CrossQG: Improving Difficulty-Controllable Question Generation through Consistency Enhancement

Anonymous ACL submission

Abstract

Automatically generating questions with controlled difficulty has great application value, especially in the field of education. Although large language models have the capability to generate questions of various difficulty levels, the generated questions often fail to align with the given target difficulty. To mitigate this issue, we propose CrossQG, a novel training-free question generation method that enhances difficulty consistency. Specifically, CrossQG consists of two steps: (1) contrast enhancement, which leverages questions from different difficulty levels to enhance the base models' understanding of the target difficulty, and (2) cross filtering, which compares generated questions across different difficulty levels and filters out those that do not meet the target difficulty. We evaluate CrossQG on three high-quality question answering datasets, applying two difficulty estimation schemata. Experimental results demonstrate that across multiple models, CrossQG significantly outperforms several mainstream methods, achieving superior consistency with target difficulty and improving question quality. Moreover, CrossQG surpasses supervised fine-tuning in various instances even without training.

1 Introduction

The task of Difficulty-Controllable Question Generation (DCQG) aims to generate questions with controlled difficulty levels. It holds significant application value in education, such as improving learning efficiency (Uto et al., 2023; Wang, 2014) and better assessing learners' abilities (Benedetto et al., 2023). The main difficulties of DCQG lie in: (1) appropriately categorizing the difficulty levels of the questions, and (2) ensuring that the generated questions match the target difficulty.

In recent years, with advancements in natural language processing, several studies in DCQG have recognized the importance of question diversity (Bi et al., 2021) and the alignment of question difficulty levels with learners' abilities (Srivastava and Goodman, 2021; Uto et al., 2023). However, current research still faces a number of challenges:

Context:
A man has a bird. ... Every day the man speaks to the bird. ... "What are you doing?" says the man. "What are you doing?" says the bird. The man is not at home one day. A thief comes in. ... "What are you doing?" The thief is very afraid, so he does not take any things and runs out of the house at last.
Target Difficulty Level: *Hard*

Prompt-based QG
How does the thief react when he hears the bird? (*Too Easy* ✗)

ICL-based QG
What is the purpose of dreaming during sleep? (*Irrelevant* ✗)

Self-refine QG
What does the man's reaction to the thief's presence reveal about his character and values? (*Unanswerable* ✗)

CrossQG
Q1: How does the man's daily interaction with the bird impact the bird's behavior towards intruders? ✓
Q2: What message do you think the story is trying to convey about the relationship between humans and animals? ✓

Table 1: An example from RACE in which questions (corresponding answers are omitted) generated by CrossQG achieve better difficulty consistency compared to other methods.

Lacking Reasonable Difficulty Estimation. On the one hand, most automatic difficulty estimation methods (Gao et al., 2019; Lin et al., 2019; Bi et al., 2021) rely on model accuracy or designed features, lacking support from educational theories (Krathwohl, 2002). On the other hand, human evaluation is relatively precise, yet subjective and time-consuming. Although the quality of generated questions has gradually improved, the definition of question difficulty has not remained consistent yet. As a result, it is crucial to find a method for difficulty estimation that is both reasonable and fast.

Low Difficulty Consistency. Difficulty consistency in DCQG measures how well the generated questions match the target difficulty level, reflecting the model's accuracy in controlling question difficulty. Most studies have primarily used difficulty consistency for evaluation, without proposing specific methods for enhancement. Moreover, most DCQG methods are designed for small language models, with limited research on difficulty consistency for large language models (LLMs). As illustrated in Table 1, the prompt-based method struggles to ensure that the generated questions match the target difficulty level, while in-context learning (ICL) and self-refine methods tend to generate irrelevant or unanswerable questions.

To address these limitations, we propose CrossQG, an LLM-based DCQG method that enhances the difficulty consistency of generated questions. We measure a question's difficulty from two aspects: answer acquisition difficulty and cognitive level. Our difficulty estimation schemata refer to the expert-annotated labels in FairytaleQA (Xu et al., 2022) and Bloom's Taxonomy (Krathwohl, 2002). For question generation, our method consists of two steps: contrast enhancement and cross filtering. During contrast enhancement, unlike typical in-context learning, we incorporate additional negative examples into the prompt to guide LLMs to regenerate distinct questions. This idea is inspired by the self-refine method (Madaan et al., 2023), which allows LLMs to reflect on the questions they generate for the target difficulty. To further improve the method, we choose to use questions of other difficulties as negative examples, exposing LLMs to the information contained in questions of various difficulties. During cross filtering, questions from different difficulty levels with high semantic similarity are removed. In most cases, the difficulty of similar questions tends to be consistent. If an LLM inconsistently rates the difficulty levels of the similar questions, it probably signifies an error in the LLM's judgment. As a result, we filter out these questions.

We conduct experiments on three question answering datasets. Experimental results indicate that CrossQG significantly outperforms prompt-based, ICL-based and self-refine methods in question difficulty consistency and quality on multiple LLMs. Moreover, our training-free method even surpasses supervised fine-tuning (SFT) in various instances.

The main contributions of this paper are summarized as follows:

- We propose CrossQG, a novel training-free LLM-based method for DCQG, which improves generated questions in terms of difficulty consistency.
- We innovatively leverage information from difficulty levels other than the target difficulty in DCQG, enhancing LLMs' understanding of target difficulty.
- We conduct extensive experiments on several datasets. Results show that CrossQG remarkably outperforms multiple training-free methods in both question difficulty consistency and quality on various LLMs.

2 Related Work

2.1 Question Difficulty Estimation

Estimating question difficulty involves quantifying the difficulty of questions, which is a crucial task in the educational field (Benedetto et al., 2023). Existing methods are mainly divided into two categories: automatic evaluation and human evaluation. Automatic evaluation methods often rely on model accuracy or designed features, lacking enough educational theoretical support. For example, DLPH-GDC (Gao et al., 2019) assesses question difficulty based on R-Net (Wang et al., 2017) and BiDAF (Seo et al., 2017). Additionally, several methods incorporate some rules or features to assist difficulty estimation. PCQG (Huang et al., 2018) proposes simple and convenient evaluation rules for questions of different categories. CCQG (Bi et al., 2021) designs five domain-independent features to measure question complexity. Compared to automatic evaluation, human evaluation is based on a more scientific rule or an existing educational theory. For instance, multi-hop QG (Cheng et al., 2021) defines difficulty as the number of inference steps required to answer a question. SkillQG (Wang et al., 2023) characterizes question difficulty from a cognitive perspective based on Bloom's taxonomy. However, human evaluation is usually subjective and time-consuming. In this paper, we train two classifiers based on expert-annotated labels in FairytaleQA and Bloom's taxonomy to assess question difficulty. Our approach is both scientific and efficient, and is suitable for difficulty estimation.

2.2 Difficulty-Controllable Question Generation

Early research on DCQG mainly focus on small language models. DLPH-GDC (Gao et al., 2019) proposes an LSTM-based model to generate questions of designated difficulty levels. Multi-hop QG (Cheng et al., 2021) introduces an iterative framework that gradually increases question difficulty through step-by-step rewriting. This method is guided by an extracted reasoning chain, and uses GPT-2 (Radford et al., 2019) for question generation. CCQG (Bi et al., 2021) uses a mixture of experts (Shen et al., 2019) as the selector of soft templates. It then leverages BiLSTM (Hochreiter and Schmidhuber, 1997) as encoder to generate questions with controlled complexity. SkillQG (Wang et al., 2023) proposes a question generation pipeline. The pipeline utilizes the skill-specific knowledge extracted by GPT-2 to generate questions with BART (Lewis et al., 2020). Recently, PFQS (Li and Zhang, 2024) proposes an LLM-guided method, which generates questions based on answer plans. However, almost all methods generate questions separately for different difficulty levels, overlooking the information implied by questions from other difficulty levels. Additionally, there is a lack of sufficient exploration of LLMs for DCQG. In this paper, we primarily focus on LLM-based methods for DCQG. When generating questions of a specific difficulty level, we leverage the questions generated at different levels to improve the performance of LLMs.

