Extracting, Detecting, and Generating Research Questions for Scientific Articles

Sina Taslimi[1,2], Artemis Capari[1], Hosein Azarbonyad[1], Zi Long Zhu[1], Zubair Afzal[1], Evangelos Kanoulas[2], George Tsatsaronis[1],

[1]Elsevier, [2]University of Amsterdam

taslimisina@yahoo.com, {a.capari,h.azarbonyad,z.zhu,zubair.afzal,g.tsatsaronis}@elsevier.com, e.kanoulas@uva.nl

Abstract

The volume of academic articles is increasing rapidly, reflecting the growing emphasis on research and scholarship across different science disciplines. This rapid growth necessitates the development of tools for more efficient and rapid understanding of these articles. Clear and well-defined Research Questions (RQs) in research articles can help guide scholarly inquiries. However, many academic studies lack a proper definition of RQs in their articles. This research addresses this gap by presenting a comprehensive framework for the systematic extraction, detection, and generation of RQs from scientific articles. The extraction component uses a set of regular expressions to identify articles containing well-defined RQs. The detection component aims to identify more complex RQs in articles, beyond those captured by the rule-based extraction method. The RQ generation focuses on creating RQs for articles that lack them. We integrate all these components to build a pipeline to extract RQs or generate them based on the articles' full text. We evaluate the performance of the designed pipeline on a set of metrics designed to assess the quality of RQs. Our results indicate that the proposed pipeline can reliably detect RQs and generate high-quality ones.

1 Introduction

Scientific articles, often characterized by their extensive and complex nature, pose challenges for users seeking specific key findings or addressed questions within the research (Zia and Mushtaq, 2023; Candal-Pedreira et al., 2023). These challenges are partially due to the absence of standardized Research Question (RQ) formulations across articles (Khongsdier, 2007), with many failing to state their findings or even feature RQs explicitly. With the rapid expansion of scholarly literature, there is a growing need for efficient and reliable methods to both detect (and annotate) RQs if present and generate them if missing within the context of articles (Zia and Mushtaq, 2023). Such high-quality questions can play an important role in motivating researchers to engage deeply with the study and think critically (Dickman, 2009; Walsh and Sattes, 2016; Mehta and Bhandari, 2016).

The primary objective of this study is to construct a pipeline to detect and annotate RQs within articles, subsequently leveraging this annotated data to formulate RQs for articles that lack them. This pipeline not only aids in annotating RQs, but also serves as a tool for formulating RQs. The proposed pipeline has three main components: **RQ extraction**, **RQ detection**, and **RQ generation**.

The **RQ extraction** component tries to identify a set of easily recognizable RQs. The primary application of this component is to annotating RQs in research articles and compiling an RQ dataset. The goal of the **RQ detection** component is to detect RQs in articles with well-defined RQs. The extracted compilation of RQs from the RQ extraction step serves as a reference dataset, which is used to train a sentence classifier to tag sentences within articles as containing an RQ or not. These RQs found in a corpus of articles by this trained model are used to augment the set of RQs extracted by the extraction component. The **RQ generation** component aims at generating RQs for articles lacking them. To build this component, the augmented dataset resulting from the RQ detection step is employed to fine-tune a set of text generation models, including Large Language Models (LLMs). By integrating all these components together, we establish a comprehensive framework to extract, identify, and generate RQs.

In summary, this study investigates the following research questions:

RQ1: How effective is a text classification approach in detecting different research question patterns in research articles?

RQ2: How do different LLMs perform on the task of generating well-defined research questions?

RQ3: How does a unified pipeline combining research question extraction, detection, and generation components perform in forming well-structured research questions for scientific articles?

We evaluate the performance of each component

Proceedings of the 31st International Conference on Computational Linguistics, pages 8573–8588
January 19–24, 2025.

on a set of articles with their RQs. For the RQ detection task, we evaluate its performance in distinguishing RQs from non-RQ sentences. Our results show that the models trained on the dataset resulting from the RQ extraction component can effectively detect RQs. We further improve the performance of this model using an Active Learning (AL) pipeline integrating LLMs into the data annotation process. Our results on the RQ generation task also indicate the capability of the fine-tuned LLMs in generating high-quality RQs for research articles.

2 Related Work

Our work touches upon several related works from different aspects: question answering and generation, RQ formulation, and fine-tuning LLMs.

2.1 Question Asking and Answering

The ability to ask and formulate a good question is an important skill that facilitates critical reasoning, leading to improved thinking and understanding (Stanlick and Strawser, 2015). Posing high-quality questions motivates researchers to think critically (Dickman, 2009; Walsh and Sattes, 2016; Mehta and Bhandari, 2016). While there is not much work on the automatic generation of such high-quality questions, there has been a lot of work around answering questions in the scientific domain (Chen and Yih, 2020; Gupta and Gupta, 2012; Lehnert, 2022; Mollá and Vicedo, 2007; Voorhees, 2001). In this work, we reverse this process and we aim to build a pipeline to generate engaging and interesting questions based on a given scientific article that are answerable by the article.

2.2 Automatic Question Generation

Automatic Question Generation (AQG) is the process of creating questions from textual content using computational methods or AI without human intervention. AQG techniques have been developed to meet the growing demand for high-quality questions, notably in educational settings where questions were often generated and ranked using logistic regression models trained on tailored datasets, significantly enhancing their acceptance rate by the annotators (Kurdi et al., 2020; Heilman and Smith, 2010). Recent surveys highlight a shift towards semantic information and transformer-based models that improve accuracy and efficiency by leveraging self-attention mechanisms (Lu and Lu, 2021; Ning et al., 2023). These models are used for various applications, such as data augmentation in Question Answering (QA) systems or reading comprehension tasks. AQG systems now predominantly use statistical methods over rule-based templates. Although there is no standard evaluation metric, automatic and human evaluations remain common.

2.3 Research Question Formulation

Ratan et al. (2019) defined the characteristics of a good RQ by the acronym "FINERMAPS", which stands for *Feasible*, *Interesting*, *Novel*, *Ethical*, *Relevant*, *Manageable*, *Appropriate*, *Potential value and publishability*, and *Systematic*. Detailing the problem statement, refining the issue under study, adding focus to the problem statement, guiding data collection and analysis, and setting the context of the research are detected as the key steps for defining a good RQ. They also highlighted the importance of developing a hypothesis that gives insight into the RQ, is testable and measurable, has a logical basis, and follows the most likely outcome. Finally, they noted that having a grounded interest in the RQ, conducting a comprehensive literature review, and having a deep understanding of the specific area or problem to be investigated are important keys to forming a good RQ. In this work, we adapt and use a subset of these characteristics to evaluate the generated RQs.

2.4 Fine-tuning Large Language Models

LLMs stand out for generating human-like text, understanding context, and performing complex language tasks. Fine-tuning has proven effective in optimizing them for specific domains (Dong et al., 2024; Huang et al., 2023). In this research, we use specific models from the BERT series for the RQ detection task and from the T5, Mistral, and Llama series for the RQ generation task due to their open-source availability.

Meta's Llama models (Touvron et al., 2023a) (Touvron et al., 2023b), trained on public datasets, outperform GPT-3 (Brown et al., 2020) in most benchmarks while using substantially lower parameters, and Llama 3 (Dubey et al., 2024) closes the gap with GPT-4 (OpenAI et al., 2024). Mistral 7B (Jiang et al., 2023) stands out for its innovative approach to complex language processing, but performs worse in linguistic accuracy in comparison to ChatGPT and Llama models (Hou and Lian, 2024). BERT (Devlin et al., 2019), with its bidirectional context understanding, excels at tasks like question answering. T5 (Raffel et al., 2020), by framing tasks as text-to-text, performs well across multiple NLP problems.

3 Methods

In this section, we commence by defining what constitutes an RQ in this work, after which we describe our pipeline for extracting, detecting, and generating such RQs.

