Graph-based Semantic Difficulty Controllable Question Generation

Hanbee Jang*, Junyeong Park*, Minho Cha*, Seohyeon Kim*
Korea Advanced Institute of Science and Technology (KAIST)

Abstract

This paper proposes several methods that can develop the performance of the existing Difficulty Controllable Question Generation (DCQG) system. In the previous DCQG system, there was a limitation in generating only syntactically complex or simple questions by defining difficulty by simply the number of inference steps. We introduce question paraphrasing and context simplification to make the difference in the semantic difficulty of questions more noticeable. In our work, we propose a new semantic difficulty controllable question generation model that can control semantic question difficulty rather than syntactical difficulty.

1 Introduction

Difficulty Controllable Question Generation (DCQG) is a QG system that generates questions with required difficulty levels. It is a system that has recently attracted the attention of researchers due to its wide range of applications, such as curriculum-learning approaches and educational purpose. Following the guidance of the extracted reasoning chain, the question difficulty is gradually increased through step-by-step rewriting.

There are few studies on the difficulty controllable QG system among the previous QG researches. In addition, in the case of the previous DCQG system(Cheng et al., 2021), a multi-hop question is generated from the context through a graph-based approach, which has several problems. There are questions which the sentence structure is not reasonable. Also, because the questions are generated based on the reasoning graph, when a multi-hop question is created, the sentence structure becomes complicated rather than difficult. To address these problems, our paper introduces question paraphrasing and context simplification.

In the existing DCQG system, questions are generated simply by going through multiple hops. However, this means that questions are created based on only the vocabularies in the context, so it is relatively easy to answer the questions by matching the words in the question to the context. Therefore, the questions that are only syntactically complex are not truly difficult questions. The difficult question should be a question that can only be solved by truly understanding the meaning of the context. Therefore, we propose question paraphrasing as a way to create more difficult questions. This is because, by introducing paraphrasing, questions could be created with new vocabularies other than the vocabularies in the original context. Thus, it requires comprehension of the semantic meaning of the question and context to answer the question.

In addition, the existing DCQG system classifies questions like the following as an easy question. *"Robert Godsell, is a retired South African businessman and the former CEO of South African gold mining company AngloGold Ashanti, a position he held from 1998 to 2007, he was the CEO of which South African gold mining company?"* In this question, the number of logical inference steps itself is small due to the small number of hops. However, the sentence length is very long and there are many modifiers, so it is hard to recognize that this is an easy question. Therefore, by leaving only the main contents of the context of the easy question through context simplification, we avoid the question of strange sentence structure. By making the sentence structure simpler and only leaving the core content, an easier and less confusing question, which can be answered at a glance can be created. These new questions will widen the gap between difficulty levels in the difficulty controllable question generation.

Finally, in baseline test using SQuAD, T5 and Bart performed better than GPT2, so we tried to improve performance by using T5 and Bart instead of GPT2.

In summary, our contributions are as follows:

- We propose a novel semantic DCQG framework that complements the existing syntactical DCQG system.
- We introduce question paraphrasing and context simplification as a way to make the semantic difficulty difference more pronounced.

2 Approach

In this paper, we propose three approaches to increase the performance of the prior work of DCQG; question paraphrasing, context simplification, and changing question generating model. The first two approaches aims to make difference in the required comprehension level of the reader to answer the question, thus providing new difficulty level and widening the gap between existing difficulty level of questions. The last approach replaces the question generation language model to other language model in attempt for better performance.

Question Paraphrasing

We paraphrased the question to make difficult question more semantically difficult. Paraphrasing was applied to $\text{QG}_{\text{initial}}$ and $\text{QG}_{\text{rewrite}}$ respectively. As the initial question is generated by using $\text{QG}_{\text{initial}}$ and repeatedly rewritten with $\text{QG}_{\text{rewrite}}$ to generate more syntactically complex question, by applying paraphrasing in those steps, more syntactically complex and semantically difficult questions are generated with more rewriting process.

Context Simplification

We simplified the context to make easy question semantically easier (Laban et al., 2021). Some existing 1-hop questions, easy questions according to the original difficulty classification, are hard to answer as mentioned in Sec.1. This is mainly because a question is generated based on the context that often contains unnecessary modifiers. Therefore, the questions were generated with unnecessary or excessive information which would only confuse the readers. Therefore, to make the question concise and only contain the semantically essential information, we applied simplification to the context.