3 Method

Given an input context text c and a specific difficulty level $d \in \mathcal{D}$, the objective of the task is to generate several question-answer (QA) pairs $(\mathcal{Q}, \mathcal{A})$, where questions in $\mathcal{Q}$ align with the difficulty level d. This process can be formalized as the following function:

$$(\mathcal{Q}, \mathcal{A}) = \mathcal{F}(c, d) \tag{1}$$

where $\mathcal{F}$ is a question generation method.

Figure 1 illustrates the overall architecture of our method. Before generation, we introduce the two difficulty estimation schemata used in our approach. During initial question generation, based on the schemata, we use LLMs to generate initial QA pairs $(\mathcal{Q}^{\text{init}}, \mathcal{A}^{\text{init}})$ with prompts tailored for different difficulty levels. During contrast enhancement, given difficulty level d, we select QA pairs from $(\mathcal{Q}^{\text{init}} \setminus \mathcal{Q}^{\text{init}}_d, \mathcal{A}^{\text{init}} \setminus \mathcal{A}^{\text{init}}_d)$ as negative examples to help LLMs avoid generating questions with non-target difficulty levels. During cross filtering, we remove questions from different difficulty levels with high semantic similarity. The following paragraphs will introduce the entire generation process in detail.

3.1 Difficulty Estimation Schemata

We estimate a question's difficulty from two aspects: answer acquisition difficulty and cognitive level. For answer acquisition difficulty, we refer to the labels annotated by experts in FairytaleQA, a well-structured question answering dataset derived from child-friendly storybooks. These labels have been proven to be scientific and reasonable by several previous works (Eo et al., 2023; Leite and Cardoso, 2024; Li and Zhang, 2024). For cognitive level, we refer to the 5-dimensional skill schema in SkillQG, which is drawn on Bloom's Taxonomy. As shown in Table 2, we define 3 difficulty levels for answer acquisition and 5 cognitive levels, with difficulty ranging from low to high. The alignment of our labels with those in FairytaleQA ("local/summary", "explicit/implicit") is as follows: (1) easy aligns with (local, explicit); (2) medium maps to both (local, implicit) and (summary, explicit); and (3) hard corresponds to (summary, implicit). To maintain a clear order of difficulty levels, we include two cases under the medium difficulty. Separating these two cases into different difficulty levels would raise ambiguity about which level is more challenging.

3.2 Initial Question Generation

During the initial question generation process, we use prompts to guide an instruction-tuned LLM in generating QA pairs based on the given context and difficulty level. Considering efficiency, we propose two methods at this stage: CrossQG and CrossQG-fast. CrossQG generates questions for varying difficulty levels with different prompts. By comparison, CrossQG-fast employs a single prompt to simultaneously generate questions across all difficulty levels within a difficulty estimation schema, improving the efficiency of generation. The detailed design of prompts can be found in Appendix A.1.

Given the context c, the difficulty level d, and the prompt template T^{init}, we obtain the complete prompt $T^{\text{init}}(c, d)$[1]. The initial QA pairs of difficulty level d can then be generated using the following expression:

$$(\mathcal{Q}^{\text{init}}_d, \mathcal{A}^{\text{init}}_d) = \text{LLM}(T^{\text{init}}(c, d)) \tag{2}$$

[1] CrossQG-fast does not require a specific difficulty level d; this formula is used here for unified expression.

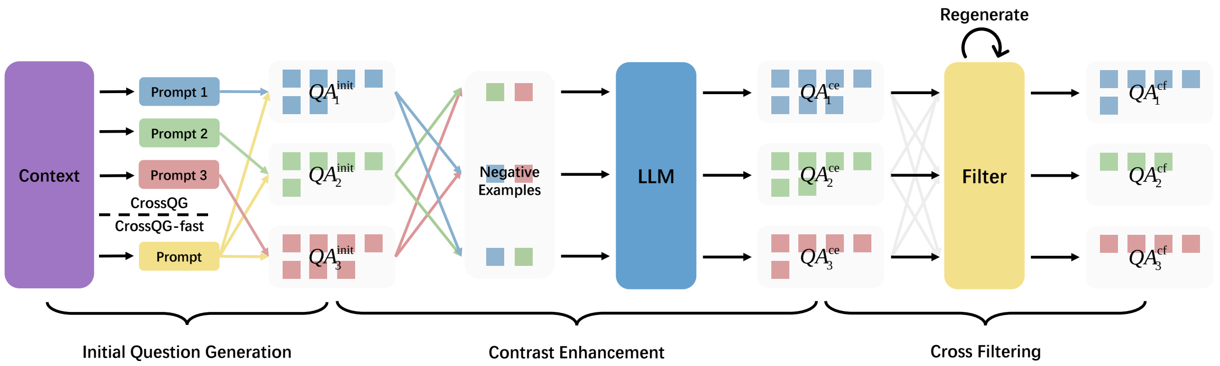


Figure 1: Overall architecture of CrossQG. The figure illustrates the optimization process with three difficulty levels. In the figure, subscript numbers indicate the difficulty of the questions, and each small square represents a QA pair.

Difficulty	Definition
Easy	Answers can be directly found in the text; getting the answer requires focusing on the local information (e.g. one single sentence) in the context.
Medium	Case 1: Answers cannot be directly found in the text; getting the answer requires focusing on the local information (e.g. one single sentence) in the context. Case 2: Answers can be found directly in the text; obtaining the answer involves synthesizing and summarizing information from multiple parts of the context.
Hard	Answers cannot be directly found in the text; obtaining the answer involves synthesizing and summarizing information from multiple parts of the context.
Remember	Questions involve directly retrieving facts from input context without any modification or analysis, the facts could be places, times, quantities, etc.
Understand	Questions involve constructing meanings from recalled facts.
Analyze	Questions involve drawing connections among facts or ideas, the connections could be causal relationship, etc.
Evaluate	Questions involve making clear judgments on something related to humans especially the author, such as feeling, intention, attitude, etc.
Create	Questions involve predicting something not clearly mentioned in the given context, in a future tense or uncertain tone.

Table 2: Definitions of difficulty levels based on answer acquisition difficulty (up) and cognitive level (below).

where $\text{LLM}(\cdot)$ represents performing an inference by LLMs.

3.3 Contrast Enhancement

After the initial process, the generated questions might not meet the expected difficulty levels, indicating that LLMs do not fully understand the difficulty requirements. To tackle this problem, we propose a component called contrast enhancement (CE), which leverages negative examples to enhance LLMs' understanding of target difficulty.

Let $\mathcal{D} \in \{\mathcal{D}_1, \mathcal{D}_2\}$ denotes difficulty levels in one difficulty estimation schema ($\mathcal{D}_1$ for answer acquisition difficulty and $\mathcal{D}_2$ for cognitive level). To enhance LLMs' understanding of difficulty level $d (d \in \mathcal{D})$, we randomly select several QA pairs of other difficulty levels to form negative examples E_d, which can be expressed as follows:

$$E_d = \{f((\mathcal{Q}_{d'}^{\text{init}}, \mathcal{A}_{d'}^{\text{init}}), n) | d' \in \mathcal{D} \backslash \{d\}\} \qquad (3)$$

where $f(\cdot, \cdot)$ represents a uniform sampling function, and n is a hyperparameter that denotes the number of questions randomly selected from each difficulty level.

