxPool and use it as the node embedding E_i in entity graph:

$$e_i = [w_l, w_{l+1}, ..., w_j] \tag{1}$$

$$E_i^0 = MaxPooling(h_l, h_{l+1}, ..., h_r) \tag{2}$$

The next step is to aggregate the information in the entity graph; here, we used a GAT to compute the multi-head attention score between two entity nodes by:

$$\alpha_{ij} = \frac{exp(\sigma(\mathbf{W}[h_i, h_j]))}{\sum_{k \in N_i} exp(\sigma(\mathbf{W}[h_i, h_k]))} \tag{3}$$

$$\sigma(x) = LeakyReLU(x) \tag{4}$$

where $\mathbf{W}$ is the trainable matrix and N_i is the neighbors of entity i.

We aggregate the information by multi-head attention at each step:

$$h_i^{t+1} = \|_{k=1}^{K} \sigma(\sum_{j \in N_i} \alpha_{ij}^k \mathbf{W}^k h_j) \tag{5}$$

where $\|$ is the concatenate operation, $\mathbf{W}^k$ is the trainable weighting matrix for the kth head and all nodes share the same parameters of $\mathbf{W}^k$. Then we obtain the updated node embedding $E^{t+1} = [h_1^{t+1}, h_2^{t+1}, ..., h_n^{t+1}]$.

To generate a multi-hop question, we need to select the key entities in complex multi-hop reasoning chains. We formulate this as a node classification task, i.e., deciding whether each node should be involved in the process of asking, i.e., appearing in the reasoning chain for raising a multi-hop question, as exemplified by Figure 2.

To this end, we add one feed-forward layer on top of the final layer of the graph encoder, taking the output node representations E^T for classification. We deem a node as a positive ground-truth to train the key entities extract task if its contents appear in the ground-truth question and optimize it by cross-entropy loss.

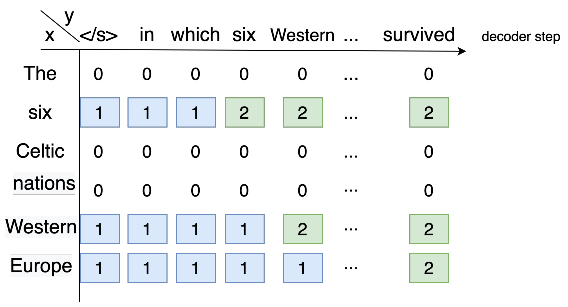
3.3 Controlled Generator with Flag Tag

In order to ensure the complexity of the generated question, the generated question must contain the key entities extracted from the entity graph. To this end, we need a controlled generator $G(Y|X, Y)$ where X is the input passage tokens and some x_i correspond to lexical constraints that must be satisfied in the generated outputs.

We describe the flag tag firstly, at decoding step t, the flag tag indicates whether each lexical constraint has been satisfied up until this step. Notably, the flag tag for each token at step t is that:

$$flag_i^t = \begin{cases} 0 & x_i \ is\ not\ a\ constrain \\ 1 & x_i \ does\ not\ appear\ in\ y_{1:t} \\ 2 & x_i \ appear\ in\ y_{1:t} \end{cases}$$

where $flag_i^t$ is the flag tag for ith input token at decoding step t, and $y_{1:t}$ is the generated tokens thus far. The tokens with the values 1 or 2 of the flag is a lexical constraint and the token with 0 is not constrained to appear in the question. Obviously, the flag tag for any token can only remain unchanged or updated to value 2. As shown in Figure 3, the input tokens X is that X = [The, six, Celtic, nations, Western, Europe] and the flag tag at the beginning is that $flag^0$ = [0,1,0,0,1,1] because the tokens are not constrained except *six*, *Western* and *Europe*. At step 4, the flags update to [0,2,0,0,0,1,1] because the token *six* has been generated but *Western* and *Europe* have not.



x \ y	</s>	in	which	six	Western	...	survived
The	0	0	0	0	0	...	0
six	1	1	1	2	2	...	2
Celtic	0	0	0	0	0	...	0
nations	0	0	0	0	0	...	0
Western	1	1	1	1	2	...	2
Europe	1	1	1	1	1	...	2

decoder step

Figure 3: An example for flag tag update.

