# REGULAR: A Framework for Relation-Guided Multi-Span Question Generation

**Jiayi Lin [1,2], Chenyang Zhang [1,2], Bingxuan Hou [1,2], Dongyu Zhang [1,2]**
**Qingqing Hong [1,2], Junli Wang [1,2*]**
[1] Key Laboratory of Embedded System and Service Computing (Tongji University),
Ministry of Education, Shanghai 201804, China.
[2] National (Province-Ministry Joint) Collaborative Innovation Center
for Financial Network Security, Tongji University, Shanghai 201804, China.
{2331908, inkzhangcy, 2151130, yidu}@tongji.edu.cn
{2332012, 2052643, junliwang}@tongji.edu.cn

## Abstract

To alleviate the high cost of manually annotating Question Answering (QA) datasets, Question Generation (QG) requires the model to generate a question related to the given answer and passage. This work primarily focuses on Multi-Span Question Generation (MSQG), where the generated question corresponds to multiple candidate answers. Existing QG methods may not suit MSQG as they typically overlook the correlation between the candidate answers and generate trivial questions, which limits the quality of the synthetic datasets. Based on the observation that relevant entities typically share the same relationship with the same entity, we propose **REGULAR**, a framework of **RE**lation-**GU**ided Mu**L**ti-Sp**A**n Question Gene**R**ation. REGULAR first converts passages into relation graphs and extracts candidate answers from the relation graphs. Then, REGULAR utilizes a QG model to generate a set of candidate questions and a QA model to obtain the best question. We construct over 100,000 questions using Wikipedia corpora, named REGULAR-WIKI, and conduct experiments to compare our synthetic datasets with other synthetic QA datasets. The experiment results show that models trained with REGULAR-WIKI achieve the best performance. We also conduct ablation studies and statistical analysis to verify the quality of our synthetic dataset. [1]

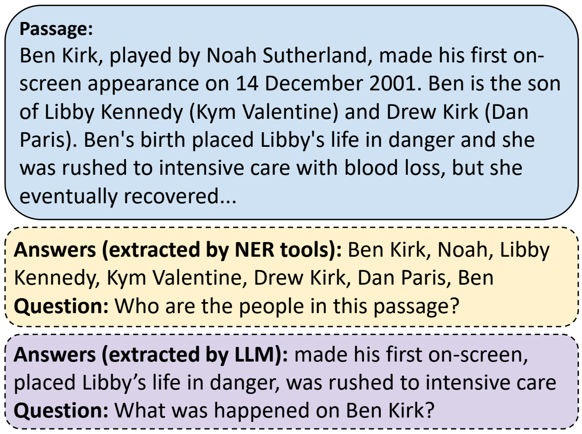


Figure 1: An example where both NER tools and LLM fail to extract reasonable entities as answers, leading to questions that are trivial or irrelevant to the answers.

## 1 Introduction

Question Answering (QA) (Rajpurkar et al., 2018; Kwiatkowski et al., 2019) requires the model to provide answers for a given question, which has wide-ranging applications like chat systems(OpenAI et al., 2024), information retrieval(Esteva et al., 2021), and AI education (Rabin et al., 2023). As a subtype of the QA task, Multi-Span Question Answering (MSQA) (Li et al., 2022; Yue et al., 2023) requires the model to extract multiple non-redundant answers from a given passage. However, the models may need a large amount of training data to facilitate either MSQA or other QA tasks. To alleviate the high cost of manually annotating QA datasets, Question Generation (QG) has been proposed, which requires the model to generate a question related to the given answer and passage.

Traditional QG methods (Guo et al., 2024a) typically train a sequence-to-sequence language model (Seq2Seq LM) (Sutskever et al., 2014) that takes a passage and an answer as input to generate the corresponding questions. When answers are unavailable, these approaches usually employ rule-based methods (Lyu et al., 2021a; Lee et al., 2023) or model-based methods (Shakeri et al., 2020) to generate answers. With recent advancements in Large Language Models (LLMs), some researches have explored employing LLMs to generate questions. For example, Guo et al. (2024b) propose the SGSH framework that enhances question generation by incorporating question prefixes in prompts. PFQS

*Corresponding author. This work was supported by the National Key Research and Development Program of China under Grant 2022YFB4501704.

[1]Our code and data are available at https://github.com/PluseLin/REGULAR.



*Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pages 69–85
December 20-24, 2025 ©2025 Association for Computational Linguistics

(Li and Zhang, 2024) decomposes the QG task into a multi-step process, requiring the LLM first to generate a planning, then generate questions with the answer and the planning.

This work primarily focuses on Multi-Span Question Generation (MSQG), where the generated question corresponds to multiple candidate answers. Unfortunately, both existing QG methods and the LLMs struggle with MSQG. Taking Figure 1 as an example, the NER tool extracts people's names as answers. However, these answers are irrelevant, resulting in a trivial question. On the other hand, LLM extracted multiple action segments, but 'was rushed to intensive care' did not occur on Ben Kirk, so the generated question is incorrect. The reason may be that these methods primarily focus on generating single-answer questions, without considering the correlation between multiple answers in the MSQG task. Although LIQUID (Lee et al., 2023) employs an additional QA model to refine the initial candidate answers, the correlation between candidate answers is still ignored.

Relation graphs, in which edges connect different entities with relation types, may help obtain relevant candidate answers as relevant entities typically share the same relationship with the same entity. We define Commonality Entity (CE) as a group of entities that share the same relation type with a specific entity in a relation graph. Then we propose **REGULAR**, a framework for **RE**lation-**GU**ided Mu**L**ti-Sp**A**n Question Gene**R**ation. For a given passage, REGULAR converts it into a relation graph and employs a graph traversal algorithm to extract CE as candidate answers. After extracting candidate answers, REGULAR utilizes a QG model to generate a set of candidate questions and a QA model to obtain the best question. Compared with existing QG methods, REGULAR considers the relevance between candidate answers, avoiding the negative impact of irrelevant answers on the synthetic datasets.

We construct the REGULAR-WIKI dataset on the Wikipedia corpus. We conducted Supervised Fine-Tuning (SFT) on multiple open-source LLMs including Llama-3 (Grattafiori et al., 2024) and Qwen-2.5(Qwen et al., 2025) using both the REGULAR-WIKI dataset and MSQA datasets synthesized by existing QG methods, followed by comprehensive evaluations on multiple MSQA benchmarks. Experiment results show that LLMs trained on the REGULAR-WIKI consistently outperform other settings, indicating the superior quality of REGULAR-WIKI. Ablation studies confirm that each step in our proposed methodology is essential for synthesizing high-quality MSQA data. Besides, we also conduct statistical analysis to verify the quality of the REGULAR-WIKI dataset.