We explicitly design prompts to enable LLMs to regenerate questions based on previous difficulty requirements and negative examples. The details of the prompts are available in Appendix A.2.

Formally, with the context c, difficulty level d, negative examples E_d, and the prompt template T^{ce}, the whole prompt is $T^{\text{ce}}(c, d, E_d)$. The QA pairs regenerated during the contrast enhancement process can be expressed as follows:

$$(\mathcal{Q}_d^{\text{ce}}, \mathcal{A}_d^{\text{ce}}) = \text{LLM}(T^{\text{ce}}(c, d, E_d)) \qquad (4)$$

where $\text{LLM}(\cdot)$ denotes performing an inference by LLMs.

3.4 Cross Filtering

In this section, we propose a component based on semantic similarity, called cross filtering (CF). The component is designed to filter out questions that may not match the target difficulty levels, further improving the difficulty consistency. Similar

Algorithm 1 Cross Filtering Algorithm

Input: List of list of QA pairs P, list of difficulty levels D, context c

Parameter: Semantic similarity threshold t, threshold for the number of filtered questions m

Output: Filtered QA pairs P'

```
Initialize $S \leftarrow P$, $P' \leftarrow \emptyset$
for all $(p, q)$ where $p \in P_i, q \in P_j, i \neq j$ do
    if $\mathrm{sim}(p, q) > t$ then
        Remove $p$ from $P_i$, remove $q$ from $P_j$.
    end if
end for
for all $P_i \in P$ do
    if $\#P_i < m$ then
        $E \leftarrow S_i \setminus P_i$        ▷ Filtered-out QA pairs
        $P' \leftarrow P' \cup \mathrm{CE}(c, d_i, E)$  ▷ Regeneration
    else
        $P' \leftarrow P' \cup P_i$
    end if
end for
return $P'$
```

questions generally have a consistent level of difficulty. Therefore, if an LLM generates semantically similar questions for different difficulty levels, we suppose it indicates a potential error in the model's evaluation of question difficulty. If this occurs, all involved questions will be removed.

The algorithm of a single round is described in Algorithm 1. In the algorithm, P and D can be formalized as $\{P_i\}_{i=1}^{n_d}$ and $\{d_i\}_{i=1}^{n_d}$, respectively. P_i represents the list of QA pairs generated by the model based on the difficulty d_i, and n_d is the number of different difficulty levels. In addition, $\#P$ denotes the number of QA pairs in P, while E is a list that stores the QA pairs remaining after filtering. In terms of functions, $\mathrm{sim}(\cdot, \cdot)$ is a semantic similarity function (for which we use cosine similarity) and $\mathrm{CE}(\cdot)$ is the function described in Equation 4.

The algorithm consists of two main steps. Firstly, we calculate the semantic similarity between questions of different difficulty levels. If the similarity exceeds a threshold t, we remove those questions. Note that we only calculate the similarity between questions rather than QA pairs. This is because the answers generated by LLMs can vary even for similar questions, and their lengths can be quite long. Both factors can interfere with the calculation of question similarity.

Secondly, we assess whether a next iteration is necessary. We denote the remaining number of QA pairs for difficulty level d_i as n_i. If $n_i \geq m$, the filtered QA pairs for d_i are obtained. Otherwise, we regenerate QA pairs for d_i by contrast enhancement. Note that the regenerated QA pairs need to undergo the next cross-filtering process. In contrast, QA pairs of other difficulties will not be updated in the next filtering phase. They will only be used to calculate similarity with questions that require filtering. The entire process will be repeated until no further QA pairs are regenerated, or it has been repeated three times.

4 Experiments

4.1 Experimental Setup

Datasets. We conduct assessments on three question answering datasets: SQuAD (Rajpurkar et al., 2016), RACE (Lai et al., 2017), and FairytaleQA (Xu et al., 2022). These datasets, sourced from Wikipedia, English exams, and stories respectively, are well-known for their quality. We select a random sample of 1,500 articles for testing, with 500 articles from each dataset.

Metrics. To evaluate the consistency of question difficulty, we first estimate the actual difficulty of the questions, and then calculate the consistency score between the actual and target difficulties.

For difficulty estimation, we initially attempt to utilize GPT-4 (Achiam et al., 2023) in a zero-shot or few-shot manner. However, the average accuracy is below 65% (see complete results in Appendix B.1). Therefore, we train a difficulty classifier based on a manually annotated dataset.

We first construct a dataset called DiffQA. Each entry in DiffQA is a quintuple (c, q, a, d_1, d_2), where c, q, a, d_1, d_2 represents the context, the question, the answer, the answer acquisition difficulty, and the cognitive level respectively. $d_1 \in \{1, 2, 3\}$ denotes {easy, medium, hard} and $d_2 \in \{1, 2, 3, 4, 5\}$ corresponds to {remember, understand, analyze, evaluate, create}. The QA pairs are derived from the above three datasets and human annotation. Considering the relatively lower difficulty of SQuAD, we compose some more challenging questions based on its contexts. In addition, we revise some questions from RACE due to their inconsistent format compared to the other two datasets. We employ five annotators to label d_1 and d_2, with each label undergoing a cross-check by two annotators. In total, we annotate 6500 entries, with 5500 serving as the training set and 1000 as

Method	**SQuAD**		**RACE**		**FairytaleQA**	
	ρ	κ	ρ	κ	ρ	κ
Llama-2-7b-chat						
Prompting	0.3799/0.6846	0.2001/0.3531	0.3252/0.5845	0.1324/0.3112	0.3672/0.6505	0.1519/0.3637
ICL	0.4189/0.6779	0.2589/0.5243	0.4117/0.5864	0.2513/0.4309	0.3385/0.6643	0.2090/**0.4960**
Self-refine	0.4464/0.7925	0.2287/0.5084	0.3619/0.7363	0.1787/0.4469	0.3011/0.7042	0.1273/0.4214
SFT (*)	0.5923/0.8115	0.3732/0.7202	0.6147/0.6938	0.3784/0.6083	0.5046/0.9653	0.3053/0.8444
CrossQG (ours)	**0.6057/0.8505**	0.3509/**0.6080**	**0.6525/0.7901**	**0.4120/0.5030**	**0.5608/0.7433**	**0.3487**/0.4562
CrossQG-fast (ours)	0.6023/0.8374	**0.3737**/0.5971	0.6316/0.7824	0.3802/0.4995	0.5515/0.7354	0.3260/0.4542
Llama-2-13b-chat						
Prompt	0.3900/0.5955	0.1506/0.3100	0.3704/0.5105	0.1566/0.2664	0.4193/0.6196	0.1633/0.3295
ICL	0.3512/0.6526	0.2438/0.5079	0.2928/0.5496	0.1769/0.3833	0.2542/0.6624	0.1585/0.4880
Self-refine	0.6211/0.8259	0.3564/0.5954	0.5954/0.7303	0.3481/0.4474	0.5927/0.6929	0.3724/0.3873
SFT (*)	0.6325/0.8106	0.4203/0.7167	0.5912/0.6578	0.3801/0.5667	0.5270/0.9638	0.3296/0.8583
CrossQG (ours)	**0.7036/0.8832**	**0.4687**/0.6728	**0.7155/0.7959**	**0.4627/0.5392**	**0.6458/0.7574**	**0.4523/0.4913**
CrossQG-fast (ours)	0.6829/0.8764	0.4606/**0.6749**	0.6926/0.7822	0.4400/0.5284	0.6175/0.7363	0.4139/0.4807
Mistral-7b-instruct						
Prompt	0.3946/0.5941	0.0954/0.2693	0.4204/0.5494	0.1457/0.2532	0.3482/0.5749	0.1199/0.2843
ICL	0.4396/0.5748	0.1905/0.3124	0.4592/0.5120	0.2443/0.2695	0.3910/0.6199	0.1885/0.3370
Self-refine	0.4365/0.7045	0.1342/0.3648	0.4491/0.6480	0.1868/0.3496	0.3790/0.6588	0.1349/0.3861
SFT (*)	0.6668/0.8979	0.4553/0.8591	0.6436/0.7252	0.4508/0.6779	0.5129/0.9700	0.3198/0.8653
CrossQG (ours)	**0.6063/0.7739**	**0.2772**/0.4617	**0.5779/0.6954**	0.3088/0.3981	**0.4257**/0.6912	**0.2316**/0.4288
CrossQG-fast (ours)	0.5661/0.7364	0.2617/**0.4893**	0.5719/0.6730	**0.3183/0.3995**	0.4104/**0.7073**	0.2230/**0.4601**

