TOWARDS LIGHTER AND ROBUST EVALUATION FOR RETRIEVAL AUGMENTED GENERATION

Alex-Răzvan Ispas
Paris-Saclay University, BNP Paribas CIB
alex.razvan.ispas@gmail.com

Charles-Élie Simon
BNP Paribas CIB
charleselie.simon@bnpparibas.com

Fabien Caspani
BNP Paribas CIB
fabien.caspani@bnpparibas.com

Vincent Guigue
AgroParisTech
vincent.guigue@agroparistech.fr

ABSTRACT

Large Language Models are prompting us to view more NLP tasks from a generative perspective. At the same time, they offer a new way of accessing information, mainly through the RAG framework. While there have been notable improvements for the autoregressive models, overcoming hallucination in the generated answers remains a continuous problem. A standard solution is to use commercial LLMs, such as GPT4, to evaluate these algorithms. However, such frameworks are expensive and not very transparent. Therefore, we propose a study which demonstrates the interest of open-weight models for evaluating RAG hallucination. We develop a lightweight approach using smaller, quantized LLMs to provide an accessible and interpretable metric that gives continuous scores for the generated answer with respect to their correctness and faithfulness. This score allows us to question decisions' reliability and explore thresholds to develop a new AUC metric as an alternative to correlation with human judgment. We also provide a script for the corrected dataset which is available at: GitHub Repository

1 INTRODUCTION

Large Language Models (LLMs) have advanced the field of Natural Language Processing (NLP) in recent years Achiam et al. (2023); Touvron et al. (2023); Jiang et al. (2024). However, some questions require information outside the knowledge scope of the model. Therefore, Retrieval Augmented Generation (RAG) Lewis et al. (2020) was proposed to enhance the quality of the answers for questions by retrieving information from a relevant knowledge base. RAG reliability remains a critical concern, particularly due to hallucinations in the generated answers. While much effort has been dedicated to improving model accuracy, a structured evaluation framework that explicitly addresses hallucination detection is still needed.

In general, we want to assess the quality of an LLM answer by comparing it to a ground truth. Previous approaches used token-based metrics such as Exact Matching (EM) Rajpurkar et al. (2016), which are fast to compute but not robust to semantic variations. Human evaluators are considered the most consistent approach for open evaluation, but they require a lot of resources Islam et al. (2023). Information extraction approaches can be used to compare the facts in the generated response with a ground truth Jayanthi et al. (2021), but such systems are quite domain-sensitive, and the metric may not be very robust to domain changes. NLI approaches Liu et al. (2021) can be used to validate statements, but typically in the context of simple sentences or using a combinatorial approach. LLM evaluators have recently risen in popularity, and they have been compared with human evaluators regarding agreement on different benchmarks such as MT-bench Zheng et al. (2024). Factual few-shot prompting evaluation methods Es et al. (2023); Zhang et al. (2023); Manakul et al. (2023) can check the level of truth of each answer statement while avoiding the search complexity of a knowledge graph Sansford et al. (2024).

While factual evaluation exhibits progress, most proposed methods rely on enterprise models such as GPT4 Es et al. (2023); Manakul et al. (2023), making the reproduction of experiments challenging

due to financial and ethical considerations, such as the protection of sensitive data. Since numerous open-source lightweight LLMs offer comparable performance and are more accessible with quantization, we explore their capabilities with respect to enterprise models in terms of human agreement. In the meantime, we look for interpretability by providing an analysis at the statement level for each answer.

Our contributions are as follows:

- Proposing a reproducible framework for evaluating RAG-generated answers based on quantized models.
- Defining a transparent and interpretable metric based on the decomposition and annotation of the response in terms of statements.
- Providing the community with a corrected dataset for RAG evaluation.
- Analyze the behaviour of the continuous metric to build a new alignment score with human judgment based on AUC.

2 Related Work

Previous research on Question-Answering (QA), such as SQuAD Rajpurkar et al. (2016), relied on deterministic approaches like exact matching (EM) and F1-score. The main advantages of deterministic metrics are that they guarantee correctness, are interpretable, and can be easily generalised. Although traditional metrics like BLEU Papineni et al. (2002) and ROUGE Lin (2004) perform well for specific tasks such as translation and summarization, they are not robust to task variations. Additionally, metrics that rely on token overlap cannot capture the semantic similarities between two answers. The standard method of evaluating text quality is through human evaluation. Human annotators have the advantage of understanding the context and human preferences much better. However, this method can introduce human bias, and the annotation process requires a significant amount of time and budget.

Large Language Models used as evaluators are designed to act as a bridge between the previous approaches. They are capable of handling a wider range of open-ended NLP tasks that are in line with human preferences and that are more difficult to evaluate using traditional metrics Zheng et al. (2024). The analysis conducted by Chiang & Lee (2023) highlights that LLMs can consistently assess text quality, like humans, when provided with the same instructions and examples. Unlike human evaluation, LLMs used as judges are a scalable alternative that minimizes the need for human involvement during the evaluation process and provides faster model iterations. However, positional bias makes the comparison of the generated answers challenging Wang et al. (2023). According to Zheng et al. (2024), there are three different ways of structuring the prompts. In *pairwise comparison*, the model receives a question and two answers and is asked to choose the better one. For *pointwise scoring*, the model is given a single answer and is asked to assign a score Manakul et al. (2023). *Reference-guided scoring* provides the judge with a reference solution in addition to the question and the answer Es et al. (2023). All of them can enhance their reasoning precision by applying chain-of-thought Wei et al. (2022).