Changing Question Generation Model

We changed question generating model. In the previous DCQG work (Cheng et al., 2021), pretrained GPT2-small language model was fine-tuned with the dataset for question generation. In the baseline test using SQuAD-1.1 dataset, T5 and Bart showed superior performance over GPT2 model. Thus, we replaced the question generation language model in this experiment with the T5 and Bart model for the better performance.

3 Experiments

3.1 Dataset

First, we experiment on extractive question answering dataset SQuAD-1.1 (Rajpurkar et al., 2016) consisting of 100K question/answer pairs. Then for multi-hop question generation experiments, we use HotpotQA (Yang et al., 2018) question answering dataset, consisting of 113K Wikipedia-based and crowdsourced question/answer pairs, including 90K training sets, 7.4K development sets and 7.4K test sets.

3.2 Implementation Details

Baseline

We test question generation with following baselines on SQuAD-1.1 dataset and HotpotQA dataset.

- **GPT2** (Radford et al., 2018) is a transformer text generation model. We used huggingface *p208p2002/gpt2-squad-qg-hl* as a baseline. The input is a context which the answer of the question is denoted with [HL] token at the start and end of the answer. The generated output consists of input text and the generated question which the generated question is concatenated after the input text.
- **Bart** (Lewis et al., 2019) is a seq2seq transformer model. We used huggingface *p208p2002/bart-squad-qg-hl* as a baseline. The input is same as GPT2 model, and the model only generates the question.
- **T5** (Raffel et al., 2019) is a text2text framework. We used huggingface *p208p2002/t5-squad-qg-hl* as a baseline. The input is the same as GPT2 model, and the model only generates the question.

The upper models are trained on SQuAD dataset. For HotpotQA dataset baseline, the upper huggingface models are fine-tuned on HotpotQA dataset for 3 epochs each.

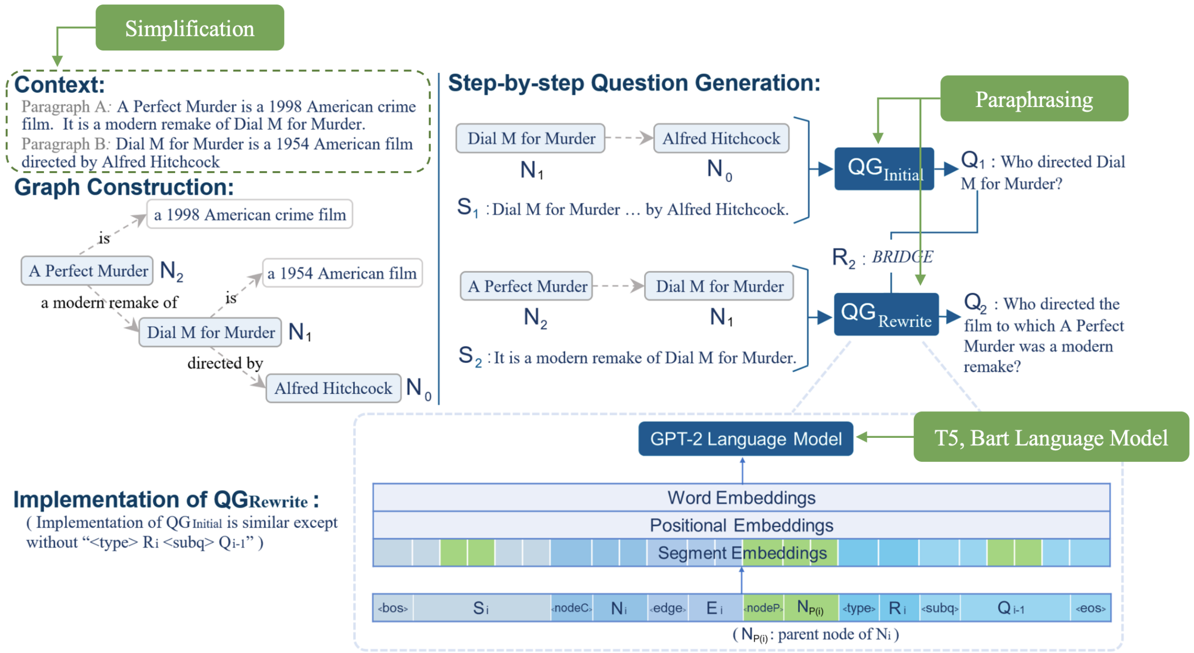
Simplification
Context:
Paragraph A: A Perfect Murder is a 1998 American crime film. It is a modern remake of Dial M for Murder.
Paragraph B: Dial M for Murder is a 1954 American film directed by Alfred Hitchcock
Graph Construction:
a 1998 American crime film
is
A Perfect Murder
N₂
a modern remake of
is
a 1954 American film
Dial M for Murder
N₁
directed by
Alfred Hitchcock
N₀
Step-by-step Question Generation:
Paraphrasing
Dial M for Murder
Alfred Hitchcock
N₁
N₀
S₁ : Dial M for Murder … by Alfred Hitchcock.
QG Initial
Q₁ : Who directed Dial M for Murder?
R₂ : BRIDGE
A Perfect Murder
Dial M for Murder
N₂
N₁
S₂ : It is a modern remake of Dial M for Murder.
QG Rewrite
Q₂ : Who directed the film to which A Perfect Murder was a modern remake?
GPT-2 Language Model
T5, Bart Language Model
Implementation of QGRewrite :
(Implementation of QG Initial is similar except without "<type> Ri <subq> Qi-1")
Word Embeddings
Positional Embeddings
Segment Embeddings
<bos>
Si
<nodeC>
Ni
<edge>
Ei
<nodeP>
NP(i)
<type>
Ri
<subq>
Qi-1
<eos>
(NP(i) : parent node of Ni)