In summary, our contributions are listed as follows:

- To obtain relevant candidate answers in MSQG, we explore extracting entities from the relation graph as candidate answers. We define CE as a group of entities that share the same relation type with a specific entity in a relation graph, and design a graph traversal algorithm to extract CE.
- We propose REGULAR, which extracts CE from graph structures as candidate answers and generates corresponding questions. We construct the REGULAR-WIKI dataset from the Wikipedia corpus.
- Experiment results demonstrate that our synthetic datasets can be used to train open-source LLMs and achieve better performance. We also conduct ablation studies and statistical analysis to validate the quality of the synthetic dataset.

## 2 Related Work

### 2.1 Question Generation

QG requires models to generate a question that matches the given passage and the answer. This work primarily focuses on MSQG where the generated question corresponds to multiple answers. In real-world applications, the answers are often unknown, so obtaining the answers is necessary first and then generating the corresponding questions.

Traditional methods typically utilize LMs or rule-based tools to extract candidate answers. Puri et al. (2020) train a BERT (Devlin et al., 2019) to extract candidate answers. Shakeri et al. (2020) use a Sequence-to-Sequence LM to end-to-end generate both questions and answers. Lyu et al. (2021a) extract summarization of the given passage and then use NER tools and syntactic parsing tools to extract candidate answers. LIQUID (Lee et al., 2023) first extracts multiple candidate answers with a summarization model and NER tool, and generates multi-answer questions, followed by iterative

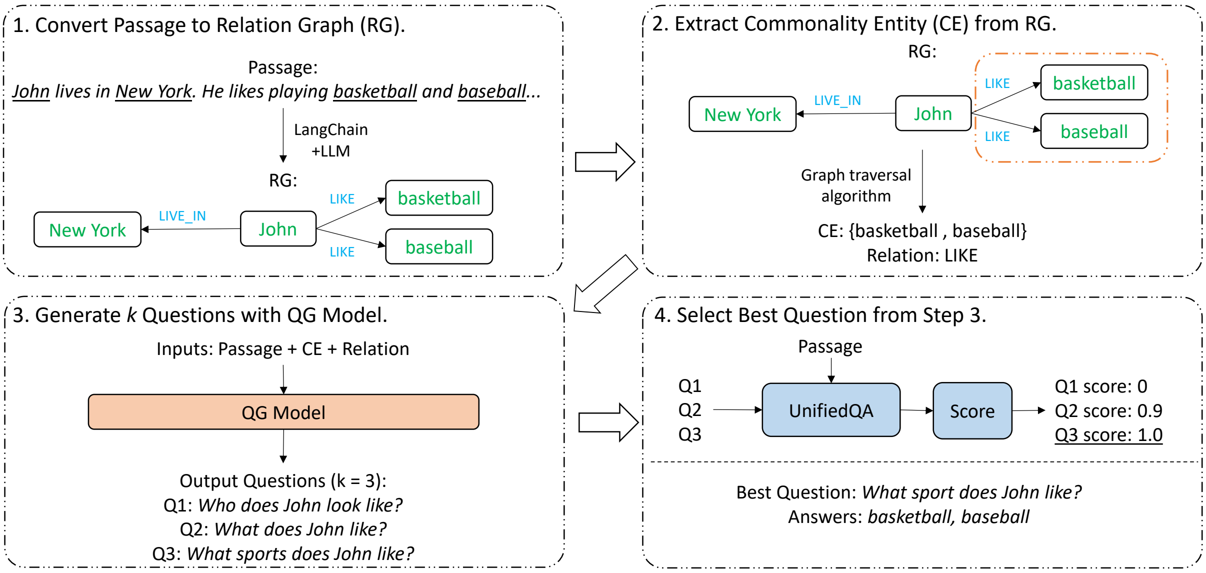


Figure 2: The pipeline of our REGULAR framework.

updates to both the questions and candidate answers. However, these methods fail to consider the correlation between candidate answers. In contrast, we extract CE in the relation graph, ensuring the correlation among the candidate answers and improving the quality of the synthetic datasets.

### 2.2 LLM-based Question Generation

Recently, LLMs (Grattafiori et al., 2024; OpenAI et al., 2024) have gained widespread attention due to their powerful language modeling and text generation capabilities. Recent studies have explored methods such as In-Context Learning (ICL) (Brown et al., 2020) and Chain-of-Thought (CoT) (Wei et al., 2022; Kojima et al., 2022) to further improve the performance of LLMs in QG tasks.

For example, TASE-CoT (Lin et al., 2024) first uses the T5 (Raffel et al., 2020) model to predict the question type and key fragments within the question, then designs a three-step CoT approach to guide the LLM in generating multi-hop questions. Similarly, SGSH (Guo et al., 2024b) addresses Knowledge Base Question Generation (KBQG) by using a fine-tuned BART (Lewis et al., 2020) model to provide the question prefix before generating questions with GPT-3.5. Li and Zhang (2024) focus on controllable question generation and propose the PFQS framework. This framework first generates an initial plan based on the question label, adjusts it with the context, and then generates the question based on the article, answer, and plan. In addition to text-only question generation, Wu et al. (2024) focus on Multi-Modal Question Generation (MMQG) and they propose SMMQG, which samples multi-modal sources and generates different types of questions with GPT-4.

In this work, we primarily utilize advanced LLMs to convert passages into relation graphs and use fine-tuned LLMs to generate questions.

## 3 Method

The MSQG task can be described as: Given a passage $p$, models are required to first extract a set of non-redundant text spans as the candidate answers $A$, and then generate the corresponding question $q$, as shown in Equation 1:

$$\begin{aligned} A &= Extract\_Answers(p) \\ q &= M_{QG}(p, A), \end{aligned} \tag{1}$$

where $M_{QG}$ refers to the QG model.

Figure 2 shows the architecture of our REGULAR framework. Specifically, the REGULAR framework consists of four steps: (1) Convert the given passage to a relation graph; (2) Extract CE from the relation graph as candidate answers; (3) Utilize a QG model to generate a set of candidate questions; (4) Score each candidate question with a QA model and select the best question with the highest score for constructing the MSQA dataset. Steps 1 and 2 ensure the relevance of the candidate answers, while steps 3 and 4 guarantee the consistency between the generated questions and the candidate answers.

Next, we will introduce the definition of CE in Section 3.1, and elaborate on each step from Section 3.2 to Section 3.4.

### 3.1 Commonality Entity

The definition of CE can be described as follows: Given a reference entity $\bar{v}$ and a relation $r$, CE is defined as a set of entities that connect to $\bar{v}$ with the edges that share the same relation $r$. The above definition can be represented by Equation 2.

$$CE(\bar{v}, r) = \{v | v \in N(\bar{v}) \wedge R(v, \bar{v}) = r\}, \quad (2)$$

where $N(\bar{v})$ represents the neighbor entities of $\bar{v}$ and $R(v, \bar{v})$ represents the relation of the edge between $v$ and $\bar{v}$.

