CSE584 Final Project Report: Improving LLM Performance on Faulty Multiple-Choice Science Questions

Bucky Park
The Pennsylvania State University
hbp5148@psu.edu

Abstract

This study investigates the robustness of large language models (LLMs) against faulty multiple-choice questions (MCQs) and proposes three prompting strategies to mitigate their susceptibility. Using a dataset spanning 15 scientific disciplines, we demonstrate that adding an explicit "None of the above" option yields the most significant improvement, while in-context learning offers a moderate alternative. Our findings highlight the critical role of prompt design in enhancing the reliability and adaptability of LLMs.

1 Introduction

The rise of large language models (LLMs) such as GPT, Gemini, or Claude has transformed numerous fields, enabling machines to achieve human-like understanding and reasoning in various tasks. Despite their impressive capabilities, LLMs remain vulnerable to ambiguity and faulty input, which can lead to incorrect or misleading responses. Addressing these vulnerabilities is critical to ensuring the reliability and robustness of LLMs in real-world applications.

One common scenario where LLMs encounter challenges is in answering faulty questions. Faulty questions, where no correct answer exists or the phrasing of the question is misleading, can easily trick even state-of-the-art models. This study investigates how prompt design can mitigate this issue and improve LLM performance when handling faulty questions, especially on multiple-choice questions (MCQs). We focus on three distinct prompting strategies: (1) adding a hint sentence to the question, (2) introducing an explicit "None of the above" option, and (3) using in-context learning with example questions.

To evaluate these strategies, we create a dataset of faulty MCQs spanning 15 scientific disciplines, including mathematics, physics, computer science, and biology. Each discipline contains 60 questions designed to mislead the LLM. We then test the effectiveness of the three proposed strategies using GPT-4o-mini as the evaluation model. The results demonstrate that explicit prompts, such as adding a "None of the above" option, significantly improve model performance, while hint sentences are less effective. In-context learning, on the other hand, provides a moderate improvement, suggesting its potential as a flexible strategy for various question types.

2 Related Work

Large language models have achieved significant milestones in natural language processing, with applications spanning text generation, question answering, and reasoning tasks. However, their susceptibility to adversarial inputs and ambiguous questions remains an active area of research.

Several studies have explored the robustness of LLMs when faced with adversarial examples or faulty input. Research by Hendrycks et al. (Hendrycks et al., 2020) highlighted the limitations of LLMs in multitask settings and emphasized the need for robust evaluation frameworks. Additionally, benchmarks like AdvGLUE have been developed to assess LLMs' resilience against adversarial prompts, providing a comprehensive suite for testing models under various adversarial scenarios. (Zhu et al., 2023)

Prompt engineering has emerged as a key strategy for enhancing LLM performance in various tasks. Techniques such as zero-shot, one-shot, and few-shot prompting have been shown to improve the reasoning capabilities of LLMs by providing contextual hints within the input. (Brown, 2020; Kojima et al., 2022; Wang et al., 2023) In-context learning, where example questions are included in the prompt, has also gained attention for its ability to guide LLMs without requiring task-specific fine-tuning. (Li, 2023)

The challenge of handling faulty questions, particularly in MCQ formats, has been addressed in limited studies. Prior work has mainly focused on developing robust question-answering models or introducing probabilistic frameworks to identify unanswerable questions. For instance, (Faldu et al., 2024) proposed RetinaQA, a knowledge base question-answering model designed to detect both answerable and unanswerable questions by discriminating over candidate logical forms. Similarly, (Sawhney et al., 2024) introduced FUn-FuSIC, a few-shot transfer learning approach that iteratively prompts large language models to generate logical forms, assessing confidence to decide answerability. However, the integration of prompt design strategies to specifically address faulty MCQs remains underexplored, motivating the contributions of this study.

Based on these related works, this study introduces a new dataset of faulty scientific MCQs and evaluates the impact of various prompt engineering techniques on LLM performance. By bridging the gap between prompt design and robustness testing, this study provides valuable insights into enhancing the reliability of LLMs in challenging situations.