In the case of RAG evaluation, Es et al. (2023) tried to overcome positional bias Wang et al. (2023) by using factual evaluation Manakul et al. (2023), increasing the faithfulness accuracy of GPT3.5 from 0.72% to 0.95%. Other research Adlakha et al. (2023) compared the deterministic metrics and the LLM evaluators by computing the correlation with the human evaluators. They emphasised that bag-of-tokens recall is highly correlated with humans when the verbose information of the answer is ignored during the evaluation. At the same time, contextual embedding transformers such as BERTScore (BertS) have a lower correlation when they are used to calculate the recall Zhang et al. (2019). Overall, GPT4 remains one of the highest-rated evaluators in most of the previous studies Zheng et al. (2024).

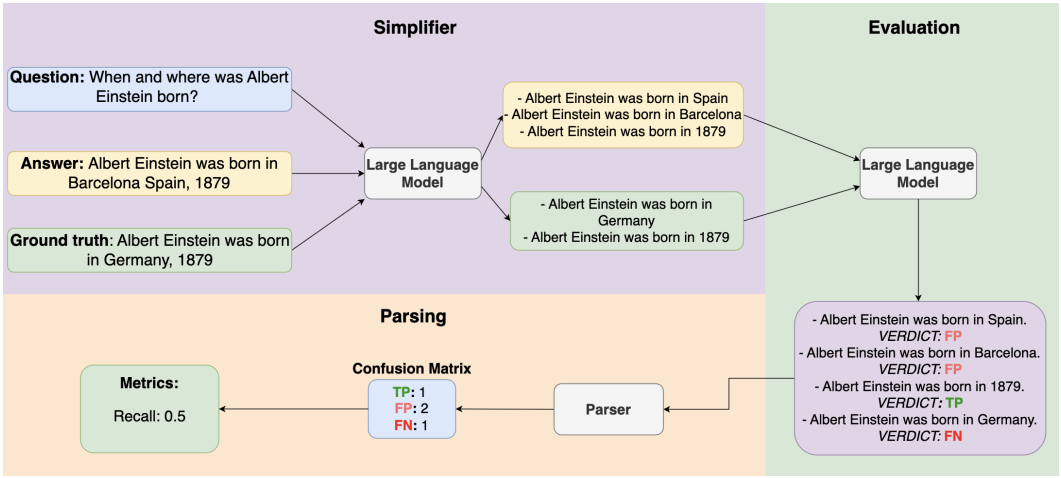

Figure 1: Evaluation pipeline for answer correctness. First, the simplifier extracts the statements of the answer and the ones of the ground truth. Afterwards, the evaluator labels the statements according to the definitions. Finally, the parser extracts the labelled statements and calculates the metric.

3 METHODOLOGY

3.1 FACTUAL EVALUATION

Following the evaluation methods emphasized by Manakul et al. (2023) and Es et al. (2023), we will evaluate the answers using metrics that can be grounded in *facts*. We define a *fact* or a *statement* as a declarative sentence that conveys information which can be either true or false. For the example shown in Figure 1, it can be noticed that the answer *Albert Einstein was born in Barcelona Spain, 1879* was split into three statements: *Albert Einstein was born in Spain*, *Albert Einstein was born in Barcelona*, *Albert Einstein was born in 1879*. The label of each statement will be assigned with respect to either a ground truth or a context passage, depending on the calculated metric. By breaking the generated answers into smaller units, we can reduce the complexity of evaluating the truthfulness of each individual fact.

3.1.1 CORRECTNESS

The first metric is the *answer correctness*. In a RAG setting, an answer is considered *correct* if the statements from the ground truth directly support the statements of the answer. Section B.1 provides the definitions for the labels which can be assigned to the statements. The final score can be calculated as either the recall or the f1 score of the classified statements. The recall is a softer version of correctness because it ignores false positive statements, which are verbose information that appears in the answer but not in the ground truth. For instance, if an LLM needs to answer the question *In which country was Albert Einstein born?*, a possible answer is *Germany*. However, an LLM would provide more information, such as Germany's population, which is irrelevant to the ground truth answer. The *harsher* version is the f1-score, which penalises answers that contain verbose information. In general, the f1-score is more suitable for domain adaptation scenarios as the answer should match the structure of the ground truth. Meanwhile, the recall is ideal for scenarios where we seek specific information without being concerned about additional details. Since we work in a few-shot setting, and the chosen dataset was not annotated without considering the verbose information, we will use the recall and ignore the additional information.

$$\text{Recall} = \frac{TP}{TP + FN} \qquad \text{F1} = \frac{TP}{TP + 0.5 \cdot (FP + FN)}$$

