Reproducibility Report for CSE 481M (Sp25)

Medha Gupta, Jerry Gao, & Saket Gollapudi
University of Washington
{medhag2,jerrygao,saket312}@cs.washington.edu

Reproducibility Summary

Motivation

Recently, Large Language Models (LLMs) have become widely used across many fields. While users often expect confident answers, LLMs can generate false information, known as hallucinations. As LLMs expand into areas like healthcare and government, ensuring accurate output is critical. Agrawal et al. (2024), explore detecting hallucinations without external fact-checking by analyzing academic citations. In this project, we replicate their work and extend it to two new domains: a structured airport dataset and a Jeopardy-style Q&A set, to test the method's robustness and generalization.

Scope of Reproducibility

The paper claims hallucinations in LLM outputs can be detected without external ground truth by checking consistency in model responses to repeated metadata queries. We reproduced this experiment and extended it to world airport data and Jeopardy trivia to test if consistency reliably indicates correctness across different factual domains.

Methodology

We first reproduced the original experiment in the paper by reimplementing the code for the experiment, using different models. We reproduced 200 computer science topics, collecting five references per topic and querying each using indirect (IQ) and direct (DQ) methods. For extensions, we applied similar methods to 1,000 airport IATA codes and locations and 1,000 Jeopardy trivia Q&A's. Reproduction took one week for code fixes and integration and execution. Extensions required 2 weeks of development runtime.

Results

Our study was not successful in reproducing the results for the original paper. Because we used much smaller models than the paper which did not, we couldn't get the same accuracy in detecting hallucinations. There was an average of a 0.3 difference in the AUC from the original paper. However, our airports extension performed extremely well, with the models detecting 100% of hallucinations and ground truths correctly and with an AUC of 1.0.

What was Easy

The easiest parts of the project were coding the extensions and integrating our ideas into the authors' codebase. The authors framework was well-documented and easy to modify, making setup and reproduction and the extension straightforward.

What was Difficult

Due to feasibility constraints, we could not replicate results involving OpenAI models and other APIs/tools, so we used alternative models instead. Integrating these into the original codebase took more time than expected. Additionally, understanding the authors' evaluation metrics and interpreting their results also took more time than expected.

Communication with Original Authors

We did not contact or communicate with the original authors of this paper.

1 Introduction

Large language models (LLMs) have become powerful tools in recent years, expanding the range of tasks they can handle. They are now used in real-world applications like summarization and complex question answering with impressive fluency. However, despite sounding accurate, LLMs often produce made-up content. As noted by (Du et al., 2023), this is called hallucination. More specifically, they define hallucination as when a model's output deviates from what is expected, given the input or task. These errors can be serious in fields like healthcare, law, and education, where accuracy is critical.

A recent study by Agrawal et al. (2024) explores whether hallucinations can be detected without ground truth. Instead of external fact-checking, they look at how models explain the citations they generate. The paper found that hallucinated citations often come with vague or incorrect metadata, which can be a clue for spotting made-up content. They show hallucinations can be detected and suggest the problem may come from the generation process rather than model training. In this extension, we replicate their work and test how well it generalizes. We first repeat their experiments using the original dataset and code. Then we try two extensions: one on a structured airport dataset and another on Jeopardy-style QA. These help us explore the strengths and limits of this approach and see if it offers a training-free way to improve LLM reliability.

2 Scope of Reproducibility

The primary claim the paper is trying to address is that hallucinated responses generated by LLMs can be detected without using any external ground truth. Specifically, in the paper, the authors show that by asking the model detailed questions about a response it generated (academic reference about a given computer science topic) repeatedly and checking if those answers are consistent, we can tell if the response is correct or hallucinated. They find that models give more consistent answers when the responses are correct and less consistent answers when the responses are wrong.

We reproduced the original experiment and tested if this approach works beyond academic references. First, we ran the method on another factual dataset: world airports. Second, we wanted to see if it works on more general queries that still have deterministic answers, by using trivia questions from a Jeopardy dataset. For these two extensions, we aimed to support the paper's claim that we can detect hallucinations in LLMs just by looking at the consistency of their responses when queried about various metadata about their responses, and without looking at the ground truth labels of the datasets. This is to help us understand if consistency is an effective proxy to detect hallucinations in different types of contexts, not just academic references.

2.1 Addressed Claims from the Original Paper

1. We can detect if LLMs are hallucinating about references for computer science topics without needing external ground truth and just by querying the model repeatedly about metadata about those references and checking for the consistency in the models' responses.

3 General Steps in Methodology

3.1 Reproduction Methodology

We reproduced the original experiment using a dataset of 200 computer science topics (Rous (2012)). For each topic, we prompted the model to generate 5 academic references (1000 total). Each reference was then queried using one indirect query (IQ) and three direct queries (DQs), as in the original paper. The IQ asked, "Who were the authors of this reference?" and the DQs were yes/no questions phrased in different ways to check for the reference's existence. Each IQ was asked 3 times, and each DQ was asked 10 times to measure the model's response consistency. A consistency score was computed as the fraction of correct outputs per query type (e.g., 7 out of 10 "yes" answers yields a score of 0.7), independent of ground truth. We used a threshold on this score to classify each reference as hallucinated or grounded, purely based on model behavior. Further evaluation using ground truth is described in §4.4.

3.2 Airports Extension Methodology

We wanted to see if we could expand the findings of this paper can be translated to a different topic that was not academic references. This experiment follows the same setup as the original experiement but uses a structured dataset of 1000 International Air Transport Association(IATA) airport codes. For each code, we asked 3 IQs (country, airport name, continent) and 1 DQ ("Does the airport with IATA code x exist?"). IQs were asked 3 times, DQs 5 times to reduce compute cost. The rest of the consistency-based analysis and classification process remains unchanged.

3.3 Jeopardy Extension Methodology

The final part of the extension was to see if the findings would translate to a less structured and more generalized data set. Essentially increasing the variance of the prompts but decreasing the bias for the model. This extension uses 1000 trivia questions instead of references. The model answers each question freely, typically with 1–2 words. Because there's no uniform metadata, we ask 0 IQs and 3 DQs: one checking the model's answer, one with the correct answer, and one with a plausible incorrect answer (generated and verified via ChatGPT). Each DQ is repeated 5 times. The rest of the analysis follows the original consistency-based method.

3.4 Evaluation Methodology

To evaluate our results, we labeled each response as a true/false positive/negative by comparing the model's classification (via consistency score thresholding) to the ground truth. For the original experiment, we used the OpenAlex API (instead of Bing, due to access issues) to validate reference existence (OpenAlex (2024)). For the airports extension, we matched generated IATA codes to an airport dataset and checked country/continent correctness using a country-to-continent map. The Jeopardy dataset already included ground truth answers.