Table 3: Main experimental results on three datasets, with ρ and κ for each method. Best results are highlighted in bold. Unlike other methods, SFT (*) requires model training, so its results don't directly participate in the comparison. If other methods outperform it, those results are marked in red. The results for answer acquisition difficulty and cognitive level are displayed on the left and right side of "/", respectively.

the test set. The details of DiffQA are shown in Appendix B.2.

Then, using DiffQA, we train two classifiers based on Roberta-base (Liu, 2019) for answer acquisition difficulty (d_1) and cognitive level (d_2) respectively, achieving an average accuracy over 84% (see complete results in Appendix B.1).

For consistency score calculation, we collect QA pairs generated by a model for all contexts and target difficulties, denoted as $\mathcal{P}$. Let $d_t(\cdot)$ and $d_a(\cdot)$ represent the target difficulty and the actual difficulty (predicted by trained classifiers) of a QA pair, respectively. We employ Spearman correlation coefficient (Spearman, 1904) ρ and Cohen's kappa coefficient (Cohen, 1960) κ to assess the correlation and consistency between $\mathcal{D}_t = \{d_t(p)|p \in \mathcal{P}\}$ and $\mathcal{D}_a = \{d_a(p)|p \in \mathcal{P}\}$ (see detailed calculations in Appendix B.3).

Baselines. In our experiments, we compare our CrossQG method with four baselines: (1) Prompt (Wei et al., 2022), which leverages the prompt derived from the initial question generation phase of standard CrossQG to generate questions. (2) ICL (Brown et al., 2020), which incorporates several in-context examples with target difficulty into the prompt to guide LLMs in generating questions. (3) Self-refine (Madaan et al., 2023), which lets LLMs reflect on the questions they generate based on the Prompt method and regenerate. (4) SFT, which finetunes LLMs using DiffQA.

Other Settings. We conduct experiments on three LLMs: Llama2-7b-chat, Llama2-13b-chat (Touvron et al., 2023), and Mistral-7b-instruct-v0.2 (Jiang et al., 2023). Additionally, two estimation schemata (for answer acquisition difficulty and cognitive level) are applied for difficulty evaluation. Regarding hyperparameter settings, the default values for n in the contrast enhancement, and t, m in the cross filtering, are 1, 0.8, and 2, respectively. To be consistent with CrossQG, ICL and self-refine methods use 2 and 4 examples for answer acquisition difficulty and cognitive level respectively.

4.2 Main Results

The results of the baselines and CrossQG are presented in Table 3. Overall, CrossQG significantly outperforms other training-free baselines in almost all settings, and even surpasses SFT in many instances (marked in red). Specifically, compared with three training-free baselines, CrossQG shows an average increase of at least 0.10 in the ρ metric and at least 0.11 in the κ metric, respectively.

Method	SQuAD		RACE		FairytaleQA	
	ρ	κ	ρ	κ	ρ	κ
			Llama-2-7b-chat			
CrossQG	**0.6057/0.8505**	**0.3509/0.6080**	**0.6525/0.7901**	**0.4120/0.5030**	**0.5608/0.7433**	**0.3487/0.4562**
w/o CE	0.5023/0.7871	0.3216/0.4867	0.4405/0.6970	0.2459/0.4248	0.4378/0.7190	0.2594/0.4321
w/o CF	0.5524/0.8110	0.2976/0.5568	0.6064/0.7428	0.3670/0.4624	0.5359/0.6946	0.3259/0.4220
			Llama-2-13b-chat			
CrossQG	**0.7036/0.8832**	**0.4687/0.6728**	0.7155/**0.7959**	**0.4627/0.5392**	0.6458/**0.7574**	0.4523/**0.4913**
w/o CE	0.5874/0.7690	0.3573/0.5188	0.5620/0.7014	0.3430/0.4578	0.5194/0.7167	0.3325/0.4337
w/o CF	0.6998/0.8583	0.4518/0.6395	**0.7165**/0.7708	0.4606/0.5078	**0.6516**/0.7192	**0.4546**/0.4486
			Mistral-7b-instruct			
CrossQG	**0.6063/0.7739**	**0.2772/0.4617**	**0.5779/0.6954**	**0.3088/0.3981**	**0.4257/0.6912**	**0.2316/0.4288**
w/o CE	0.5485/0.7177	0.2424/0.4173	0.5258/0.6296	0.2482/0.3404	0.4033/0.6322	0.2075/0.3668
w/o CF	0.5590/0.7447	0.2054/0.4154	0.5368/0.6666	0.2672/0.3733	0.3983/0.6525	0.1929/0.3970

Table 4: Ablation results for CrossQG, where *CE* denotes contrast enhancement and *CF* denotes cross filtering. We report ρ and κ for each method. The results for answer acquisition difficulty and cognitive level are displayed on the left and right side of "/", respectively.

Moreover, compared to CrossQG, the overall performance of CrossQG-fast shows a slight decline, but it exhibits improvements in a few cases. However, in terms of efficiency, CrossQG-fast is superior to CrossQG. Specifically, CrossQG requires at least 2 LLM inferences to generate questions for each difficulty level. By contrast, CrossQG-fast only needs $1 + 1/n_d$ inferences, where n_d is the number of difficulty levels. In other words, CrossQG-fast significantly enhances inference efficiency while only undergoing a slight decrease in performance. In addition, CrossQG outperforms SFT in many cases for answer acquisition difficulty. This suggests that the finetuned models sometimes struggle to learn effectively when it comes to deeper semantic features of questions. Nevertheless, CrossQG still exhibits strong performance in this case.

4.3 Ablation Study

In Table 4, we investigate the impact of contrast enhancement (CE) and cross filtering (CF) on our method. The results indicate that combining CE and CF usually yields the best outcomes. Comparing the results lacking CE and CF, it is evident that CE is more crucial for the performance of LLMs. The absence of CE leads to a significant decline in LLM performance, while CF has a comparatively minor impact on it.