During the training of models, all the constraints have been satisfied before stopping the generation. This is a strong signal for the model to satisfy all the constraints. In addition, the flag tag is simple enough, which only adds the embedding with three tokens.

To utilize the rich information in flag tag, we employ a Transformer-based decoder as a generator to incorporate it and construct a simple controlled generation framework. We inject the flag tag into the embedding vector and use this embedding as the relative position embedding to bridge the decoder and the encoder.

In particular, at decoding step t, we incorporate the flag tag embedding by cross-attention in decoder. The conventional cross-attention module is computed by:

$$Cross(Q, K, V) = softmax(\frac{Q^{\top} K}{\sqrt{d_k}})V \quad (6)$$

where Q is the decoder states, K and V are encoder states and d_k is the dimensions of K vectors.

We introduce the flag tag at step t $F^t \in \mathbb{R}^{3*lenP}$ where $lenP$ is the length of the input passage, to transformer decoder as relative position embedding to compute the cross attention at step t as follows:

$$\alpha^t_{cross} = softmax(E^t) \quad (7)$$

$$E^t = \frac{Q^t(K + R^t)^{\top}}{\sqrt{d}} \quad (8)$$

$$R^t = Embedding(F^t) \quad (9)$$

where Q^t is the states of decoder at step t and the K is the outputs of encoder. And then the outputs of cross module is:

$$Cross(Q^t, K, V, F^t) = \alpha^t_{cross} V \quad (10)$$

where V is the outputs of encoder.

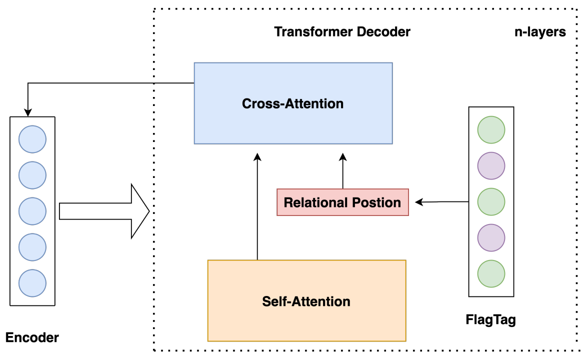


Figure 4: The controlled Transformer-based decoder where incorporate the flag tag as relational position embedding.

We train our model by the negative log likelihood for the target sequence y:

$$L = \frac{1}{T}\sum_{t=1}^{T} logP(\tilde{y}_t = y_t) \quad (11)$$

4 Experiments

4.1 Data and Metrics

To evaluate the model's ability to generate multi-hop questions, we conduct experiments on HotpotQA (Yang et al., 2018), which contains about 100,000 crowd-sourced questions that require reasoning over separate Wikipedia articles. Each question has two supporting documents that contain the necessary evidence to infer the answer. In this paper, we take the fact supporting sentences with the answer as inputs to generate the multi-hop questions. We follow the split of the original dataset including 90,447 and 7405 examples for training and developing respectively. Because the test set is not available publicly, so we set the original developing set as the test set and extract 500 samples from the training set as the developing set. Overall, we use the 89,947/500/7405 samples as training set, developing set and testing set, respectively.