4 Methodology

Here we will discuss the specific models and datasets we used for this experiment. In summary, we took inspiration from the original code and wrote our own code for each extension that models the experiment setup from the papers. The setup and code structure built by the authors was reused for parts of the extension, and we added unique prompts and functions that were applicable for each extension's objective. The hardware used to run these experiments were an M4 MacBook pro and M1 MacBook Pro.

4.1 Model Descriptions

The original paper used:

1. gpt-3, (text-davinci-003); transformer-based model; 175 billion parameters
2. gpt-3.5-turbo; transformer-based model; approx. 100–175 billion parameters
3. gpt-4; transformer-based model; about 1 trillion parameters
4. llama-2 chat series: L2-7B, L2-13B, and L2-70B; Transformer decoder-only architecture; 7/13/70 billion parameters respectively

The objective of using all these models was to run the experiment and determine hallucination rate. Unfortunately, due to feasability concerns, it was difficult for us to access the gpt models to replicate some of the results in the paper. Instead we used slightly different models found on Ollama for the reproduction and extensions to account for cost and GPU constraints.

1. gemma3 4b; transformer decoder-only architecture; 4 billion parameters
2. llama2 7b; transformer decoder-only architecture with SwiGLU activation; 7 billion parameters
3. llama3.2 3b; Transformer decoder-only architecture; 3 billion parameters
4. mistral 7b; Transformer decoder-only architecture; 7 billion parameters

The learning objective of using all these models was to run the experiment and determine hallucination rate.

4.2 Datasets

1. We used a dataset from the Association for Computing Machinery (ACM) to get list of 200 computer science topics. We obtained this dataset from the Github repo of the original paper itself, which did not link the original data source. The dataset size was 200 and it was not annotated in any way. This dataset was simply a csv file with a column for 'title'.
2. For the airports extension, we used this dataset from Kaggle (Muhammad (2021)). We obtained it by searching for reliable world airports datasets on Kaggle. The size of the dataset is 5454 and we didn't annotate it. It is a CSV file including columns for the airport name, IATA code, and airport country.
3. Additionally, because one IQ for the airports extension requires knowing the continent that each airport is in, we have this dataset of country to continent mappings for all countries in the world (Altan (2019)). We obtained it through Kaggle. The size of the dataset is 197, and we didn't annotate it. It is a CSV file including columns for country and continent.
4. For the jeopardy extension, we used this dataset to get the trivia questions (Tunguz (2019)). We obtained it through Kaggle. The size of it is 200000. This dataset includes many columns for the question, answer, jeopardy category, and jeopardy game metadata; additionally, many of the questions referenced links to images. So, we pruned it to remove all invalid questions (that include links), to get 1000 random entries, and to only include the question and answer columns.

4.3 Hyperparameters

We didn't have any hyperparameters in terms of implementing ML models. However, below are hyperparameters/decisions we made in the experiment setup.

1. For the reproduction, we used 200 CS topics and asked the model for 5 references per topic (1000 references total), and these values came from the Github repo from the paper.
2. The original paper ran each IQ 3 times and each DQ 10 times. In our reproduction, we did the same.
3. For the airports extension, we ran each IQ 3 times and each DQ 5 times. For the jeopardy extension, we ran each DQ 5 times. We decided on these values to closely resemble the original paper's but to account for compute/time constraints.
4. For the airports extension, we took the 5 continents and asked the model for 200 airports per continent to get 1000 airports in total. For the jeopardy extension, we queried the model with 1000 trivia questions. We did this to mimic the original paper.
5. The 'temperature' field when calling the model APIs was set to 0.2. This value came from the code provided by the paper and we used it for all the experiments. We reduced it to make sure the model outputted answers in the format we expected.
6. For the extensions, we set the 'max_gen_len' field to 200 when calling the model APIs. This was because we knew the outputs should never exceed 200 charactres regardless of the query and it helped reduce the time to run it and make sure the model outputted answers would not be unexpectedly long.

4.4 Evaluation

We used 2 metrics to evaluate the results of all our experiments (including the extensions), both of which are the same metrics that the original paper used.

The first metric was **Receiver Operating Characteristic (ROC) curves**. The purpose of these is to visualize the tradeoff between accuracy on G (how often grounded responses were labeled as G) and accuracy on H (how often hallucinated responses were labeled as H). For each experiment's results, for each generated response (reference, IATA code, or trivia answer), we have a consistency score between 0 and 1 for each different type of query we asked on that response. We discussed previously how we can set a threshold for the score to classify that reference as being grounded (G) or hallucinated (H). Thus, for each generated response and each type of query, we can get a classification of G/H. So, for each experiment, we created ROC curves for each type of query (all IQs, all DQs). We generated this curve by trying many different thresholds between 0 and 1 (see

§A.1 for how we chose these thresholds), and for each one, calculated the classification for all the references and then calculated the True Positive Rate and False Positive Rate (definitions are §A.2) of the results. Thus, for each threshold we got a (TPR, FPR) data point which was graphed into the ROC curve. This curve helped visualize how effective each type of query's (as well as combinations of queries) consistency scores were in classifying hallucinations, compared to with the ground truth labels. The overall detection performance was summarized with the Area **Under the Curve (AUC)** of the ROC curve, which is a value between 0 and 1, where 1 means better hallucination detection.

The second metric was **False Discovery Rate (FDR) Curves**. A false discovery rate is the

$$\frac{\text{false positives}}{\text{true positives} + \text{false positives}}$$

This is the number of responses labeled grounded that were actually hallucinated, over the total number of responses labeled grounded. The FDR curve graphs the fraction of responses preserved on the x-axis, and the false discovery rate on the y-axis for each model and each type of query at different classification thresholds (each data point is a {fraction of responses preserved, FDR} for a given threshold). This is beneficial because users may have a certain rate of tolerance for hallucinations, and would like to maximize the number of generated responses under that constraint. So, seeing the FDR curves helps visualize how effective the consistency proxy is over detecting hallucinations simply with ground truth labels, and whether it will work under different users' constraints.

4.5 Implementation

Much of the extension code was implemented by ourselves. The code to format the data, run the prompts and data through the models, and store the results were all written by us. We drew inspiration from the original paper's code so we could understand the implementation details as well as seamlessly integrate our code with the repo's environment, specifically when calculating the evaluation metrics to generate the graphs. The original paper's Github repo can be found at https://github.com/microsoft/hallucinated-references and our Github repo is https://github.com/asdf6921/hallucinated-references, which was forked from the original. In order to run everything, one does not need to look at the original Github repo at all - everything is in our extended Github repo; it is documented well with setup and running instructions.

4.6 Computational Requirements

Prior to running the experiments, we estimated that we would need a separate GPU like on the Hyac servers and about 30GB with 2-3 GPU hours per experiment. However, in implementation we learned that the Mac Pro laptops are sufficient. The hardware we actually used was an M4 MacBook Pro with 12-core CPU and a 16-core GPU for the reproductions, and an M1 MacBook Pro with 10-core CPU and a 16-core GPU was used for the extensions. Each experiment actually took 2 GPU-hours and 20GB memory total.