### 3.2 Extracting CE as candidate answers

In MSQG tasks, selecting multiple candidate answers is important because unrelated candidate answers may result in low-quality questions (Lyu et al., 2021b; Lee et al., 2023). Existing methods (Lee et al., 2023) typically utilize NER tools (e.g., SpaCy [2]) to extract named entities. However, these approaches fail to consider the correlations among candidate answers, thereby limiting the quality of the synthetic data.

We propose extracting CE as candidate answers, considering that CE in a relation graph is connected to a specific entity through the same edges, ensuring relevance among these entities. This process contains two steps: converting passages into relation graphs and extracting CE in the relation graph.

**Converting Passages into Relation Graphs** We utilize *LangChain LLMGraphTransformer*[3] to convert passages into relation graphs. This process can be described as Equation 3:

$$G = LangChain(p), \quad (3)$$

where $p$ refers to the passage and $G$ refers to the relation graph and $LangChain()$ refers to the LangChain tool.

**Extracting CE in the Relation Graph** We design a graph traversal algorithm that identifies CE by counting the 1-hop neighbors of each node. We extract CE with two or more entities as candidate answers $A$. This process can be described as Equation 4:

$$A = Extract\_Answers(G), \quad (4)$$

where $G$ refers to the relation graph. We provide a detailed algorithm in Appendix A.

[2] https://spacy.io/

[3] https://python.langchain.com/api_reference/experimental/graph_transformers/langchain_experimental.graph_transformers.llm.LLMGraphTransformer.html

### 3.3 Generating Questions

**Generating Questions with CE** We utilize a generative LM $M_{QG}$ as the QG model to generate questions. The inputs of $M_{QG}$ are the passage $p$, the candidate answers $A$, reference entity $v$, and relation $r$. We sample $k$ candidate questions $Q = \{q_1, ..., q_k\}$, where $k$ is the number of generated questions, shown in Equation 5:

$$Q = M_{QG}(p, A, v, r), \quad (5)$$

**Extracting Relations for Training the QG Model** Existing MSQA datasets such as MultiSpanQA(Li et al., 2022) and MA-MRC(Yue et al., 2023) do not include commonality relation we need. Intuitively, we could use a prompted LLM to extract the commonality relation from the question. However, this may introduce bias between training and generating. To address this problem, we first prompt an LLM to convert the question-answer pairs into declarative sentences. Then, following the method proposed in Section 3.2, we check whether the answers satisfy the definition of CE. If the candidate answers are CE, we add the corresponding commonality relation $r$ to the training data, otherwise, we discard this data.

### 3.4 Obtaining Optimal Question

Existing QG researches (Lee et al., 2023; Mohammadshahi et al., 2023) typically employ a QA model to validate the generated questions. In this work, we employ a QA model $M_{QA}$ fine-tuned on the MSQA datasets to score the candidate questions generated in Section 3.3 and select the question with the highest score. For each candidate question $q_i \in Q$ and its corresponding passage $p$, we predict its answers with $M_{QA}$. Then we calculate the F1 score of the predicted answers and obtain the best question $\hat{q}$ that maximizes the F1 score.

This process can be described as Equation 6:

$$\begin{aligned} O_i &= M_{QA}(p, q_i) \\ s_{q_i} &= F1\_Score(O_i, A) \\ \hat{q} &= \underset{q_i \in Q}{argmax}(s_{q_i}), \end{aligned} \quad (6)$$

where $F1\_Score(O_i, A)$ refers to the F1 score of $O_i$ when $A$ is used as the reference[4].

Finally, we construct synthetic dataset $D$ with the candidate answers $A$ and the generated question

[4] When calculating the F1 score, we take the average of the Exact Match F1 and Partial Match F1 scores. Details of Exact Match and Partial Match are shown in Section 4.1

$\hat{q}$, shown in Equation 7:

$$D = \{(p, A, \hat{q})\}, \tag{7}$$

where $n$ refers to the question number of $D$.

## 4 Experiments

Based on the hypothesis that higher-quality synthetic datasets yield more capable models, we systematically compare datasets generated by our REGULAR framework and conventional QG methods. We perform supervised fine-tuning (SFT) on LLMs using each synthetic dataset, followed by out-of-domain (OOD) evaluation on human-annotated MSQA benchmarks. Furthermore, we design ablation studies to examine the contribution of key components in the REGULAR framework and validate their rationality.

### 4.1 Experimental Setup

**Synthetic Dataset** We select the open-source corpus **Wikipedia** [5] and construct the REGULAR-WIKI dataset with our proposed framework. The REGULAR-WIKI dataset contains over 100,000 questions. To save computation cost, we randomly sample 5,000 high-quality questions for our experiment.

**MSQA Benchmarks** We select the **MultiSpanQA** (Li et al., 2022), **MA-MRC** (Yue et al., 2023), and **QUOREF** (Dasigi et al., 2019) for our experiments. Considering that the MA-MRC dataset contains over 8,000 questions in the validation set, we randomly sample 1,000 questions for evaluation to reduce computational cost. Details of the MSQA benchmarks are shown in Appendix B.1.

**Models** We select Llama3.2-3B, Llama3.1-8B (Grattafiori et al., 2024) [6], Qwen2.5-3B, and Qwen2.5-7B (Qwen et al., 2025) for our experiments. We download the model checkpoints from huggingface[7].

**Baselines** We set both training-free baselines (zero-shot and few-shot) and training-required baselines (QAGen-LLM and LIQUID) for our experiments:

- **Zero-shot**: We prompt LLM to extract answers from the given passage. The prompt only contains the task definition, passage, and question.
- **Few-shot**: Building upon the zero-shot setting, we enhance the prompt with examples containing a passage, a question, and gold answers. In our experiment, we utilize the BM25 retriever (Robertson and Walker, 1994) and select 3 examples for each question in the validation set. The prompt for zero-shot and few-shot is shown in Appendix Table 7.
- **QAGen-LLM** (Shakeri et al., 2020) use a generative LM to generate questions and answers. In this work, we employ GPT-4o [8] to generate question-answer pairs from given passages. We add 2 examples in the prompt to facilitate the generation of multi-answer questions and their corresponding answers. The prompt for QAGen-LLM is shown in Appendix Table 9.
- **LIQUID** (Lee et al., 2023) first uses a summarization model and NER tools to extract named entities as candidate answers. Then, LIQUID employs a QG model to generate questions, and the questions and candidate answers are updated through multiple iterations. We download the LIQUID-WIKI datasets from `https://github.com/dmis-lab/LIQUID` for our experiments.

**Evaluation Metrics** Following (Li et al., 2022), we use **Exact Match (EM)** and **Partial Match (PM)** as the main metrics. EM assigns a score of 1 when a prediction fully matches one of the gold answers and 0 otherwise, while PM considers the overlap between the predictions and gold answers. We report F1 scores in our experiments.