Figure 1: An overview of our proposed framework

Question Generation

Figure 1 shows the overall architecture of the question generation model. From the input context, answer and the difficulty level d, a question is generated. First, a context graph is constructed with given context (Fan et al., 2019). The <subject, relation, object> triples are extracted from context sentences, and each subject and object becomes the nodes and relation becomes the edge. The nodes from different sentences are merged by coreference resolution. Second, to select a reasoning chain, we sample a connected subgraph of size d from the context graph. The answer of the question becomes the root node of the connected subgraph. Third, we generate question by step-by-step rewriting. We first use $\mathrm{QG}_{\mathrm{initial}}$ to generate initial simple question based on the root node and the following node. Then, we use $\mathrm{QG}_{\mathrm{rewrite}}$ to rewrite question into more complex question by sampling bigger subgraph. In the implementation, the $\mathrm{QG}_{\mathrm{initial}}$ and $\mathrm{QG}_{\mathrm{rewrite}}$ are initialized with pretrained GPT2-small model, and fine-tuned with our dataset.

The generated question $\mathrm{QG}_{\mathrm{initial}}$ is compared with the generated question of baselines.

Multi-hop Question Generation

We also generate multi-hop question with question rewriting $\mathrm{QG}_{\mathrm{rewrite}}$. As question is rewritten, it becomes more complex and difficult because the question rewriting step adds inference steps to answer the question. There are two rewriting type patterns, Bridge and Intersection. Bridge type rewrites question by replacing an entity with a modified clause. Alternatively, the Intersection type rewrites question by adding another restriction to an existing entity in the question.

Dataset Construction

Most questions in HotpotQA dataset require two hops of reasoning, each concerning one paragraph. We automatically decompose these question. First, we distinguish the reasoning type of the question and filter out those that are not Bridge and Intersection. Then decompose the question into two sub-questions based on linguistic rules. These sub-questions and type of rewriting are used as an additional information for training the model.

Question Paraphrasing

For paraphrasing, we used huggingface *ramsrigouthamg/t5_paraphrase* model on question to generate paraphrased question. Paraphrasing is done between question rewriting. The generated $\mathrm{QG}_{\mathrm{initial}}$ is paraphrased before rewrited. Also, the generated $\mathrm{QG}_{\mathrm{rewrite}}$ is paraphrased before next rewriting step.

Context Simplification

For simplification, we used huggingface *philippelaban/keep_it_simple* model on the context. The simplified context is used to generate question with the same process.

Changing Question Generation Model
Instead of using GPT2-small model, we tried using Bart-base and T5-base model for better performance. We used pre-trained Bart-base and T5-base model and fine-tuned with our dataset. In question generation, we generated one word at a time with those models by top filtering strategy without cache.