Some factors that led us to estimate higher computational requirements were that the models used in the paper were very large, such as gpt-4 and llama 70B. Additionally, because models' responses are non-deterministic, they can sometimes output very unexpected or long responses after a lot of usage; this could be because they are set to high temperatures as well. We made some efforts to reduce the requirements by considering these factors in our experiment. Firstly, we used smaller models and chose models without think blocks, to make sure generating responses is faster. We also set a max generation length and reduced the temperature to 0.2 to encourage concise responses in the correct format. Finally, we refactored our code to be written in such a way that we can rerun it from any point rather than needing to run it from start to finish each time, in case one part of the experiment didn't work correctly or the model didn't work.

5 Results

Our work supports the claims that it is possible to detect hallucinations in LLMs without external fact-checking, and here we will quantitatively discuss how effective the indirect and direct queries were at detecting hallucinations.

Firstly, here is the true hallucination rate of all the models we ran for the 1000 generated references during our reproduction.

5.1 The Reproduction Results

5.1.1 ROC curves

Here are the ROC curves for Gemma3 4b in the reproduction of the original experiment. All other plots for other models are in the appendix §B to account for space constraints.

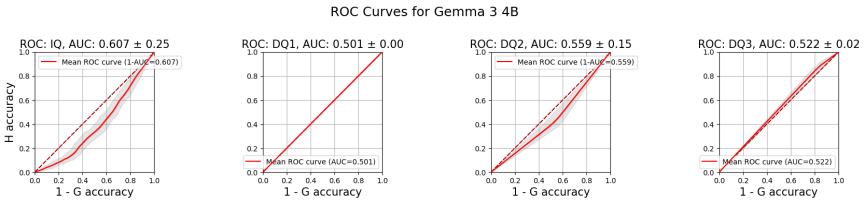

Figure 1: ROC curves for individual queries with Gemma3 4B

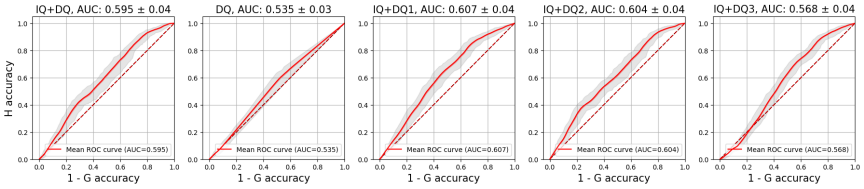

Figure 2: More ROC curves for combinations of queries with Gemma3 4B

In an ROC curve, the diagonal red line is the mean ROC curve for if a model randomly generated an answer that had equal probability of being hallucinated or grounded. So, the results in figures 1 and 2 show that for the individual queries for Gemma3 4B, the consistency scores approximated a random model - they also had an AUC of about 0.5 and 0.6. When combining the direct and indirect queries, they performed better with a curve higher than the mean line and AUC's still at around 0.6 but higher than before.

5.1.2 FDR curves

Here are the FDR curves for the models.

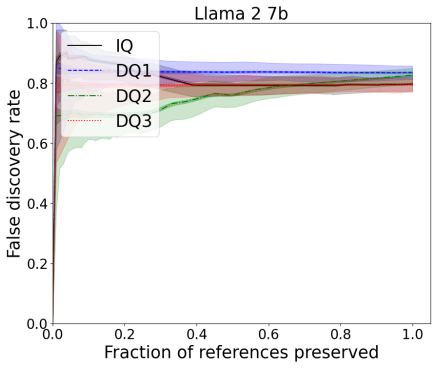

Figure 3: FDR curve for Llama2 7B

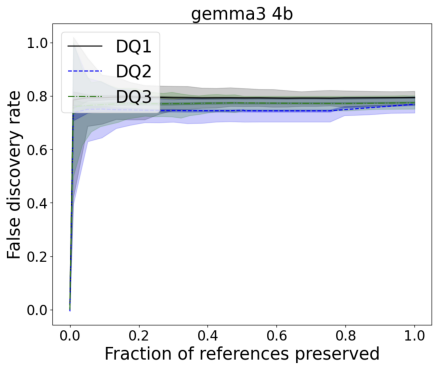

Figure 4: FDR curve for Gemma3 4B

For Figure 3, this shows that as the fraction of references preserved increases, the false discovery rate shoots up quite fast, meaning many of the references it labels as grounded are actually hallucinated. The results for Gemma3 4B (Figure 4) are much better, with the indirect query having the FDR increase gradually.

5.2 Extension Results

5.2.1 Airports Extension

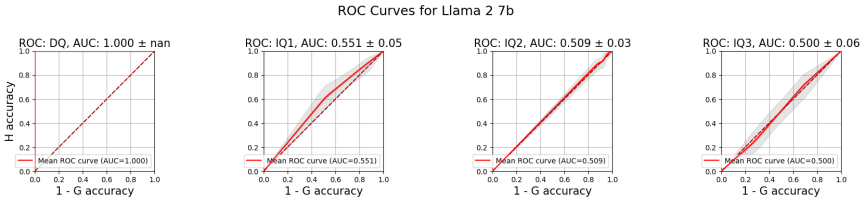

Figure 5: ROC curve for Llama2 7B for airports

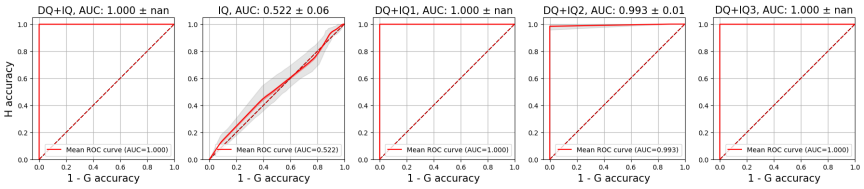

Figure 6: More ROC curves for Llama2 7B for airports

Figures 5 and 6 show how the ROC curve for DQ and the DQs and IQs combined was perfect - the model correctly detected hallucinations. It worked close to random for the individual IQs, but the direct query was very effective in detecting ground truth vs hallucinated airport codes. We see similar results for the rest of the models, as seen in the appendix §B.

Here is the plot of the FDR rate for Llama2 7B:

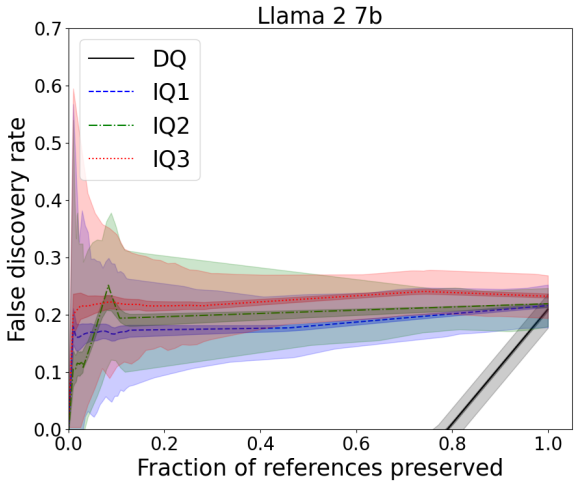

Figure 7: FDR curve for Llama2 7B for airports

Figure 7 shows a better FDR rate as the fraction of references preserved increases, the false discovery rate stays almost static. So, the model doesn't wrongly detect grounded references as it discovered more grounded references.