Additionally, the inclusion of CE consistently enhances LLM performance, providing stable improvements in the ρ and κ metrics across different LLMs and difficulty estimation schemata. By contrast, the enhancements brought by CF are less stable. For instance, in the context of the Llama-2-13b-chat model, when considering answer acquisition difficulty, CF has negligible impact on model performance. This is because CF involves the calculation of semantic similarity, which does not strongly correlate with difficulty. Semantically similar questions sometimes vary in difficulty, potentially leading to ineffective filtering.

Method	ρ	κ	acc
	Llama-2-7b-chat		
Self-refine	0.369/0.741	0.177/0.458	0.673/0.529
CE	0.566/0.746	0.332/0.479	0.768/0.866
	Llama-2-13b-chat		
Self-refine	0.603/0.746	0.359/0.477	0.663/0.545
CE	0.686/0.783	0.456/0.533	0.770/0.856
	Mistral-7b-instruct		
Self-refine	0.418/0.670	0.155/0.367	0.681/0.574
CE	0.495/0.691	0.225/0.396	0.761/0.854

Table 5: Results integrating three datasets, where *acc* denotes the accuracy of negative examples. The results for answer acquisition difficulty and cognitive level are displayed on the left and right side of "/", respectively.

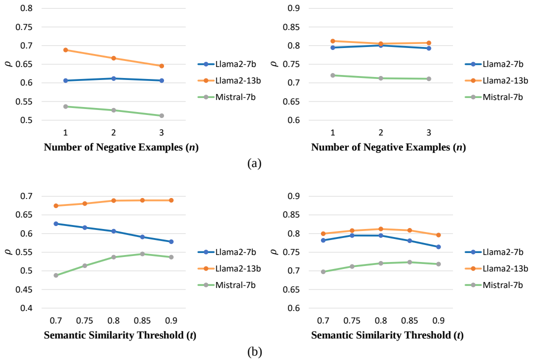
To further verify the quality of negative examples selected by the CE component, we compare it with the self-refine method. Experimental results are shown in Table 5. For the convenience of explanation, we refer to the negative examples used in CE and the questions used for reflection in self-refine collectively as negative examples. For fairness, the negative examples used in CE and self-refine are all sourced from questions generated by the Prompt method, and are consistent in quantity. Note that the difficulty of questions in negative examples may match the target level, making them not true negatives. Utilizing a classifier trained with DiffQA, we assess the accuracy of negative examples in both methods to measure its correlation with model performance. It is evident that CE's negative examples show significantly better accuracy than those from self-refine. Furthermore, when it comes to answer acquisition difficulty, model performance is remarkably influenced by negative example accuracy. Given that the answer acquisition difficulty generally involves deeper semantic features of the questions, we infer that LLMs require examples of higher quality in this case.

4.4 Analysis

Impact of Negative Examples Count. In contrast enhancement, we randomly select n QA pairs for each other difficulty level to form negative examples. After increasing the number of negative examples, it is likely that LLMs gain a better understanding of the target difficulty, but it also introduces additional noise. We study the impact of different values of n on model performance, and the results are shown in Figure 2(a). It is observable that n affects the performance of different models differently. For Llama-2-13b-chat, the model performs better at $n = 2$, but the overall difference is not significant. However, for the other two models, performance deteriorates as n increases. Therefore, $n = 1$ is a better parameter choice.

Impact of Semantic Similarity Threshold. In cross filtering, the semantic similarity threshold t reflects the strictness of the filtering process. We investigate the impact of different values of t on model performance, and the results are displayed in Figure 2(b). Overall, at $t = 0.8$, all three models exhibit competitive performance, thus we select 0.8 as the value for parameter t. Additionally, model performance is more sensitive to t when it comes to answer acquisition difficulty.

Human Evaluation for Question Quality. In the question quality study, we first randomly select 150 articles from three datasets. Then, we perform uniform sampling on QA pairs of various difficulty levels generated by different models for human evaluation. Three dimensions are rated from 1 (worst) to 5 (best): (1) correctness—whether the question and answer match and are semantically correct; (2) relevancy—whether the question is relevant to the given context; (3) diversity—whether the QA pairs are diverse from each other. As shown in Table 6, CrossQG outperforms the Prompt and ICL methods across three quality metrics. Additionally, it also surpasses the self-refine method in terms of the relevance and diversity of the generated questions. This indicates that our method not only enhances the consistency of question difficulty but also improves the quality of questions.



(a)

(b)

Figure 2: The ρ performance of models under different numbers of negative examples (a) and semantic similarity thresholds (b). Results for answer acquisition difficulty and cognitive level are shown on the left and right side, respectively. The results integrates three datasets.

Method	Correctness	Relevancy	Diversity
Prompt	4.284	4.379	2.025
ICL	3.794	3.619	2.071
Self-refine	**4.602**	4.428	2.428
CrossQG (ours)	4.546	**4.451**	**2.645**

Table 6: Human evaluation on correctness, relevancy and diversity.

5 Conclusion

In this paper, we propose CrossQG, a novel training-free DCQG method aimed at optimizing the difficulty consistency of generated questions. CrossQG leverages information from various difficulty levels, which is often overlooked in previous research, to assist in generating questions at the target difficulty. It first employs contrast enhancement to select questions from different difficulty levels as negative examples. Then, it utilizes cross filtering to filter out semantically similar questions of varying difficulties. We conduct experiments on three datasets. Results across multiple LLMs demonstrate that CrossQG achieves superior difficulty consistency and quality compared to three training-free baselines. In addition, it surpasses SFT in many instances. Future research will explore methods to enhance the robustness of CrossQG against noisy data.

Limitations

The cross filtering component used in CrossQG filter out questions based on semantic similarity. However, semantic similarity does not fully correlate with difficulty, resulting in limited improvements from this component. In future work, we will attempt to design a filtering component targeted at question difficulty. Additionally, CrossQG is a training-free method. In the future, we will explore Parameter-Efficient Fine-Tuning (PEFT) methods to further enhance model performance while ensuring efficiency.

Ethics Statement

In this paper, we propose a novel DCQG method aimed at enhancing the difficulty consistency of questions generated by large language models. The three question answering datasets and all base models are publicly available. In addition, all references derived from prior works are marked with citations. During the experiments, random seeds are selected entirely at random and maintained consistently across different model configurations. In this way, we minimize bias and discrimination in our experiments. Lastly, the QA pairs are generated based on the text in datasets and do not include any harmful content. Overall, we avoid any ethical concerns in our research.

We employ 5 annotators with undergraduate degrees to perform annotations. We pay $12 USD per 100 annotations, which includes both question difficulty and quality estimation. To ensure the anonymity and privacy of the annotators, we exclude all personal identifiers and retain only the annotation results.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Luca Benedetto, Paolo Cremonesi, Andrew Caines, Paula Buttery, Andrea Cappelli, Andrea Giussani, and Roberto Turrin. 2023. A survey on recent approaches to question difficulty estimation from text. *ACM Computing Surveys*, 55(9):1–37.

Sheng Bi, Xiya Cheng, Yuan-Fang Li, Lizhen Qu, Shirong Shen, Guilin Qi, Lu Pan, and Yinlin Jiang. 2021. Simple or complex? complexity-controllable question generation with soft templates and deep mixture of experts model. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4645–4654, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

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.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Sugyeong Eo, Hyeonseok Moon, Jinsung Kim, Yuna Hur, Jeongwook Kim, SongEun Lee, Changwoo Chun, Sungsoo Park, and Heuiseok Lim. 2023. Towards diverse and effective question-answer pair generation from children storybooks. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6100–6115, Toronto, Canada. Association for Computational Linguistics.

Yifan Gao, Lidong Bing, Wang Chen, Michael R Lyu, and Irwin King. 2019. Difficulty controllable generation of reading comprehension questions. In *IJCAI*, pages 4968–4974.

