A.M.P at SciHal2025: Automated Hallucination Detection in Scientific Content via LLMs and Prompt Engineering

Khoa Nguyen-Anh Le[1,2], Dang Van Thin[1,2],
[1]University of Information Technology-VNUHCM, Ho Chi Minh City, Vietnam
[2]Vietnam National University, Ho Chi Minh City, Vietnam
23520742@gm.uit.edu.vn, thindv@uit.edu.vn

Abstract

This paper presents our system developed for SciHal2025: Hallucination Detection for Scientific Content. The primary goal of this task is to detect hallucinated claims based on the corresponding reference. Our methodology leverages strategic prompt engineering to enhance LLMs' ability to accurately distinguish between factual assertions and hallucinations in scientific contexts. Moreover, we discovered that aggregating the fine-grained classification results from the more complex subtask (subtask 2) into the simplified label set required for the simpler subtask (subtask 1) significantly improved performance compared to direct classification for subtask 1. This work contributes to the development of more reliable AI-powered research tools by providing a systematic framework for hallucination detection in scientific content. The implementation of our system is available on GitHub. [1]

1 Introduction

Nowadays, the rapid advancement of generative AI has revolutionized academic research practices, introducing AI-powered research assistants capable of synthesizing information and responding to complex scientific queries (Glickman and Zhang, 2024). These systems leverage large language models (LLMs) such as Llama (Touvron et al., 2023) and DeepSeek (Xiong et al., 2025) to generate highly accurate answers in a very fast time. Although these tools offer efficiency in knowledge synthesis, they face a critical challenge: models generate text that sounds correct but is actually false or made up, this problem is called **Hallucination** (Ji et al., 2023). Hallucinations in scientific content are particularly problematic as they can propagate misinformation, undermine research integrity, and lead to flawed scientific conclusions.

[1]https://github.com/LeNguyenAnhKhoa/Hallucination-Detection

Question	What temperature does water boil at?
Answer	Water boils at 90 degrees Celsius which is equivalent to 194 degrees Fahrenheit
Claim	Water boils at 90 degrees Celsius
Reference	Water boils at 100 degrees Celsius
Label	Contradiction
Justification	Numeric Error, water boils at 100 degrees Celsius, not 90

Table 1: Example data point from the training dataset.

Given these problems, the SciHal2025 tasks focus on the detection of hallucination from the claim that is extracted from the answer of LLM. In this paper, we present a methodology that combines state-of-the-art language models with advanced prompt engineering techniques to identify and classify different types of hallucination.

2 Data and Task

2.1 Data

The full provided data is divided into four batches, three batches for training, and one batch for testing (batch1/batch2/batch3/test, 500/1592/1500/1000). All dataset samples have the following fields: Question (questions users ask LLM), Answer (answer generated by GenAI-powered research assistant), Claim (one or more sentences extracted from the generated answer that answers the question) and Reference (one or more references, each being an abstract from GenAI-powered research assistant). The training dataset has two additional fields: Label (classification labels are typed by SME[2] annotator) and Justification (reasoning provided by SMEs for assigning the label). SMEs received the claims, references, and detailed guidelines, including hallucination type definitions and a decision tree (as shown in Figure 1) to annotate. Every instance was labeled by one SME, ensuring baseline human judgment for all samples. Additionally, batch3 and

[2]SME = Subject Matter Expert

Proceedings of the Fifth Workshop on Scholarly Document Processing (SDP 2025), pages 328–335

the test set are annotated by three different SME annotators. This ensures high-quality, consensus-based annotations, making batch 3 and the test set more challenging and reliable. Table 1 shows the overview of the data.

2.2 Task

This task is a multiclass classification task to determine the claim extracted from the answer containing any hallucinated content based on the references. For subtask 1, the task is to determine whether the references entail, contradict, or are unverifiable to the claim. Subtask 2 is more complex, the task is to determine whether the references entail, are unrelated and unverifiable, are related but unverifiable, misrepresentation, missing information, contain a numeric error, contain an opposite meaning, or contain an entity error to the claim. For example, in Table 1, the claim has a "Numeric Error" when water boils at 100 degrees C and not 90 degrees C, which also leads to the reference contradicting the claim. According to Figure 1, subtask 1 is a more compact version of subtask 2 with only three labels, while subtask 2 has eight labels. Weighted F1-score (Harbecke et al., 2022) is the main benchmark for this task, and we also use this score to evaluate methods.