5.2.2 *Jeopardy Extension*

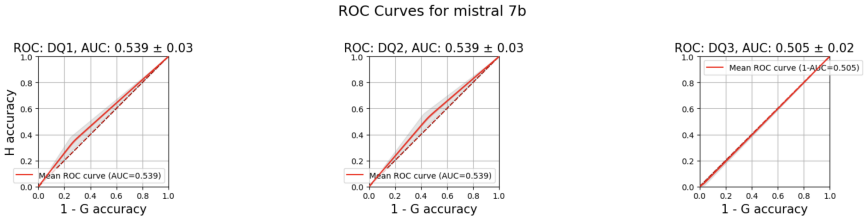

Figure 8: ROC curves for Mistral 7b for jeopardy

Figure 8 shows the ROC curves for mistral 7B for the jeopardy extension. Notice how for this extension, the consistency scores did not work well in detecting hallucinations, and the detection rate was about equal to what a random model would do. The other models performed similarly not well, as seen in the extension appendix §B.

Here in the figure is the FDR rate for the jeopardy extension for Mistral 7b.

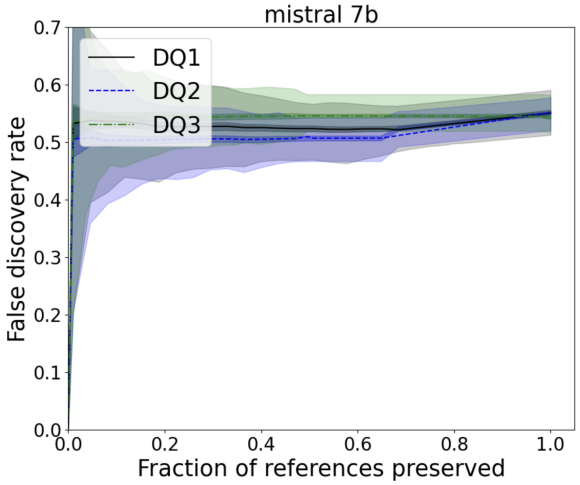

Figure 9: FDR curve for mistral 7B for jeopardy

This shows that the llama 7b model did not perform as well as hoped on the jeopardy set, The False Discovery rate is very high, around 0.6, which is not that great. But it is not as bad as expected, but does uncover some of the concerns of dealing with an unstructured/open ended data set like trivia questions.

6 Discussion

So what can we learn from the results. First off, we were not able to exactly reproduce the results of the paper. One major factor is that we could not run the experiment on the gpt4 or gpt3.5 models because of cost constraints (the paper mentioned it cost them $412). So, we ran the experiment on many smaller models, but these are much less advanced and performed worse than gpt models. We were able to support our original claims using the airport extension. We found that the direct query was very effective in detecting hallucinations. This could be because it is easier for models to know the IATA codes of popular world airports. However, we were not able to support our original claim with the jeopardy extension. It performed extremely poorly, with a very high FDR rate and the ROC curves approximating the mean. One reason for this is a major limitation with our experiment setup. Because LLM responses are non-deterministic and it is very difficult to moderate them into a specific format, in many of the responses the LLM simply responded with ”Sure, I can answer this question” without properly answering it. So, it automatically got counted as incorrect and wasn’t

able to generate a proper answer to the trivia question. But again, the unstructured nature of the trivia data set was something we thought would affect the model negatively, and the experiments showed it did.

We believe that a strength of our approach is attempting to use different types of datasets and run the claim on different queries. However, a weakness is that because we are trying to evaluate and compare LLM-generated results, they are very non-deterministic and skew our conclusions based on errors in generation. We conclude that in order to completely successfully reproduce the results of the paper, we need to use more advanced models like gpt-4o and gpt-o1, and run it on many more iterations to get rid of edge-case errrored generations. We do think that there is a potential of showing that hallucinations can be detected through consistency of responses, if we can find a way to consistently and accurately compare multiple responses.

6.1 What was Easy

The easiest part of the this project would be the process of developing the extensions and integrating them into the provided code base. The code base was well-documented, and the hierarchy of the repo was very well defined making the reproduction and the extension straightforward. Additionally, the authors did a very good job in the paper of not only building the motivation for why we it is important to explore hallucinations within LLM's, but also discussing how we can develop a process/metric to easily determine how often LLM's hallucinate.

6.2 What was Difficult

The hardest part of the experiment was reproducing the paper's results. Although the authors provided a well-documented and structured codebase, we couldn't access several APIs, models, and tools due to feasibility issues. This meant we couldn't replicate results involving OpenAI models or use the Bing API key as in the original work. Much of our effort went into finding alternative models to reproduce the findings without drastically altering the results. We eventually used the models described in **Section 4.1** from Ollama, but integrating them was difficult and took more time than expected. This limitation with OpenAI likely contributed to why we couldn't get strong results across multiple models, especially for the Jeopardy extension. Another difficulty was understanding the metrics and terms the authors used. Their explanations were not very clear, so more detail would help future readers. Overall, limited access to tools and a lack of detail in the paper made reproducibility a big challenge.

6.3 Recommendations for Reproducibility

Some recommendations we have for reproducing this report is:

1. As mentioned prior, it would be be helpful to include additional compatible models or clear guidance on substituting models to handle cases where original APIs (e.g., OpenAI) are unavailable or impractical.
2. Use widely recognized and clearly explained evaluation metrics. Include detailed explanations and examples in the paper to help others interpret results consistently.
3. Provide more detail on the environment specs they used to produce their results including hardware specs and software versions, to facilitate replication instead of just mentioning "we used a node with 8 V100 GPUs". Having metrics and checkpoints when generating the results would have given us metrics to look out for during the extensions.

Communication with Original Authors

We did not have any contact or communication with the original authors.

References

Ayush Agrawal, Mirac Suzgun, Lester Mackey, and Adam Tauman Kalai. Do language models know when they're hallucinating references? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL 2024)*, 2024. URL https://arxiv.org/pdf/2305.18248v3.pdf.

Serdar Altan. Countries by continent. https://www.kaggle.com/datasets/hserdaraltan/countries-by-continent, 2019. Accessed: 2025-06-02.