3.1.2 FAITHFULNESS

Instead of quantifying the hallucination of an answer, we are going to quantify its inverse: the *faithfulness* score. We consider that an answer is *faithful* if its claims can be directly inferred in the retrieved context. For the example in Figure 4, the context passage that mentions information about Einstein's childhood in Germany makes the statements that mention Barcelona and Spain not faithful. The final score can be calculated as the precision of the *passed* and *failed* statements. Section B.2 provides the definitions of the two labels.

$$Precision = \frac{Passed}{Passed + Failed}$$

3.2 FORMALIZING

Our LLM evaluation framework consists of three components. The first one is the *simplifier* S, which transforms a text into a set of elementary statements. The second one is the *evaluator* E, which assesses the matching between the system's response and the ground truth. Both the *simplifier* and the *evaluator* phases are performed by an LLM. The third one is the *parser* P, which extracts the classified statements to calculate the final score. Figure 1 showcases the steps taken by the evaluation pipeline to calculate the answer's correctness.

Given a RAG answer t and ground truth t':

- S : Transform t and t' into lists of facts as in Manakul et al. (2023); Es et al. (2023), which are expressed in the form of elementary sentences that we call *statements*. $S : t \mapsto \{s_i\}_{i=1}^{N}$
- E : Use few-shot examples to compute semantically which facts are supported or not between the two texts. By default, s is binary since statements can be supported in both directions. However, we are switching to a 3-label system because non-support is not interpreted the same way in both directions: (TP: $s_i \in \{s'_j\}_{j=1}^{N'}$), (FP: $s_i \notin \{s'_j\}_{j=1}^{N'}$), (FN: $s'_j \notin \{s_i\}_{i=1}^{N}$).
- P : As the E step is performed by an LLM, the TP, FP, FN labels are in text format and must be extracted before we can calculate the final metric.

The final score of **correctness** is the recall of the labelled statements. This score will be compared with the binary human judgment.

In the case of **faithfulness**, given a context c and one faithful and unfaithful answer, t_g and t_p, the task is to distinguish the most faithful answer.

- S : Simplify t_g and t_p to get their statements $\{s_{g,i}\}_{i=1}^{N}$ $\{s_{p,j}\}_{j=1}^{M}$
- E : Evaluate the statements individually with respect to the given context $(\{s_{g,i}\}_{i=1}^{N}, c)$ and $(\{s_{p,i}\}_{i=1}^{M}, c)$ to get binary labels :(TP: $s_i \Leftarrow c$), (FP: $s_i \not\Leftarrow c$)
- P : Parse the final labels and compute the precision scores to rank the two answers. This setting may lead to *ties* which are discussed in Section 4.2

We are mainly focused on cases where the golden passage was provided to generate the answer. Like this, any inaccuracy or hallucination in the generated answer can be attributed to the LLM rather than the retriever. The prompts used in each phase can be consulted in Section D of the Appendix.

3.3 PARSER

The chain-of-thought can increase the reasoning abilities of the LLMs Wei et al. (2022). However, there can still be variations in the final results, especially when working with quantized LLMs that have lower precision. Therefore, instead of letting the LLM evaluator calculate the final metric, we parse the labelled statements and manually compute the metrics.

We propose two options to parse the output. The first option is the deterministic approach, which tries to match the labelled statements through a regular expression. This method is fast to compute but can create errors in the final result if the generated answers have variations. The second approach

is the constrained generation, which forces the LLM to generate an output that respects a JSON schema. Given a JSON structure, a finite automata is created. During the generation process of each token, a mask will be applied on each invalid token during the respective step in the automata. The process continues until the automata reaches an end state which represents a valid generation that respects the structure. Once the statements are extracted, the confusion matrix can be calculated deterministically. This method implies an additional generation. Consequently, it can take more time to calculate the final metric. At the same time, it is non-deterministic due to its sampling nature. In this work, we will perform constrained generation with the help of the Outlines library Willard & Louf (2023). Figure 5 offers an example of the two approaches. The JSON schemas for *correctness* and *faithfulness* are presented in Section C.2.

4 DATASETS

4.1 NATURAL QUESTIONS

For evaluating answer correctness, we used InstructQA Adlakha et al. (2023), which contains questions extracted from three QA datasets: Natural Questions Kwiatkowski et al. (2019), HotpotQA Yang et al. (2018) and TopioCQA Adlakha et al. (2022). More precisely, we used the subset from Natural Questions, which included 100 questions answered by four different LLMs, generating 400 samples in the oracle setting of RAG. Subsequently, a group of students annotated the LLM answers as either correct or incorrect, reaching an inter-annotator agreement of 92.42%. As the provided data contained mismatching labels, we carefully reviewed all 400 samples and rectified the discrepancies in the annotations. Some of the most common errors included labelling questions as correct, even when the answer and the ground truth contained completely different information. Furthermore, we excluded one question due to its unclear formulation, rendering it unanswerable even for humans. The final reannotated dataset comprised 396 samples.

In this setting, given the question, the generated answer and the ground truth, the evaluator needs to identify if the answer is correct or incorrect. This dataset was used to calculate the correlation with respect to the human annotators. Thus, we will use the Spearman and Kendall correlation. Unlike Pearson correlation, Spearman's ρ and Kendall's τ are non-parametric rank correlation measures. That means we can assess the relationship between the human and the LLM score by using a monotonic function rather than assuming a linear relationship.