**Implementation Details** When converting passages to relation graphs, we utilize the LangChain *LLMGraphTransformer* [9] and invoke *GPT-4o-mini* [10]. When generating questions, we select Llama3.1-8B as the QG model and train it on MultiSpanQA and MA-MRC datasets. The prompt for the QG model is shown in Appendix Table 8. When selecting the best question, we choose *UnifiedQA-T5-large*[11] and fine-tune it on MultiSpanQA and MA-MRC datasets. Training details are shown in Appendix B.2.

[5] `https://www.wikipedia.org/`
[6] For simply expression, we refer Llama-3.2 and Llama-3.1
[7] `https://huggingface.co/`
[8] `https://openai.com/api/`
[9] `https://python.langchain.com/`
[10] `https://openai.com/api/`
[11] `https://huggingface.co/allenai/unifiedqa-t5-large`

| | MultiSpanQA | | MA-MRC | | QUOREF | | Average | |
|---|---|---|---|---|---|---|---|---|
| | EM F1 | PM F1 | EM F1 | PM F1 | EM F1 | PM F1 | EM F1 | PM F1 |
| **Llama3.2-3B** | | | | | | | | |
| Zero-Shot | 57.31 | 75.23 | 56.40 | 71.42 | 63.47 | 75.71 | 59.06 | 74.12 |
| Few-Shot | 65.03 | 80.03 | 69.64 | 80.65 | 64.34 | 72.06 | 66.34 | 77.58 |
| SFT (QAGen-gpt4o) | 69.06 | 83.30 | 67.40 | 80.04 | 64.83 | 77.34 | 67.10 | 80.23 |
| SFT (LIQUID) | 59.75 | 77.45 | 62.38 | 75.50 | 55.23 | 70.05 | 59.12 | 74.33 |
| SFT (REGULAR) | **70.49** | **84.45** | **73.75** | **83.85** | **67.04** | **81.76** | **70.43** | **83.35** |
| **Llama3.1-8B** | | | | | | | | |
| Zero-Shot | 58.41 | 76.66 | 62.51 | 75.61 | 73.02 | 82.80 | 64.65 | 78.36 |
| Few-Shot | 68.79 | 84.16 | 71.79 | 81.65 | 74.08 | 83.81 | 71.55 | 83.21 |
| SFT (QAGen-gpt4o) | 70.16 | 85.45 | 69.01 | 81.15 | 74.59 | 84.43 | 71.25 | 83.68 |
| SFT (LIQUID) | 68.45 | 84.20 | 69.85 | 80.77 | 70.27 | 81.75 | 69.52 | 82.24 |
| SFT (REGULAR) | **72.12** | **86.23** | **74.60** | **83.98** | **76.79** | **85.66** | **74.50** | **85.29** |
| **Qwen2.5-3B** | | | | | | | | |
| Zero-Shot | 59.45 | 76.24 | 57.56 | 71.76 | 64.42 | 73.75 | 60.48 | 73.92 |
| Few-Shot | 65.22 | 79.61 | 65.52 | 77.76 | 64.43 | 74.70 | 65.06 | 77.36 |
| SFT (QAGen-gpt4o) | 67.73 | 82.63 | 62.54 | 77.52 | 67.93 | 80.61 | 66.07 | 80.25 |
| SFT (LIQUID) | 67.11 | 82.44 | 66.25 | 78.33 | 67.31 | 78.17 | 66.89 | 79.65 |
| SFT (REGULAR) | **69.06** | **82.45** | **72.11** | **82.70** | **69.15** | **81.54** | **70.11** | **82.23** |
| **Qwen2.5-7B** | | | | | | | | |
| Zero-Shot | 68.06 | 82.79 | 60.12 | 73.26 | 75.88 | 84.19 | 68.02 | 80.08 |
| Few-Shot | 70.58 | 84.59 | 68.02 | 79.89 | 76.17 | 84.01 | 71.59 | 82.83 |
| SFT (QAGen-gpt4o) | 70.55 | 84.14 | **73.31** | **83.45** | 74.71 | 85.40 | 72.86 | 84.33 |
| SFT (LIQUID) | 69.59 | 83.61 | 64.08 | 76.72 | 64.29 | 76.21 | 65.99 | 78.85 |
| SFT (REGULAR) | **71.23** | **85.28** | 72.59 | 83.28 | **78.79** | **85.75** | **74.20** | **84.77** |

Table 1: Exact Match and Partial Match F1 scores of the LLMs in the training-free and training-required settings. "SFT (QAGen-LLM)", "SFT (LIQUID)", and "SFT (REGULAR)" refer to the models trained with QAGen-LLM, LIQUID, and REGULAR-WIKI datasets, respectively. The best results are in **bold**.

### 4.2 Main Results

The main results are shown in Table 1. Based on these results, the following conclusions can be made: **(1) Incorporating some demonstrations improves the performance of LLMs.** For instance, on the MultiSpanQA dataset, the EM F1 score of Llama-3B in the few-shot setting improves by 8 points compared to the zero-shot setting. The improvements on the QUOREF dataset are limited, probably due to the excessive length of the demonstrations, which constrained the LLM's performance. **(2) Traditional QG methods struggle with generating high-quality MSQA datasets.** We observe that after training on the LIQUID dataset, the performance of the LLM only slightly surpassed the zero-shot setting. Moreover, even models trained on data directly generated by GPT-4o exhibit a decline in performance. **(3) The quality of the REGULAR dataset exceeds that of other synthetic datasets.** In our experiments, LLMs trained on REGULAR achieve best performance in most settings, with particularly significant improvements observed in the 3B model. This is because the REGULAR framework extracts CE from the relation graph, ensuring the correlation between candidate answers, and thereby improving the quality of the synthetic dataset.

| | Llama3.2-3B | | Llama3.1-8B | |
|---|---|---|---|---|
| | EM F1 | PM F1 | EM F1 | PM F1 |
| REGULAR | **70.49** | **84.45** | **72.12** | **86.23** |
| **Step 1-2** | | | | |
| NER ans | 61.47 | 77.00 | 63.82 | 79.85 |
| LLM ans | 59.46 | 74.86 | 62.29 | 77.52 |
| **Step 3** | | | | |
| w/o context | 69.61 | 83.90 | 71.74 | 85.65 |
| w/o relation | 69.84 | 83.53 | 71.17 | 85.96 |
| **Step 4** | | | | |
| Random Q | 68.30 | 82.27 | 70.34 | 84.94 |
| Worst Q | 65.89 | 79.68 | 68.85 | 81.35 |

Table 2: Ablation Study on the validation set of MultiSpanQA. The best results are in **bold**.

Besides the OOD experiments, we also compare LLMs trained with REGULAR-WIKI and human-annotated datasets. Results and analysis are shown in Appendix C.1.