3.3 Evaluation Metric

The main metric we used are BLEU3, BLEU4 (Papineni et al., 2002), METEOR (Lavie and Agarwal, 2007), CIDEr (Vedantam et al., 2015). Metrics measure similarity between sentences regarding n-gram. Among the metrics, BLEU3, BLEU4, and CIDEr focus on exact word matching while METEOR also considers stemming and synonym matching along with the standard exact word matching. Additionally, we used BertScore (Zhang et al., 2019) and MoverScore (Zhao et al., 2019) for evaluating question paraphrasing and context simplification. BertScore and MoverScore considers factors other than the exact word matching. BertScore computes a similarity score for each token in the candidate sentence with each token in the reference sentence. MoverScore is a metric combining contextualized embedding and Earth Mover's Distance so that it considers semantics rather than surface forms. Therefore, for experiments which needs semantic matching evaluation, we added MoverScore and BertScore to base metrics. In this paper, we used BLEU3, BLEU4, and CIDEr for exact word matching and METEOR, BertScore, and MoverScore for semantic word matching.

4 Result

4.1 Question Generation

These are the evaluation results of question generation on SQuAD-1.1 and HotpotQA dataset. We test GPT2, T5, and Bart model, as they are frequently used in text generation task.

SQuAD-1.1
The evaluation result of SQuAD-1.1 dataset is shown in Table 1. The previous DCQG model (Cheng et al., 2021) was based on GPT2 model, and we used huggingface models for GPT2, T5, and Bart. T5 and Bart showed good performance on SQuAD-1.1, and their evaluation results were higher than the previous model and GPT2 in the most evaluation metrics. The performance of T5 and Bart was better than GPT2.

Model	BLEU3	BLEU4	METEOR	CIDEr
GPT2	11.30	7.75	17.18	0.77
T5	22.50	17.76	24.06	1.77
Bart	**24.25**	**19.40**	**25.31**	**1.93**
DCQG (Ours)	20.40	15.26	19.83	1.25

Table 1: Evaluation result on SQuAD-1.1 dataset

HotpotQA
We also trained models on HotpotQA dataset which is used for multi-hop question generation. We used T5 and Bart model which are pretrained on SQUAD-1.1 dataset. Then, we fine-tuned them on 1/3 of HotpotQA dataset with 3 epochs each. Also, we implemented multi-hop question generation model on GPT2 based on HotpotQA dataset. The result is shown in Table 1.

Model	BLEU3	BLEU4	METEOR	CIDEr
T5	16.43	13.03	18.23	1.39
Bart	15.80	12.87	17.75	1.39
DCQG (Ours)	14.51	10.88	17.72	0.78
DCQG (Cheng et al., 2021)	**20.98**	**15.59**	**24.19**	**1.46**

Table 2: Evaluation result on HotpotQA dataset

4.2 Question Paraphrasing

The evaluation results are shown in Table 3 and Table 4. For this experiment, we also used BertScore and MoverScore for evaluating semantic matching. A n-hop question is rewritten based on the not paraphrased (n-1)-hop question. With paraphrased question as input, the generated question consists of different vocabularies and have different structure compared to the original generated question. The generated questions with paraphrasing has low performance in BLEU and CIDer scores, decreasing over 35%. However, the METEOR, BertScore, and MoverScore showed less performance loss.

	BLEU3	BLEU4	CIDEr
2-hop	18.98	15.18	1.10
2-hop (paraphrasing)	11.94 (-37%)	8.85 (-42%)	0.68 (-38%)
3-hop	20.00	15.93	1.04
3-hop (paraphrasing)	13.06 (-35%)	8.58 (-46%)	0.64 (-39%)

Table 3: Evaluation result of question paraphrasing with exact matching evaluation metrics

The evaluation metric which measure exact word matching showed huge performance loss while the evaluation metric which measure semantic word matching showed less performance loss. This suggest that by paraphrasing, the vocabularies and sentence structure are changed from those in the context just like our intention.

Also, the model sometimes unexpectedly added more context to question. According to the Table 5, not only vocabularies and sentence structure are changed, but also question contains more context than generated question with original method. Therefore, to answer the question, we should understand the meaning of question not only find the same vocabularies in the context. We conclude that the question become semantically difficult through paraphrasing.