S Hochreiter and J Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Yi-Ting Huang, Meng Chang Chen, and Yeali S Sun. 2018. Development and evaluation of a personalized computer-aided question generation for english learners to improve proficiency and correct mistakes. *arXiv preprint arXiv:1808.09732*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

David R Krathwohl. 2002. A revision of bloom's taxonomy: An overview. *Theory into practice*, 41(4):212–218.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Bernardo Leite and Henrique Lopes Cardoso. 2024. On few-shot prompting for controllable question-answer generation in narrative comprehension. In *Proceedings of the 16th International Conference on Computer Supported Education, CSEDU 2024, Angers, France, May 2-4, 2024, Volume 2*, pages 63–74. SCITEPRESS.

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, Online. Association for Computational Linguistics.

Kunze Li and Yu Zhang. 2024. Planning first, question second: An LLM-guided method for controllable question generation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4715–4729, Bangkok, Thailand. Association for Computational Linguistics.

Li-Huai Lin, Tao-Hsing Chang, and Fu-Yuan Hsu. 2019. Automated prediction of item difficulty in reading comprehension using long short-term memory. In *2019 International Conference on Asian Language Processing (IALP)*, pages 132–135. IEEE.

Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, pages 46534–46594. Curran Associates, Inc.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Tianxiao Shen, Myle Ott, Michael Auli, and Marc'Aurelio Ranzato. 2019. Mixture models for diverse machine translation: Tricks of the trade. In *International conference on machine learning*, pages 5719–5728. PMLR.

C Spearman. 1904. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101.

Megha Srivastava and Noah Goodman. 2021. Question generation for adaptive education. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 692–701, Online. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Masaki Uto, Yuto Tomikawa, and Ayaka Suzuki. 2023. Difficulty-controllable neural question generation for reading comprehension using item response theory. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 119–129, Toronto, Canada. Association for Computational Linguistics.

Tzu-Hua Wang. 2014. Developing an assessment-centered e-learning system for improving student learning effectiveness. *Computers & Education*, 73:189–203.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 189–198, Vancouver, Canada. Association for Computational Linguistics.

Xiaoqiang Wang, Bang Liu, Siliang Tang, and Lingfei Wu. 2023. SkillQG: Learning to generate question for reading comprehension assessment. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13833–13850, Toronto, Canada. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Ying Xu, Dakuo Wang, Mo Yu, Daniel Ritchie, Bingsheng Yao, Tongshuang Wu, Zheng Zhang, Toby Li, Nora Bradford, Branda Sun, Tran Hoang, Yisi Sang, Yufang Hou, Xiaojuan Ma, Diyi Yang, Nanyun Peng, Zhou Yu, and Mark Warschauer. 2022. Fantastic questions and where to find them: FairytaleQA – an authentic dataset for narrative comprehension. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 447–460, Dublin, Ireland. Association for Computational Linguistics.

A Design of Main Prompts

A.1 Initial Question Generation

In this section, we show prompts which are used to guide LLMs to generate initial QA pairs with given difficulty levels. For CrossQG, we design a prompt template as shown in Table 7.

Prompt Template for CrossQG
<s>[INST]
Your task is to generate pairs of questions and answers according to the following context, meeting all the requirements.
Context: [Context]
Requirements:
1. [Difficulty Definition]
2. Answers must be clear, concrete, and well-justified based on the context.
[Supplement]
Output Format:
- Q1: {question}
- A1: {answer}
- Q2: {question}
- A2: {answer}
- Continue as needed...
[/INST]

Table 7: Prompt template used for the initial question generation in CrossQG. In the template, [Context], [Difficulty Definition] and [Supplement] need to be substituted.

In the template, we replace the [Context] token with the given context. In addition, we substitute the [Difficulty Definition] token with the corresponding difficulty definitions shown in Table 2 based on the difficulty level. For easy, medium, and hard levels, we replace the [Supplement] token respectively according to the mapping rules shown in Table 8.

Difficulty	**Supplement**
Easy	Ensure that the answers can be directly and unambiguously located within the text.
Medium	Make sure that each question distinctly follows either Case 1 or Case 2 as defined.
Hard	Ensure that the questions demand a deep understanding and interaction with the context, leading to comprehensive and insightful answers.

Table 8: Mapping rules for difficulty levels in initial question generation.

For CrossQG-fast, we design two prompt templates for answer acquisition difficulty and cognitive level, as shown in Tables 9 and 10 respectively.

A.2 Contrast Enhancement

In this section, we present prompts used to let LLMs generate QA pairs with required difficulty

Prompt Template for CrossQG-fast
<s>[INST]
Your task is to generate pairs of questions and answers according to the following context, meeting all the requirements.
Context: [Context]
Requirements:
1. A question should test any difficulty level of given difficulties, and all generated questions are expected to cover all difficulty levels.
2. Answers must be clear, concrete, and well-justified based on the context.
Difficulties:
- EASY: Answers can be directly found in the text; getting the answer requires focusing on the local information (e.g. one single sentence) in the context.
- MEDIUM: Case 1: Answers cannot be directly found in the text; getting the answer requires focusing on the local information (e.g. one single sentence) in the context. Case 2: Answers can be found directly in the text; obtaining the answer involves synthesizing and summarizing information from multiple parts of the context.
- HARD: Answers cannot be directly found in the text; obtaining the answer involves synthesizing and summarizing information from multiple parts of the context.
Ensure that:
- For easy questions, the answers can be directly and unambiguously located within the text.
- For medium questions, each question distinctly follows either Case 1 or Case 2 as defined.
- For hard questions, the questions demand a deep understanding and interaction with the context, leading to comprehensive and insightful answers.
Output Format:
EASY:
- Q1: {question}
- A1: {answer}
- Q2: {question}
- A2: {answer}
- Continue as needed...
MEDIUM:
- Q1: {question}
- A1: {answer}
- Q2: {question}
- A2: {answer}
- Continue as needed...
HARD:
- Q1: {question}
- A1: {answer}
- Q2: {question}
- A2: {answer}
- Continue as needed...
[/INST]

Table 9: Prompt template for answer acquisition difficulty used in the initial question generation phase of CrossQG-fast. In the template, [Context] needs to be substituted with the given context.

Prompt Template for CrossQG-fast
<s>[INST]
Your task is to generate pairs of questions and answers according to the following context, meeting all the requirements.
Context: [Context]
Requirements:
1. A question should test any ablity of given abilities, and all generated questions are expected to cover all abilities.
2. Answers must be clear, concrete, and well-justified based on the context.
Abilities:
1. REMEMBER, involving directly retrieving facts from input context without any modification or analysis, the facts could be places, times, quantities, etc.
2. UNDERSTAND, involving constructing meanings from recalled facts.
3. ANALYZE, involving drawing connections among facts or ideas, the connections could be causal relationship, contrast relationship, etc.
4. EVALUATE, involving making clear judgments on something related to humans especially the author, such as feeling, intention, attitude, etc.
5. CREATE, involving perdicting something not clearly mentioned in the given context, in a future tense or uncertain tone.
Output Format:
REMEMBER:
- Q1: {question}
- A1: {answer}
- Q2: {question}
- A2: {answer}
- Continue as needed...
UNDERSTAND:
- Q1: {question}
- A1: {answer}
- Q2: {question}
- A2: {answer}
- Continue as needed...
ANALYZE:
- Q1: {question}
- A1: {answer}
- Q2: {question}
- A2: {answer}
- Continue as needed...
EVALUATE:
- Q1: {question}
- A1: {answer}
- Q2: {question}
- A2: {answer}
- Continue as needed...
CREATE:
- Q1: {question}
- A1: {answer}
- Q2: {question}
- A2: {answer}
- Continue as needed...
[/INST]

Table 10: Prompt template for cognitive level used in the initial question generation phase of CrossQG-fast. In the template, [Context] needs to be substituted with the given context.