The human annotators assign a score of 1 for correct and 0 for incorrect, while the LLM evaluator assigns a score between 0 and 1 depending on the level of truth of each statement. Therefore, we will calculate the average F1 score for different threshold levels (F1 AUC). Unlike the Area Under the Curve of Precision and Recall, which shows the balance between the two metrics, F1 AUC can quantify how well the evaluator can separate the score distributions of the correct and incorrect answers by penalising the outliers at each threshold:

$$F1_{\text{AUC}} = \frac{1}{10}\sum_{i=0}^{10} F1\left(d\left(\mathbf{r}, \frac{i}{10}\right), \mathbf{h}\right), \qquad d(r, \text{th}) = \begin{cases} 1, & \text{if } r \geq \text{th}, \\ 0, & \text{if } r < \text{th}. \end{cases}$$

where r is the correctness score of the pipeline, h is the ground truth and th is the current threshold.

4.2 WIKIEVAL

For evaluating faithfulness, we will use the dataset proposed by Es et al. (2023). The dataset contains questions formulated from 50 Wikipedia pages created in 2022. GPT3.5 was used to answer the questions once given context and once not given any context. Afterwards, two human annotators had to decide which one of the two answers was more faithful, reaching a human agreement of 95%.

In the setting of WikiEval, given a question, the two answers and the context, the evaluator needs to identify which one of the two answers is more faithful. Since we want to compare with the human evaluators, the accuracy score is the number of times the LLM evaluator succeeded in identifying the faithful response divided by the total number of questions. That could also be quantified as the total number of times the good answer's faithfulness score was greater than the faithfulness score of the poor answer.

One issue that can be encountered are the *ties*. A *tie* is encountered when the faithfulness scores of the *good answer* and the *poor answer* are **the same**. Therefore, we propose a score system to handle this problem. Whenever the faithfulness score of the good answer is greater than the faithfulness score of the poor answer, it receives 1 point. If the scores are equal, we assign a partial point of 0.5. Otherwise, it receives 0 points. We provide scores for three scenarios. The worst case is calculated by assigning 1 point only if the faithfulness score of the good answer is strictly greater than the poor ones. The middle case of faithfulness is the sum of all points divided by the number of questions. The best case assigns 1 point even for the *tie* scenarios:

$$s(a_1, a_2) = \begin{cases} 1 & \text{if } faith(a_1) > faith(a_2) \\ 0.5 & \text{if } faith(a_1) = faith(a_2) \\ 0 & \text{if } faith(a_1) < faith(a_2) \end{cases}$$

$$worst_{faith} = \frac{\sum_{i=1}^{n} 1_{[faith(good_i) > faith(poor_i)]}}{n}$$

$$middle_{faith} = \frac{\sum_{i=1}^{n} s(good_i, poor_i)}{n}$$

$$best_{faith} = \frac{\sum_{i=1}^{n} 1_{[faith(good_i) \geq faith(poor_i)]}}{n}$$

5 EXPERIMENTS

The experiments were conducted only on quantized LLMs. For 4-bit precision, we used V100, while the 16 float precision LLMs were running on an A100. The temperature was set to 1 for each LLM evaluator to favour sampling. For each phase of the evaluation pipeline, the LLMs received few-shot examples to learn how to split the answers into statements and how to give a verdict. We implied only two families of LLMs: Llama and Gemma. For the parsing procedure, we used two regular expressions and one constrained generation procedure presented in C. Since the constrained generation implies copying each token of a statement in a list, that would require too much time during the sampling process. Therefore, we map each statement to a single token to reduce the sampling process, as shown in Section C.3. An experiment lasted between 4 and 8 hours, depending on the chosen parsing strategy and the size of the model. model. Each LLM evaluator was compared with at least one deterministic metric on human agreement. We consider an evaluator to be robust if its judgement aligns with that of human annotators through correlation and if it has a high separability between the scoring distribution of the good and bad answers through the F1-AUC for correlation and the score system defined in Section 4.2 for faithfulness.

Results Correctness Table 1 displays the experiments conducted for answer correctness on the Natural Questions subsample of InstructQA. The baseline is the bag-of-tokens recall, which we recalculated to emphasise that the correlation was not altered drastically compared to the original dataset results declared by Adlakha et al. (2023). In the original paper, the Natural Question subsample has a Spearman ρ of 55.02%, while our reannotated data has a correlation score of 56.89% for the same subsample. Since the correlation with the human annotators does not emphasise how well an evaluator can separate the distribution of correct and incorrect answers, we calculate the F1 AUC to estimate how well the evaluator performs at different threshold levels. We prefer the F1 score over the Precision-Recall AUC because it captures the capability of the evaluator to minimize the False Positives and False Negatives rather than the ability to balance the tradeoff between precision and recall. The best configurations have the mean and the standard deviation calculated on five distinct reruns.

For the parsing procedure, it could be noticed that constrained generation is not always the optimal solution. While Llama3 8B benefited from constrained generation parsing, reaching an AUC F1 of 90.57% and a Spearman correlation of 44.41, the other LLMs had better results using regular expressions. This result emphasises that deterministic parsing can be a faster alternative which does not require guided generated answers.