3 System Overview

In this section, we describe the system in detail. We first noticed that the claim and reference contained quite a few encoding errors, so we used the **ftfy** library to fix these encoding errors. We then performed prompt engineering on large language models to make predictions. In addition, we discovered a simple, efficient two-step method that yields better results.

3.1 Prompt Engineering

First of all, we used LLMs as a black-box detection system, so we used the entire training set as a validation set to test the models and evaluate the methods. Next, we selected versions of the gemini models (Imran and Almusharraf, 2024), smaller versions of OpenAI's o3 and o4 (Ramachandran, 2024) models to make direct predictions.

We continued with the tuning prompt, the most optimal prompt for subtask 1 is shown in Figure 2. The first part of the prompt is to define the LLM's role and task, this helped the LLM understand the specific role to be undertaken and the task to be performed, thereby focusing on the right goal and giving feedback appropriate to the context of the request and its effectiveness has been proven at (Shanahan et al., 2023). The second part of the prompt is that we explain what each label means to help the LLM distinguish between potentially confusing concepts and apply them in the correct context for each use case, this explanation is based on the decision tree as shown in Figure 1. For the next part of the prompt, we apply few-shot learning (Schick and Schütze, 2022) to improve the model's ability to understand and perform tasks. Few-shot learning allows the model to learn from a limited number of examples, helping it quickly adapt to specific requirements and orient the model towards the desired output format. For each label, we randomly select an example for the LLM to understand better, in Figure 2 we choose the example sample for the label 'Contradiction'. Finally, we set output requirements that require the model to produce a 'justification' and an 'answer'. The 'justification' part first explains why it predicted that label, forcing the model to think before making a final decision. The 'answer' part must only respond to a single label, helping the model produce the answer with the highest probability and go straight to the point. For subtask 2, we add more explanation for the remaining labels and give more examples, we can see the sample prompt in Figure 3. The results of this approach are presented in Table 2.

Model	Subtask 1	Subtask 2
gemini-2.0-flash	0.580	0.572
gemini-2.5-flash	**0.719**	**0.635**
o3-mini	0.708	0.626
o4-mini	0.693	0.617

Table 2: Weighted F1-score on validation set among models. Best results are in bold.

3.2 Two-step approach

As shown in Figure 1, we can see that the label nodes of subtask 1 are parents of the label nodes of subtask 2 in the decision tree. So instead of directly predicting three labels for subtask 1, we can predict eight labels for subtask 2 and then reduce this result to three labels for subtask 1. The converted labels can be seen in detail in Figure 4. With this method, we reduced the direct prediction from two