Prompt Template for Contrast Enhancement
[Initial Prompt]
[Negative Examples]</s>
<s>[INST]
[Error Analysis]
[Supplement]
Output Format:
- Q1: {question}
- A1: {answer}
- Q2: {question}
- A2: {answer}
- Continue as needed...
[/INST]

Table 11: Prompt template for answer acquisition difficulty applied in contrast enhancement. In the template, [Initial Prompt], [Negative Examples], [Error Analysis] and [Supplement] are special tokens and need to be substituted.

Prompt Template for Contrast Enhancement
[Initial Prompt]
[Negative Examples]</s>
<s>[INST]
Some of the above questions you generate cannot meet the requirements, please generate questions different from them.
Ensure that [Difficulty Definition]
Output Format:
- Q1: {question}
- A1: {answer}
- Q2: {question}
- A2: {answer}
- Continue as needed...
[/INST]

Table 12: Prompt template for cognitive level applied in contrast enhancement. In the template, [Initial Prompt], [Negative Examples] and [Difficulty Definition] are special tokens and need to be substituted.

levels given negative examples of other levels. The prompt templates for answer acquisition difficulty and cognitive level are shown in Tables 11 and 12 respectively.

Given the input context c, target difficulty level d, for both templates, we first substitute the [Initial Prompt] token with $T^{\text{init}}(c, d)$ (relative prompt template shown in Table 7). Then, we replace the [Negative Examples] token with E_d computed by Equation 3. For answer acquisition difficulty, we choose the prompt shown in Table 11 and replace the [Error Analysis] and [Supplement] tokens according to the mapping rules shown in Table 13. For cognitive level, we choose the prompt shown in Table 12, and substitute the [Difficulty Definition] token with corresponding difficulty definitions shown in Table 2.

A.3 Others

In this section, we show prompts which are used in the main experiments. The prompt templates used for ICL and SFT are presented in Table 14 and 15 respectively.

In both templates, we replace the [Context], [Difficulty Definition] and [Supplement] tokens according to the substitution rules shown in Appendix A.1. For ICL, we need to replace the [Example i] token with "Example i:\n Context: c_e\n Q: q_e\n A: a_e\n", where c_e, q_e and a_e denote the context, question and answer of the example, respectively. For SFT, LLMs utilize the prompt as input and are fine-tuned to generate one QA pair at a time. Consequently, it is necessary to adjust the descriptions of "question" and "answer" in the prompt to the singular form.

B Details of Metrics

B.1 Experiments of Difficulty Estimation

For difficulty estimation, we compare the accuracy of GPT-4 and our classifiers on the test split of DiffQA, as shown in Table 16. Our classifiers utilize the tuple (c, q, a) as input and are independently trained to predict the answer acquisition difficulty (d_1) and the cognitive level (d_2) of the given questions, respectively. It is evident that our classifiers outperform GPT-4 under both difficulty estimation schemata, with an average accuracy exceeding 84%.

B.2 Details of DiffQA

The proportion distribution of d_1 and d_2 in train split and test split of DiffQA are shown in Figure 3 and 4 respectively.

B.3 Details of Calculation

Consistent with the symbols used in the main text, $\mathcal{P} = \{p_i\}_{i=1}^{N}$ represents the QA pairs generated by the LLM. Let $d_t(\cdot)$ and $d_a(\cdot)$ represent the target difficulty and the actual difficulty (predicted by trained classifiers) of a QA pair, respectively.

Spearman correlation coefficient ρ can be calculated by the following formula:

$$\rho = 1 - \frac{6 \sum_{i=1}^{N} (d_a(p_i) - d_t(p_i))^2}{N(N^2 - 1)} \tag{5}$$

Let $A = [a_{ij}]_{n_d \times n_d}$ be the confusion matrix between the target difficulty levels and the actual difficulty levels, where $a_{ij} = |\{p|p \in \mathcal{P}, d_t(p) =$

Difficulty	Error Analysis	Supplement
Easy	The above questions are too hard to answer. Answers were not adequately justified with direct text references or focused on multiple text areas instead of local information. Please generate easier questions which meet the requirements.	Ensure that the questions only require simple information extraction and superficial understanding with the context, leading to easy and direct answers.
Medium	The above questions are either too easy or too hard to answer. Answers were justified with direct text references (too easy) or focused on multiple text areas and required summarization (too hard). Please generate questions which meet the requirements.	Ensure that the questions require a moderate (not too deep, not too simple) understanding and interaction with the context.
Hard	The above questions are too easy to answer. Answers were justified with direct text references or focused on local information instead of multiple text areas. Please generate questions which meet the requirements.	Ensure that the questions demand a deep understanding and interaction with the context, leading to comprehensive and insightful answers.

Table 13: Mapping rules for difficulty levels in contrast enhancement.

Prompt Template for ICL
<s>[INST] Your task is to generate pairs of questions and answers according to the following context, meeting all the requirements. Context: [Context] Requirements: 1. [Difficulty Definition] 2. Answers must be clear, concrete, and well-justified based on the context. [Supplement] Here are several examples. [Example 1] [Example 2] ... Output Format: - Q1: {question} - A1: {answer} - Q2: {question} - A2: {answer} - Continue as needed... [/INST]

Table 14: Prompt template used for ICL. In the template, [Context], [Difficulty Definition], [Supplement] and [Example i] tokens need to be substituted.

Prompt Template for SFT
<s>[INST] Your task is to generate a pair of question and answer according to the following context, meeting all the requirements. Context: [Context] Requirements: 1. [Difficulty Definition] 2. Answer must be clear, concrete, and well-justified based on the context. [Supplement] Output Format: - Q: {question} - A: {answer} [/INST]

Table 15: Prompt template used for SFT. In the template, [Context], [Difficulty Definition] and [Supplement] tokens need to be substituted.

Model	acc of d_1	acc of d_2
GPT-4 (zero-shot)	0.548	0.472
GPT-4 (few-shot)	0.619	0.527
Roberta (Ours)	**0.817**	**0.873**

Table 16: Accuracy results on the test split of DiffQA. d_1 and d_2 denote answer acquisition difficulty and cognitive level, respectively.

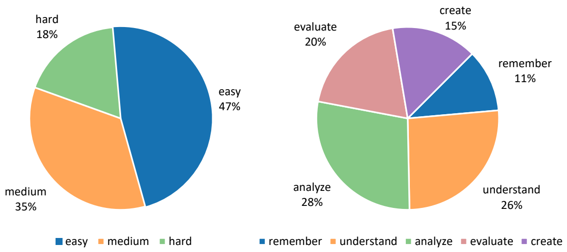


Figure 3: The proportion distribution of d_1 (left) and d_2 (right) in train split of DiffQA.

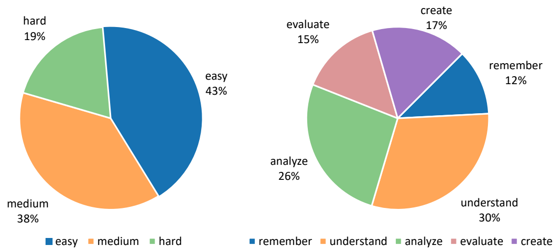


Figure 4: The proportion distribution of d_1 (left) and d_2 (right) in test split of DiffQA.