Although the bag-of-tokens recall has a high correlation with human annotators, the AUC F1 is much lower compared to Gemma2 9B and Llama3 70B. In Figure 2, we can see the distribution of correct and incorrect answers for the evaluators, which use the second regular expression as the parser, and the distribution of the bag-of-tokens recall. While the bag-of-tokens recall can separate the distribution of the incorrect answers, the scores of the correct answers are more dispersed, resulting in a lower F1 AUC when the threshold is increased. Bag-of-tokens is a lexically based metric, and if

Table 1: Correctness and faithfulness experiments. The evaluator represents either a deterministic metric or an LLM. BoT stands for bag-of-token. L3 stands for Llama3 and G2 for Gemma2. The Parsing column denotes the parsing strategy: R1 for the first regular expression, R2 for the second one and C for constrained generation. F1 AUC is the area under the curve of the F1-score. ρ and τ are the Spearman and Kendall correlation. The last columns are the lower bound, the mean and the upper bound for the Wikieval questions. The best configurations are in bold

Pipeline		Correctness			Faithfulness		
Evaluator	Parsing	F1 AUC	ρ	τ	Worst	Middle	Best
BoT Recall	N/A	88.78	56.89	52.89	N/A	N/A	N/A
RAGAS (GPT3.5-Turbo)	N/A	N/A	N/A	N/A	N/A	N/A	0.95
K-precision	N/A	N/A	N/A	N/A	N/A	N/A	0.96
L3 8B 4 bit	R1	87.44	30.21	28.54	0.74	0.84	0.94
L3 8B 4 bit	R2	89.62	37.36	36.54	0.78	0.85	0.92
L3 8B 4 bit	C	90.57	44.41	43.28	0.72	0.89	1.0
L3.1 8B 4 bit	R1	86.14	36.89	34.34	0.74	0.79	0.84
L3.1 8B 4 bit	R2	86.47	40.02	37.74	0.78	0.82	0.86
L3.1 8B 4 bit	C	75.33	30.84	29.50	0.72	0.83	0.94
G2 9B 4 bit	R1	92.20	52.55	50.21	**0.92**	**0.94**	**0.96**
G2 9B 4 bit	R2	**93.83 ±0.27**	**62.06 ±1.83**	**60.49 ±1.54**	**0.82**	**0.88**	**0.94**
G2 9B 4 bit	C	89.05	55.01	52.10	0.82	0.90	0.98
L3 70B 16 bit	R1	86.42	49.44	45.41	0.94	0.95	0.96
L3 70B 16 bit	R2	**92.72 ±0.20**	**63.59 ±1.51**	**60.55 ±1.39**	**0.94**	**0.95**	**0.96**
L3 70B 16 bit	C	77.21	40.52	37.23	0.88	0.91	0.94

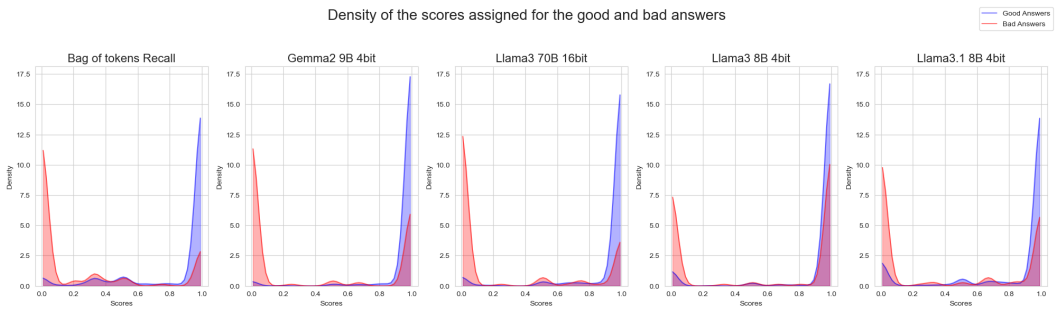

Figure 2: The density distribution plots of the correctness evaluators that use the second regular expression as parser. The distribution of the correct and incorrect answers are marked with blue and red, respectively. The labels were chosen according to the human annotations.

the generated answer does not use the exact words as the ones in the ground truth, it cannot capture the semantic similarities, resulting in a penalised score.

One impressive result is the Gemma2 9B, which uses the deterministic parser, as it performs similarly to Llama3 70B, although it is about seven times lighter and has four times lower precision. In Figure 2, we can observe that Gemma2 is usually very confident in its decisions, assigning a score of either 0 or 1 almost all the time. However, it misclassified a considerable percentage of incorrect answers by giving them a score of 1. Llama3 70B minimizes the number of wrong answers that have a score of 1, but the distribution of the incorrect answers remains dispersed. Nevertheless, both Gemma2 9B and Llama3 70B can maintain the distribution of good answers above the score of 0.5. Unfortunately, Llama3 8B cannot distinguish between correct and incorrect answers, assigning a score of 1 almost every time. Llama3.1 has both distributions sparsed between 0 and 1.