3 Datasets

3.1 Multiple-Choice Question Dataset

In this study, we focus on scientific questions in a multiple-choice format. To generate faulty science questions, we utilize the existing MCQ dataset.

The MMLU dataset[1] (Hendrycks et al., 2020) is an extensive multitask benchmark consisting of multiple-choice questions across a wide range of disciplines. It spans 57 disciplines including elementary mathematics, US history, computer science, law, and more.

From this dataset, we select 15 scientific disciplines for our study: Anatomy, Astronomy, College Biology, College Chemistry, College Computer Science, College Mathematics, College Medicine, College Physics, Computer Security, Econometrics, Electrical Engineering, Machine Learning, Medical Genetics, Nutrition, and Virology.

3.2 Data Generation Strategy

To generate faulty questions, we implement the following data generation strategy.

[1] https://huggingface.co/datasets/Stevross/mmlu

3.2.1 Answer Removal

For each MCQ in the MMLU dataset, we remove the correct answer from the set of choices. We selected 15 scientific disciplines, as described in the previous subsection.

The script `generate_mcq_data.py` extracts MCQ data from the MMLU dataset. During this process, the 15 disciplines mentioned earlier are specified, and the extracted data is stored in the `mcq_data.json` file. Each question is saved in the following format:

```
{
"index": 405,
"question": "A plant that grows along a trellis exhibits",
"choices": [
    "thigmotropism",
    "phototropism",
    "gravidotropism",
    "negative gravidotropism"
],
"answer": 0,
"discipline": "college_biology",
"no": 78
}
```

- **`index`**: The index of the question in the original dataset.
- **`question`**: The text of the question.
- **`choices`**: An array containing the four possible answer choices.
- **`answer`**: The correct answer to the question.
- **`discipline`**: The discipline to which the question belongs.
- **`no`**: The question number within the specified discipline.

After extracting questions, the questions is processed to remove the correct answer from the available choices. The script `remove_answer.py` reads the `mcq_data.json` file, removes the correct answer from the choices in each question, and stores the question to a file named `removed_answer_mcq_data.json` in the following format:

```
{
"index": 405,
"question": "A plant that grows along a trellis exhibits",
"choices": [
    "phototropism",
    "gravidotropism",
    "negative gravidotropism"
],
"discipline": "college_biology",
"no": 78
}
```

3.2.2 LLM Testing

Each question, after removing the correct answer, is formatted into a text prompt in the following format: *"Answer the following multiple choice question. Question:* `[Question]` *Choices:* `[Choices]`". The script `data_textualize.py` does this process and stores textualized questions to a file named `textual_mcq_data.json` in the following format.

```
{
"index": 405,
"discipline": "college_biology",
"no": 78,
"text": "Answer the following multiple choices questi
on.\nQuestion: A plant that grows along a trellis exhi
bits\nChoices:\n0: phototropism\n1: gravidotropism\n
2: negative gravidotropism\n"
}
```

These text prompts is then input into a LLM for evaluation in the script `ask_llm.py`. We use **GPT-4o-mini** for this project. The model then stores responses to a file named `response_mcq_data.json` in the following format.

```
{
"index": 405,
"discipline": "college_biology",
"no": 78,
"text": "Answer the following multiple choices questi
on.\nQuestion: A plant that grows along a trellis exhi
bits\nChoices:\n0: phototropism\n1: gravidotropism\n
2: negative gravidotropism\n",
"response": "The correct answer is: 0: phototropism.
\n\nPhototropism is the growth of a plant in response
to light, often causing it to grow towards a light sou
rce, which is common for plants that climb trellises."
}
```

3.3 Collecting Desired Questions

If the LLM selects a specific choice for a given question, it indicates that the LLM has been fooled by the faulty question, as it should not provide an answer when the correct choice is missing. By applying this method, we systematically generate and analyze faulty science questions that can fool the model.