Following the previous work, we employ BLEU (Papineni et al., 2002), ROUGE-L (Lin, 2004) and METEOR (Lavie and Agarwal, 2007) as automated evaluation metrics. BLEU measures the average n-gram overlap on a set of reference sentences. Both METEOR and ROUGE-L specialize BLEU's n-gram overlap idea for machine translation and text summarization evaluation, respectively.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L
Seq2Seq + attn (Bahdanau et al., 2014)	32.97	21.11	15.41	11.81	18.19	33.48
NQG++ (Zhou et al., 2018)	35.31	22.12	15.53	11.50	16.96	32.01
ASs2s (Kim et al., 2019)	34.60	22.77	15.21	11.29	16.78	32.88
s2sa-at-mp-gsa (Zhao et al., 2018)	38.74	24.89	17.88	13.48	18.39	34.51
UniLM (Dong et al., 2019)	42.37	29.95	22.61	17.61	25.48	40.34
MuLQG (Su et al., 2020)	40.15	26.71	19.73	15.20	20.51	35.30
Semantic Graph (Pan et al., 2020)	40.55	27.21	20.13	15.53	20.15	36.94
IGND (Fei et al., 2021)	41.22	24.71	18.99	16.36	24.19	38.34
BART (Lewis et al., 2020)	41.41	30.90	24.39	19.75	25.20	36.13
Strong Transformers (Sachan et al., 2020)	-	-	-	20.02	22.40	39.49
CQG	**49.71**	**37.04**	**29.93**	**25.09**	**27.45**	**41.83**

Table 1: Automatic evaluation results on HotpotQA. Our CQG achieves the best performance and have significant improvement in all metrics.

4.2 Baselines

We compare our proposed model against several strong baselines on question generation.

Seq2Seq + attn: (Bahdanau et al., 2014) the basic sequence-to-sequence (Seq2Seq) model with attention, which takes the document as input to decode the question.

NQG++ (Zhou et al., 2018): a Seq2Seq model with feature-enrich encoder.

s2sa-at-mp-gsa (Zhao et al., 2018): employs a gated attention encoder and a maxout pointer decoder to deal with long text inputs.

ASs2s (Kim et al., 2019): proposes an answer separated Seq2Seq model by replacing the answer in the input sequence with some specific words.

Semantic Graph (Pan et al., 2020): a graph-to-seq model for MQG, which constructs a semantic graph to capture the global information.

MuLQG (Su et al., 2020): a graph-to-seq model employs an encoder reasoning gate to capture the entity graph information.

IGND (Fei et al., 2021): a graph-to-seq model that introduces the copy tag and iterative graph-based decoder, it is the state-of-the-art model for shallow QG. We construct the graph following (Pan et al., 2020) to match the HotpotQA dataset.

BART (Lewis et al., 2020): The strong pre-training generation model that obtains the state-of-the-art performance on shallow question.

UniLM (Dong et al., 2019): Another strong pre-training generation model.

Strong Transformers (Sachan et al., 2020): the state-of-the-art model for MQG, which propose a series of strong Transformer models for MQG.

4.3 Implementation Details

We use the BERT base model loaded from transformers in huggingface library [2]. The embedding size and head hidden size of the flag tag are 64. The number of heads in BERT, transformer-based decoder and GAT attention is 8. The number hop of GAT in the entity graph is 3. As for entity extracting, if the number of key entities is more than 5, we use the top-5 entities with the highest probability. We use the AdamW (Loshchilov and Hutter, 2017) as the optimizer and the learning rate is set to 2e-5. We stop the training if the validation BLEU-4 score stops improving for 10 epochs. We clip the gradient at length 10. The batch size is 128 and the beam search width 5. All hyperparameters are tuned on the development set. We implement all models in MindSpore.

4.4 Main Results

Table 1 shows the experimental results of the HotpotQA dataset. In terms of BLEU-4 regarded as the main evaluation metric for text generation, our model greatly improves performance; it outperforms the strong Transformers about 25% by 5 BLEU points. We achieve state-of-the art results on HotpotQA. Not only in BLEU-4, our CQG achieves the best performance and shows significant improvement in all metrics.

4.5 Human Evaluation

Metrics for automatic evaluation based on n-grams may not truly reflect the quality of generated questions. Hence, we further randomly sample 300 examples in the test set for human evaluation. Following by (Pan et al., 2020), we conduct human