	METEOR	BertScore	MoverScore
2-hop	19.94	77.24	30.15
2-hop (paraphrasing)	16.32 (-18%)	74.82 (-3%)	26.25 (-13%)
3-hop	20.61	77.00	29.87
3-hop (paraphrasing)	17.23 (-16%)	74.62 (-3%)	24.98 (-16%)

Table 4: Evaluation result of context simplification with semantic matching evaluation metrics

1-hop	Who is the artist features vocals on the song "Rock City"?	
	original	paraphrasing
2-hop	Who is the artist that features vocals on the song "Rock City"?	Who sings on "Rock City" and was a member of the duo Bad Meets Evil
3-hop	Who is the singer on "Rock City" who is a member of the duo Bad Meets Evil?	Who sings on "Rock City" and was a member of the duo Bad Meets Evil that is composed of Detroit-based rappers, Royce da 5’9" and who?

Table 5: Example of question generated by paraphrasing

4.3 Simplification

This is the evaluation result of context simplification. The model show good performance with context simplification, especially in semantic matching evaluation metrics. Also, from the example, the model successfully generate simplified questions by context simplification. This suggest that the context simplification help generating easier questions by removing the unnecessary words but keeping the important semantic content.

DCQG	What is the name of the river divides the country into eastern and western halves?
DCQG + simp	What river divides the country into eastern and western halves?

Table 6: Example of question generated by context simplification

Model	BLEU3	BLEU4	CIDEr
DCQG	**19.39**	**16.00**	1.37
DCQG + simp	19.32	15.97	**1.40**

Table 7: Evaluation result of context simplification with exact matching evaluation metrics

Model	METEOR	BertScore	MoverScore
DCQG	20.21	78.32	31.13
DCQG + simp	**20.49**	**78.37**	**31.14**

Table 8: Evaluation result of context simplification with semantic matching evaluation metrics

4.4 Change Question Generation Model

We tried various ways to change the language model to T5 and Bart, but we failed in the end. The output of model() and model.generate() method generated the original context, which is the same as input. We also tried simplifying the input sequence in the training process to <*sos*> *question* <*answer*> *answer* <*question*> *question*, but it was not effective. Finally, we tried generating one word at a time by top filtering strategy without cache. In the example, the question itself is generated well. The model generates question well which starts with ‘where’, ‘what’, and ‘how’, and ends with a question mark. However, in the example, while the context and original question is about the director of the romance comedy film, the generated question mentions Hollywood's debut album, which is unrelated to the context. Like this, the model tends to ignore the input context and generate unrelated question. We suspect that the top filtering strategy did not work well.

Context	Big Stone Gap is a 2014 American drama romantic comedy film written and directed by Adriana Trigiani and produced by Donna Gigliotti ...
Original Question	The director of the romantic comedy "Big Stone Gap" is based in what New York City?
Generated Question	What other parties are considered in what category (Hollywood's debut album Best Number C) is considered in what era?

Table 9: Example of question generated by changing question generation model to Bart-base model

5 Conclusion

We challenged the difficulty-controllable question generation task. We proposed a novel semantic DCQG framework that complements the existing syntactic DCQG system. Question paraphrasing and context simplification was proposed to control semantic difficulty of the generated question. The questions are paraphrased in the step-by-step question rewrite process to generate semantically difficult questions by introducing vocabularies that do not exist in the original context. The context is simplified to generate less confusing, simpler and easier question. Extensive evaluations show that we have generated semantically difficult and easy questions and keep high question quality at the same time.

6 Limitations & Future Work

In this work, the generated questions are evaluated by comparison to the golden question. Even though we introduced semantic matching metrics for extensive evaluation, comparing the question to the golden question is still lacking in measuring the completeness and difficulty of the question. Especially, we need a metric for measuring the semantic difficulty. We believe we need human-evaluation more convincing result.

In changing the question generation model to T5 and Bart, the top filtering strategy failed. We believe finding appropriate top p and top k parameters or trying other text generation strategies such as beam search could be effective.

References

Yi Cheng, Siyao Li, Bang Liu, Ruihui Zhao, Sujian Li, Chenghua Lin, and Yefeng Zheng. 2021. Guiding the growth: Difficulty-controllable question generation through step-by-step rewriting. 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 5968–5978, Online. Association for Computational Linguistics.

Angela Fan, Claire Gardent, Chloe Braud, and Antoine Bordes. 2019. Using local knowledge graph construction to scale seq2seq models to multi-document inputs.

Philippe Laban, Tobias Schnabel, Paul Bennett, and Marti A. Hearst. 2021. Keep it simple: Unsupervised simplification of multi-paragraph text.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. pages 228–231.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461.

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.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4566–4575.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance.