LFKQG: A Controlled Generation Framework with Local Fine-tuning for Question Generation over Knowledge Bases

Zichu Fei[1,2], **Xin Zhou**[1,2], **Tao Gui**[3], **Qi Zhang**[1,2], **Xuanjing Huang**[1,2,3]
School of Computer Science, Fudan Unviersity[1]
Shanghai Key Laboratory of Intelligent Information Processing, Shanghai, China[2]
Institute of Modern Languages and Linguistics, Fudan University, Shanghai, China[3]
{zcfei19, xzhou20, qz, tgui,xjhuang}@fudan.edu.cn

Abstract

Question generation over knowledge bases (KBQG) aims to generate natural questions about a subgraph that can be answered by a given answer entity. Existing KBQG models still face two main challenges: (1) Most models often focus on the most relevant part of the answer entity, while neglecting the rest of the subgraph. (2) There are a large number of out-of-vocabulary (OOV) predicates in real-world scenarios, which are hard to adapt for most KBQG models. To address these challenges, we propose LFKQG, a controlled generation framework for Question Generation over Knowledge Bases. (1) LFKQG employs a simple controlled generation method to generate the questions containing the critical entities in the subgraph, ensuring the question is relevant to the whole subgraph. (2) We propose an optimization strategy called local fine-tuning, which makes good use of the rich information hidden in the pre-trained model to improve the ability of the model to adapt the OOV predicates. Extensive experiments show that our method outperforms existing methods greatly on three widely-used benchmark datasets SimpleQuestion, PathQuestions, and WebQuestions [1].

1 Introduction

Question generation (QG) aims to endow machines with the ability to ask relevant and to-the-point questions for a given form of data such as text (Du et al., 2017a; Song et al., 2018a), image (Li et al., 2018), table (Bao et al., 2018) and knowledge bases (KB) (Elsahar et al., 2018). KBQG aims to generate natural language questions given a subgraph in the KB, i.e. a set of connected triples of the form <subject, predicate, object>. KBQG is an effective approach to generating high-quality QA pairs that can significantly address the data scarcity issue for Knowledge Base Question Answering (KBQA). In addition, KBQG can be applied for educational purposes by producing practice assessments (Heilman and Smith, 2010) and can help dialog systems have more engaging conversations (Mostafazadeh et al., 2016).

Triple: <Alice Betty Stern, children, Otto Frank>, <Otto Frank, religion, *jew*>
Reference: What type of religion does Alice Betty Stern's heir have?
Q1: What religion does Otto Frank's children have ?

Triple: <12343 Martinbeech, discoverer or inventor, *Spacewatch*>
Reference: Who is the discovery organization that discovered 12343 Martinbeech?
Q2: What does 12343 Martinbeech discover?

Figure 1: The examples of two main challenges in KBQG where the questions are generated from the subgraph and the red texts are the answer entity. **Q1** only focus on the second triple but neglect the first.**Q2** does not contain any semantic of *discoverer or inventor*, which is an OOV predicatge.

Current KBQG systems follow an attention-based sequence-to-sequence structure (Elsahar et al., 2018; Kumar et al., 2019). To make use of the rich structure information hidden in the subgraph, (Chen et al., 2020) proposes the graph-to-sequence framework. However, these models face two main challenges: (1) Most models often focus on the most relevant part of the answer entity, while neglecting the rest of the subgraph (Bi et al., 2020; Chen et al., 2020). (2) There are many out-of-vocabulary (OOV) predicates in the real knowledge bases, which are unseen at the training time. Most KBQG models are hard to adapt to OOV predicates, which makes these models difficult to use in real-world scenarios (Elsahar et al., 2018). Figure 1 illustrates the examples of these two problems. **Q1** only focuses on the second triple but neglects the first. As for **Q2**, *discoverer of inventor* is an OOV

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

predicate, and the model is unable to handle this type of predicate, so the **Q2** does not contain any semantic relating to it.

To address these challenges, we propose LFKQG, a controlled generation framework. (Sun et al., 2018a; Fei et al., 2021) claim that the entity words in the given input play a decisive role in the semantics of the whole question. We can see that **Q1** only focuses on the second triple and miss the critical entity *Alice Betty Stern* in the first triple. Intuitively, all the critical entities should appear in the generated questions to ensure the generated questions contains the semantics of the whole subgraph. To this end, we introduce the controlled generation method to KBQG task. We use flag tag (Wang et al., 2021), a simple but effective lexical constraint for generation at each decoding step, to achieve the controlled generation. In detail, in decoding progressing, each input token is provided with a flag tag that indicates whether the constraint of this token has been satisfied. It is a strong incentive for the model to try to satisfy all constrains. Furthermore, the fine-tuning method distorts the pre-trained features for OOV samples, because the model over-fits the features for training data while removing the OOV features that were originally hidden in the pre-trained models (Zhang et al., 2021; Kumar et al., 2021). To address the OOV problem in KBQG, we propose a novel optimization strategy called local fine-tuning, which can retain the OOV features in the pre-trained model.

Extensive experiments show that our LFKQG model outperforms existing methods greatly on three widely-used benchmark datasets SimpleQuestion, PathQuestions, and WebQuestions. In addition, we conduct experiments on OOV data, and the results show that our local fine-tuning greatly improves the performance in this challenging scenario.

Our main contributions are summarized as follows:

1. We propose LFKQG, which employs a controlled generation framework for KBQG to make model generate questions relevant to the whole subgraph. We are the first one to introduce the controlled generation methods to the KBQG task.

2. We propose a novel optimization strategy called local fine-tuning to utilize the rich information hidden in the pre-trained features to address the OOV problem.

3. Experimental results show that our model greatly improves the performance. The experimental results on OOV data prove that local fine-tuning is able to improve the performance of a pre-trained generation model on OOV data.

2 Related Work

2.1 Question Generation

Early works on QG (Mostow and Chen, 2009; Heilman and Smith, 2010) focused on the rule-based approaches that rely on heuristic rules or hand-crafted templates, with low generalizability and scalability. Recent works adopted the attention-based sequence-to-sequence neural model for QG tasks, taking answer sentence as input and output the question (Du et al., 2017b), which proved to be better than rule-based methods. To generate a question for a given answer, (Sun et al., 2018a; Kim et al., 2019; Song et al., 2018b) applied various techniques to encode answer location information into an annotation vector corresponding to the word positions, thus allowing for better quality answer focused questions.

Recently, there is an increasing interest in Question Generation over Knowledge Bases (KBQG), sequence-to-sequence neural framework with RNN or Transformer have been applied to this task and are end-to-end trainable (Serban et al., 2016; Indurthi et al., 2017; Kumar et al., 2019; Chen et al., 2020). However, these works suffer the semantic drift problem (Zhang and Bansal, 2019). To solve the problem, (Elsahar et al., 2018) introduces a set of textual contexts paired through distant supervision. (Bi et al., 2020) employs grammarical information and introduce auxiliary information to model. These works focus on introduce extra knowledge information but do not exploit the rich knowledge information hidden in the pre-trained model. In this work, we design the local fine-tuning method to exploit the rich information hidden in the pre-trained model and solve semantic drift problem from a controlled generation perspective.

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

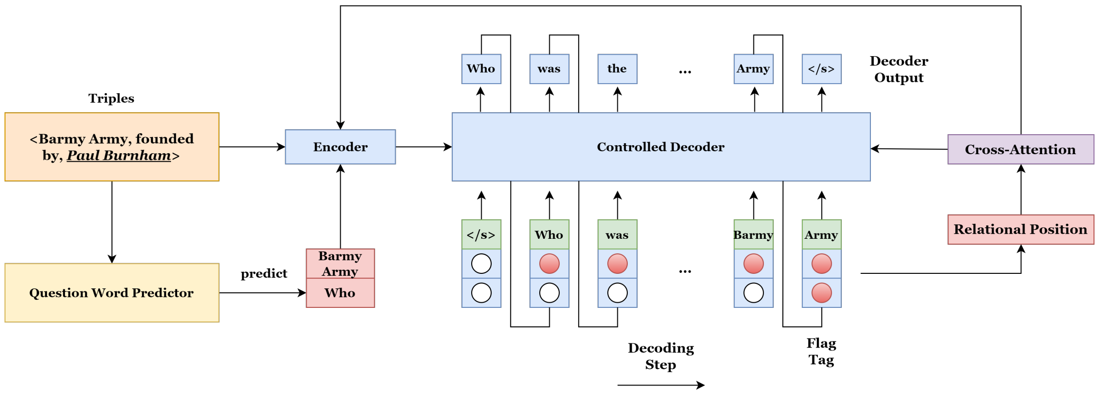


Figure 2: The overall architecture of our LFKQG. We firstly predict the question word and input both triples and question word to the controlled generator with flag tag to generate the questions.

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 (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. Furthermore, recent work uses stochastic search (Sha, 2020) or mention flag (Wang et al., 2021) to achieve hard control.

3 Methodology

In this section, we formalize the question generation over knowledge bases (KBQG) task and introduce our methodology. In particular, we describe our controlled generation framework for KBQG and show the overall architecture in Figure 2. Following this, we describe our local fine-tuning method and we show it in Figure 4.

3.1 Problem Formulation

The input to the KBQG task is a subgraph from the knowledge bases, which is a set of connected triples $X = \{T_1, .., T_n\}$ where the n is the number of triples and $T_i = \{S_i, P_i, O_i\}$ is a triple of the form $\{subject, predicate, object\}$. The desired goal of KBQG is to generate a question $Y = [y_1, ..., y_t]$ about entity S_1 and the answer is entity O_n conditioned on the whole subgraph.

3.2 Controlled Generation Framework

According to the existing research on question generation (Sun et al., 2018b; Bi et al., 2020; Fei et al., 2021), the entity word in the given input and question word are vital for the semantics of generated question. The critical entity and question word must appear in the generated question to make questions relevant to the whole subgraph. To this end, we need a controlled generator $G(Y|X, W, E)$ where X is the input sub-graph, W is the question word, and E is the critical entity for the subgraph, which must appear in the generated question. In this section, we first describe the model to predict question words and then describe a Transformer-based controlled generator.

3.2.1 Question Word Predictor

It is essential to predict the correct question word to control the question type and semantics (Zhou et al., 2019). We count the number of different question words on three KBQG datasets' testing set, and report the results in Table 1. We divide question words into 9 categories, including 8 common question words and an additional type "Others".

We use a BERT model (Devlin et al., 2018) to predict the question word. We joint the triples and answer entity with '[SEP]' and input it into the BERT model. The question word predictor predicts the question word as follow:

$$H = BERT(X) \tag{1}$$

$$h^q = H_{cls} \tag{2}$$

$$P(Q_w) = softmax(W_q h^q) \tag{3}$$

where X is the input token, W_q is a trainable matrix, and we use the hidden state in CLS to predict the question word and train the model as follow:

$$L_q = -log(P(Q_w)) \tag{4}$$

where L_q is the loss of question word prediction and Q_w is the target question word.

3.2.2 Transformer-based Controlled Generator

We employ the flag tag (Wang et al., 2021) to achieve controlled generator. In detail, 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:

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

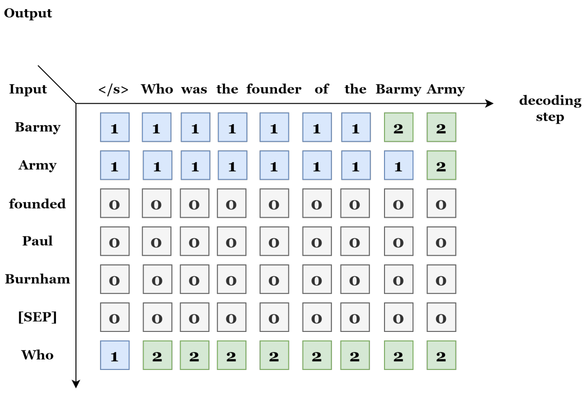
where $flag_i^t$ is the flag tag for ith input token at decoding step t, and $y_{1:t}$ is the generated tokens thus far. The tokens with the values 1 or 2 of the flag is a lexical constraint and the token with 0 is not constrained to appear in the question. Obviously, the flag tag for any token can only remain unchanged or updated to value 2.