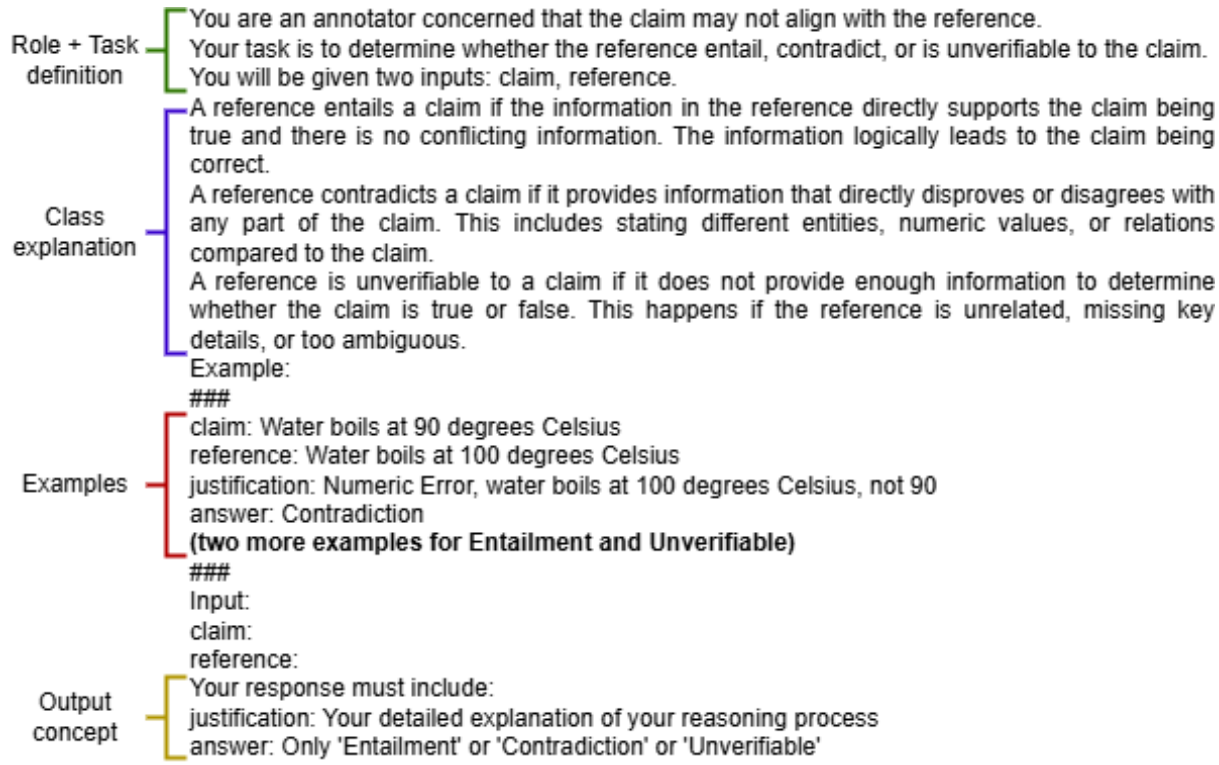

Figure 2: Example prompt with one example for each label for subtask 1.

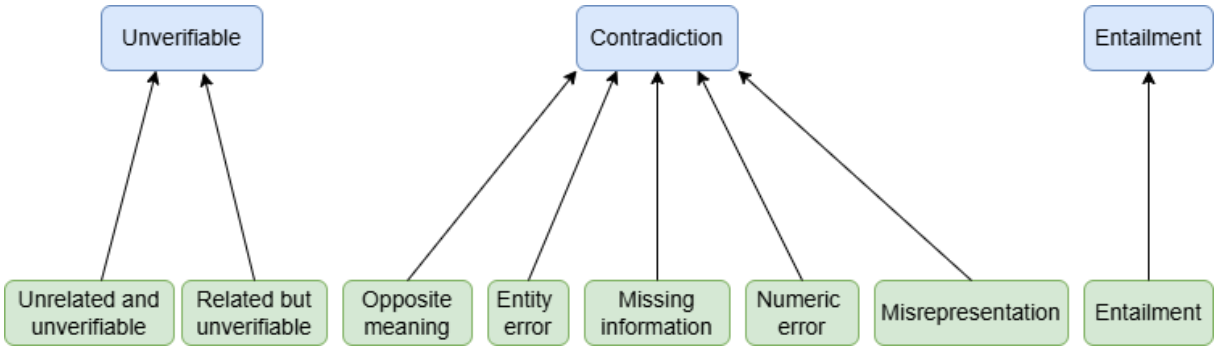

Figure 4: Two-step approach. The label of subtask 1 is light blue. The label of subtask 2 is light green.

Model	Directly	Two-step
gemini-2.0-flash	0.580	0.694
gemini-2.5-flash	0.719	0.723
o3-mini	0.708	0.714
o4-mini	0.693	0.703

Table 3: Weighted F1-score on validation set of subtask 1 between directly and two-step approach.

times (each time one subtask) to a single prediction for subtask 2, while subtask 1 only requires simple operations to be able to make the prediction. Finally, this approach gives better results in all models shown in Table 3.

4 Experimental Setup

We used large language models through APIs, which allow us to make predictions quickly and test multiple methods without the need for powerful hardware. However, sometimes due to network errors, we have to find specific patterns to re-predict, which costs a bit of money. For Gemini models, we cast the output to JSON format so that the output has a specific format and we can process the output more easily. Also, we leave the **temperature** coefficient as 0 so that the model gives the highest probability result. Gemini's API documents are available at Google AI Studio [3]. For OpenAI models, we cannot set the temperature coefficient

[3] https://ai.google.dev/gemini-api/docs

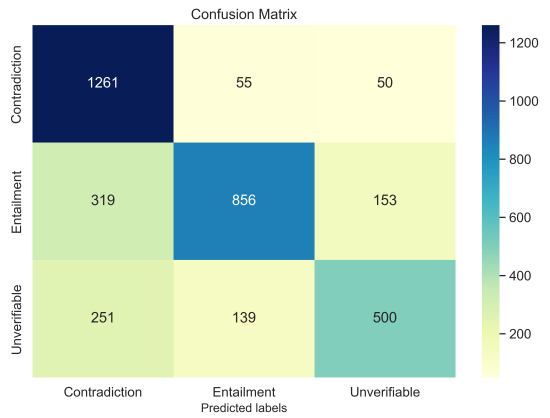

Figure 4: Confusion matrix of subtask 1 uses gemini-2.5-flash model to make prediction using two-step method.