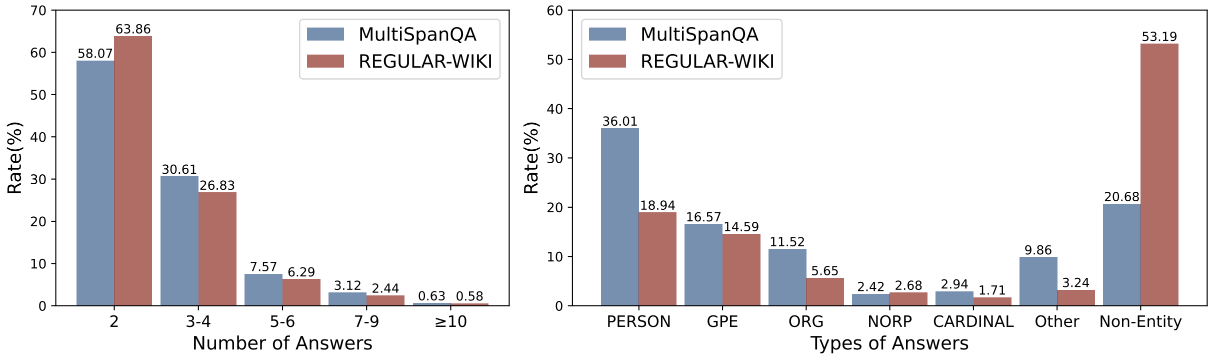


Figure 3: Left: Number of answers in the MultiSpanQA and the REGULAR-WIKI datasets; Right: Types of answers in the MultiSpanQA and the REGULAR-WIKI datasets. The numbers in the figures represent the percentage.

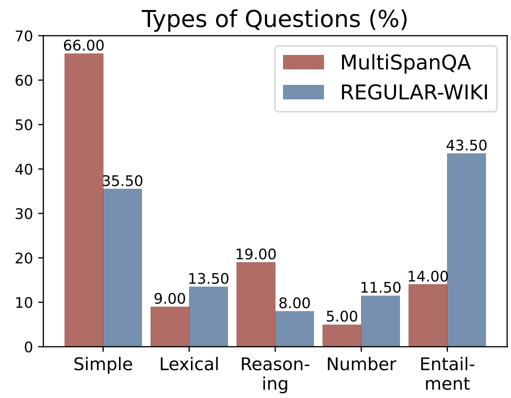


Figure 4: Types of questions in the MultiSpanQA and the REGULAR-WIKI datasets. One question may contain two or more question types.

### 4.3 Ablation Study

We hypothesize that each step in REGULAR contributes to constructing a higher-quality synthetic dataset. To validate this, we perform ablation studies on each REGULAR synthetic step and evaluate the validation set of the MultiSpanQA dataset. We implement the following ablation strategies: (1) **NER ans**: Use NER tools to extract candidate answers from the passage. (2) **LLM ans**: Directly prompt the LLM to extract candidate answers from the passage along with the commonality relation. The prompt is shown in Appendix Table 10 (3) **w/o context**: Remove the passage when generating questions. (4) **w/o relation**: Remove the commonality relation and key entity when generating questions. (5) **Random Q**: Randomly select a candidate question instead of the highest-scoring question. (6) **Worst Q**: Select the lowest-scoring question instead of the highest-scoring question.

As shown in Table 2, all ablation settings lead to a decline in model performance. Specifically, the ablations of Step 1 and Step 2 cause EM/F1 scores of Llama3.2-3B to decrease by 9 and 11 points, respectively. This suggests that using NER tools and prompting the LLM to extract answers does not yield high-quality results. On the other hand, training on datasets constructed with the worst questions (worst Q) also results in a performance decline, indicating that selecting best questions is beneficial for better LLM training.

## 5 Analysis on the Synthetic Dataset

In this section, we statistically analyze the answer types, number of answers, and question types in the REGULAR-WIKI and MultiSpanQA datasets. We also conduct a case study to compare the REGULAR-WIKI dataset with the QAGen-LLM dataset.

### 5.1 Number of Answers

We analyze the number of answers for each question in the MultiSpanQA and REGULAR-WIKI datasets, as shown in Figure 3. Compared with the MultiSpanQA dataset, the REGULAR-WIKI dataset has a higher proportion of questions with 2 answers and a lower proportion with more than 3 answers. This may be because REGULAR extracts answers with specific topological structures (i.e. CE), limiting the number of answers.

### 5.2 Types of Answers

We use SpaCy to analyze the answer types in the MultiSpanQA and REGULAR-WIKI datasets. Figure 3 shows the proportion of named entity answers with top-5 frequencies and non-named entity answers. Surprisingly, we observe that the proportion of non-entity answers in REGULAR-WIKI was much higher than in MultiSpanQA. This may be because both named and non-named entities were

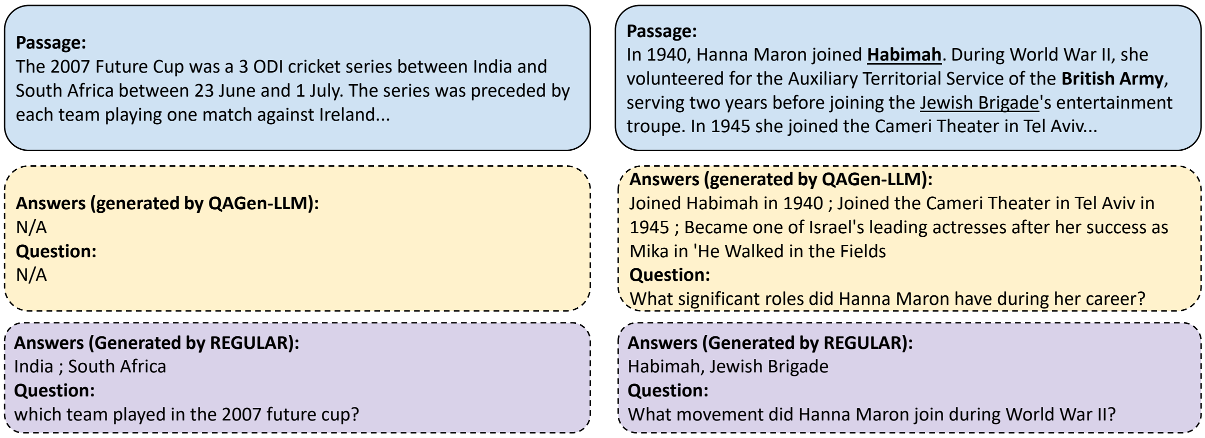


Figure 5: Case study. The examples are selected from the QAGen-LLM and REGULAR-WIKI datasets.

included as nodes during the relation graph extraction process. The reason may be that incorporating more non-named entities as candidate answers helps enhance the diversity of questions and answers.

| | Time (1,000 data) |
|---|---|
| Step 1 | 51 min. |
| Step 2 | <1s |
| Step 3 | 17 min. |
| Step 4 | 42 min. |
| Total | 110 min. |

Table 3: Inference time of REGULAR. REGULAR takes about 110 minutes to generate 1,000 questions.