3.1 Research Question Definition

An RQ within a study serves as a statement that summarizes the goal of the study or identifies the main problem that the study aims to answer with a potential solution. It may be framed as a question or a declarative statement. Based on this definition, we define a snippet of text to "contain an RQ" if an RQ can be constructed solely using the information present in that snippet.

Based on the characterization of RQs discussed by Ratan et al. (2019), a good RQ possesses the "FINERMAPS" characteristics. For our research, we simplify these characteristics to "FIRE", standing for *Feasible*, *Interesting*, *Relevant*, and *Ethical*. *Feasible* means that the question or problem should be solvable and able to be realistically answered. *Interesting* implies that the RQ should engage and curiosity of researchers. *Relevant* requires the RQ to be aligned with the topic of the article and to raise a question or problem that the article is seeking to address. We assess relevance based on the RQs connection to the specific content and focus of the article. Lastly, *Ethical* mandates the RQ to adhere to ethical standards and guidelines in research and respect the rights, well-being, and privacy of all individuals or groups involved. We exclude the other characteristics primarily because they either overlap with the selected characteristics or are difficult to measure. For instance, *Manageable* is essentially synonymous with *Feasible*, and assessing attributes like *Novel* or the version of *Relevant* provided by Ratan et al. (2019) would require analyzing the relevant literature review for each article, which is beyond the scope of this study.

3.2 Research Question Pipeline

The goal of the RQ pipeline is to seamlessly handle the entire process of identifying or generating an appropriate RQ for a given article. The framework takes an article as input and either extracts a suitable RQ from the text or generates one if none can be found. Figure 1 illustrates the architecture of our pipeline. The process begins with RQ extraction, where we extract RQs from articles using a set of predefined regular expressions. If a sentence in the article matches one of these patterns, the pipeline concludes having successfully extracted the RQ. If no patterns match, the pipeline proceeds to the RQ detection phase. The RQ detection component classifies a context of three sentences as containing an RQ or not. Consequently, the classifier processes each three-sentence segment of the article. If an RQ-containing context is identified, it is reformulated into an RQ using the RQ generation component. If no explicit RQs are detected in the text, we use sections where RQs are frequently introduced, such as "Abstract" or "Introduction". [1]

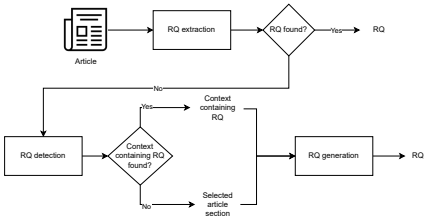


Figure 1: Overview of the RQ processing pipeline: extraction, detection, and generation.

3.3 Research Question Extraction

The first step for training RQ detection models is to create a dataset for this task. Manually creating large-scale datasets is resource-intensive. Therefore, we create a weakly-labeled dataset using a set of regular expressions. The goal is to extract RQs that are easily identifiable within the text to form an initial dataset. To do this, we first perform an extensive qualitative analysis of the structure of RQs on a set of articles. In doing so, we try to specify the commonalities of RQs and then convert such commonalities to a set of regular expressions. These regular expressions are subsequently used to detect RQs from a large set of articles. As a result, these extracted RQs will serve as our initial corpus to train RQ detection models, which will accordingly make the identification of more complex RQs possible.

For the purpose of RQ extraction, we match the article text with regular expressions (regex patterns) that are likely to contain or be followed by an RQ. For example, a phrase such as *"...aim of this study is..."* is expected to be followed by an RQ. To gather negative samples, we aim to find non-RQ sentences that closely resemble RQs to enhance our classification models' ability to accurately distinguish between the classes. To achieve this, we select the negative sentences close to where the RQs are extracted, specifically within the range of 20 to 10 sentences before and after the RQ, while ensuring that they do not contain any RQ patterns.

[1] As we show in Appendix B, RQs appear most frequently in the "Abstract" or "Introduction" sections" of the articles. Therefore, we use these sections to generate an RQ when an RQ-containing context is not identified.

Furthermore, we extract negative samples based on regex patterns, similar to the approach used for RQ extraction. Occasionally, it is observed that articles refer to RQs from other studies, particularly in the "Related work" or "Background" sections. In such instances, we classify those sentences as negative and refer to them as negative RQs, since they do not present an RQ from the article itself. Identifying and labeling these cases as negative samples is crucial, as it helps the model to more accurately distinguish genuine RQs within the article and improves its ability to detect real RQs in shorter texts. The details of the regex patterns are given in Appendix A.

3.4 Research Question Detection

The RQ detection component identifies RQs in an article by dividing it into three-sentence contexts and determining if each contains an RQ. To achieve this, a sentence classifier is trained on data from the RQ extraction process. The SciBERT model (Devlin et al., 2019) is fine-tuned for sequence classification, with three-sentence contexts used as input, where the potential RQ is placed in the middle sentence. Incorporating the surrounding context of a sentence provides more information, leading to a more accurate and effective model for RQ detection. For half of the data, we remove the RQ patterns (the strings that matched the regex pattern in RQ extraction) to maintain a level of complexity in the training data that encourages the model to develop a deeper understanding of what constitutes an RQ. To improve the generalizability of the model, we further extract negative samples using an Active Learning (AL) process.

Active Learning The negative samples extracted for RQ classification are randomly selected from contexts adjacent to where the RQs are located within the articles. To enhance the robustness and generalization of our model, we need more hard negative samples.

To extract hard negatives, we devise an algorithm using the AL framework: we first train a classifier on an initial dataset, which is then used to classify a new, unseen set of samples. The samples detected as positive (i.e., containing an RQ) by the model are subsequently fed into an LLM, accompanied by specific instructions, to verify whether an RQ is indeed present in the given context (a detailed description of the prompt can be found in Appendix D). We then retain the samples that the LLM identifies as negative. We consider these samples as hard negatives, as they closely resemble texts containing an RQ but are incorrectly flagged as positive by the initial RQ detector. Next, the hard negative samples identified

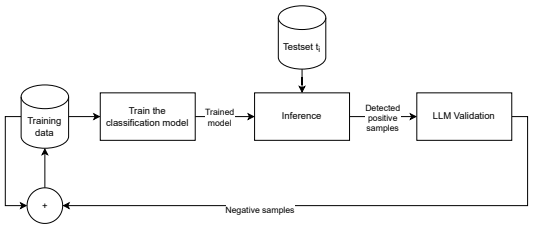


Figure 2: Active Learning process to identify and refine negative samples for RQ detection.

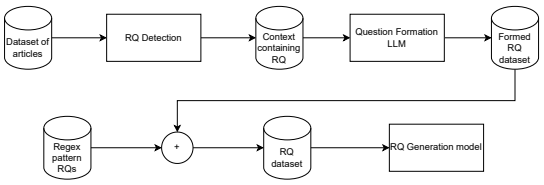


Figure 3: Workflow for generating RQs using detected contexts and LLM fine-tuning.

by the LLM are incorporated into our training set and labeled as negative. This process of training and extracting hard negatives is iterative, allowing us to continually expand the dataset with additional hard negatives over multiple training cycles. The primary aim is to develop a robust dataset enriched with high-quality negative samples, which are then used to train a lightweight classification model. This model is more efficient and cost-effective than using LLMs for classification. Figure 2 shows the process of AL for extracting hard negative samples. We repeat the AL loop four times in our pipeline.

3.5 Research Question Generation

The RQ detection method focuses on identifying explicit RQs or contexts that closely resemble RQs within articles. However, some articles may lack clearly defined and structured RQs or contexts that include RQs. To address this gap and ensure that all articles have well-formulated and precise RQs, it becomes necessary to generate these RQs for them.

Our RQ generation approach uses the contexts and RQs detected by the RQ detection method to train models that, given a context within an article, generates an RQ describing the context in a well-defined question-like format.