Category	Precision	Recall	F1-score
Contradiction	0.69	0.92	0.79
Entailment	0.82	0.64	0.72
Unverifiable	0.71	0.56	0.63
Accuracy	—	—	0.73
Macro Avg	0.74	0.71	0.71
Weighted Avg	0.74	0.73	0.72

Table 4: Classification report for subtask 1 using gemini-2.5-flash model to make prediction using two-step method.

or cast the output, but instead we can adjust the **reasoning_effort** coefficient to "high" to make the model think more carefully before giving the final answer. OpenAI API documentation can be found at OpenAI Platform [4]. For all models, I set my lucky **random seed** to 13 so that each run of the models gives the same results.

5 Results

Based on Table 2 and Table 3, we decided to make a direct prediction using the prompt in Figure 3 for subtask 2 and reduce this result to predict for subtask 1. We also selected the only best model (gemini-2.5-flash) to make predictions on the test set.

5.1 Subtask 1 result

Our two-step system demonstrated moderate performance with an overall accuracy of 73% and a weighted F1-score of 0.72, exhibiting notable class-wise performance disparities according to Table 4.

[4] https://platform.openai.com/docs/overview

Category	Precision	Recall	F1-score
Entailment	0.82	0.64	0.72
Entity error	0.54	0.78	0.64
Misrepresentation	0.40	0.48	0.43
Missing information	0.00	0.00	0.00
Numeric error	0.83	0.83	0.83
Opposite meaning	0.64	0.96	0.77
Related but unverifiable	0.61	0.45	0.52
Unrelated and unverifiable	0.50	0.51	0.50
Accuracy	—	—	0.64
Macro avg	0.54	0.58	0.55
Weighted avg	0.66	0.64	0.63

Table 5: Classification report for subtask 2 using gemini-2.5-flash model to make prediction directly.

The model achieved good performance in contradiction detection, with a precision of 0.69, recall of 0.92, and F1-score of 0.79, indicating effective identification of contradictory statements with minimal false negatives. The most challenging category is unverifiable content classification, achieving the lowest F1-score of 0.63 with a precision of 0.71 and recall of 0.56. The confusion matrix (can be viewed at Figure 4) reveals significant misclassification patterns, particularly 251 unverifiable instances incorrectly predicted as contradictions, indicating the model's tendency to over-predict the contradiction class.

5.2 Subtask 2 result

Based on the classification report (as shown in Table 5) and the confusion matrix (as shown in Figure 5), our system exhibits moderate performance with 64% accuracy and a weighted F1-score of 0.63. Additionally, we have a strong performance in detecting numeric inconsistencies (F1-score: 0.83) and opposite meaning contradictions (F1-score: 0.77, recall: 0.96). However, the model encounters significant limitations with subtler hallucination types, most notably complete failure in missing information detection (zero performance across all metrics) and poor performance in misrepresentation identification (F1-score: 0.43). The confusion matrix reveals substantial misclassification patterns, with entailment cases frequently confused with other categories (856 correct and 472 misclassified instances), and notable confusion between related categories such as 'Related but unverifiable' and 'Entailment' (132 misclassifications).

5.3 Final result

In the test set evaluation, the o3-mini model demonstrated superior performance on both sub-tasks, al-

Models	Subtask 1	Subtask 2
o3-mini	**0.59**	**0.48**
gemini-2.5-flash	0.57	0.47
o4-mini	0.52	0.43
gemini-2.0-flash	0.49	0.4

Table 6: Performance of our models on the test set. Best result are in bold.

System	Subtask 1	Subtask 2
ScaDS.AI x sebis	**0.6**	0.5
YupengCao	0.59	**0.51**
Ours	0.59	0.48
Crivoi Carla	0.51	0.43
JB	0.25	0.49

Table 7: Weighted F1-score on test set on the leaderboard. Best results are in bold.

though the gemini-2.5-flash model performed better on the validation set (full models' performance in Table 6. The model achieved a weighted F1-score of 0.59 for subtask 1 and 0.48 for subtask 2, representing the highest scores among all evaluated models. We can see the results on the test set in Table 7, our system is ranked 3rd in subtask 1 and 4th in subtask 2.