### 5.3 Types of Questions

We further analyze the distribution of question types in REGULAR-WIKI and MultiSpanQA datasets. We adopt the categories proposed by Lee et al. (2023): Simple Questions, Lexical Variation, Inter-sentence Reasoning, Number of Answers, and Entailment, where a question may correspond to multiple types. We sample 200 questions and use GPT-4o to classify each question. Detailed definitions of the five types of questions can be found in Appendix D.

The statistical results are shown in Figure 4[12]. Compared with the MultiSpanQA dataset, the REGULAR-WIKI dataset contains fewer Simple Questions. These questions typically have answers within a single sentence, but the answers in REGULAR-WIKI are derived from relation graphs and might span multiple sentences. On the other hand, REGULAR-WIKI contains more Entailment questions, perhaps because the generated questions implicitly contain prior knowledge from the QG model. Overall, the question distribution in REGULAR-WIKI is more balanced, suggesting that the REGULAR framework can generate a wider variety of questions.

[12]Due to differences in sampling data and evaluation methods, the analysis results may differ from the results in (Lee et al., 2023).

### 5.4 Case Study

We conduct a case study demonstrating that the REGULAR method can generate better synthetic datasets. Figure 5 shows examples of questions and answers generated by QAGen-LLM and REGULAR for the same passage. In the first case, QAGen-LLM fails to provide a question and answers. However, "India" and "South Africa" are countries that joined in "the 2007 Future Cup". In contrast, REGULAR extracts these countries and provides a question that is relevant to the answers. In the second case, although QAGen-LLM provide a correct question, the answers extracted by QAGen-LLM are relatively long and could be simplified. The answers generated by REGULAR are more concise, which is beneficial for training. These examples demonstrate that the REGULAR method, by extracting CE, can generate higher-quality questions and answers.

### 5.5 Time Cost

We calculate the time cost of constructing 1,000 questions using the REGULAR pipeline. The result is shown in Table 3. REGULAR takes about 2 hours to generate 1,000 questions, where the main time expenses are in the first and fourth steps. The time cost could be further reduced by adopting parallel technology and faster APIs.

## 6 Conclusion

This work focuses on the MSQG task and proposes REGULAR, a framework for relation-guided Multi-Span Question Generation. REGULAR converts passages into relation graphs and extracts CE as the candidate answers. Then, REGULAR utilizes a QG model to generate a set of candidate questions and a QA model to obtain the best question. We construct over 100,000 questions using the Wikipedia corpora, named REGULAR-WIKI. We conduct SFT experiments where we compare models trained with REGULAR-WIKI and models trained with other synthetic datasets. The experiment results show that models trained with the REGULAR-WIMI dataset achieve best performance in most settings, indicating that the quality of the REGULAR datasets is higher than other synthetic QA datasets. Besides, we also conduct ablation studies and statistical analysis to validate the quality of the synthetic dataset.

## 7 Limitations and Future Work

In this work, we utilize LangChain to convert passages into relation graphs. However, this step relies on advanced LLMs (e.g., GPT-4o-mini), which may incur significant costs. Although we assume that advanced LLMs have mastered the ability to extract relation graphs during their training, we have not explicitly addressed the potential errors that may occur. On the other hand, we primarily focus on generating multi-answer questions. We do not consider other types of question generation (e.g., multi-hop reasoning questions, multiple-choice questions, etc.).

In future work, we plan to improve the ability of LLMs to extract relation graphs with the open-source LLMs (e.g., Llama, Qwen). Additionally, we will explore how this method can be applied to generate other types of questions.

## References


Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Pradeep Dasigi, Nelson F. Liu, Ana Marasović, Noah A. Smith, and Matt Gardner. 2019. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. 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 5925–5932, Hong Kong, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Andre Esteva, Anuprit Kale, Romain Paulus, Kazuma Hashimoto, Wenpeng Yin, Dragomir Radev, and Richard Socher. 2021. Covid-19 information retrieval with deep-learning based semantic search, question answering, and abstractive summarization. *NPJ digital medicine*, 4(1):68.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal

Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad

Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Shash Guo, Lizi Liao, Cuiping Li, and Tat-Seng Chua. 2024a. A survey on neural question generation: methods, applications, and prospects. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, IJCAI '24.

Shasha Guo, Lizi Liao, Jing Zhang, Yanling Wang, Cuiping Li, and Hong Chen. 2024b. SGSH: Stimulate large language models with skeleton heuristics for knowledge base question generation. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4613–4625, Mexico City, Mexico. Association for Computational Linguistics.

Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213. Curran Associates, Inc.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*.

Seongyun Lee, Hyunjae Kim, and Jaewoo Kang. 2023. Liquid: A framework for list question answering dataset generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(11):13014–13024.

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 *ACL:2020:main*, pages 7871–7880, Online. acl.

Haonan Li, Martin Tomko, Maria Vasardani, and Timothy Baldwin. 2022. MultiSpanQA: A dataset for multi-span question answering. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1250–1260, Seattle, United States. 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.

Zefeng Lin, Weidong Chen, Yan Song, and Yongdong Zhang. 2024. Prompting few-shot multi-hop question generation via comprehending type-aware semantics. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3730–3740, Mexico City, Mexico. Association for Computational Linguistics.

Chenyang Lyu, Lifeng Shang, Yvette Graham, Jennifer Foster, Xin Jiang, and Qun Liu. 2021a. Improving unsupervised question answering via summarization-informed question generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4134–4148, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chenyang Lyu, Lifeng Shang, Yvette Graham, Jennifer Foster, Xin Jiang, and Qun Liu. 2021b. Improving unsupervised question answering via summarization-informed question generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4134–4148, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alireza Mohammadshahi, Thomas Scialom, Majid Yazdani, Pouya Yanki, Angela Fan, James Henderson, and Marzieh Saeidi. 2023. RQUGE: Reference-free metric for evaluating question generation by answering the question. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6845–6867, Toronto, Canada. Association for Computational Linguistics.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik

Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Raul Puri, Ryan Spring, Mohammad Shoeybi, Mostofa Patwary, and Bryan Catanzaro. 2020. Training question answering models from synthetic data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5811–5826, Online. Association for Computational Linguistics.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Roni Rabin, Alexandre Djerbetian, Roee Engelberg, Lidan Hackmon, Gal Elidan, Reut Tsarfaty, and Amir Globerson. 2023. Covering uncommon ground: Gap-focused question generation for answer assessment. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 215–227, Toronto, Canada. Association for Computational Linguistics.

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(140):1–67.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

S. E. Robertson and S. Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR '94*, pages 232–241, London. Springer London.

Siamak Shakeri, Cicero Nogueira dos Santos, Henghui Zhu, Patrick Ng, Feng Nan, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2020. End-to-end synthetic data generation for domain adaptation of question answering systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5445–5460, Online. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 28th International Conference* *on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3104–3112, Cambridge, MA, USA. MIT Press.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Ian Wu, Sravan Jayanthi, Vijay Viswanathan, Simon Rosenberg, Sina Khoshfetrat Pakazad, Tongshuang Wu, and Graham Neubig. 2024. Synthetic multimodal question generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 12960–12993, Miami, Florida, USA. Association for Computational Linguistics.