We prepare a dataset for RQ generation by running an inference of the RQ detection model (enhanced via the AL process) on a set of articles and extracting the positive detections. This dataset comprises contexts that include an RQ, but these contexts often lack well-formatted RQs necessary to train an RQ generation model. These contexts are transformed into well-defined RQs prior to

model training. We use an LLM to convert the detected contexts into well-defined RQs. The process involves feeding the context into the LLM with specific instructions to generate questions based on the context. These instructions include a description of the task and a definition of RQs. Assuming we have a set of articles D in our dataset, we first extract $C = Model_{AL}(D)$ as the set of contexts containing RQs detected by the model from the AL experiment. The set of RQs is formed as $R_{formed} = LLM(C)$. For a detailed description of the prompt used, refer to Appendix E.

In addition to generating RQs for articles with detected RQs, we also extract a set of question-type RQs from those obtained through regex patterns. These extracted RQs are incorporated into the dataset, resulting in a comprehensive collection of question-type RQs that enhance the dataset's robustness and diversity. Assuming that R_{regex} is the set of question-type RQs extracted using regex patterns, the combined set of question-type RQs is: $Q = R_{formed} \cup R_{regex}$.

To have a model that can generate an RQ from an article, we need to use a relatively big and important part of it as input to the model. Therefore, to train our model, we extract the specific paragraphs or sections that contain the original RQs or the parts of the text from which the RQs are formed. We define P and S as the set of paragraphs and sections (respectively) in D that contain the text from which we found or formed the RQs in Q. After constructing this enriched dataset (Q, P, S), we proceed to fine-tune several LLMs for the task of RQ generation. We consider two types of input tasks: paragraph-level input, where P serves as the input, and section-level input, where S serves as the input. In both cases, the output of the models is Q, the set of well-formed RQs. The process of RQ generation is shown in Figure 3. For a detailed description of the prompt used, refer to Appendix E.

4 Experimental Setup

4.1 Article Corpus

As the article corpus, we use the full text of articles published in peer-reviewed journals from Elsevier's ScienceDirect[2] database. This database contains over 19 million full-text articles published in different science domains. We specifically consider a single domain, Computer Science (CS), to narrow the range of RQ possibilities and to maintain consistency in their formulation. There are in total 555,688 articles in the CS domain with full text accessible.

[2] https://www.sciencedirect.com/

4.2 Benchmark Set

For evaluation, we designate a subset of the extracted RQs from our dataset as our standard test set, though its proximity to the training data may introduce bias. To tackle this problem, we create an additional test set consisting of 40 articles with manually identified RQs. We refer to this as the benchmark set. These articles are randomly selected and are completely distinct from the data used for RQ detection. To ensure accuracy and consistency, four data scientists, experienced in working with scientific articles and conducting scientific research, particularly in the CS domain, meticulously extract all RQs from these articles. The findings are cross-checked to resolve any discrepancies. A total of 159 RQs are identified in these articles, which serve as the gold standard for evaluating the performance of our models across various experiments.

4.3 Experimental Setup

RQ Detection For training different RQ detection methods, we use a dataset of 27,137 articles containing RQs and negative samples resulting from the method described in Section 3.3. We take 20% of the samples for testing and split the remaining samples into sets of 80% train and 20% validation.

Next, we separate the positive RQ and negative RQs to ensure that all sentences from a single article remain within the same training, validation, or test set, preventing any potential overlap between sets. We end up with 299,431 train, 75,117 validation, and 94,002 test data points. Out of 299,431 train samples, $n_p = 17,367$ are positive. Then, based on the *training negative/positive proportion (TNP)* value, we sample $n_n = \text{TNP} \times n_p$ negative data. This results in a total of $n_p \times (\text{TNP}+1)$ samples.

We further experiment with adding the samples containing negative patterns to the train data. For this experiment, there are 30,807 data points achieved using the method outlined in Section 3.3. We denote the inclusion of negative pattern data with the *WNEG* variable. Consequently, the total number of training samples is given by the following expression:

$$n_p \times (\text{TNP}+1) + \text{WNEG} \times 30{,}807. \quad (1)$$

We evaluate the models on our test split and the benchmark set. For the evaluation on the test set, we consider *accuracy*, *precision*, *recall*, and *F1* metrics. The latter three metrics are macro averages over the positive and negative labels. For the evaluation on the benchmark set, we consider *precision*, *recall*, and *F1* metrics on positive labels only.

We use a batch size of 32 for our experiments, a warm-up step of 400, and a weight decay of 0.01.

We train the models for 2 epochs and use a learning rate of 3×10^{-7}.

Active Learning For the initial model training of the AL algorithm, we take 5,000 training samples as the initial training set. We train the classifiers with TNP $= 1$ and a learning rate of 3×10^{-7} for 30 epochs. We use *GPT-3.5-turbo* for validating RQs, with its temperature, frequency penalty, and presence penalty values set to zero. We run the AL algorithm for four iterations, fine-tuning a SciBERT model for RQ classification each time. We consider a large test set consisting of 95,848 data points for all experiments, along with separate smaller test sets for each iteration.

RQ Generation For RQ generation, we consider 13,359 articles containing 940,279 three-sentence contexts as the data on which we perform RQ detection. These articles are selected from a larger set that excludes those identified using regex patterns for RQ extraction. We employ the AL experiment model after the final training iteration to perform RQ detection on this dataset. After applying the model, we identify 7,116 positive samples that contain RQs from 3,922 articles. These 7,116 three-sentence contexts are then given to *GPT-3.5-turbo* to form RQs. Temperature, frequency penalty, and presence penalty are set to zero. We then extract RQs in question format from our set of RQs containing regex patterns. As a result of this process, we extract a total of 7,258 RQs. We then add this data to the previously formed RQs dataset obtained from the RQ detection and formation components, resulting in a combined total of 14,374 RQs. For each RQ, we also extract the corresponding paragraph and the smallest subsection of the article in which the RQ was found, to be used as input for our RQ generation models.

We conduct our experiments on *Flan-T5*[3] (Chung et al., 2024), *Mistral 7B*[4](Jiang et al., 2023), and *Llama-3-8B* [5]. We use *METEOR*, *BLEU*, *ROUGE-1*, *ROUGE-2*, *ROUGE-L*, *ROUGE-LSUM*, and *BERT scores* to evaluate these models. For each model and input type, we conduct two experiments: one without any training (evaluating the zero-shot model) and one where we fine-tune the model for 2 epochs. We take 15% of the data as the test split, then divide the remaining data into 85% training and 15% validation sets for all the experiments.

For the *T5* model, we set a warm-up ratio of 0.05, a weight decay of 0.01, and a learning rate of 3×10^{-5}. The model is trained with a maximum generation length of 512 tokens. We evaluate the model every 500 steps, using the *ROUGE-L* metric to select the best model from the evaluations. We select a maximum length of 512 tokens for paragraph inputs and 2048 tokens for section inputs (refer to Appendix H). Respectively, we use a training batch size of 4 and 1.

For *Llama-3-8B* and *Mistral 7B*, we use the Unsloth library[6], which enables fast and low memory-intensive fine-tuning of LLMs. We utilize QLoRA (Dettmers et al., 2024) to use and fine-tune the LLMs efficiently on a single GPU, along with 4-bit quantized model weights that we access from the same library. Similar to the *T5* model, we use a maximum token length of 512 for paragraph inputs and 2048 for section inputs for the tokenizer, a learning rate of 3×10^{-5}, and a warm-up ratio of 0.05. We use a training batch size of 4, a weight decay of 10^{-4}, and 8-bit AdamW (Loshchilov and Hutter, 2019) optimizer. We choose *ROUGE-L* as the metric to select the best model among evaluations, which we do every 500 steps. Finally, we use similar but slightly different prompt templates for fine-tuning and evaluation (RQ generation) for the LLMs, which we elaborate on in Appendix F.