6 Conclusion

In this paper, we introduced a good system to detect various types of hallucinations produced by LLMs. The key point of our system is to design an optimal prompt with the following components: role and task definition, class explanation, examples, and output concepts so that the model can understand the concept and make accurate predictions. In addition, we introduce a two-step method to make efficient and fast predictions for both subtasks. In summary, the A.M.P system was competitive with the other systems submitted for evaluation.

Acknowledgements

This research was supported by The VNUHCM-University of Information Technology's Scientific Research Support Fund.

References

Mark Glickman and Yi Zhang. 2024. Ai and generative ai for research discovery and summarization.

David Harbecke, Yuxuan Chen, Leonhard Hennig, and Christoph Alt. 2022. Why only micro-f1? class weighting of measures for relation classification.

Muhammad Imran and Norah Almusharraf. 2024. Google gemini as a next generation ai educational tool: a review of emerging educational technology. *Smart Learning Environments*, 11(1):22. Open Access.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Anand Ramachandran. 2024. Revolutionizing research and engineering openai o3's transformative role in scientific discovery and global innovation.

Timo Schick and Hinrich Schütze. 2022. True few-shot learning with prompts—a real-world perspective. *Transactions of the Association for Computational Linguistics*, 10:716–731.

Murray Shanahan, Kyle McDonell, and Laria Reynolds. 2023. Role play with large language models. *Nature*, 623(7987):493–498.

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. 2023. Llama: Open and efficient foundation language models.

Luolin Xiong, Haofen Wang, Xi Chen, Lu Sheng, Yun Xiong, Jingping Liu, Yanghua Xiao, Huajun Chen, Qing-Long Han, and Yang Tang. 2025. Deepseek: Paradigm shifts and technical evolution in large ai models. *IEEE/CAA Journal of Automatica Sinica*, 12(5):841–858.

A Decision Tree

The figure below presents the decision tree guideline used by SME annotators during the annotation process. It also adds an explanation of the classes for LLMs to make predictions.