Results Faithfulness Table 1 displays the results for the faithfulness experiments conducted on WikiEval. Each experiment has three calculated scores according to the worst, middle and best case. Since we had minimal information regarding how the final accuracy score was calculated in the original paper Es et al. (2023), we assumed that the best-performing seed was reported, that being the best case score. We also assume that a good evaluator should have a high accuracy score for the lower bound, and it needs to minimize the difference between the best and worst case. The baseline with whom we compare the LLM evaluators are the results declared by Es et al. (2023) and

the Knowledge-Precision (K-Precision) metric, which counts how many tokens from the generated answer can be found in the retrieved context.

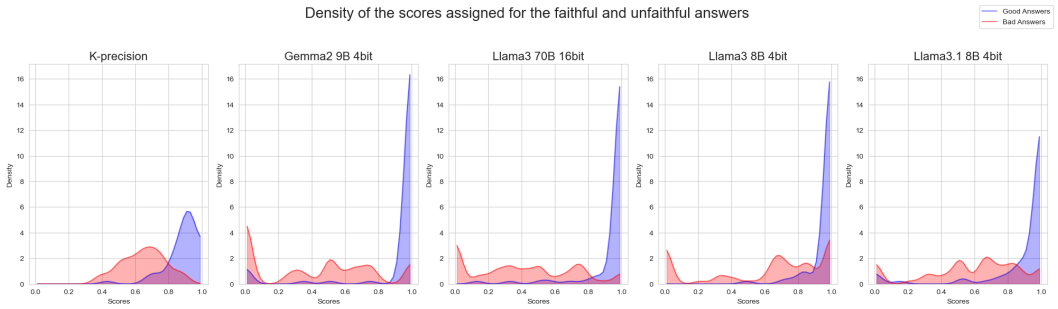

Figure 3: The density distribution plots of the faithfulness evaluators that use the second regular expression as parsing. The distribution of the faithful and unfaithful answers scores are marked with blue and red, respectively. The labels were chosen according to the human annotations.

If we look at the experiments which use constrained generation, it could be noticed that they have a considerably high score for the worst case. However, the difference between the worst and the best case is the highest among all experiments, which means that it encounters many scenarios where it assigns the same score for both the faithful and unfaithful answers. The deterministic parsers have a much lower difference, regardless of the evaluator. Llama3 70B is the highest-performing evaluator, reaching the lowest possible difference and the highest score for the worst case: 0.94%. Once again, Gemma2 9B is reaching the second-highest worst case score despite having a lower precision and size than Llama3 70B.

By observing the density distributions of the scores assigned to the faithful and unfaithful answers in Figure 3, we can see that the K-precision succeeds in maintaining the scores of the faithful answers above 0.5. Unfortunately, the distribution of the unfaithful answers is shifted towards the middle of the scale, making it difficult to estimate if the answer is wrong just by looking at the score. An answer can contain similar tokens to those existing in the context. However, the semantics of the phrases can be different. Gemma2 9B and Llama3 70B assign a score of 1 almost all the time for the faithful answers, while the scores for the unfaithful answers are more sparse. That is expected, considering that an unfaithful answer can contain some statements that can be inferred from the context. Nevertheless, few unfaithful answers receive a score of 1, making the two distributions separable. Unfortunately, Llama3 8B and Llama3.1 8B have the unfaithful answers scores dispersed across the whole scale.

If we compare with RAGAS, we notice that even the lower bound and the middle accuracy of our best configuration is close to the score of the RAGAS framework, which uses GPT3.5-Turbo as an evaluator. That reinforces the fact that lighter LLMs could become an alternative for having evaluation performance close to one of the enterprise LLMs. K-precision has high accuracy, but it should be taken into account that, in this setting, the faithful answer was selected by comparing its score with the unfaithful one. Therefore, it is difficult to decide if an answer is either faithful or unfaithful just by looking at the score of the answer due to the overlap of the two distributions.

6 Conclusion

To conclude, we benchmarked the capabilities of lighter and quantized LLMs to evaluate the correctness and faithfulness of answers generated by the RAG pipelines, and we highlighted that quantized Llama3 70B and Gemma2 9B perform similarly to the enterprise LLM evaluators. We also provided statistical interpretations by observing the distribution of the scores assigned to the good and bad answers and we highlighted that the best-performing configurations have a high separability for the two distributions. In general, deterministic parsing approaches should be favoured whenever possible, while constrained generation should be utilised as a last resort. According to our analysis, wrong answers are more challenging to detect just by looking at their score distribution. For future work, we would like to explore mixtures of LLM evaluators to assess if they can outperform individual evaluators in terms of human agreement and reduce bias

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Vaibhav Adlakha, Shehzaad Dhuliawala, Kaheer Suleman, Harm de Vries, and Siva Reddy. Topiocqa: Open-domain conversational question answering with topic switching. *Transactions of the Association for Computational Linguistics*, 10:468–483, 2022.

Vaibhav Adlakha, Parishad BehnamGhader, Xing Han Lu, Nicholas Meade, and Siva Reddy. Evaluating correctness and faithfulness of instruction-following models for question answering. *Transactions of the Association for Computational Linguistics*, 2023.

Cheng-Han Chiang and Hung-yi Lee. Can large language models be an alternative to human evaluations? *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 2023.

Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. Ragas: Automated evaluation of retrieval augmented generation. *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, 2023.

Pranab Islam, Anand Kannappan, Douwe Kiela, Rebecca Qian, Nino Scherrer, and Bertie Vidgen. Financebench: A new benchmark for financial question answering. *arXiv preprint arXiv:2311.11944*, 2023.

Sai Muralidhar Jayanthi, Varsha Embar, and Karthik Raghunathan. Evaluating pretrained transformer models for entity linking in task-oriented dialog. *arXiv preprint arXiv:2112.08327*, 2021.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33: 9459–9474, 2020.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pp. 74–81, 2004.

Hanmeng Liu, Leyang Cui, Jian Liu, and Yue Zhang. Natural language inference in context-investigating contextual reasoning over long texts. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 13388–13396, 2021.

Potsawee Manakul, Adian Liusie, and Mark JF Gales. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *Conference on Empirical Methods in Natural Language Processing*, 2023.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318, 2002.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *Conference on Empirical Methods in Natural Language Processing*, 2016.

Hannah Sansford, Nicholas Richardson, Hermina Petric Maretic, and Juba Nait Saada. Grapheval: A knowledge-graph based llm hallucination evaluation framework. *arXiv preprint arXiv:2407.10793*, 2024.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. *arXiv preprint arXiv:2305.17926*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Brandon T Willard and Rémi Louf. Efficient guided generation for large language models. *arXiv e-prints*, 2023.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*, 2018.

Tianhua Zhang, Hongyin Luo, Yung-Sung Chuang, Wei Fang, Luc Gaitskell, Thomas Hartvigsen, Xixin Wu, Danny Fox, Helen Meng, and James Glass. Interpretable unified language checking. *arXiv preprint arXiv:2304.03728*, 2023.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *International Conference on Learning Representations*, 2019.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36, 2024.

A FAITHFULNESS PIPELINE

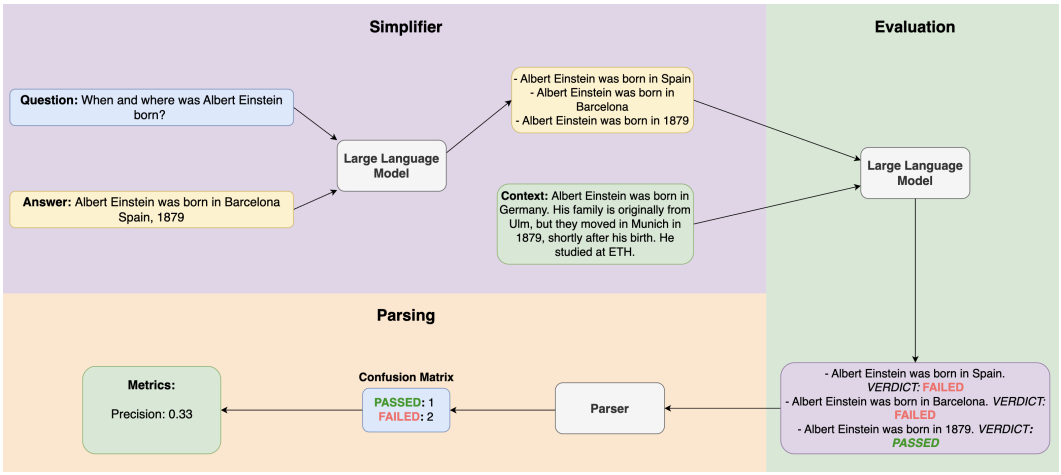

Figure 4: The evaluation pipeline for faithfulness. Firstly, the LLM extracts the statements of the answer. Afterwards, given the context, the statements are labelled according to the definition from Section B.2. Finally, the parser extracts the label matches and calculates the metric.

B STATEMENTS

B.1 STATEMENTS CORRECTNESS

While calculating correctness, a statement can be labelled as:

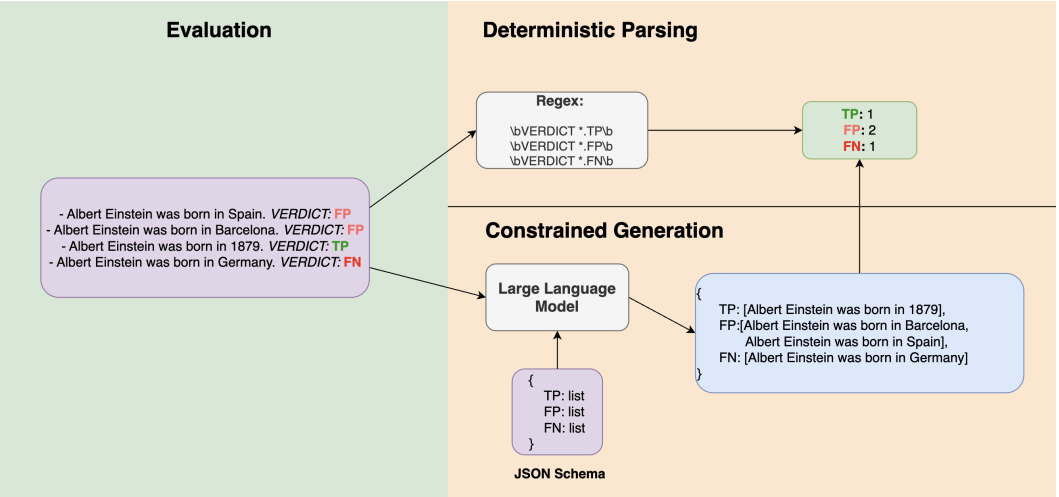

Figure 5: The parsing strategies for extracting the labelled statements. The deterministic parsing uses regular expressions to match the labels. The constrained generation parsing collects the labelled statements in a JSON schema and then returns the number of matches for each label.