RQ Pipeline To evaluate our pipeline, we use an approach leveraging an LLM with an extensive input capacity, namely *GPT-4*, allowing the full text of an article to be processed at once. We design a prompt that instructs the LLM to evaluate the proposed RQ of our pipeline for the considered article based on the four *FIRE* criteria (feasible, interesting, relevant, and ethical). In the prompt, we define the task and provide explanations of these characteristics. We further include instructions to assign a score of 1, 2, or 3 for each criterion that reflects the quality of the proposed RQ in each of the *FIRE* aspects. For a detailed description of the prompt, refer to Appendix G. For RQ generation, the "Abstract" sections of the articles are used as input. The pipeline is evaluated on 50 chosen papers using *GPT-4* as its evaluator, with its temperature set to zero for a more deterministic outcome. The generated RQs are then evaluated against the corresponding articles by running our custom-designed prompt through the LLM, yielding four scores based on the *FIRE* criteria.

5 Results

In this section, we present the results of the RQ detection and generation components as well as the RQ pipeline.

[3] https://huggingface.co/google/flan-t5-base
[4] https://huggingface.co/unsloth/mistral-7b-bnb
[5] https://github.com/llama3/tree/main
[6] https://github.com/unslothai/unsloth

Experiment	w-neg	TNP	Test Split 4.3				Benchmark Set 4.2		
			Acc	P	R	F1	T_P	T_R	T_{F1}
N2	No	2	96.85	83	94	88	78	31	44
N4	No	4	98.07	89	95	92	65	16	26
N8	No	8	99.78	100	99	99	41	8	14
W0	Yes	0	95.42	86	90	88	82	26	39
W1	Yes	1	98.99	96	99	97	60	18	28
W2	Yes	2	96.11	80	92	85	88	38	54
W4	Yes	4	99.44	98	99	98	21	4	6
W8	Yes	8	99.13	98	97	98	23	4	7

Table 1: RQ classification experiments. **w-neg** (with negative) denotes whether the negative samples extracted from negative patterns were used for training. **TNP** (training negative/positive proportion) shows the ratio of negative to positive samples. The experiment name is defined as {w-neg}{TNP}. Metrics are expressed as percentages (%).

5.1 Research Question Detection

Table 1 shows the results of different experiments for the RQ classification task. We observe overfitting patterns in experiments except for N2, N4, W0, and W2, as the metrics for the test split are considerably high, in contrast to the benchmark set. Through various experiments, we find that using high TNP values, such as 4 or 8, consistently worsens the model's performance, sometimes resulting in very low precision, recall, and F1 scores. Although a high TNP value more accurately reflects real-world conditions——where the ratio of RQ sentences to non-RQ sentences is significantly low——it negatively impacts the model's learning ability. For instance, in our benchmark set, we identified 159 RQs in 8,178 sentences, equating to a 1.94% ratio. Consequently, the TNP value for this set exceeds 50.

While our initial objective was to experiment with high TNP values to emulate real-world data distributions, we discovered that the model learns better with a more balanced ratio of negative to positive samples, such as a TNP value of 2.

In experiments with high TNP values, such as 4 or 8, incorporating the negative data extracted based on regex patterns disrupts the training process. This results in an excessive number of negative samples in the training data, causing the model to focus on identifying negative samples rather than learning to detect RQs. Consequently, experiments W4 and W8 achieve very low precision, recall, and F1 scores on the benchmark set.

For low TNP values, adding negative data based on regex patterns can noticeably improve performance. In the W0 experiment, the TNP value of 0 means that there are no ordinary negative samples used in the training, and only the negative data based on regex patterns are incorporated. In this case, the model achieves a precision of 0.82 on the benchmark set, which is higher than that of N2, an experiment with a similar total number of negative samples. The recall and F1 score are lower, but given our interest in higher precision, using negative data based on regex patterns could prove more useful here. Maintaining ordinary negative samples while incorporating the negative data based on regex patterns in experiment W2 results in significantly higher precision, recall, and F1 score on the benchmark set, suggesting that using both types of negative data can further improve performance. Therefore, we conclude that incorporating negative data based on regex patterns is beneficial.

Overall, we observe that the model based on the W2 setting performs the best, achieving 88.4% precision, 38.36% recall, and 53.5% F1 score on the benchmark set. Also, the results indicate that a TNP value of 2 is the most effective.

The results of the AL experiments can be seen in Table 2. We observe that after the second training, the model's precision improves significantly, and although the recall decreases, the overall F1 score improves by 2%. Across all experiments, precision increases and recall drops for both the test split and the benchmark set, except for the benchmark set in the fourth training. After the third training, the model achieves the highest precision of 93%. Despite a decrease in recall and F1 score, this high precision is beneficial for detecting RQs that are not typically identified through regex patterns, aiding in training RQ generation models in other experiments. After the fourth training, the test split metrics remain constant, while the benchmark set metrics decline. This is due to the fact that adding negative samples to the training data reduces the number of positive samples detected, as shown in the "pos" column of Table 2. Consequently, after the fourth iteration, the model detects fewer true positive samples in the benchmark set, but the number of false positives remains the same, resulting in lower precision.

Overall, we find that AL significantly improves the model's performance in terms of precision, making it highly useful for our use case. Additionally, our results show that using an LLM for validating whether an RQ is present in a context is practical and beneficial.

5.2 Research Question Generation

Table 3 presents the results of RQ generation experiments, comparing the performance of models using paragraph and section inputs. For T5, we observe that additional training significantly enhances performance compared to the zero-shot model. The fine-tuned T5 models achieve significantly higher

Train	# Train	Test Split 4.3				Benchmark Set 4.2			Large Test Set 4.3
Iteration	Samples	Acc	P	R	F1	T_P	T_R	T_{F1}	pos
1	5000	96.36	87	96	91	38	25	30	3.13
2	5389	96.58	88	95	91	81	20	32	1.32
3	5617	96.85	90	94	92	93	13	22	0.85
4	5710	96.93	90	94	92	92	11	19	0.75

Table 2: Results of the Active Learning (AL) experiment. **pos** denotes the percentage of samples detected as positive. Metrics are expressed as percentages (%).

performance compared to the Llama-3 and Mistral models, with the one using section inputs being the overall best model across all metrics. Notably, in the zero-shot setting, the T5 models achieve significantly higher BERT scores than Mistral and Llama-3, suggesting they have a better semantic understanding. Although T5 is pre-trained with a maximum token length of 512, it adapts effectively to a maximum token length of 2048, even achieving a better performance.

5.3 Research Question Pipeline

Table 4 presents the results of the evaluation for our pipeline. Due to the novelty of the task, no established baseline method is available for direct comparison. All test samples achieved a full score of 3 out of 3 for both the *Feasibile* and *Ethical* criteria. For the *Relevant* criterion, only three samples scored a non-perfect 2, while 11 samples received a score of 2 for the *Interesting* criterion. None of the samples achieved a score of 1 in any criteria. These results indicate that the pipeline is highly effective in identifying or generating well-defined RQs.

6 Conclusion

In this study, we have addressed the critical need for efficient and reliable methods to detect and generate Research Questions (RQs) in academic articles, particularly within the domain of computer science. We proposed a comprehensive framework for extracting, detecting, and generating RQs for scientific articles. For RQ extraction, by developing a rule-based methodology using a set of regular expressions, we were able to extract a subset of articles with common RQ patterns. We then proposed RQ detection, a method to identify RQs in the article by fine-tuning a BERT-based classifier to detect if any three-sentence pieces of the article contain an RQ. We devised the Active Learning (AL) algorithm to enhance the performance of our model. The results demonstrated that using AL significantly improved the precision of the classifier. We achieved a high performance on our proposed benchmark set, validating the effectiveness of our detection method.

Regarding RQ generation, using the dataset of RQs from the extraction and detection components, we trained several Large Language Models (LLMs) to generate RQs from paragraphs or sections of articles. We found Flan-T5 to perform the best when trained on section inputs, excelling across all evaluation metrics and being able to closely predict the RQs from the given section of the article. Consequently, by combining the components of RQ extraction, detection, and generation, we developed a novel end-to-end framework that either identifies or generates a suitable RQ from a given article. We provided a definition for RQs and specified evaluation criteria using the FIRE characteristics. We then demonstrated the strong performance of our framework through our proposed LLM-based evaluation method.