[2]huggingface.co/transformers

Model	Short Contexts			Medium Contexts			Long Contexts			Average		
	Flu.	Rel.	Cpx.	Flu.	Rel.	Cpx.	Flu.	Rel.	Cpx.	Flu.	Rel.	Cpx.
MulQG (Su et al., 2020)	3.78	3.56	3.49	3.53	3.47	3.44	3.39	3.36	3.26	3.56	3.47	3.39
Semantic Graph (Pan et al., 2020)	3.79	3.55	3.51	3.54	3.46	3.42	3.40	3.37	3.28	3.57	3.46	3.40
UniLM (Dong et al., 2019)	4.27	4.11	3.86	4.23	4.06	3.84	4.18	4.01	3.82	4.22	4.06	3.84
BART (Lewis et al., 2020)	4.32	4.16	3.94	4.29	4.15	3.92	4.25	4.11	3.88	4.28	4.14	3.91
Our CQG	**4.41**	**4.28**	**4.21**	**4.40**	**4.26**	**4.18**	**4.38**	**4.27**	**4.17**	**4.39**	**4.27**	**4.18**
Ground Truth	4.94	4.92	4.97	4.93	4.93	4.96	4.89	4.92	4.98	4.91	4.93	4.97

Table 2: The human evaluation for different models.

Model	BLEU
CQG	25.09
CQG w/o entity graph	24.12
CQG w/o inference dynamical flag tag	22.96
CQG w/o controlled decoder	20.87
CQG w/o key entities + controlled decoder	19.89

Table 3: The ablation study for CQG.

evaluations on 300 random test samples consisting of 100 short (<50 tokens), 100 medium (50-200 tokens), and 100 long (>200 tokens) documents. We ask three workers to rate the 300 generated questions as well as the ground-truth questions between 1 (poor) and 5 (good) on three criteria: (1) fluency, which indicates whether the question follows the grammar and accords with the correct logic; (2) relevance, which indicates whether the question is answerable and relevant to the passage; (3) complexity, which indicates whether the question involves reasoning over multiple sentences from the document. We average the scores from raters on each question and report the performance of UniLM, MuLQG Semantic Graph, BART and our CQG. Workers were unaware of the identity of the models in advance. We show the results in Table 2.

We can see that the performance of pre-training generation models is much better than MulQG and Semantic Graph. Our CQG model shows the best performance for all three criteria and all lengths of context. Furthermore, CQG is outstanding in complexity where other models are weak in it, and this result proves that our model is effective in solving the complexity control issue of MQG task.

4.6 Ablation Study

To further evaluate and investigate the performance of different components and strategies in our model, we perform the ablation study in the HotpotQA test set and show the results in Table 3.

CQG w/o entity graph The model removes the entity graph and employs the context embedding passed BERT to extract the entity, which does not change the setting of the controlled generator.

CQG w/o controlled decoder The model removes the controlled decoder and employs the standard transformer model, where the BERT encoder encodes both input passage and key entities and feeds then into the decoder.

CQG w/o inference dynamical flag tag The model does not update flag tag in **inference stage**, which means all the values of flag tag at the last step are the same as those at the first step.

CQG w/o key entities + controlled decoder The model removes the key entities extractor and controlled generator; we can see it as a baseline model consisting of a BERT encoder and a Transformer decoder.

First of all, there is a huge gap between CQG and CQG w/o key entities + controlled decoder, which demonstrates that our controlled generation framework plays an important role. Comparing between CQG and CQG w/o controlled decoder, we find that the controlled generator with the flag tag is the critical module in CQG.

Secondly, CQG is higher than CQG w/o entity graph 0.97 of BLEU points. We can see that the entity graph constructed from the input passage contains rich structure information among entities and captures the information by GAT, which can improve the performance for CQG.

Thirdly, although the CQG w/o dynamical inference flag tag is worse than CQG, it is much higher than CQG w/o controlled decoder. This phenomenon shows that the flag tag is a strong signal that prompts the model to satisfy as many constraints as possible in the training stage. Although CQG w/o dynamical inference flag tag does not update the flag tag in the inference stage, the model also tries to generate the key entities to improve the performance.