Li Du, Yequan Wang, Xingrun Xing, Yiqun Yao, Xiang Li, Xin Jiang, and Xuezhi Fang. Quantifying and attributing the hallucination of large language models via association analysis. In *arXiv preprint arXiv:2309.05217*. arXiv, 2023. URL https://arxiv.org/pdf/2309.05217.pdf.

Assad Muhammad. Airports worldwide with iata codes. https://www.kaggle.com/datasets/assadmuhammad/airports-worldwide-with-iata-codes, 2021. Accessed: 2025-06-02.

OpenAlex. Openalex api overview. https://docs.openalex.org/how-to-use-the-api/api-overview, 2024. Accessed: 2025-06-02.

Bernard Rous. Major update to acm's computing classification system. *Communications of the ACM*, 55(11):12, 2012.

Bojan Tunguz. 200,000+ jeopardy! questions. https://www.kaggle.com/datasets/tunguz/200000-jeopardy-questions, 2019. Accessed: 2025-06-02.

A Appendix: More Details on Evaluation Metrics

A.1 Determining Thresholds to plot in Graphs

The evaluation metrics involved generating FDR and ROC curves. These plotted values on different thresholds for the G/H classification. Here we would like to clarify what "different thresholds" we used. For each generated reference and type of query, we have consistency scores. So, for each type of query, there are 1000 consistency scores across all the references. We used each consistency score as a value for the threshold, as well as the expected [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0] values. This allowed us to plot an entire curve as we had many different consistency scores.

A.2 Calculating True and False Positive Rates

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

Where TP = true positives = the number of generated responses that were labeled grounded that were actually grounded, FP = false positives = the number of generated responses that were labeled grounded that were actually hallucinated, TN = the number of generated responses that were labeled hallucinated that were actually hallucinated, and FN = the number of generated responses that were labeled hallucinated that were actually grounded.

B More Extension Results

B.1 Airports Extension Results

These are the graphs for the other models we ran the airport extension on. The ROC curves are approximately the linear diagonal line, meaning they do not detect hallucinations better than a randomly generating model would. Mistral 7b does work slightly better, with higher curves for the indirect queries and an AUC of 0.671. The FDR curve is also better, as the false discovery rate doesn't increase much as the fraction of references preserved increases.