We input the subgraph and question word into the controlled generator, so we set the question word and the critical entity S_1 in triples as the constrained tokens. As shown in Figure 3, the input tokens X is that X = [Barmy, Army, founded, Paul, Burnham, [SEP], Which] and the flag tag at the beginning is that $flag^0$ = [1,1,0,0,0,0,1] because the tokens are not constrained except key entity *Mendoza*, and question word *Which*. At step 2, the flags update to [1,1,0,0,0,0,2] because the token *Who* has been generated but *Barmy* and *Army* have not.

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 the decoder. The conventional cross-attention module is computed by:



Input	</s>	Who	was	the	founder	of	the	Barmy	Army
Barmy	1	1	1	1	1	1	1	2	2
Army	1	1	1	1	1	1	1	1	2
founded	0	0	0	0	0	0	0	0	0
Paul	0	0	0	0	0	0	0	0	0
Burnham	0	0	0	0	0	0	0	0	0
[SEP]	0	0	0	0	0	0	0	0	0
Who	1	2	2	2	2	2	2	2	2

Figure 3: An example for updates of flag tag.

$$Cross(Q, K, V) = softmax(\frac{Q^\top K}{\sqrt{d_k}})V \tag{5}$$

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

$$\alpha^t_{cross} = softmax(E^t) \tag{6}$$

$$E^t = \frac{Q^t(K + R^t)^\top}{\sqrt{d}} \tag{7}$$

$$R^t = Embedding(F^t) \tag{8}$$

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:

$$Cross(Q^t, K, V, F^t) = \alpha^t_{cross}V \tag{9}$$

where V is the outputs of encoder.

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

$$L(y_t, \tilde{y}_t) = -\frac{1}{T}\sum_{t=1}^{T} logP(\tilde{y}_t = y_t) \tag{10}$$

3.3 Local Fine-tuning Method

Some works on KBQG (Elsahar et al., 2018) claim that there are many out-of-vocabulary (OOV) predicates that are unseen at the training time in the real knowledge bases, but most KBQG models are hard to adapt. The OOV problem makes

Question Word	What	Which	Where	Who	Whose	Why	When	How	Others
SimpleQuestion	59.70 %	13.72 %	11.30 %	10.98 %	0.01 %	0.04 %	0.5 %	0.01 %	3.98 %
WebQuestion	52.35 %	13.20 %	5.85 %	11.60 %	0 %	0 %	0.7 %	0.2 %	16.10 %
PathQuestion	71.00 %	0.4 %	0.4 %	6.2 %	0 %	0.2 %	0 %	0.1 %	21.70 %

Table 1: Proportions of each type of questions on three datasets.

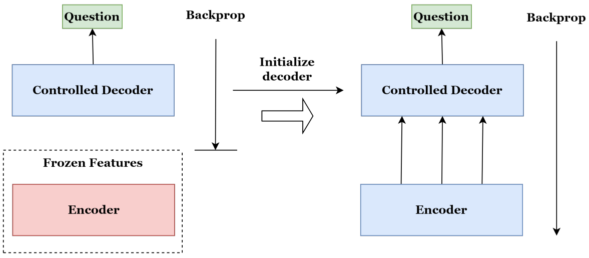


Figure 4: The describe of our local fine-tuning method.

most KBQG models difficult to use in real-world scenarios.

Pre-trained generation model like BART (Lewis et al., 2020) may be a good way to solve the OOV problem. As we know, the pre-trained model has seen a large number of predicates in the pre-training stage. In a word, the information of most OOV predicates is hidden in the pre-trained features. However, the performance of the pre-trained model on OOV data is still inferior. Recent works (Zhang et al., 2021; Kumar et al., 2021) claim the standard fine-tuning method has bad performance for few-sample task. (Kumar et al., 2021) studys the OOV samples in classifier tasks, and proves that the standard fine-tune method distorts the pre-trained features for OOV data, because the models over-fit the features for training data while removing the OOV features that were originally hidden in the pre-trained models. Motivated by this, we propose an optimization strategy for pre-trained generation models called local fine-tuning to retain the OOV features in the pre-trained models and address the OOV problem. We show the two-stage of our local fine-tuning method in Figure 4.