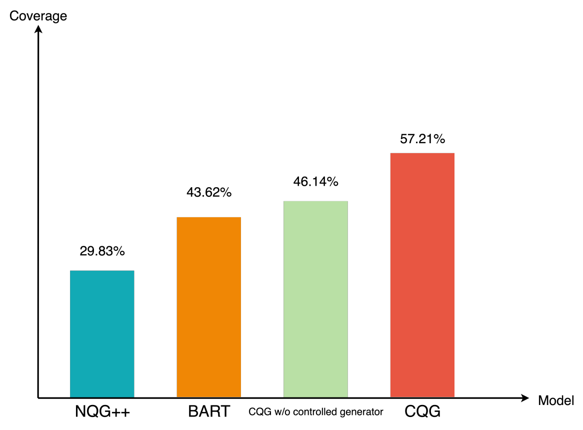


Figure 5: The coverage percentage of key entity

CQG w/o controlled decoder removes the hard controlled generator and employs the soft controlled method, which encodes the key tokens and feeds them to the decoder. CQG w/o controlled decoder is 0.98 higher than CQG w/o key entities + controlled decoder, which shows the soft controlled method is effective but is far from the hard method in CQG.

4.7 Analysis for controlled generator

We conduct some experiments to analyze the controlled generator in this section. At first, we compare the key entity coverage percentage for different models. In particular, we compute the coverage percentage of the appeared key entity in question generated by different models, where we think all the entities that appear in the gold question are key entities. This metric reflects the complexity of generated questions because the multi-hop reasoning chains are composed of these key entities. As shown in Figure 5, we can find that the coverage of CQG is much higher than in the other models, and this improvement is from the controlled generator according to the comparison between CQG and CQG w/o controlled generator. This result shows that our CQG improves the control of the model generation process.

4.8 Case Study

We present a case study to show the control ability of our model and compare the strong baseline BART model, CQG and the gold. The cases are presented in Table 4.

It is clearly shown the BART model generates the question only involved paragraph A, which is not the multi-hop question. As for CQG, we provide three examples with the different key entity and the questions generated by CQG contain the given key entity. The given key entity can control the semantic of the generated question, and we can see that the question in the first example, where the given key entities are the same entity as gold, have the same semantic as the gold question. The examples demonstrate that our CQG can be controlled to generate the high-quality multi-hop question with the given key entity.

Paragraph A: Letters to Cleo Letters to Cleo are an alternative rock band from Boston, Massachusetts, best known for the 1994 single, "Here & Now, from their full-length debut album, "Aurora Gory Alice". The band's members are Kay Hanley, Greg McKenna, Michael Eisenstein, Stacy Jones, Scott Riebling, and later, Tom Polce. **Paragraph B: Screaming Trees** Screaming Trees was an American rock band formed in Ellensburg, Washington in 1985 by vocalist Mark Lanegan, guitarist Gary Lee Conner, bass player Van Conner and drummer Mark Pickerel. Pickerel had been replaced by Barrett Martin by the time the band reached its most successful period. **Answer:** Letters to Cleo **Gold Question:** Which band, Letters to Cleo or Screaming Trees, had more members?
BART: Which band's members are Kay Hanley, Greg Mckenna, Michael Eisenstein, Stacy Jones, Scott Riebling, and Tom Polce ? **Key Entity:** Letters to Cleo, Screaming Trees **CQG:** Which band has more members, Letters to Cleo or Screaming Trees? **Key Entity:** Letters to Cleo, Kay Hanley **CQG:** Is Kay Hanley the member of Letters to Cleo's member ? **Key Entity:** Boston **CQG:** Which rock band are from Boston ?

Table 4: Case study of one example from HotpotQA test set. We indicate the key entity by different color.

5 Conclusion

The MQG task is more challenging and worthy of exploration compared with conventional shallow QG. To address the complexity control problem of MQG, we propose a simple control framework CQG, which consists of a GAT-based key entity extractor and a controlled generated. CQG greatly improves the performance and we hope our model will help researchers to study the MQG task.

6 Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by National Natural Science Foundation of China (No. 61976056, 62076069), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103). This research was supported by Meituan, Beijing Academy of Artificial Intelligence(BAAI), and CAAI-Huawei MindSpore Open Fund.