In conclusion, this study offers a robust solution for the detection and generation of RQs in scientific articles. By integrating advanced text classification and generation techniques, we have created a framework that enhances the identification of RQs, simplifying the process of reading and comprehending scientific articles for researchers and students. The framework lays the foundation for future advancements in managing and understanding scholarly literature. Future work could enhance the framework, generalize it across domains, or evaluate it on larger benchmarks.

Limitations

While our framework has shown to be capable of detecting and generating well-defined RQs, several areas for future research and improvement remain:

Generalizability Across Domains: Extending the framework to other domains beyond computer science to assess its generalizability and adaptability is crucial. Testing these methods on articles from diverse fields, including medicine, social sciences, and engineering, would be necessary.

Exploring Different Context Lengths for RQ Detection: For RQ classification, we included the surrounding context of the RQ to provide the model with additional information, making it easier to determine whether an RQ is present. In our experiments, we used a context length of three sentences, but we did not test with longer contexts. Future research can explore different context lengths to assess their impact on the performance of detection models.

Improving Data Used for Training and Evaluation: The data we gather for RQ detection and generation is primarily based on regex patterns. This introduces potential bias in the data. Although we used techniques such as synthesizing data with LLMs,

Experiment	METEOR	BLUE	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-LS	BERT-P	BERT-R	BERT-F1
T5-para-ep0	23.18	10.47	28.99	14.95	25.22	25.35	85.45	84.63	84.97
T5-para-ep2	57.51	40.62	62.37	48.42	57.36	57.57	93.39	92.98	93.17
T5-section-ep0	20.79	4.64	27.00	11.27	22.19	22.50	86.51	86.60	86.49
T5-section-ep2	**65.89**	**45.21**	**70.44**	**58.61**	**66.06**	**66.26**	**94.92**	**94.38**	**94.64**
Mistral-para-ep0	31.44	9.79	20.76	13.81	18.44	18.49	63.77	68.04	65.82
Mistral-para-ep2	39.14	12.61	27.02	19.19	24.25	24.21	65.82	72.89	69.13
Mistral-section-ep0	27.11	10.15	22.38	15.72	20.27	20.40	56.35	59.67	57.94
Mistral-section-ep2	32.19	11.06	22.88	15.24	20.06	20.67	58.27	64.20	61.05
Llama3-para-ep0	28.26	12.47	27.41	17.96	24..42	24.45	63.73	65.22	64.42
Llama3-para-ep2	37.06	10.27	23.16	15.84	20.60	20.48	70.30	75.70	72.88
Llama3-section-ep0	30.23	13.96	29.13	19.99	26.32	26.19	66.22	68.09	67.09
Llama3-section-ep2	35.85	11.39	26.58	17.62	22.98	23.40	69.83	74.19	71.90

Table 3: RQ generation experiments. Experiment names specify the training model (T5, Mistral, or Llama-3), input type (paragraph or section), and number of fine-tuning epochs (*ep0* for zero-shot and *ep2* for two epochs). All values are expressed as percentages (%).

Criterion	Score
Feasible	100
Interesting	92.67
Relevant	98
Ethical	100

Table 4: Pipeline evaluation results based on the FIRE criteria.

there remains room for improvement in extracting more diverse RQs from articles. For evaluation, we created a benchmark set; however, it is limited to 40 papers. Expanding this effort by manually annotating RQs for a larger set of articles would greatly enhance the dataset for a more robust evaluation.

Improving RQ Generation in the Pipeline: Currently, the pipeline uses the "Abstract" section of articles as input for the RQ generation component. However, incorporating the "Introduction" section could offer improvements. A study could be conducted to determine the optimal sections or combinations of sections that yield higher-quality generated RQs. Using the full article as the input context is yet another possibility.

Developing an RQ Ranking Framework: The RQ detection component can be employed to detect multiple RQs as opposed to the current pipeline, which is limited to one. Consequently, multiple RQs can be generated for an article. Beyond detection and generation, developing a ranking framework to assign scores to RQs based on relevance and informativeness would be beneficial. This tool could measure how relatable the detected or generated RQs are to the article's content. Specifically, it can help identify the best RQ and suggest improvements if the detected RQ is not optimal.

References

Tom B. Brown et al. 2020. Language models are few-shot learners. In Proceedings of the 34th International Conference on Neural Information Processing Systems.

Cristina Candal-Pedreira, Julia Rey-Brandariz, Leonor Varela-Lema, Mónica Pérez-Ríos, and Alberto Ruano-Ravina. 2023. Challenges in peer review: how to guarantee the quality and transparency of the editorial process in scientific journals. Anales de Pediatría (English Edition), 99(1):54–59.

Danqi Chen and Wen-tau Yih. 2020. Open-domain question answering. In Proceedings of the 58th annual meeting of the association for computational linguistics: tutorial abstracts, pages 34–37.

Hyung Won Chung et al. 2024. Scaling instruction-finetuned language models. Journal of Machine Learning Research, 25(70):1–53.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: efficient finetuning of quantized llms. In Proceedings of the 37th International Conference on Neural Information Processing Systems.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In North American Chapter of the Association for Computational Linguistics.

Nathan Eric Dickman. 2009. The challenge of asking engaging questions. Currents in Teaching & Learning, 2(1).

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2024. How abilities in large language models are affected by supervised fine-tuning data composition. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 177–198.

Abhimanyu Dubey et al. 2024. The llama 3 herd of models. Preprint, arXiv:2407.21783.

Poonam Gupta and Vishal Gupta. 2012. A survey of text question answering techniques. International Journal of Computer Applications, 53(4).

Michael Heilman and Noah A Smith. 2010. Good question! statistical ranking for question generation. In Human language technologies: The 2010 annual conference of the North American Chapter of the Association for Computational Linguistics, pages 609–617.

Guangyu Hou and Qin Lian. 2024. Benchmarking of commercial large language models: Chatgpt, mistral, and llama. PREPRINT (Version 1) available at Research Square.

Kai Huang, Xiangxin Meng, Jian Zhang, Yang Liu, Wenjie Wang, Shuhao Li, and Yuqing Zhang. 2023. An empirical study on fine-tuning large language models of code for automated program repair. In 2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE), pages 1162–1174.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. Preprint, arXiv:2310.06825.

Romendro Khongsdier. 2007. Bio-cultural approach: The essence of anthropological study in the 21st century. Anthropologist, 3:39–50.

Ghader Kurdi, Jared Leo, Bijan Parsia, Uli Sattler, and Salam Al-Emari. 2020. A systematic review of automatic question generation for educational purposes. International Journal of Artificial Intelligence in Education, 30:121–204.

Wendy G Lehnert. 2022. The process of question answering: A computer simulation of cognition. Routledge.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In International Conference on Learning Representations.

Chao-Yi Lu and Sin-En Lu. 2021. A survey of approaches to automatic question generation: from 2019 to early 2021. In Proceedings of the 33rd Conference on Computational Linguistics and Speech Processing (ROCLING 2021), pages 151–162.

Bharati Mehta and Bharti Bhandari. 2016. Engaging medical undergraduates in question making: a novel way to reinforcing learning in physiology. Advances in Physiology Education, 40(3):398–401.

Diego Mollá and José Luis Vicedo. 2007. Question answering in restricted domains: An overview. Computational Linguistics, 33(1):41–61.

Zhansheng Ning, Prasanth Venugopal, Gert Rietveld, and Thiago Batista Soeiro. 2023. Transformer neural network for early battery capacity prediction based on electrochemical impedance spectroscopy. In 2023 IEEE International Conference on Metrology for eXtended Reality, Artificial Intelligence and Neural Engineering (MetroXRAINE), pages 329–334. IEEE.