Zhiang Yue, Jingping Liu, Cong Zhang, Chao Wang, Haiyun Jiang, Yue Zhang, Xianyang Tian, Zhedong Cen, Yanghua Xiao, and Tong Ruan. 2023. Ma-mrc: A multi-answer machine reading comprehension dataset. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 2144–2148, New York, NY, USA. Association for Computing Machinery.

# A Algorithm for Extracting Commonality Entity

The algorithm for extracting CE is shown in Algorithm 1. Specifically, for a given relation graph $G = \{V, E\}$, we first initialize its adjacency matrix $M_G$. Then, for each node $v \in V$, we count its 1-hop neighbor nodes and the types of edges connecting them. If node $v$ is connected to a set of neighbor nodes $\bar{V}$ via edges of the same type, or if $\bar{V}$ point to $v$ using edges of the same type, then $\bar{V}$ are considered as CE.

# B Experimental Setup

## B.1 MSQA Datasets

**MultiSpanQA (Li et al., 2022)** MultiSpanQA focuses on questions with more than one answer. The raw questions and contexts are extracted from the Natural Question dataset (Kwiatkowski et al., 2019).

**MA-MRC (Yue et al., 2023)** MA-MRC is a large-scale dataset containing over 100,000 questions, including both multi-span questions and single-span questions. In this work, we randomly sample 10,000 training data and 1,000 validation data and obtain **MA-MRC-10k** for our experiment.

**QUOREF (Dasigi et al., 2019)** The QUOREF dataset is sourced from Wikipedia and contains over 4,700 passages and more than 24,000 questions. The QUOREF dataset requires the model to possess certain co-reference resolution and reasoning abilities. In this work, we select questions with multiple answers for our experiment.

Since the official test sets of these datasets are not public, we report the performance on validation sets. Some statistics about the four datasets are shown in Table 4.

## B.2 Implementation Details

We utilize Huggingface's trl [13] to conduct SFT. We train our model with 4 V100 GPUs (32GB). Training hyper-parameters are shown in Table 6.

# C Additional Experiment and Analysis

## C.1 Comparisons with Human-annotated Datasets

We compare the performance of LLMs trained with REGULAR-WIKI and human-annotated datasets. The training implementation details are the same as the main experiments. The results are shown in Table 5. Due to the domain gaps between the REGULAR-WIKI dataset and the human-annotated dataset, the performance of LLMs trained with REGULAR-WIKI is slightly lower than LLMs trained on human-annotated datasets. However, LLMs trained with REGULAR-WIKI achieve second-best performance in some settings. For example, Llama-3B trained with REGULAR-WIKI performs better than Llama-3B trained with QUOREF and MultiSpanQA datasets. This indicates that the REGULAR-WIKI dataset could improve the generalization ability of LLMs and achieve similar results compared to human-annotated datasets.

# D Definition of the Types of Question

Lee et al. (2023) proposes a category for question types based on the reasoning required to answer these questions, listed as follows:

- **Simple questions**: Questions simply derived from evidence texts with few lexical variations.
- **Lexical variation**: Questions created with lexical variations using synonyms and hypernyms.

[13] https://github.com/huggingface/trl

| | #train | #dev | average answer number | average context length | average question length |
|---|---|---|---|---|---|
| **MultiSpanQA** | 5,230 | 658 | 2.89 | 279 | 10 |
| **MA-MRC (10k)** | 10,000 | 1000 | 2.31 | 77 | 10 |
| **QUOREF** | 1,963 | 215 | 2.45 | 431 | 19 |

Table 4: Dataset Statistic.

| | **MultiSpanQA** | | **MA-MRC** | | **QUOREF** | | **Average** | |
|---|---|---|---|---|---|---|---|---|
| | **EM F1** | **PM F1** | **EM F1** | **PM F1** | **EM F1** | **PM F1** | **EM F1** | **PM F1** |
| **Llama3.2-3B** | | | | | | | | |
| Oracle (MultiSpanQA) | **76.28** | **88.97** | 66.63 | 82.32 | <u>75.14</u> | <u>87.53</u> | <u>72.68</u> | **86.27** |
| Oracle (MA-MRC) | 65.14 | 80.64 | **80.02** | **87.57** | 70.23 | 81.80 | 71.80 | 83.34 |
| Oracle (QUOREF) | <u>75.52</u> | <u>85.35</u> | 65.23 | 78.56 | **83.12** | **90.71** | **74.62** | <u>84.87</u> |
| SFT (REGULAR) | 70.49 | 84.45 | <u>73.75</u> | <u>83.85</u> | 67.04 | 81.76 | 70.43 | 83.35 |
| **Llama3.1-8B** | | | | | | | | |
| Oracle (MultiSpanQA) | <u>77.47</u> | **89.69** | 67.96 | 83.28 | 75.39 | <u>88.36</u> | 73.61 | <u>87.11</u> |
| Oracle (MA-MRC) | 67.67 | 82.02 | **82.95** | **89.23** | 72.42 | 83.75 | 74.35 | 85.00 |
| Oracle (QUOREF) | **79.53** | <u>88.28</u> | 72.28 | 82.16 | **85.4** | **92.66** | **79.07** | **87.70** |
| SFT (REGULAR) | 72.12 | 86.23 | <u>74.6</u> | <u>83.98</u> | <u>76.79</u> | 85.66 | <u>74.50</u> | 85.29 |
| **Qwen2.5-3B** | | | | | | | | |
| Oracle (MultiSpanQA) | <u>75.16</u> | **87.78** | 66.79 | 82.44 | <u>72.38</u> | <u>85.68</u> | 71.44 | **85.30** |
| Oracle (MA-MRC) | 64.66 | 80.50 | **79.76** | **87.12** | 68.21 | 81.19 | 70.88 | 82.94 |
| Oracle (QUOREF) | **75.18** | <u>84.57</u> | 65.66 | 76.61 | **80.9** | **88.02** | **73.91** | <u>83.07</u> |
| SFT (REGULAR) | 69.06 | 82.45 | <u>72.11</u> | <u>82.70</u> | 69.15 | 81.54 | 70.11 | 82.23 |
| **Qwen2.5-7B** | | | | | | | | |
| Oracle (MultiSpanQA) | <u>76.22</u> | **88.89** | 65.84 | <u>81.69</u> | 73.53 | 86.86 | 71.86 | <u>85.81</u> |
| Oracle (MA-MRC) | 65.64 | 81.15 | **80.55** | **87.89** | 70.23 | 82.04 | 72.14 | 83.69 |
| Oracle (QUOREF) | **78.98** | <u>87.41</u> | <u>74.98</u> | 84.15 | **87.01** | **92.83** | **80.32** | **88.13** |
| SFT (REGULAR) | 71.23 | 85.28 | 72.59 | 83.28 | <u>78.79</u> | <u>85.75</u> | <u>74.20</u> | 84.77 |