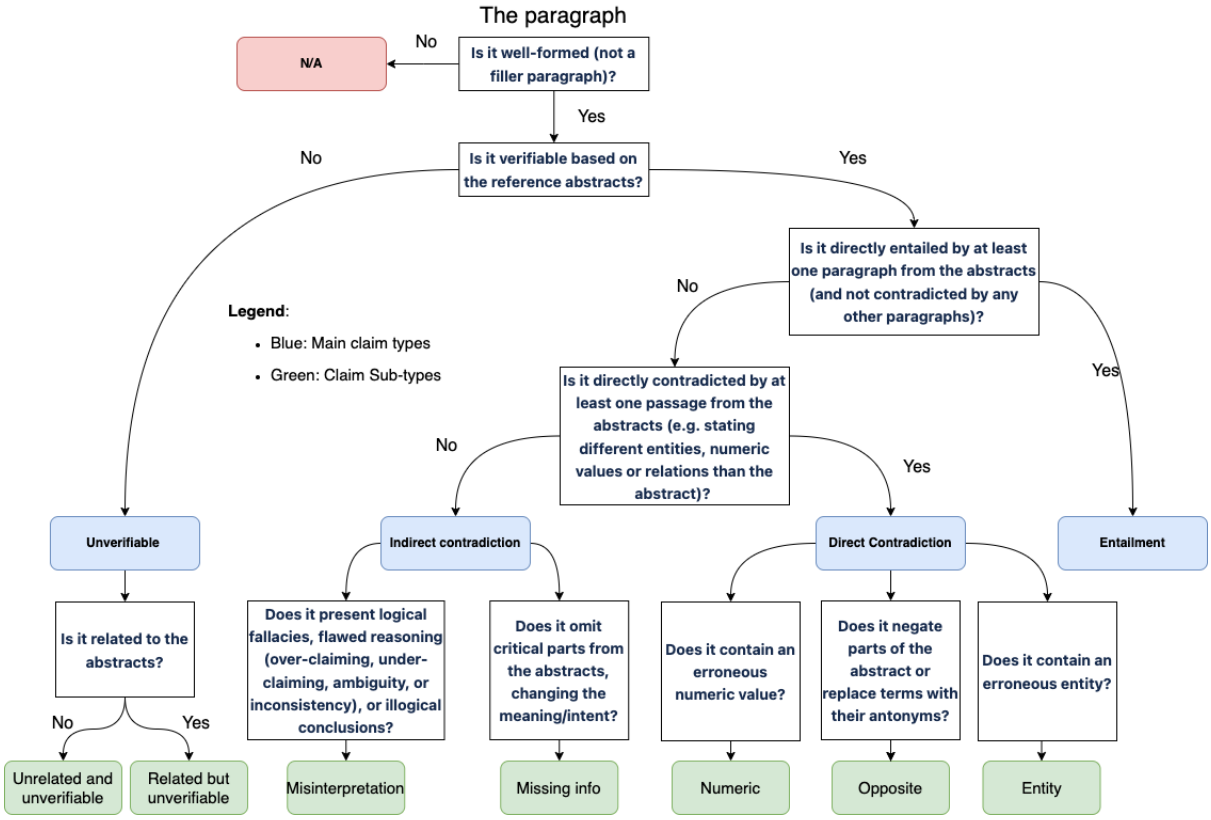

Figure 1: Decision Tree guideline for SME annotators.

B Subtask 2 prompt

Prompt for subtask 2, the components are similar to the prompt for subtask 1 and are explained in detail above. The difference is in the 'Class explanation' and 'Examples' sections as subtask 2 has more labels than subtask 1.

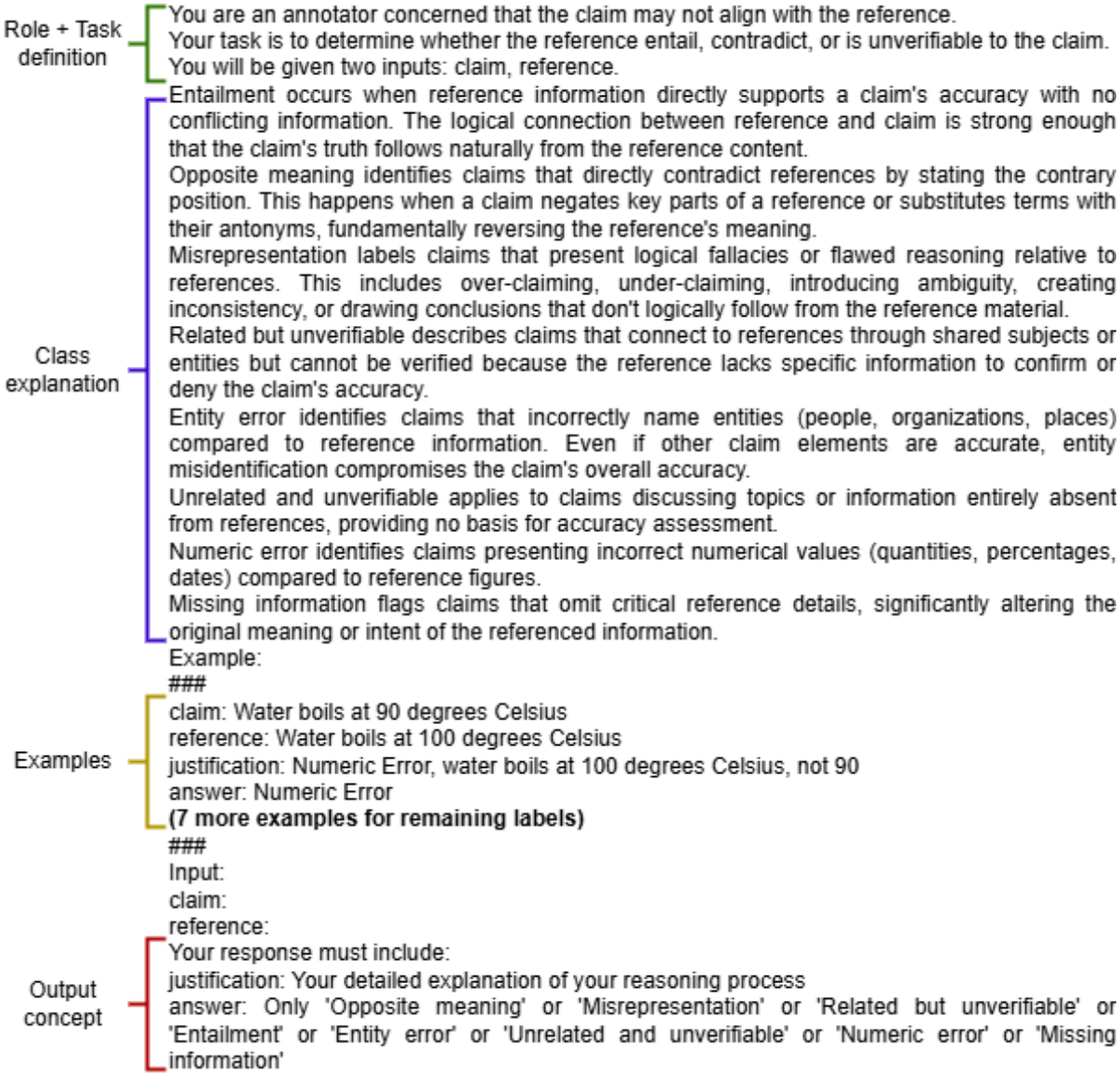

Figure 3: Example prompt with one example for each label for subtask 2.

C Subtask 2 confusion matrix

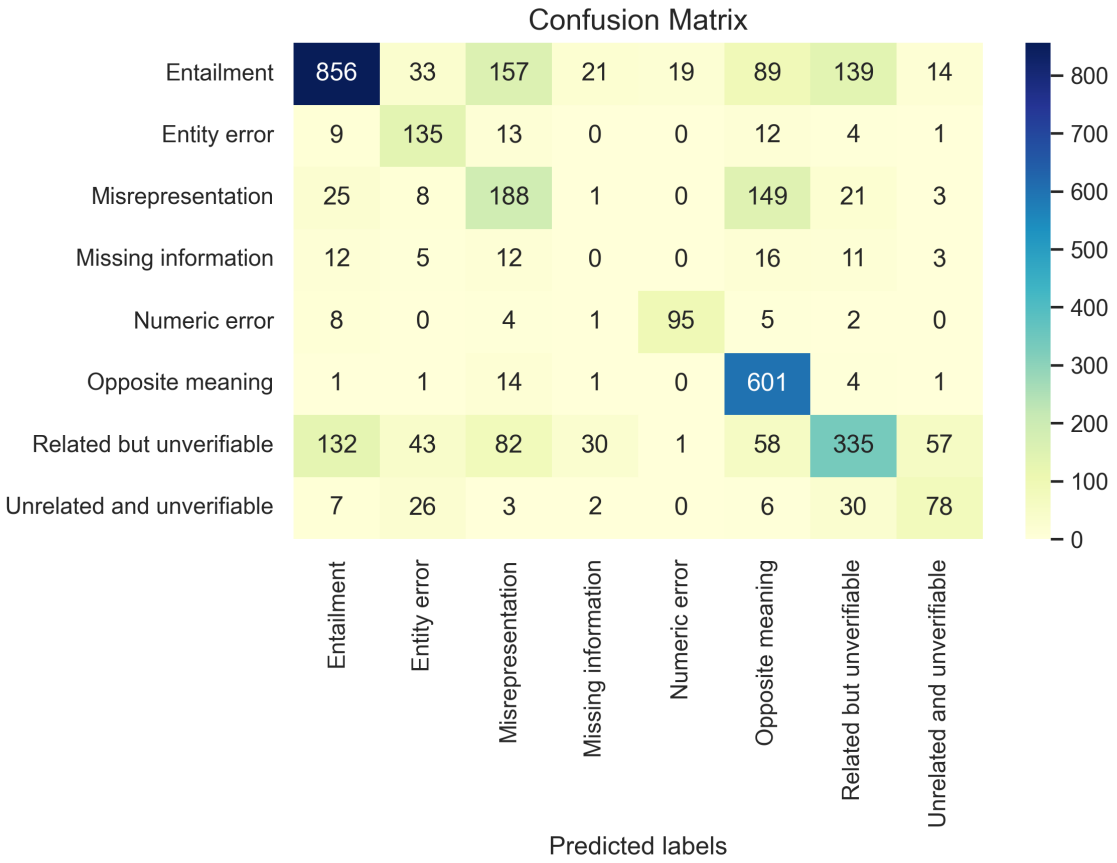

Figure 5: Confusion matrix of subtask 2 uses gemini-2.5-flash model to make prediction directly.