OpenAI et al. 2024. GPT-4 technical report. Preprint, arXiv:2303.08774.

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

Simmi K Ratan, Tanu Anand, and John Ratan. 2019. Formulation of research question–stepwise approach. Journal of Indian Association of Pediatric Surgeons, 24(1):15.

Nipun Sadvilkar and Mark Neumann. 2020. Pysbd: Pragmatic sentence boundary disambiguation. In Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS).

Nancy A Stanlick and Michael J Strawser. 2015. Asking good questions: Case studies in ethics and critical thinking. Hackett Publishing.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. Preprint, arXiv:2302.13971.

Hugo Touvron et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. Preprint, arXiv:2307.09288.

Ellen M Voorhees. 2001. The trec question answering track. Natural Language Engineering, 7(4):361–378.

Jackie Acree Walsh and Beth Dankert Sattes. 2016. Quality questioning: Research-based practice to engage every learner. Corwin Press.

Sadaf Zia and Muzamil Mushtaq. 2023. Assessment and content analysis of highly cited publications in reference services. Journal of Data Science, Informetrics, and Citation Studies, 2(3):203–211.

Examples matching RQ regex patterns
The following research question will guide our study: How does social media influence consumer behavior?
The goal of this research is to determine the impact of climate change on agriculture.
The purpose of our study is to explore the effects of remote work on employee productivity.
The objective of the research is to analyze the relationship between diet and mental health.
RQ1: What are the key factors affecting customer satisfaction in e-commerce?

Table 5: Examples of RQ sentences identified using regex patterns.

Examples matching hypothesis-related regex patterns
The following hypothesis is central to our study: Increased screen time negatively impacts sleep quality in teenagers.
We hypothesize that regular physical exercise can reduce symptoms of anxiety and depression.
Our hypothesis is that there is a positive correlation between employee motivation and workplace productivity.
Our main hypotheses are: 1. Dietary changes can prevent chronic diseases. 2. Mindfulness practices reduce stress levels.

Table 6: Example sentences matching hypothesis-related regex patterns.

A RQ Extraction Details

The regex pattern we consider for extracting RQs consists of four types of rules as follows:

```
following research question|
(goal|purpose|objective|aim) of
(this|our|the)
(study|research|article) is)|
RQ\d(:|.|-| -)|
research question(s?)(:| (is|are)
```

To optimize the extraction process, we filter out sections of the articles that are unlikely to contain RQs. Specifically, we exclude the following sections from the articles: “highlights”, “literature”, “related work”, “background”, “ethical considerations”, “conflict of interest”, “acknowledgments”, and “computing power”.

Table 5 shows some examples of text that matches the specified regular expression.

Moreover, we include regex patterns that capture the definition of research hypotheses, which constitute another type of RQ reformulation. Phrases such as *“We hypothesize...”* are highly likely to define the RQ of the article. The full regex consists of three types of rules:

```
following (hypothesis|hypotheses)|
(W| w)e hypothesis(s|z)e|
((T| t)he|(O| o)ur|(M| m)ain)
(hypothesis|hypotheses) (is|are)
```

Table 6 demonstrates examples of text that match the specified regular expression.

It is also possible that the hypothesis is merely a secondary theory rather than the primary aim of the article. Additionally, such patterns might include definitions of mathematical theorems, leading to false positive detections. To address this issue, we limit the extraction of hypothesis-related regex patterns to the “Abstract” and “Introduction” sections of the articles. Sentences matching the defined regular expression in these sections are highly likely to contain an RQ.

To extract negative samples based on regex patterns, we propose regex patterns similar to RQ patterns, specifically as below:

```
((T|t)(heir|hey)|
((P|p)revious|ast|rior))
(stud(y|ies)|research|
article|goal|purpose|
objective|aim|hypothesi(s|z)e|
hypothes(is|es))
```

For example, this pattern covers phrases such as “Their aim”, “They study”, “Previous studies”, and “prior hypotheses”.

B Statistics of Extracted RQs

After obtaining the article corpus in Section 4.1, we use the proposed RQ extraction method, explained in Section 3.3, to extract sentences containing RQs. We use the PySBD (Sadvilkar and Neumann, 2020) library in Python to perform sentence segmentation. This process leads to a dataset of 39,128 articles with sentences containing RQs.

Subsequently, the negative RQ extraction step based on regex patterns is performed to obtain the negative samples. Out of 555,688 articles, 30,807 are extracted that contain a sentence that matches the pattern. The combined set of positive and negative samples is used to train our RQ detection methods.

After RQ extraction, we perform an analysis to gain insight into the distribution of matched regex patterns along with the distribution of the placement of RQs across different sections of the articles.

First, we place the regex patterns presented in A into the following categories: “Research Questions”, “Objective”, “Purpose”, “Aim”, “Goal”, “Hypothesis”, and “Research Question (RQ-id)”. As shown in Figure 4, we find that the most frequent patterns are in the "Research Questions" category, containing patterns:

```
following research question|
research question(s?)(:| (is|are)
```

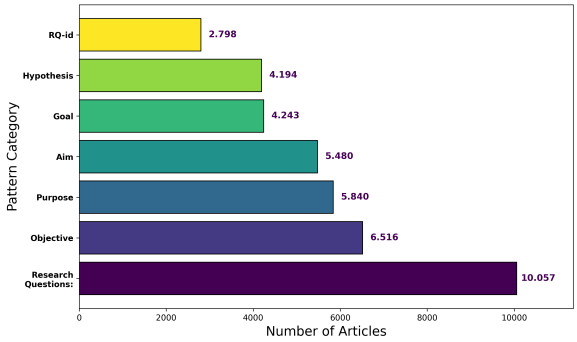


Figure 4: Distribution of RQ patterns across different categories in the dataset

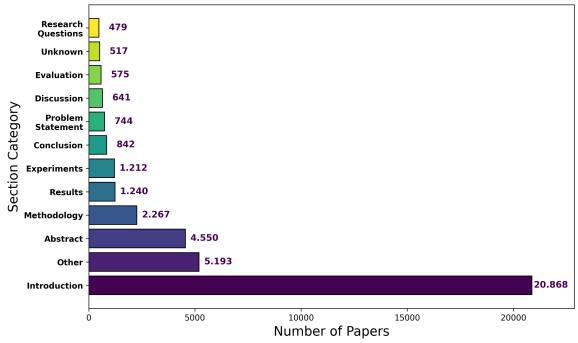


Figure 5: Sections with the highest frequency of RQs in scientific articles

. The least frequent pattern, on the other hand, is "RQ-id" (`RQ(:|.|-| -)|`).

In order to verify that our defined patterns correctly identify RQs, we manually annotate a set of extracted RQs. We randomly select a set of 30 extracted RQs for each of the 7 pattern categories, after which 3 human evaluators manually evaluate whether they are indeed RQs. We found that 187 out of 210 samples were considered correct, resulting in a precision of 0.891. While this evaluation does not consider the recall, we must also consider the purpose of the RQ pattern detection method using regular expressions. The set of detected positive and negative samples will simply be used as the training set for a model that detects RQs. This model is trained with the intent to generalize well and detect RQs beyond the scope of our previously defined exact pattern matches. We therefore argue that recall in this evaluation can be omitted.

Next, we analyze the frequency and distribution of the RQs found in different sections of the articles. To achieve this, we locate the section where each RQ is found and count the number of articles containing an RQ in that section. As authors use various titles for the same type of section, we categorize the section titles into the following categories: "Introduction", "Abstract", "Methodology", "Results", "Experiments", "Conclusion", "Problem Statement", "Discussion", "Evaluation", and "Research Questions".

As shown in in Figure 5, we find that the "Introduction" section contains the most RQs by far, with 20,868 articles. The "Abstract" section follows with 4,550 articles, and the "Methodology" section ranks third with 2,267 articles. Interestingly, 479 articles are found to have a dedicated section for their RQs, labeled as "Research questions", "Research question", or "Research questions and hypotheses", which occupy the tenth position in the ranking.