$i, d_a(p) = j\}|$ ($|\cdot|$ denotes the number of elements in a set), and n_d is the number of difficulty levels. $a_{i+} = \sum_j a_{ij}$, $a_{+j} = \sum_i a_{ij}$.

Cohen's kappa coefficient κ can be expressed as follows:

$$\kappa = \frac{p_o - p_e}{1 - p_e} \tag{6}$$

where

$$p_o = \frac{\sum_{i=1}^{n_d} a_{ii}}{N} \tag{7}$$

$$p_e = \frac{\sum_{i=1}^{n_d} a_{i+} a_{+i}}{N^2} \tag{8}$$

We use the spearmanr function from the SciPy library and the cohen_kappa_score function from the scikit-learn library to calculate ρ and κ, respectively.

C Implementation Details

In cross filtering, we utilize the commonly used sentence-transformer model, all-MiniLM-L6-v2 (Reimers and Gurevych, 2019) to encode questions and compute cosine similarity between them. For difficulty estimation, we fine-tune Roberta with the following parameter settings: learning rate = 1e-5; batch size = 32; and epoch = 5. In the main experiments, the prompt templates used for ICL and SFT are presented in Appendix A.3. The prompt template employed in the self-refine method is identical to that in the CE component, but the approach to selecting negative examples differs. When fine-tuning the LLM, the hyperparameters are as follows: learning rate = 1e-5; batch size per device = 8; and epoch = 3.

Our code is implemented based on Huggingface (Wolf et al., 2020), whereas AdamW (Loshchilov and Hutter, 2019) is used for optimization. All LLMs are loaded and used for inference on 1 Nvidia-A100-40G GPU and trained on 8 Nvidia-A100-40G GPUs. For each configuration of our method and all compared methods, we conduct 5 independent runs and report the average score.

D Case Study

Table 17 illustrates a complete example of question generation using our CrossQG method. In the initial question generation, the difficulty consistency of the generated questions is poor. The questions expected to be of medium difficulty all turn out to be easy. After applying contrast enhancement (CE), the difficulty consistency of the regenerated questions is significantly improved. Specifically, questions targeted at an easy difficulty level consistently match expectations. In addition, generated questions at other difficulties also achieve better difficulty consistency than before. Then, following cross filtering (CF), questions highlighted in yellow are removed due to high similarity. The operation results in an insufficient number of medium difficulty questions, making it necessary for LLM to regenerate them. Finally, it is evident that the final set of generated questions aligns well with the expected difficulty levels.

E List of Software and Data Licences Used in this Work

Main dependencies in this paper are as follows. They are all public and free for research use.

- Huggingface Transformers: https://github.com/huggingface/transformers/blob/master/LICENSE, under an Apache License 2.0.
- Huggingface Datasets: https://github.com/huggingface/datasets/blob/master/LICENSE, under an Apache License 2.0.
- Pytorch: https://github.com/pytorch/pytorch/blob/main/LICENSE, Misc.
- Llama 2: https://github.com/facebookresearch/llama/blob/main/LICENSE, under the LLAMA 2 Community License.
- Mistral: https://github.com/mistralai/mistral-src/blob/main/LICENSE, under an Apache License 2.0.
- SciPy: https://github.com/scipy/scipy, under the BSD-3-Clause license.
- Scikit-learn: https://github.com/scikit-learn/scikit-learn, under the BSD-3-Clause license.

Context: The investors showed little interest in Tesla's ideas for new types of motors and electrical transmission equipment and also seemed to think it was better to develop an electrical utility than invent new systems. They eventually forced Tesla out leaving him penniless. He even lost control of the patents he had generated since he had assigned them to the company in lieu of stock. He had to work at various electrical repair jobs and even as a ditch digger for $2 per day. Tesla considered the winter of 1886/1887 as a time of "terrible headaches and bitter tears." During this time, he questioned the value of his education.		
Initial Question Generation:		
Q1: Why did the investors think it was better to develop an electrical utility than invent new systems? (*Medium*) Q2: What did Tesla consider the winter of 1886/1887 as in relation to his education? (*Easy*) Q3: What happened to Tesla after he was forced out of the company? (*Easy*) Q4: How did Tesla feel about his situation during this time? (*Hard*)	Q1: What was the reason why the investors showed little interest in Tesla's ideas for new types of motors and electrical transmission equipment? (*Easy*) Q2: What did Tesla have to do to survive after he was forced out of the company? (*Easy*) Q3: What was the outcome of Tesla's decision to assign his patents to the company in lieu of stock? (*Easy*) Q4: What did Tesla consider the winter of 1886/1887 as in relation to his education? (*Easy*)	Q1: How did Tesla feel about the value of his education during the winter of 1886/1887? (*Medium*) Q2: What was Tesla's job at the time of the winter of 1886/1887? (*Easy*) Q3: What did Tesla learn from his experiences during the winter of 1886/1887? (*Hard*) Q4: Why did Tesla assign his patents to the company in lieu of stock? (*Medium*)
Negative Examples:		
Q1: What did Tesla have to do to survive after he was forced out of the company? Q2: What was Tesla's job at the time of the winter of 1886/1887?	Q1: Why did the investors think it was better to develop an electrical utility than invent new systems? Q2: What did Tesla learn from his experiences during the winter of 1886/1887?	Q1: What did Tesla consider the winter of 1886/1887 as in relation to his education? Q2: How did Tesla feel about the value of his education during the winter of 1886/1887?
Contrast Enhancement:		
Q1: What did the investors not show interest in? (*Easy*) Q2: Why did Tesla lose control of his patents? (*Easy*) Q3: What was Tesla's job during the winter of 1886/1887? (*Easy*) Q4: How much did Tesla earn from his job as an electrical repairman? (*Easy*)	Q1: What were some of the challenges that Tesla faced in trying to convince investors to fund his ideas for new types of motors and electrical transmission equipment? (*Hard*) Q2: How did Tesla's experiences during the winter of 1886/1887 affect his view of the value of his education? (*Hard*)	Q1: What were the investors' reasons for not showing interest in Tesla's new motor and electrical transmission ideas, and how did this impact Tesla's decision-making? (*Medium*) Q2: How did Tesla's experience working as a ditch digger impact his perspective on the value of his education? (*Medium*) Q3: How did Tesla's relationships with his colleagues and superiors at the company impact his experience and decision-making during this time, and what lessons did he learn from these interactions? (*Hard*)
Cross Filtering:		
Q1: What did the investors not show interest in? (*Easy*) Q2: Why did Tesla lose control of his patents? (*Easy*) Q3: What was Tesla's job during the winter of 1886/1887? (*Easy*) Q4: How much did Tesla earn from his job as an electrical repairman? (*Easy*)	Q1: How did Tesla feel about the situation he was in after being forced out of his company? (*Medium*) Q2: What were some of the personal and financial challenges that Tesla faced in the winter of 1886-1887? (*Medium*)	Q1: How did Tesla's experience working as a ditch digger impact his perspective on the value of his education? (*Medium*) Q2: How did Tesla's relationships with his colleagues and superiors at the company impact his experience and decision-making during this time, and what lessons did he learn from these interactions? (*Hard*)

Table 17: A complete example of question generation using CrossQG, with corresponding answers omitted. In the table, the target difficulty levels of the questions in the three columns from left to right are easy, medium, and hard, respectively. The similarity between the two questions highlighted in yellow is relatively high.