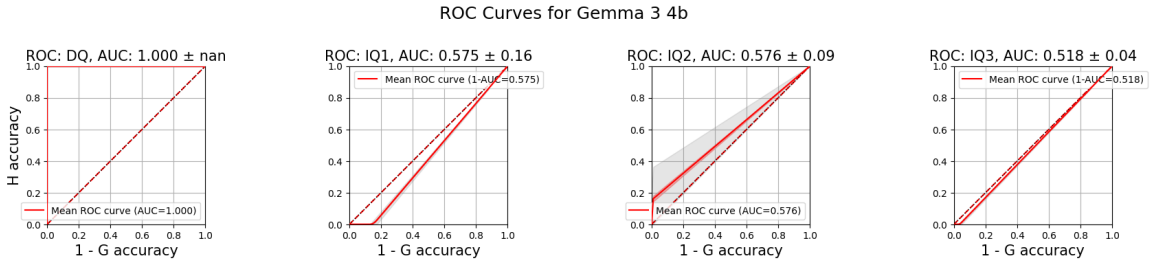

Figure 10: ROC curve for Gemma3 4B for airports

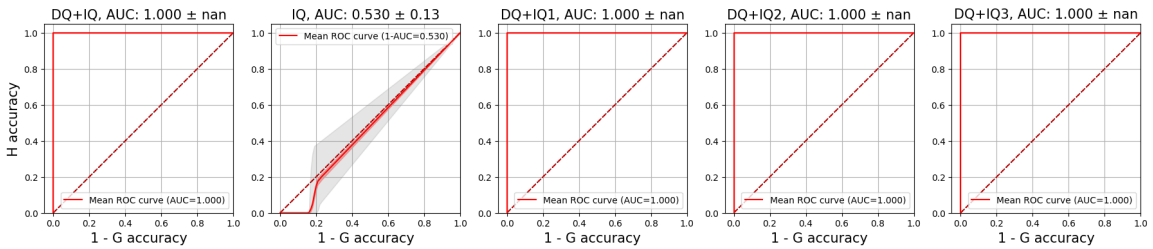

Figure 11: More ROC curves for Gemma3 4B for airports

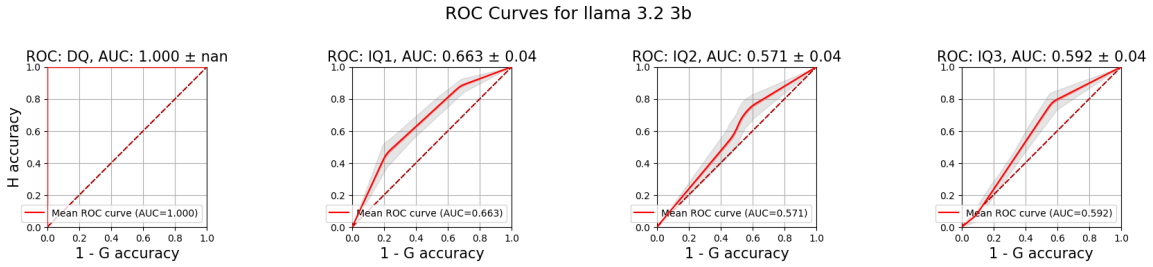

Figure 12: ROC curve for Llama 3.2 3b for airports

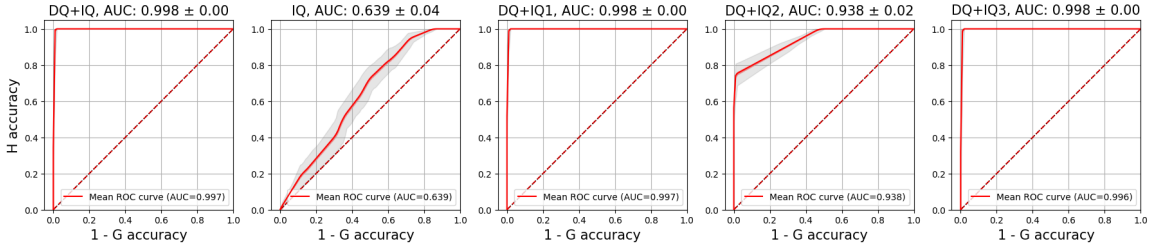

Figure 13: More ROC curves for Llama 3.2 3b for airports

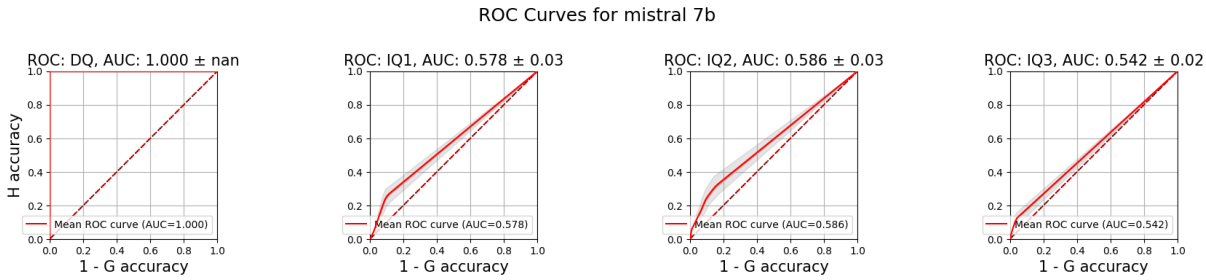

Figure 14: ROC curve for Mistral 7b for airports

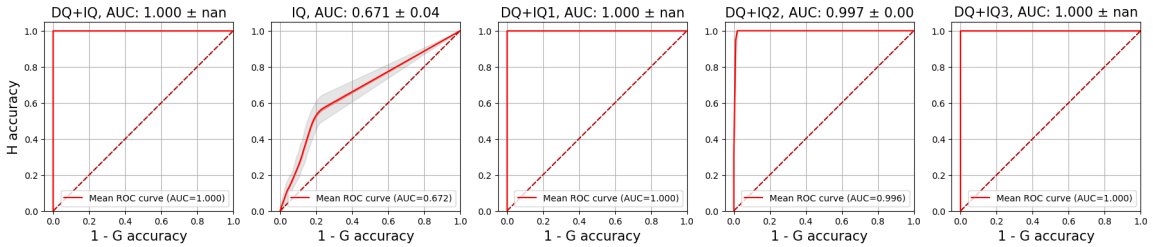

Figure 15: More ROC curves for Mistral 7b for airports

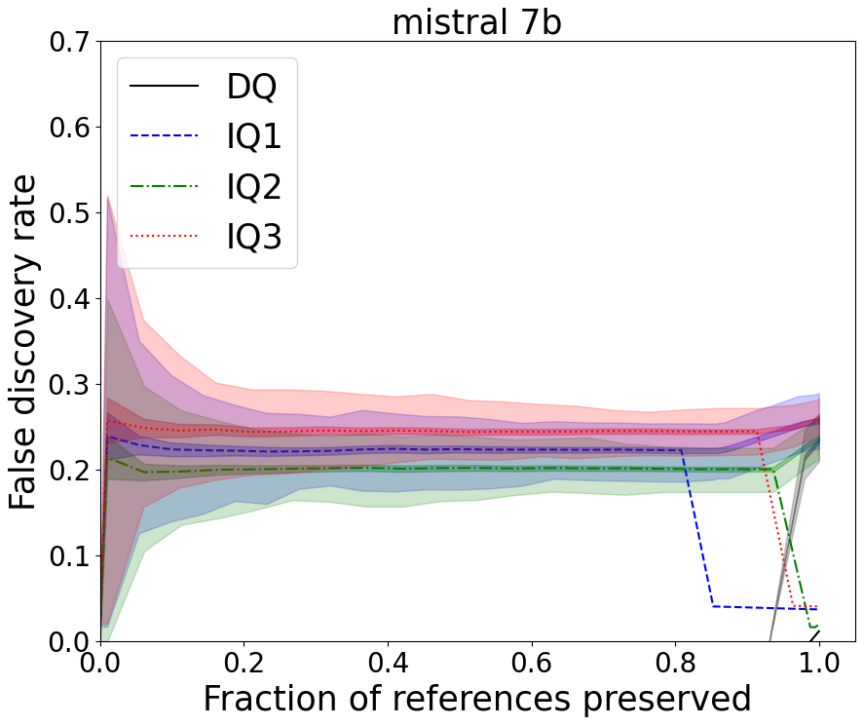

Figure 16: FDR curve for mistral 7b for airports

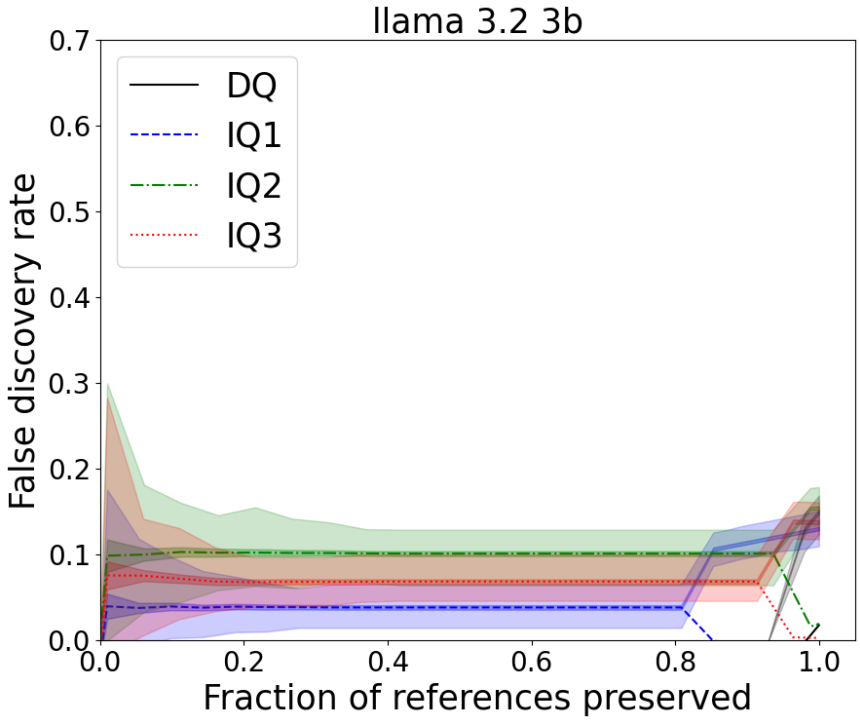

Figure 17: FDR curve for llama 3.2 3b for airports

B.2 Jeopardy Extension Results

These are the graphs for the other models we ran the jeopardy extension on. The ROC curves are approximately the linear diagonal line, meaning they do not detect hallucinations better than a randomly generating model would. The FDR rate is also extremely high (figure 23) because of issues with comparing generated responses with LLMs, as they didn't return deterministic responses.