C Data Augmentation with LLM

Our dataset, constructed using the method outlined in Section 3.3, is inherently limited by a specific set of regex patterns. As a result, it may lack the diversity of RQ variations needed to effectively train a model that can generalize well. To address this issue, we aim to enrich the dataset by including RQ samples that do not necessarily conform to the predefined regex patterns. In order to achieve this goal, we propose a novel approach that leverages an LLM to generate a set of new context samples containing RQs, thereby augmenting our dataset. We utilize a specifically adjusted prompt as input to the LLM, which explains the task of generating text samples with embedded RQs. The prompt specifies the desired structure of the output and provides detailed instructions for avoiding the use of RQ patterns in the samples and circumventing certain generation restrictions. To further clarify the task for the model, we include manually selected examples of text containing RQs sourced from legitimate articles. The prompt concludes with reminders to emphasize key instructions and help the model stay focused on the defined task, even when processing a large volume of text.

For creating the training data for the RQ classification model, we use the following prompt:

```
## TASK ##

- You are an AI assistant that generates
research questions. Generate research
questions and the important context around
them, in 3 sentences, just as in a natural
scientific paper.


## INSTRUCTIONS ##

- Use the following template:
[
      {{"rq_with_context": "..."}},
      {{"rq_with_context": "..."}},
```

```
      ...
]
- Remember to make it a JSON array. Use
  double quotes every time.
- The research question is not necessarily
  a question, but can also be summarizing
  the goal of the paper.
- Use exactly 3 sentences for each example.
  The actual research question, aka the goal
  of the paper, should come in the middle
  sentence. The length of the sentences does
  not matter.
- Try to avoid obvious patterns that show
  the example contains a research question,
  such as "The objective of this study is ..."
  or "We aim to solve the following research
  question: ...".
- The research question and the context around
  it don't need to be at the start or end of a
  paragraph. It can and is encouraged to be in
  a manner that can be continued.


## EXAMPLES ##
{example}

For instance, the middle sentences in the
examples are considered research questions
even though they are not literal questions.


## REMINDERS ##

- Generate 5 examples.
- The research questions should preferably
  not be in a question format.
- Use different patterns for different
  examples. In other words, avoid similarity
  between any two examples you provide.
- The last sentence from the 3 sentences, which
  preferably comes after the research question
  and thus does not contain one, must be
  completely different among the examples.
- Remember that the 3 sentences are not
  necessarily a whole paragraph or coming
  from the start or end of a paragraph.
- The middle sentence should contain the
  research question.
- For each example try to vary sentence
  structure.
```

We use five sets of different examples in place of `{example}` in the prompt, each with a JSON array containing two samples.We use *GPT-3.5-turbo* to generate RQs with context, consisting of three sentences containing an RQ. We set the temperature to 1 in order to encourage diversity in generated samples while maintaining coherence, and we set both the frequency and presence penalty to 0. Inside the prompt, we instruct the LLM to generate five RQs with context. We do this 250 times and with five different examples inside the prompt, resulting in 6,250 RQs with contexts. After removing generated samples with errors, such as those not properly formatted in the requested JSON format, we are left with 6,174 valid samples.

In this experiment, we investigate the impact of adding LLM-generated samples to the training data for RQ classification. In our approach, we train a model once with the standard training data and once with the addition of LLM-generated samples. We then compare the performance of these models. We use the experimental setup from N2, with the data consisting of three sentence contexts, a TNP value of 2, and a learning rate of 3×10^{-7}. We conduct two sets of experiments: one using the entire training split and the LLM-generated samples, and another where we limit the data to 3,250 samples but increase the training duration. In the first setup, we have N2 and N2G experiments, with the letter G indicating the inclusion of LLM-generated samples. In the second setup, we have the N2_lim and N2G_lim experiments, similar to the previous experiments but with limited data. We evaluate the experiments on the test split and the benchmark set. For the test split, we assess accuracy, precision, recall, and F1 score, using the macro average for precision, recall, and F1 score. For the benchmark set, we evaluate precision, recall, and F1 score, but calculate these metrics based only on the positive samples. Table 7 shows the results for these experiments.

In the limited version, we take 3000 samples randomly from the train split for both N2_lim and N2G_lim. We then add 250 random samples from the train split to N2_lim and 250 LLM-generated samples to N2G_lim to keep the training data equal for both experiments. Due to the reduced data size, we increase the number of training epochs to 30. We observe that the model with LLM-generated samples achieves slightly lower performance on the test split across all metrics. However, it achieves significantly higher scores on the benchmark set for all evaluated metrics. This suggests that the model with LLM-generated samples has learned more general properties and is less overfitted to the training data.

For the N2 and N2G experiments, where all the data is included for training, we train the models for 2 epochs. As with the first set of experiments, we observe a similar pattern for both the test split and the benchmark set. Despite the ratio of LLM-generated samples to normal training data being approximately 1:50—much lower than the 1:12 ratio in the limited experiment—we still observe a significant improvement of nearly 7% in both precision and recall on the benchmark set.

Another interesting finding is that although we limited the data to only 3250 samples in the first set of experiments, the experiments achieved almost the same F1 score as in the second set of experiments. Specifically, the precision was lower, but a higher recall was observed in both normal data and data augmented with LLM. This suggests that increasing

Experiment	# Train Samples	Test Split 4.3				Benchmark Set 4.2		
		Acc	P	R	F1	T_P	T_R	T_{F1}
N2_lim	3250	96.47	82	94	87	62	35	44
N2G_lim	3000+250	95.55	78	93	84	71	42	52
N2	299431	96.85	83	94	88	78	31	44
N2G	299431+6174	96.17	80	94	86	85	38	52

Table 7: Impact of LLM-augmented data on RQ detection model performance

the training data can result in the model becoming stricter in detecting the RQs, but also more precise.

Overall, we note that augmenting the training data with LLM-generated samples can result in significantly better models and improved performance.

D RQ Validation Prompt

For RQ validation in the Active Learning experiment, the main prompt that we use is the following:

```
## TASK ##

- You are an AI assistant. Your task is to
  determine if there are any research questions
  inside a piece of text or not.
- You are given a text consisting of 3
  sentences. The text is part of a scientific
  paper for which you want to say if a research
  question can be seen in that part.
- The research question is not necessarily a
  question, but can also be one or multiple
  sentences defining a problem that the paper
  sets out to answer.


## INSTRUCTIONS ##

Perform the following actions:
1 - If the given text defines a research
    question, try to formulate the question
    or problem that the paper wants to solve.
    Do not try to guess what the research
    question might be, only formulate it if
    it is clearly found in the given text.
2 - If you could generate a well-defined
    problem statement (research question),
    answer yes.
3 - If the given text doesn't define a research
    question or you are not able to form a proper
    research question, answer no.


Now answer for the following text:

'''
{example}
'''

## REMINDERS ##
- Only answer with yes or no.
```

We replace `{example}` in the prompt with the sample that we want to validate.

E RQ Formation Prompt

For the task of RQ formation from a given three-sentece context, we experiment with the following prompt:

```
## TASK ##

- You are a scientific researcher. Your task is
  to write clear research questions for
  scientific papers.
- In order to do that, you are given a snippet
  of text that contains the goal of a research
  paper. You will rewrite this text into a
  research question.


## INSTRUCTIONS ##

Perform the following actions:
1 - summarize the problem that the paper is
    addressing.
2 - Based on that and the given text generate
    a research question that best describes
    the aim of the paper.
3 - Revise the research question. Does it
    make sense as a standalone question?
    Does it seem like a high quality insightful
    research question? Is it understandable or
    does it need some more explanations?
    Try to improve it.
4 - your output should be in the following
    format:
'''
problem summary: ...
initial research question: ...
research question: ...
'''

Now answer for the following text:

'''
{example}
'''
```

The placeholder `{example}` in the prompt is replaced by the sample from which we want to form an RQ. After the LLM generates the output in the requested format, we take the text generated after `"research question:"` as the final RQ.