References

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

Ying-Hong Chan and Yao-Chung Fan. 2020. A recurrent BERT-based model for question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 154–162, Hong Kong, China. Association for Computational Linguistics.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. 2020. Reinforcement learning based graph-to-sequence model for natural question generation. In *Proceedings of the 8th International Conference on Learning Representations*.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2018. Question answering by reasoning across documents with graph convolutional networks. *arXiv preprint arXiv:1808.09920*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13063–13075.

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 (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada. Association for Computational Linguistics.

Zichu Fei, Qi Zhang, and Yaqian Zhou. 2021. Iterative GNN-based decoder for question generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2573–2582, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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, Los Angeles, California. Association for Computational Linguistics.

J. Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. 2019. Improved lexically constrained decoding for translation and monolingual rewriting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 839–850, Minneapolis, Minnesota. Association for Computational Linguistics.

Divyansh Kaushik and Zachary C. Lipton. 2018. How much reading does reading comprehension require? a critical investigation of popular benchmarks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5010–5015, Brussels, Belgium. Association for Computational Linguistics.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Yanghoon Kim, Hwanhee Lee, Joongbo Shin, and Kyomin Jung. 2019. Improving neural question generation using answer separation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6602–6609.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

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

I. Loshchilov and F. Hutter. 2017. Decoupled weight decay regularization.

C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, and D. Mcclosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd*

Annual Meeting of the Association for Computational Linguistics: System Demonstrations.

Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Margaret Mitchell, Xiaodong He, and Lucy Vanderwende. 2016. Generating natural questions about an image. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1802–1813, Berlin, Germany. Association for Computational Linguistics.

Jack Mostow and Wei Chen. 2009. Generating instruction automatically for the reading strategy of self-questioning.

Liangming Pan, Yuxi Xie, Yansong Feng, Tat-Seng Chua, and Min-Yen Kan. 2020. Semantic graphs for generating deep questions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1463–1475.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA. Association for Computational Linguistics.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324, New Orleans, Louisiana. Association for Computational Linguistics.

Lin Qiu, Yunxuan Xiao, Yanru Qu, Hao Zhou, Lei Li, Weinan Zhang, and Yong Yu. 2019. Dynamically fused graph network for multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6140–6150.

Devendra Singh Sachan, Lingfei Wu, Mrinmaya Sachan, and William Hamilton. 2020. Stronger transformers for neural multi-hop question generation. *arXiv preprint arXiv:2010.11374*.

Lei Sha. 2020. Gradient-guided unsupervised lexically constrained text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8692–8703, Online. Association for Computational Linguistics.

Koustuv Sinha, Shagun Sodhani, Jin Dong, Joelle Pineau, and William L. Hamilton. 2019. CLUTRR: A diagnostic benchmark for inductive reasoning from text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4506–4515, Hong Kong, China. Association for Computational Linguistics.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018. Leveraging context information for natural question generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 569–574, New Orleans, Louisiana. Association for Computational Linguistics.

Dan Su, Yan Xu, Wenliang Dai, Ziwei Ji, Tiezheng Yu, and Pascale Fung. 2020. Multi-hop question generation with graph convolutional network. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4636–4647, Online. Association for Computational Linguistics.

Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. 2018. Answer-focused and position-aware neural question generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3930–3939, Brussels, Belgium. Association for Computational Linguistics.

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*.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Yufei Wang, Ian Wood, Stephen Wan, Mark Dras, and Mark Johnson. 2021. Mention flags (MF): Constraining transformer-based text generators. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 103–113, Online. Association for Computational Linguistics.

Yuxi Xie, Liangming Pan, Dongzhe Wang, Min-Yen Kan, and Yansong Feng. 2020. Exploring question-specific rewards for generating deep questions. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2534–2546.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Jianxing Yu, Xiaojun Quan, Qinliang Su, and Jian Yin. 2020. Generating multi-hop reasoning questions to improve machine reading comprehension. In *Proceedings of The Web Conference 2020*, pages 281–291.

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.

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 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*, pages 3901–3910, Brussels, Belgium. Association for Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2018. *Neural Question Generation from Text: A Preliminary Study*, pages 662–671.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.