In detail, we first tune the parameters in the decoder but freeze parameters in the encoder to prompt the model to have the ability of KBQG based on the original encoder with rich pre-trained features as follow:

$$\theta_{decoder} = \underset{\theta_{decoder}}{\arg\min} L(y_t, \tilde{y_t}) \tag{11}$$

Then we tune all the model parameters with the decoder adapt to the original encoder as follow:

$$\theta_{model} = \underset{\theta_{model}}{\arg\min} L(y_t, \tilde{y_t}) \tag{12}$$

Since the model with the original encoder fits the KBQG task, the fine-tuning method only changes the pre-trained features a bit.

Local fine-tuning is a simple but effective method for the OOV problem on the KBQG task, and we analyze it in section 4.7.

4 Experiment

In this section, we conduct extensive experiments to evaluate the effectiveness of our proposed model for the KBQG task.

4.1 Data and Metrics

We conduct experiment on three widely-used benchmark datasets: SimpleQuestion (Bordes et al., 2015), PathQuestions (Zhou et al., 2018), and WebQuestions (Kumar et al., 2019).

SimpleQuestion consists of over 108,000 samples, and the entities are represented by their Freebase IDs (Bollacker et al., 2008). Each instance in SimpleQuestion contains a triple and a natural language question where the answer is the object entity in triple. Following (Bi et al., 2020) we first map these Freebase IDs to Wikidata IDs and transfer them to the natural language by Wikidata (Vrandečić and Krötzsch, 2014), then we extract the samples whose entity can not be found in Wikidata. We randomly selected 70% of these samples for training, 10% for validation, and 20% for testing.

WebQuestions and PathQuestions use Freebase as the underlying. The WebQuestions dataset combines examples for WebQuestionsSp (Yih et al., 2016) and ComplexWebQuestions (Talmor and Berant, 2018) where both of them are KBQA benchmarks that contain natural language questions, corresponding SPARQL queries, and answer entities. The PathQuestions dataset is similar to WebQuestions except that the KG subgraph in PathQuestions is a path between two entities that

Model	**SimpleQuestion**			**WebQuestions**			**PathQuestions**		
	BLEU-4	**METEOR**	**ROUGE-L**	**BLEU-4**	**METEOR**	**ROUGE-L**	**BLEU-4**	**METEOR**	**ROUGE-L**
RNN-based (Indurthi et al., 2017)	19.98	28.43	46.02	-	-	-	25.78	33.17	50.78
Zero-shot (Elsahar et al., 2018)	22.71	30.39	51.07	-	-	-	29.44	38.12	56.94
MHQG (Kumar et al., 2019)	25.98	34.14	56.03	11.57	29.69	35.53	25.99	33.16	58.94
BiGraph2Seq (Chen et al., 2020)	31.12	39.23	62.14	29.45	30.96	55.45	61.48	44.57	77.72
T5 (Raffel et al., 2020)	35.32	40.16	64.97	28.78	30.55	55.12	58.95	44.72	76.58
IGND (Fei et al., 2021)	32.67	41.62	65.74	30.62	31.42	55.82	61.69	45.11	77.28
LFKQG	**38.35**	**42.06**	**66.59**	**31.66**	**32.69**	**56.75**	**63.92**	**46.91**	**78.40**

Table 2: The automatic evaluation for different models on three datasets.

span two or three hops. (Kumar et al., 2019) releases these two dataset. PathQuestion dataset contains 9,793/1,000/1,000 and WebQuestions contains 18,989/2,000/2,000 examples.

Following previous works (Elsahar et al., 2018; Chen et al., 2020), we use BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005) and ROUGE-L (Lin, 2004) as automatic evaluation metrics. BLEU and METEOR were designed for evaluating machine translation systems, and ROUGE-L was developed for evaluating text summarization systems.