F RQ Generation Prompt

For Llama-3 and Mistral LLMs we use for RQ generation, we consider the following prompt:

```
Below is a snippet of text from a scientific
paper. Write the research question that the
paper aims to answer. A research question is
a clearly articulated question that identifies
the specific issue or problem that a research
study aims to address.

### Text:
{text}

### Research question:
{RQ}
```

The placeholder `{text}` is replaced by the paragraph or section that we want to generate an RQ

from, and {RQ} is replaced by the actual RQ (label). The prompt is followed by an End-Of-Sequence (EOS) token in order to prevent the model from generating unlimited output.

For evaluation, no RQ is given in the prompt, and there would not be an EOS token in the end. The fine-tuned model is responsible for generating the RQ using the rest of the prompt. Consequently, the prompt used for evaluation is as follows:

```
Below is a snippet of text from a scientific
paper. Write the research question that the
paper aims to answer. A research question is
a clearly articulated question that
identifies the specific issue or problem that
a research study aims to address.

### Text:
{text}

### Research question:
```

G RQ Evaluation Prompt

To evaluate a proposed RQ for an article based on the FIRE criteria, we use the following prompt:

```
## TASK ##
- You are an AI evaluator specializing in
  evaluating the quality of research
  questions formulated based on the full text
  of research articles.
- You are given a scientific article and a
  research question. Your task is to evaluate
  the quality of the research question based
  on a set of quality metrics.
- You need to do the evaluation in terms of
  feasibility, interestingness, relevance,
  and ethical.
- The generated question is targeting readers
  to understand the article better. You
  penalize the generated question if it does
  not fully capture the main ideas or
  contributions of the article.

## INSTRUCTIONS ##
- Make sure you read and understand the
  question, the article, and the evaluation
  criteria carefully.
- Make sure you capture the main ideas and
  novelties of the article. Such ideas and
  novelties should be reflected in the
  research questions as well, otherwise the
  research question should get a low score.
- For each criterion (feasibility,
  interestingness, relevance, and ethics),
  you need to assign a score of 1 (low), 2
  (moderate), or 3 (high).
- Feasibility reflects that the research
  question or problem should be solvable and
  able to be realistically answered given the
  available resources and constraints.
  Feasibility score of 1 would mean that the
  question is not feasible and realistic in
  the context of the article, while a score
  of 3 would reflect the the question is
  realistic and feasible.
- Interestingness denotes that the research
  question should capture the attention and
  curiosity of researchers or practitioners
  in the relevant field. A score of 1 would
  mean that the question lacks novelty or
  relevance and fails to engage the intended
  audience, while a score of 3 would reflect
  that the question is highly engaging,
  thought-provoking, and likely to stimulate
  further inquiry or discussion in the field.
- Relevance requires the RQ to be aligned
  with the subject matter of the article and
  to raise a question or problem that the
  article is seeking to address. A score of 1
  would mean that the research question is
  completely irrelevant to the subject matter
  and goals of the article, making it
  unrelated and unimportant in the given
  context, while a score of 3 would reflect
  that the research question is highly
  pertinent to the subject matter of the
  article, directly addressing key issues or
  problmes that the article aims to explore.
- Ethical is a prerequisite condition for a
  research question to be valid, and mandates
  that the research question respects privacy
  and adheres to ethical standards and
  ↪ guidelines in research. A score of 1 would
  mean that the question raises significant
  ethical concerns or violates ethical
  standards, making the research potentially
  harmful or unethical, while a score of 3
  would indicate that the question fully
  complies with ethical guidelines, ensuring
  that the research can be conducted
  responsibly and without harm.

- Your output should be in the following
  format:
'''
Feasibility score: [1-3]
Interestingness score: [1-3]
Relevance score: [1-3]
Ethical score: [1-3]
'''


Now evaluate the following research question
in relation to the given article:


Article:
'''
{article}
'''

Research question:
'''
{rq}
'''



## REMINDERS ##
- Only answer in the requested format.
```

The placeholder {article} is substituted by the desired article, and {rq} is replaced by the RQ to be evaluated for that article.

H Supplementary RQ Generation Results

After RQ detection is performed, we reformulate detected samples into RQs to prepare the data

Detected context containing RQ	Formed RQ
By applying sentiment analysis using LSTM, the model achieved an impressive accuracy of 92% in recommending the most suitable medication for a patient's condition. The novelty of this research lies in integrating deep learning techniques, specifically LSTM, with heterogeneous networks for medication recommendation. This approach combines the power of deep learning in understanding and extracting patterns from textual reviews with the versatility of heterogeneous networks in handling diverse data types.	How can the integration of deep learning techniques, specifically LSTM, with heterogeneous networks improve medication recommendation?
The influence of water content on the mechanical characteristics and wear resistance of hyaluronic acid hydrogels was investigated and analyzed in this paper. All-atomic molecular dynamics (MD) simulation was employed to simulate the network of hyaluronic acid hydrogels with varied water content. The goal of this work is to investigate and analyze the molecular space distribution after different compression strains, the mechanical properties affected by different water content, and the wear resistance of hydrogels affected by different friction environments under the influence of different water content.	How does the water content affect the molecular space distribution, mechanical properties, and wear resistance of hyaluronic acid hydrogels under different compression strains and friction environments?
Both Internet users and the industry suffer tremendous economic losses as a result [1]. Consequently, it is imperative to develop practical solutions that provide high levels of accuracy and fast response times when it comes to detecting phishing attacks. In this study, we develop a practical method to detect phishing websites based on machine learning approaches.	What machine learning approaches can be used to develop a practical method for detecting phishing websites with high accuracy and fast response times?

Table 8: Examples of detected context containing RQs after RQ detection, along with their formed RQ by GPT-3.5-turbo.

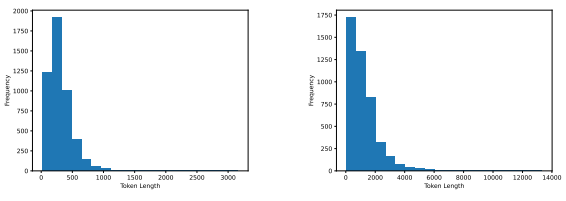
Prediction	Label
How can the concept of distance-field regression be used to accurately delineate hc from US images, and what is the effectiveness of this approach in automatic hc delineation from US images?	How can distance-field regression be used to accurately delineate hc from US images, and what are the potential benefits of this approach compared to other methods?
What are the individual trust influencing factors in social media communication?	What are the individual trust influencing factors in social media communication?
How can deep learning techniques, specifically LSTM, be integrated with heterogeneous networks to improve the accuracy and effectiveness of medication recommendations?	How can the integration of deep learning techniques, specifically LSTM, with heterogeneous networks improve medication recommendation?

Table 9: Examples of generated RQs (Prediction) and the actual RQ (Label) for the Flan-T5 model fine-tuned on section data.

we need for training our RQ generation models. Table 8 shows some examples of detected context containing RQs along with their formed RQ by the LLM, *GPT-3.5-turbo*.

Table 9 shows some examples from evaluation by comparing generated RQs with their corresponding labels, on the best-performing model, the Flan-T5 model fine-tuned on section inputs. Most of the generated RQs are highly similar or even identical to the labels, resulting in the high performance of the model.

To determine the optimal maximum token length (number of tokens) for the tokenizer, we measure the token lengths for paragraph and section inputs after tokenization. Figure 6 presents histograms showing the token lengths for both input types. Based on this analysis, we select a maximum length of 512 tokens for paragraph inputs and 2048 tokens for section inputs.



(a) Paragraph inputs (b) Section inputs

Figure 6: Histogram plots of token lengths (number of tokens) for paragraph and section inputs after tokenization.