This process was done manually. We observed that they often included phrases such as *"the correct answer is"* or *"the correct choice is"*. Using these keywords, we filtered the corresponding questions. This has been done in the script `filtering.py`.

However, there were some questions that contained these phrases but indicated that no correct answer existed, for example, responses like *"the correct answer is not in the provided choices"* or *"the correct answer is none of the above"*. Therefore, we reviewed the entire responses and excluded those responses manually.

There was a notable pattern. Disciplines such as math and physics had significantly fewer filtered cases. However, we discovered many questions where responses provided answers without including the phrases above. These cases were also included in the final dataset.

As a result, we generated 60 faulty questions per discipline, designed to mislead the LLM, across all 15 selected disciplines. The dataset is available in the Google Sheet.[2]

4 Three Research Scenarios for Improving LLM Performance

We propose three interesting research scenarios aimed at improving the LLM's performance. These scenarios involve modifications to the prompting strategy, designed to prevent the model from being tricked by the faulty scientific questions.

4.1 Scenario 1: Adding a Hint in the Question

In this scenario, we include the sentence *"There may be no right answer."* in the given text. For example, the prompt

```
Answer the following multiple-choice question.
Question: A plant that grows along a trellis exhibits
Choices:
0: phototropism
1: gravidotropism
2: negative gravidotropism
```

becomes

```
Answer the following multiple-choice question.
There may be no right answer.
Question: A plant that grows along a trellis exhibits
Choices:
0: phototropism
1: gravidotropism
2: negative gravidotropism
```

By including the additional sentence in the given text, we expect it to indicate that there might be no correct answer to the question, preventing the LLM from being fooled.

4.2 Scenario 2: Adding a choice

In this scenario, we add another option, *"3: None of the above"*, to the choices. For example, the prompt

```
Answer the following multiple-choice question.
Question: A plant that grows along a trellis exhibits
Choices:
0: phototropism
```

[2] https://docs.google.com/spreadsheets/d/15m83rfH7xvT8_nyAe1_D5IdGDN4mhVDlLkLRdPsUiVc/edit?gid=0#gid=0

```
1: gravidotropism
2: negative gravidotropism
```

becomes

```
Answer the following multiple-choice question.
Question: A plant that grows along a trellis exhibits
Choices:
0: phototropism
1: gravidotropism
2: negative gravidotropism
3: None of the above
```

By adding a choice indicating that there is no correct answer for the given question, we expect it to encourage the model to judge that there is no right answer and to select this option.

4.3 Scenario 3: In-Context Learning

In this scenario, we do not provide any additional hints in the question or the choices. However, inspired by in-context learning, we show two simple examples before introducing the given question. One example is a valid question with a correct answer, and the other is a faulty question without a correct answer. For instance, the prompt:

```
Answer the following multiple-choice question.
Question: A plant that grows along a trellis exhibits
Choices:
0: phototropism
1: gravidotropism
2: negative gravidotropism
```

becomes

```
Example 1)
Question: What is 1+1?
Choices:
0: 1
1: 2
2: 3
Response: The correct answer is 1: 2
Example 2)
Question: What is the last digit of pi?
Choices:
0: 1
1: 3
2: 5
Response: The correct answer is not in the provided choices
Answer the following multiple-choice question.
Question: A plant that grows along a trellis exhibits
Choices:
0: phototropism
1: gravidotropism
2: negative gravidotropism
```

We expect that the two example questions before the given question will serve as hints, presenting the possibility that the question may have no correct answer.

5 Results

Using the dataset we generated, we first created three prompt messages based on the scenarios described in Section 4. The script `prompt_setup.py` is used to generate these prompt messages and store them in `prompted_mcq_data.json`. Once the prompts are created for each question, they are input into the LLM, GPT-4o-mini, and the corresponding responses are collected in `experiment.py`.