Table 5: Exact Match and Partial Match F1 scores of the LLMs trained with REGULAR-WIKI and human-annotated datasets. "SFT (REGULAR)" refers to the models trained with REGULAR-WIKI. "Oracle (MultiSpanQA)", "Oracle (MA-MRC)", and "Oracle (QUOREF)" refer to the models trained with MultiSpanQA, MA-MRC, and QUOREF datasets, respectively. The best results are in **bold** and the second-best results are in <u>underline</u>.

| **Hyperparameter** | **Value** |
|---|---|
| Learning Rate | 5e-5 |
| Warmup Steps | 20 |
| Batch Size | 24 |
| epochs | 2 |
| Max Input Length | 2048 |
| Max Output Length | 128 |
| Random Seed | 1111 |
| Optimizer | Adam |
| LoRA rank | 32 |
| LoRA alpha | 32 |
| LoRA Dropout | 0.1 |

Table 6: Training Hyper-parameters. "1st-tune" and "2nd-tune" refer to the first step and the second step of the 2-step fine-tuning strategy, respectively.

- **Inter-sentence reasoning**: Questions that require high-level reasoning such as anaphora, or answers that are distributed across multiple sentences.
- **Number of answers**: Questions that specify the number of answers, which is a characteristic of a list of questions.
- **Entailment**: Questions that require textual entailment based on the evidence texts and commonsense.

**Algorithm 1:** Extracting Commonality Entities

**Input:** $G = \{V, E\}$ : Knowledge Graph
**Output:** $CE\_list$ : Commonality Entities List
**Function** `ExtractCommonalityEntities`($G$)**:**
    $CE\_list \leftarrow \emptyset$;
    `/* Initialize adjacency matrix` $M$ `of` $G$. `*/`
    $M \leftarrow$ adjacency_matrix($G$);
    `/* Find commonality entites with the structure like` $B \leftarrow A \rightarrow C$ `or` $B \rightarrow A \leftarrow C$. `*/`
    **foreach** *entity* $v$ *in* $V$ **do**
        `/* Initialize` $Groups_1$ `and` $Groups_2$ `as a map. */`
        $Groups_1 \leftarrow map()$;
        $Groups_2 \leftarrow map()$;
        **foreach** *entity* $u$ *in* $V$ **do**
            `/* If there exists edge from` $v$ `to` $u$, `then` $M[u][v] > 0$. `*/`
            $r_1 \leftarrow M[v][u]$;
            $r_2 \leftarrow M[u][v]$;
            **if** $r_1 > 0$ **then**
                $Groups_1[r_1] \leftarrow Groups_1[r_1] \cup m$;
            **if** $r_2 > 0$ **then**
                $Groups_2[r_1] \leftarrow Groups_2[r_1] \cup n$;
        **foreach** *group in* $Groups_1$ **do**
            `/* Add groups with more than 2 elements to` $CE\_list$ `*/`
            **if** $len(group) > 2$ **then**
                $CE\_list \leftarrow CE\_list \ \cup \ group$
        **foreach** *group in* $Groups_2$ **do**
            **if** $len(group) > 2$ **then**
                $CE\_list \leftarrow CE\_list \ \cup \ group$
    **return** $CE\_list$;

---

# Task Definition

Answering the following question which contains one or more answers. You should extract the answer spans from the given context. Use '#' to separate each answers,for example, if the answers are 'Tom' and 'Jerry', you should output 'Tom # Jerry'. Your reply should not contain any explanation.

# Examples
Example 1:
Inputs:
Question: question1
Passage: passage1
Outputs:
Answers: answers1

...

Inputs:
Question: question
Passage: passage
Outputs:
Answers:

Table 7: Prompts for MSQA task

# Role:
You are an English exam question designer specializing in generating corresponding questions based on given articles and candidate answers.

# Input:
Context: Contains multiple sentences.
Answers: Includes multiple text fragments or phrases.
Key Entity: All candidate answers share the same relationship type with this entity.
Relation: The shared relationship between the key entity and each candidate answer.

# Output:
Question: An interrogative sentence that must meet the following conditions:
1. Answerability: The question should be answerable by reading the article, with all candidate answers serving as correct responses.
2. Relevance: The question must relate to the key entity and specifically inquire about the corresponding relationship type.
3. Fluency: The question must be grammatically correct and free of errors or awkward phrasing.

Additional notes:
The generated question should naturally elicit all provided candidate answers when answered correctly.
The relationship between the key entity and answers should be clearly reflected in the question's formulation.
Avoid yes/no questions to ensure answers require the provided text fragments.

# Example:
Input:
Answers: basketball ; football
Relation: LIKE
key_entity: John
Context: John lives in New York. He likes playing basketball and football.

Output:
Question: What sports does John like?

Table 8: Prompts for the question generation step of REGULAR.

# Task Instruction:
You will be given an article. Your task is to:
1. Generate a question based on the article's content.
2. Provide answers to the question using multiple direct text snippets extracted from the article.

# Output Requirements:
- Format the output as JSON with the following keys:
- "question": The generated question (value: string).
- "answers": A list of answer snippets copied verbatim from the article (value: array of strings).

Here are some examples for you:
Example 1:
Input:
Passage: passage
Output:
{
"question":{question}
"answers:[
{answer1},{answer2}
]
}

Table 9: Prompts for the QAGen-LLM.

Task Instruction:
You will be given an article. Your goal is to identify all "commonality entities" in the text. Here, Entity A and Entity B are defined as "commonality entities" if they share the same relation type with Entity C (referred to as the "key entity"). Note that an article may contain multiple such entities, and your output must list all of them. You should also note that the "commonality entities" and "key entities" must be in the given passage.

Output Format:
Provide the results in JSON format with the following keys:
- ‘"commonality_entities"‘: A list of all identified commonality entities (value: array). The entities may contain multiple words and you should split each word with space.
- ‘"key_entity"‘: The key entity (value: string).
- ‘"relation"‘: The shared relation type (value: string).
Your output should start with "{" and end with "}".

Here is an example for you:
Input: Idiopathic nonspecific inflammatory disease of the orbit (orbital pseudotumor) was diagnosed detected in a cat. The cat had progressive lagophthalmia, keratitis, and decreased motion of the right eye. Four months later, the left eye was affected in a similar manner. Response to antibiotics and immunosuppressive agents was not detected. Computed tomography of the brain and orbits revealed bilateral thickening of the sclera and episcleral tissues. Bilateral exenteration of the eyes was required because of worsening clinical signs or corneal perforation. Histologic examination revealed proliferation of spindle cells and fibrovascular tissue within and adjacent to the sclera.
Output: {"commonality_entities": ["Lagophthalmia", "Keratitis","Decreased Motion Of The Right Eye"], "key_entity": "Cat", "relation": "HAS_SYMPTOM"}

Table 10: Prompts for the QAGen-LLM.