4.2 Experimental settings

We use the BART-base model loaded from transformers in huggingface library [2]. The embedding size and head hidden size of the flag tag are 64. We use the AdamW (Loshchilov and Hutter, 2018) 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 8 epochs. We clip the gradient at length 10. The batch size is 64 and the beam search width 5. All hyperparameters are tuned on the development set.

4.3 Baselines

We compare our method with the following baseline models.

RNN-based: a RNN-based question generation model to generate natural language question-answer pairs from a knowledge graph (Indurthi et al., 2017).

Zero-Shot: a zero-shot KBQG model for OOV predicates and entity types (Elsahar et al., 2018).

MHQG: an end-to-end neural network-based method for automatic generation of complex multi-hop questions over knowledge graphs (Kumar et al., 2019).

BiGraph2Seq: a novel bidirectional Graph2Seq model to generate questions from a KB subgraph and target answers (Chen et al., 2020).

Model	**Syntactic**	**Complexity**	**Relevance**
T5	4.23	3.26	3.14
BiGraph2Seq	3.61	3.56	3.47
IGND	3.72	3.65	3.52
LFKQG	**4.31**	**3.81**	**3.96**
Ground Truth	4.89	4.92	4.87

Table 3: The human evaluation results.

T5: A strong pre-trained language model that is a unified framework that converts every language problem into a text-to-text format (Raffel et al., 2020).

IGND: A QG model that propose a novel iterative graph-based decoder to use the rich structure information hidden in the text (Fei et al., 2021).

4.4 Main Results

The results of the automatic evaluation are shown in Table 2. We compare our proposed models against other state-of-the-art methods on SimpleQuestion, WebQuestions, and PathQuestions test sets. We can see that our models outperform all QG baselines by a large margin on all benchmarks, which verifies the effectiveness of our model. Our model achieves the state-of-the-art on three benchmarks. Not only in BLEU-4, but our model also 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 of SimpleQuestion dataset for human evaluations.

Generated questions are rated in the range 1-5 based on whether they are syntactically correct, complexity, and relevant to the given sub-graph. Following (Chen et al., 2020), we ask 5 human evaluators to give feedback on the quality of

[2] huggingface.co/transformers

Model	**BLEU-4**
LFKQG	38.35
LFKQG w/o controlled decoder	35.19
LFKQG w/o local fine-tuning method	37.29
LFKQG w/o fine-tuning all the model parameters	37.64
LFKQG w/o controlled decoder + local fine-tuning	34.21

Table 4: The ablation test results on SimpleQuestion dataset.

questions generated by different models. For each sample, given a sub-graph, target answers, and model output, we ask the evaluators to rate the quality of the generated questions to answer the following three questions: 1) is this generated question syntactically correct? 2) is this generated question need all the information in the subgraph to answer? And 3) is this generated question relevant to the sub-graph and target answers? The rating scale is from 1 to 5 to measure the quality of questions, and a higher score means better quality.) We average the scores from raters on each question and report the performance of Ground-truth, IGND, T5, BiGraph2Seq, and our model. Workers were unaware of the identity of the models in advance. We show the results in Table 3.

We can see that pre-trained model T5 has much better than BiGraph2Seq and IGND syntactically. But Bigraph2Seq and IGND employ a graph network to use the rich structure information hidden in the subgraph, so they can understand the input sub-graph better and generate the questions with the higher score in relevance and complexity. Our controlled generation framework with local fine-tuning achieves the best performances in all aspects. Our model guarantees that the critical entity and correct question words appear in the questions, significantly improving relevance and complexity performance. Local fine-tuning help the model understand input better and improve relevance performance.

Model	BLEU-4
BART with fine-tuning	20.12
BART only fine-tuning decoder	20.51
BART with local fine-tuning	**22.29**
LFKQG with fine-tuning	21.65
LFKQG Generator only fine-tuning decoder	22.17
LFKQG with local fine-tuning	**24.62**

Table 5: The results of different optimization strategy for OOV samples.