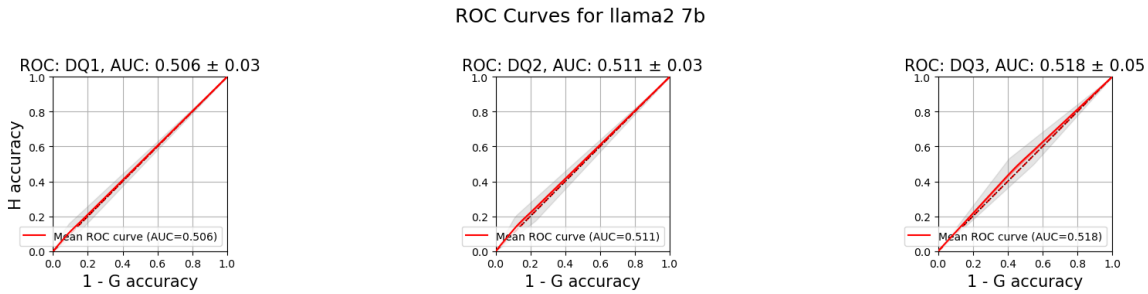

Figure 18: ROC curve for llama 7b for jeopardy

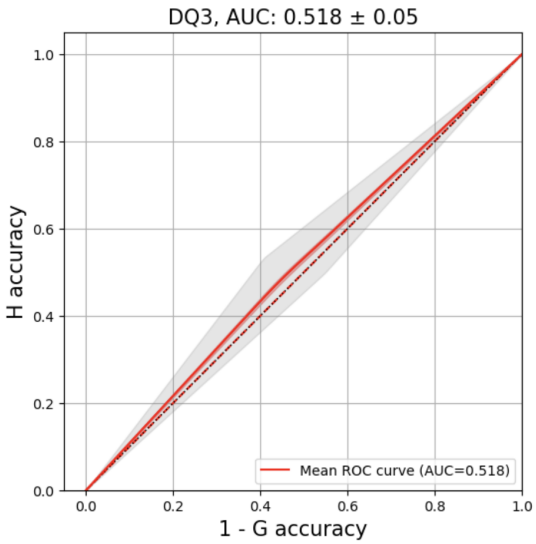

Figure 19: More ROC curves for llama 7b for jeopardy

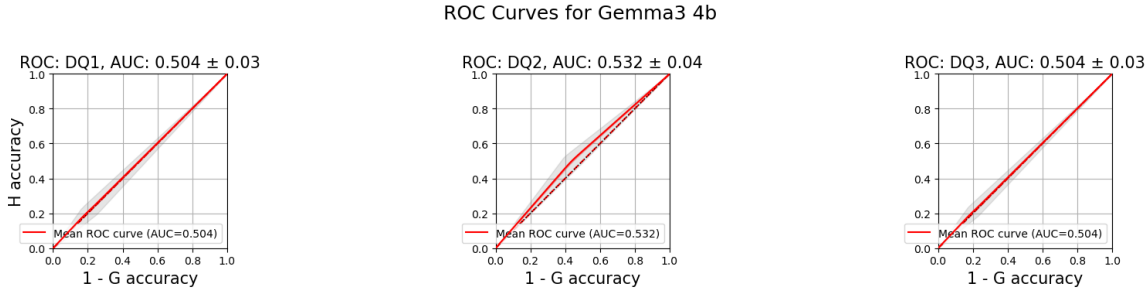

Figure 20: ROC curve for Gemma3 5b for jeopardy

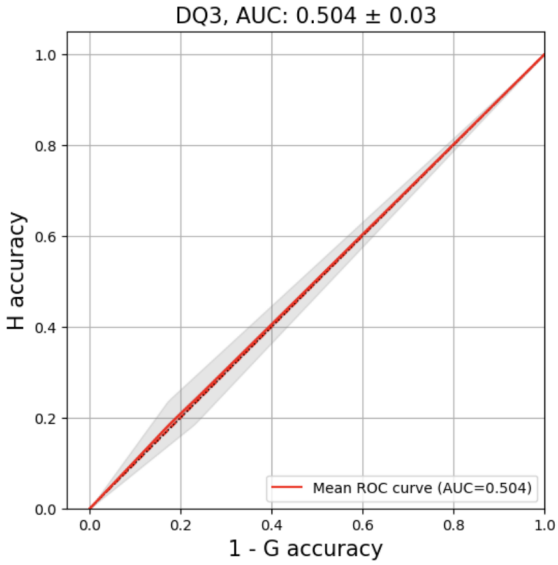

Figure 21: More ROC curves for Gemma3 4b for jeopardy