Our dataset consists of 15 different disciplines, each containing 60 questions. After obtaining responses from all questions across the three scenarios, we manually counted the number of cases where the LLM did not provide a specific answer. The responses for all questions are available in the Google Sheet.[3] Table 1 shows the number of problems for which the LLM did not provide a specific answer, indicating that it was not fooled by the faulty question, for each discipline and scenario.

Discipline	**Scenario 1**	**Scenario 2**	**Scenario 3**
Anatomy	5	17	12
Astronomy	2	15	7
College Biology	6	27	15
College Chemistry	2	18	14
College Computer Science	8	26	8
College Math	8	24	7
College Medicine	2	20	8
College Physics	7	26	15
Computer Security	3	16	16
Econometrics	1	9	1
Electrical Engineering	1	10	5
Machine Learning	3	10	4
Medical Genetics	4	25	19
Nutrition	4	14	11
Virology	0	11	9
Total	56	268	151

Table 1: Number of questions where the LLM responded without a specific answer across scenarios and disciplines

6 Discussion

From the results, we observed that Scenario 4.2 achieved the most significant improvement in the performance of the LLM, while Scenario 4.1 achieved the least improvement. This tendency is consistent across all discplines and shows the impact of different prompting on the LLM's performance.

In Scenario 4.1, we added a hint sentence, *"There may be no right answer."*. Although this hint slightly improved the performance, the scale of improvement was not significant. From the responses, we observed that the LLM often acknowledged the possibility of no correct answer, but it still attempted to provide the best possible option. This

[3] https://docs.google.com/spreadsheets/d/14HVKbefbHhXehV3FCm0T9G83rfM2gHFgRK_FrP80Qrg/edit?gid=0#gid=0

implies that while the hint sentence encourages the model to consider the absence of a correct answer, it is not enough to keep it from choosing an answer. That is the reason why this scenario showed the least improvement in performance. This may be due to the sentence not being strong enough. For instance, instead of using *"There may be no right answer"*, a more explicit statement such as *"If you believe there is no correct answer, then there is no correct answer"* could have been more effective.

On the other hand, we observed the most significant improvement in Scenario 4.2. Unlike Scenario 4.1, when the LLM recognizes that there is no correct answer, it selects the additional option, *"3: None of the above"*. This explicit option representing the possibility of no answer is more effective than the additional sentence used in the previous scenario. This approach reduces the reliance of the model on implicit reasoning and simplifies the decision-making process for the model, leading to consistently higher performance across all disciplines. This result demonstrates the importance of clear and direct prompting strategies in improving the performance and robustness of LLMs when they face with faulty questions.

In Scenario 4.3, we applied in-context learning by showing two examples before the main question: one with a valid question that has a correct answer and another with a faulty question that has no correct answer. Unlike Scenario 4.1 or 4.2, we did not provide any additional sentence or option in the prompt. Instead, the two examples serves as contextual hints, helping the LLM consider the possibility of no correct answer. While the improvement was not as significant as in Scenario 4.2, the in-context examples are still more effective than the hint sentence in Scenario 4.1. Providing more complicated and numerous examples would improve the model's performance. This demonstrates the effect of in-context learning on behavior of LLMs without the need for modifications to the question format.

7 Conclusion

In conclusion, this study generated the faulty scientific question dataset that can fool LLMs in terms of multiple-choice questions, and established three distinct prompting strategies to mitigate this issue based on our dataset.

The results demonstrated that the strategy of adding an additional choice, *"None of the above"*, yielded the most significant improvement in the performance and robustness of LLMs. On the other hand, adding a hint sentence to the question provided the least improvement, highlighting the importance of clarity and explicitness in prompt design. Clarity and explicitness in prompt design are crucial in enhancing the reliability and decision-making capabilities of LLMs. Adding a clear option in the prompt significantly reduces ambiguity and enhance the strenghts of LLMs, which enables them to handle faulty questions better.