Model	**Question Word Accuracy**	**Key Entity Converage Percentage**
BART	61.34%	67.92%
LFKQG	74.61%	81.74%

Table 6: Experiments of the question word accuracy and key entity converage percentage.

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 SimpleQuestion test set and show the results in Table 4.

LFKQG w/o controlled decoder The model removes the controlled decoder and employs the standard BART model with the local fine-tuning method.

LFKQG w/o local fine-tuning method We fine-tune all parameters in our model with a controlled decoder rather than two-stage local fine-tuning.

LFKQG w/o fine-tuning all the model parameters We only fine-tune the parameters in the decoder but freeze parameters in the encoder.

LFKQG w/o controlled decoder + local fine-tuning The model removes the controlled decoder and local fine-tuning method.

Firstly, there is a huge gap between LFKQG and LFKQG w/o controlled decoder + local fine-tuning, demonstrating that our controlled generation framework with the local fine-tuning method plays an important role. Comparing LFKQG and LFKQG w/o controlled decoder, we find that the controlled decoder is the critical module in our model.

Secondly, LFKQG is higher than LFKQG w/o local fine-tuning method 1.06 of BLEU-4 points. We can find that the local fine-tuning method remain the OOV features hidden in the pre-trained models and improves OOV samples' performance.

Thirdly, LFKQG w/o fine-tuning all the model parameters is lower than our model, only 0.71 of BLEU-4, and it is even higher than LFKQG w/o local fine-tuning method. This exciting comparison shows the pre-trained features in the encoder without fine-tuning are good enough for KBQG, and the fine-tuning is not the best optimization strategy for KBQG.

4.7 Analysis for Local Fine-tuning Method

In this section, we analyze the effectiveness of the local fine-tuning Method for OOV samples. At first we mimic the real world to construct the OOV

Input:	<I Saw the Light, lyrics by, Hank Williams>
Gold:	Who was the lyricist from I Saw the Light?
Baseline:	Who wrote I Saw the Light ?
LFKQG:	Who is the lyricist of I Saw the Light?
Input:	<Mendoza, contains administrative territorial entity, Lavalle Department>
Gold:	Which location is the administrative child of Mendoza province ?
Baseline:	What is the Mendoza's territorial entity?
LFKQG:	Which location is the administrative of Mendoza province ?
Input:	<Alice Betty Stern, children, Otto Frank>, <Otto Frank, religion, jew>
Gold:	What type of religion does Alice Betty Stern's heir have?
Baseline:	What religion does Otto Frank's children have ?
LFKQG:	What religion does Alice Betty Stern's child have?

Table 7: Case study of three examples from SimpleQestion and PathQuestions test set. We indicate the key entities by blue, the OOD predicates by cyan, and the answer entity by red.

dataset based on the annotated dataset. In detail, we extract the samples whose predicates are never seen in the training set from the SimpleQuestion testing set. Then we conduct some experiments to evaluate the performance of different optimization strategies on the OOV dataset. We show the results of two models, our controlled generator and BART-base model (Lewis et al., 2020), with different optimization strategies in Table 5.

We can see that the local fine-tuning method improves the performance of OOV on both two models significantly. In addition, compared to the models with fine-tuning, the models only tuning decoder also obtain a higher BLEU-4 score for OOV. We think the results prove the phenomenon, fine-tuning method distorts the pre-trained features that happened in the classifier task, also appear in the KBQG task. The results also show our local fine-tuning method retains the OOV features in the pre-trained models to improve the performance of OOV.

4.8 Analysis for Controlled Generator

We conduct some experiments to analyze the controlled generator on SimpleQuestion dataset in this section. We evaluate different models in terms of question word accuracy. This metric measures the ratio of the generated questions that share the same beginning word with the references which begin with a question word Similarly, we evaluate the critical entity coverage percentage, which measures the ratio of the critical entity S_1, we describe in section 3.1, appears in the generated questions. The two metrics can show the ability of controlled generator, and we report the results in Table 6. We can find that our model's two metrics are much higher than other models. This result shows that our controlled generator improves the control of the model generation process.