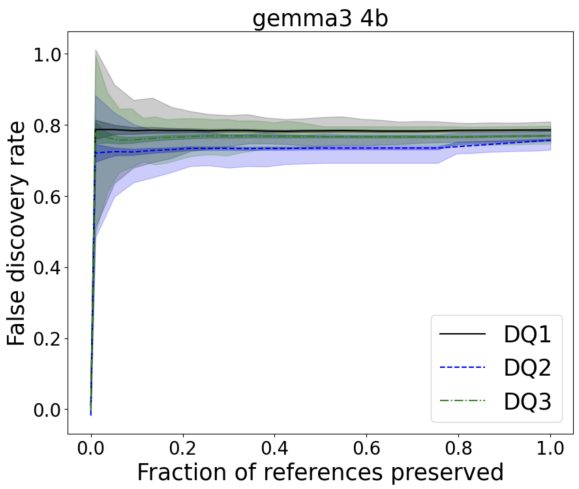

Figure 22: FDR curve for gemma3 4b

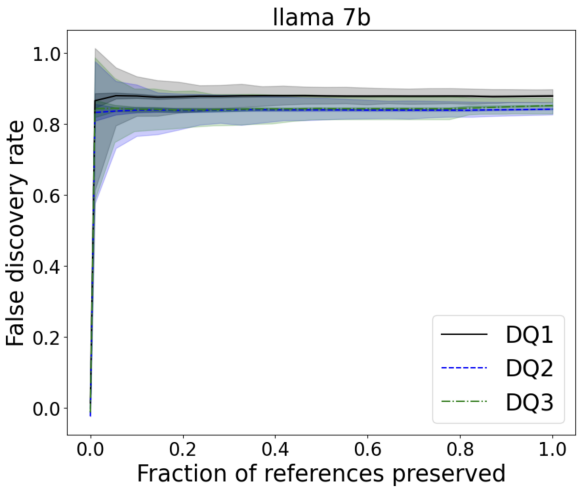

Figure 23: FDR curve for llama 7b

C Previously Missed Reproduction Images

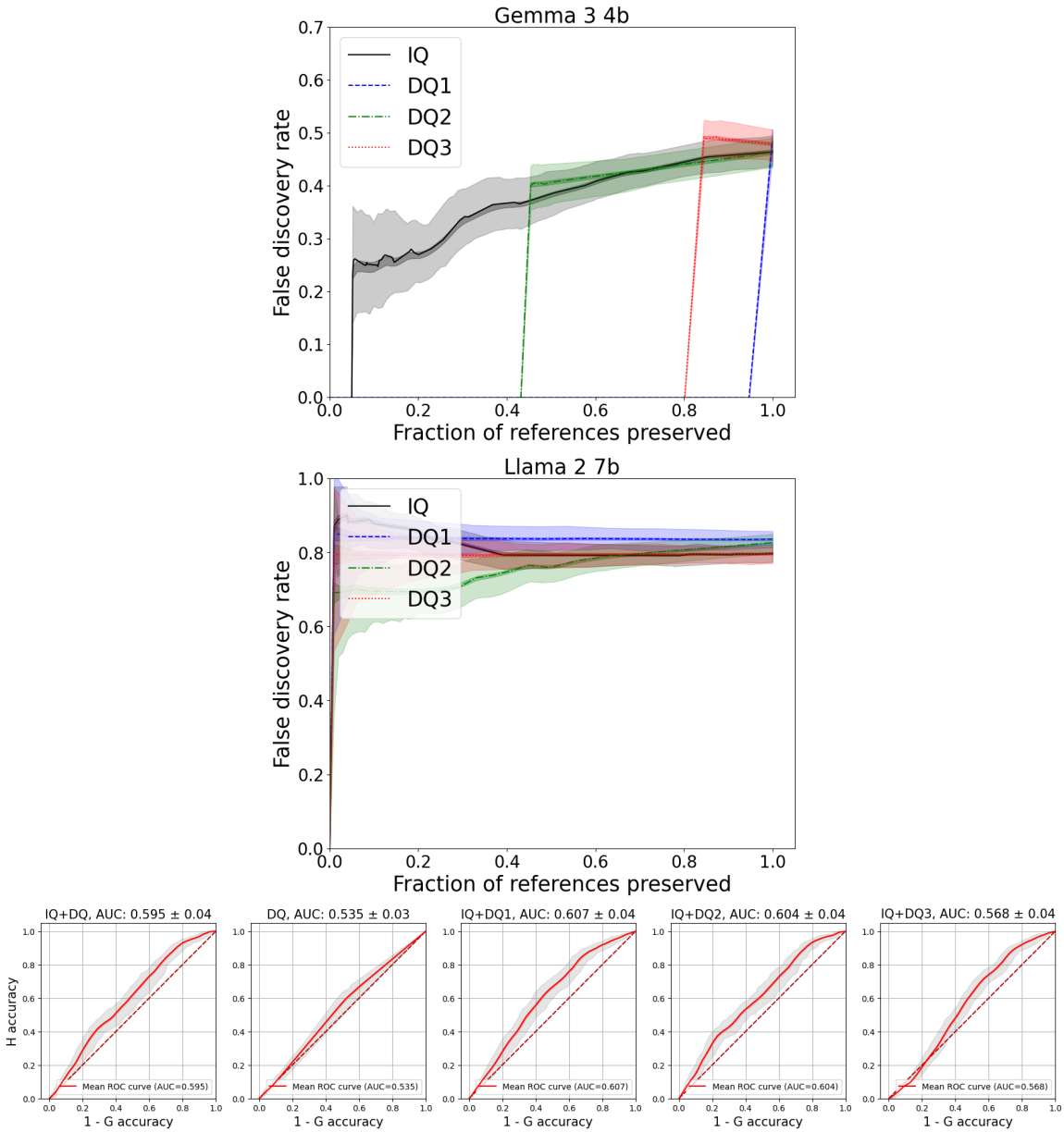

Figure 24: All missed photos

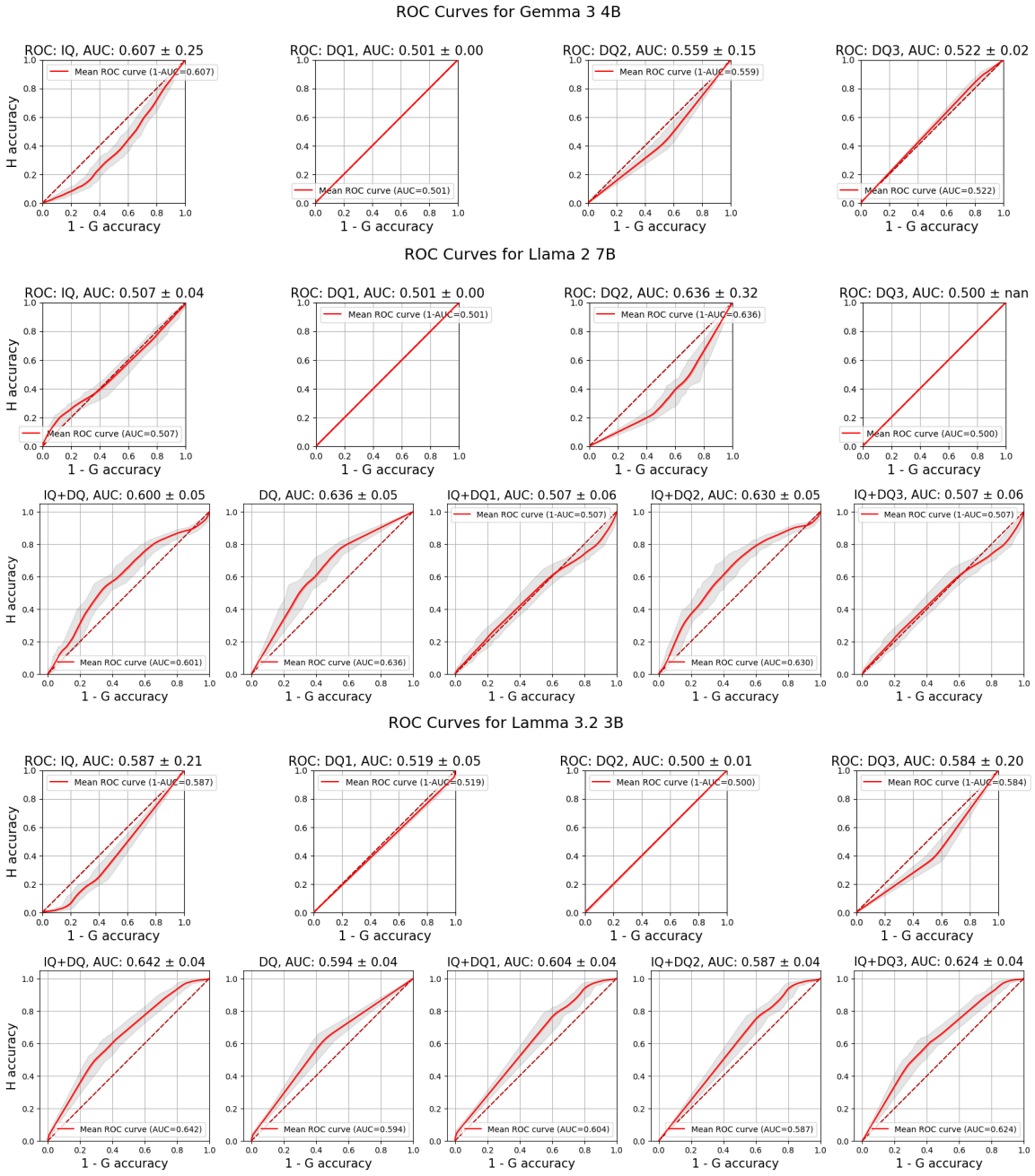

Figure 25: All missed photos