Furthermore, we showed that in-context learning is a promising tool for prompting strategies. By providing example questions to establish context, we achieved improvements in LLM performance without changing the structure of the prompt itself. While this approach was less effective than providing explicit options, it demonstrated great advantages over the hint sentence approach.

8 Limitations and Future Works

8.1 Limitations

One of the primary limitations is that the classification of responses from the LLM was done manually, which possibly gives a slight inaccuracies in the results. This manual process led to ambiguities in certain cases. For example, when the model considered that there might be no correct answer but ultimately selected the most plausible answer, we classified it as the case where the model get tricked because it still provided an answer.

Another limitation comes from disciplines with heavy calculation, such as math and physics. In these disciplines, the model often selected the closest answer or the approximated answer. This tendency shows the need for more precise criteria to evaluate the LLM's response. Establishing clearer criteria for distinguishing between being fooled and reasonable approximations would enhance the accuracy of analyses in these disciplines.

Furthermore, the scope of this study was limited to multiple-choice questions. This restriction was made to maximize the number of faulty questions and the number of disciplines in a short period. This constraint means that Scenario 4.2, showing the most significant improvement by introducing an additional option, is not directly applicable to non-MCQ formats. Despite this limitation, other strategies, such as explicitly stating that no correct answer exists (as in Scenario 4.1) or utilizing in-context learning techniques (as in Scenario 4.3), could potentially be adapted for more general question formats.

8.2 Future works

While this study primarily focused on prompt design, there are several alternative designs that could be constructed in future research to enhance the robustness of LLMs against faulty questions. These approaches go beyond prompt engineering and introduce novel techniques inspired by probabilistic reasoning, iterative querying, and extensions to non-MCQ formats.

- Probability-based Answer Classification: Inspired by classification problems, this approach asks the LLM to output a probability distribution over all possible choices for a given MCQ. If a correct answer exists, the model should assign a high probability to one of the choices. By setting a specific threshold, any choice exceeding this threshold would be considered as the correct answer. In contrast, if no choice meets the threshold, the question is considered to have no correct answer. This probabilistic framework would provide a more quantitative way to assess the LLM's confidence in its responses.

- Monte Carlo-inspired Iterative Querying: Inspired by Monte Carlo methods, this approach involves querying the LLM multiple times with the same question. If the answers provided by the model are inconsistent across trials, the question can be viewed as having no correct answer. By aggregating the results from multiple trials, it is possible to determine whether the LLM's responses converge to a consistent answer or suggest uncertainty.

- Extension to Non-MCQ Formats: For questions of non-MCQ format, a hybrid approach could be employed. Similar to the second method inspired by Monte Carlo method, the same question is queried to the LLM multiple times. If the model generates different answers across trials, a "None of the above" option can be added to the previously generated answers to create a new MCQ version of the question. This newly created question is then input to the LLM. This iterative process bridges the gap between non-MCQ and MCQ formats, making it possible to apply existing methods such as Scenario 4.2.

- Testing Strategies on Different Architectures: We can also apply the prompt strategies to different LLM architectures. It would help generalize the findings and identify architecture-specific strengths and limitations. This would provide a more comprehensive understanding of how these strategies perform in varied settings.

References

Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Prayushi Faldu, Indrajit Bhattacharya, et al. 2024. Retinaqa: A robust knowledge base question answering model for both answerable and unanswerable questions. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6643–6656.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Yinheng Li. 2023. A practical survey on zero-shot prompt design for in-context learning. *arXiv preprint arXiv:2309.13205*.

Riya Sawhney, Indrajit Bhattacharya, et al. 2024. Robust few-shot transfer learning for knowledge base question answering with unanswerable questions. *arXiv preprint arXiv:2406.14313*.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2609–2634.

Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Yue Zhang, Neil Gong, et al. 2023. Promptrobust: Towards evaluating the robustness of large language models on adversarial prompts. In *Proceedings of the 1st ACM Workshop on Large AI Systems and Models with Privacy and Safety Analysis*, pages 57–68.