4.9 Case Study

To intuitively show the generation quality of our model, we provided some generated cases in Table 7. Our model can generate high-quality texts that describe the knowledge graph more completely and faithfully.

It is clearly shown the three questions generated by the baseline model face the two main challenges for KBQG. In contrast, our model generates the questions without these problems. These three examples show our model can 1) retain the pre-trained features to handle the OOV data as shown in the first example and second example, 2) predict the correct question word and make it appear in the question to control the type of question as shown in the second example, 3) make the critical entity appear in the question relevant to the whole subgraph as shown in the third example.

5 Conclusion

The KBQG task is challenging and worthy of exploration. To address the two main challenges of KBQG, we propose LFKQG, including the controlled generation framework and local fine-tuning method. The controlled generation framework makes the given question word, and critical entity in the subgraph appear in the question to control the semantic and the type of question. The local fine-tuning method can retain the OOV features hidden in the pre-trained models. In addition, we find that the phenomenon that fine-tuning method distorts the pre-trained features also appears in the KBQG task. It may be an exciting way to study the pre-trained generation models.

Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially National Natural Science Foundation of China (No. 62076069, 61976056), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103). This work was sponsored by CCF-Tencent Open Fund.

References

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

Junwei Bao, Duyu Tang, Nan Duan, Zhao Yan, Yuanhua Lv, Ming Zhou, and Tiejun Zhao. 2018. Table-to-text: Describing table region with natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Sheng Bi, Xiya Cheng, Yuan-Fang Li, Yongzhen Wang, and Guilin Qi. 2020. Knowledge-enriched, type-constrained and grammar-guided question generation over knowledge bases. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2776–2786.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*.

Yu Chen, Lingfei Wu, and Mohammed J Zaki. 2020. Toward subgraph guided knowledge graph question generation with graph neural networks. *arXiv preprint arXiv:2004.06015*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Xinya Du, Junru Shao, and Claire Cardie. 2017a. 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.

Xinya Du, Junru Shao, and Claire Cardie. 2017b. 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.

Hady Elsahar, Christophe Gravier, and Frederique Laforest. 2018. Zero-shot question generation from knowledge graphs for unseen predicates and entity types. *arXiv preprint arXiv:1802.06842*.

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.

Sathish Reddy Indurthi, Dinesh Raghu, Mitesh M Khapra, and Sachindra Joshi. 2017. Generating natural language question-answer pairs from a knowledge graph using a rnn based question generation model. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 376–385.

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.

Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. 2021. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *International Conference on Learning Representations*.

Vishwajeet Kumar, Yuncheng Hua, Ganesh Ramakrishnan, Guilin Qi, Lianli Gao, and Yuan-Fang Li. 2019.

Difficulty-controllable multi-hop question generation from knowledge graphs. In *International Semantic Web Conference*, pages 382–398. Springer.

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.

Yikang Li, Nan Duan, Bolei Zhou, Xiao Chu, Wanli Ouyang, Xiaogang Wang, and Ming Zhou. 2018. Visual question generation as dual task of visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6116–6124.

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

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

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.

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

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 of the Association for Computational Linguistics*, pages 311–318.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. *arXiv preprint arXiv:1804.06609*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Iulian Vlad Serban, Alberto Garcia-Duran, Çağlar Gulçehre, Sungjin Ahn, Sarath Chandar, Aaron Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 588–598.

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.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018a. 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.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018b. 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.

Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. 2018a. 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.

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

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. 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 641–651.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

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.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206.

Shiyue Zhang and Mohit Bansal. 2019. Addressing semantic drift in question generation for semi-supervised question answering. 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 2495–2509.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2021. Revisiting few-sample {bert} fine-tuning. In *International Conference on Learning Representations*.

Mantong Zhou, Minlie Huang, and Xiaoyan Zhu. 2018. An interpretable reasoning network for multi-relation question answering. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2010–2022.

Wenjie Zhou, Minghua Zhang, and Yunfang Wu. 2019. Question-type driven question generation. 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 6032–6037.

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*.