- **True Positive (TP)**: if the statement appears in the answer and it is directly supported by a statement from the ground truth
- **False Positive (FP)**: if the statement appears in the answer but is not directly supported by a statement from the ground truth
- **False Negative (FN)**: if it appears in the ground truth but does not support any statement from the answer

B.2 Statements Faithfulness

While calculating faithfulness, a statement can be labelled as:

- **PASSED**: if the statement can be inferred from the context
- **FAILED**: if the statement cannot be inferred from the context

C Parsing Strategies

C.1 Deterministic Parsing

The first regex (**Regex 1**) is trying to match the VERDICT keyword as well as the corresponding label. Therefore, depending on the calculated metric, we apply as many regular expressions as labels that need to be extracted.

For answer correctness:

- **True Positive**: `\b`VERDICT: TP`\b`
- **False Positive**: `\b`VERDICT: FP`\b`
- **False Negative**: `\b`VERDICT: FN`\b`

For faithfulness:

- **PASSED**: `\b`VERDICT: PASSED`\b`
- **FAILED**: `\b`VERDICT: FAILED`\b`

The second regex (**Regex 2**) matches the same pattern as the first regex, with the exception that it allows the possibility of having additional characters between the VERDICT keyword and the corresponding label.

For answer correctness:

- **True Positive**: \bVERDICT: .*TP\b
- **False Positive**: \bVERDICT: .*FP\b
- **False Negative**: \bVERDICT: .*FN\b

For faithfulness:

- **PASSED**: \bVERDICT: .*PASSED\b
- **FAILED**: \bVERDICT: .*FAILED\b

We propose the second version as a dynamic way of handling the scenarios when the LLM evaluator does not respect the first regex pattern.

C.2 JSON SCHEMAS

Outlines requires a JSON schema from which a finite automata will be created. At each step during generation, the JSON schema will constrain the logits of the LLM evaluator by masking the invalid tokens. The process continues until the generation reaches an end state. Figure 6 shows an example of the final output.

The JSON schema for correctness is:

```
{
    TP : list
    FP : list
    FN : list
}
```

The JSON schema for faithfulness is:

```
{
    PASSED : list
    FAILED : list
}
```

C.3 SIMPLIFIED CONSTRAINED GENERATION

Figure 6 displays a simplified constrained parsing. Given few-shot examples and a JSON schema, the LLM evaluator maps the labelled statements to a single token and appends them to the corresponding list.

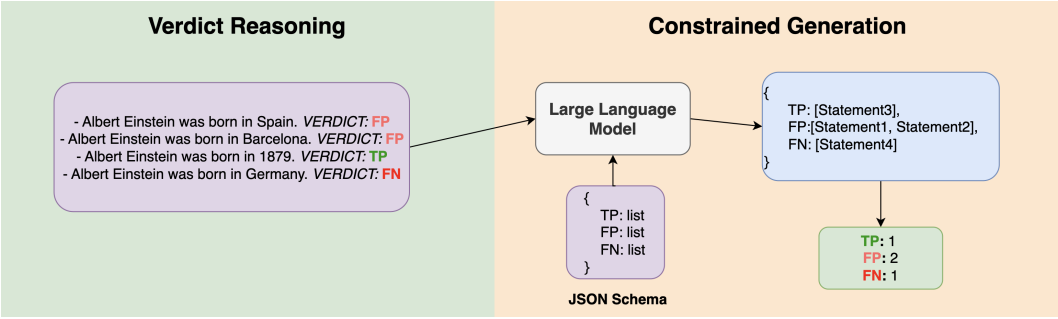

Figure 6: Outlines simplified generation. Rather than copying each token of the statement in the list, the generation process is simplified by replacing the statement with a single token.

D PROMPTS

D.1 STATEMENT EXTRACTION

Given a question, an answer, and sentences from the answer analyze the complexity of each sentence given under 'sentences' and break down each sentence into one or more fully understandable statements while also ensuring no pronouns are used in each statement. Format the output of the statements as a list with hyphens

Examples: [
{
"question":
"Who was Albert Einstein and what is he best known for?",
"answer": "He was a German-born theoretical physicist, widely acknowledged to be one of the greatest and most influential physicists of all time. He was best known for developing the theory of relativity, he also made important contributions to the development of the theory of quantum mechanics.",
"sentences":
" 0:He was a German-born theoretical physicist, widely acknowledged to be one of the greatest and most influential physicists of all time.
1:He was best known for developing the theory of relativity, he also made important contributions to the development of the theory of quantum mechanics. ",
"statements":
" - Albert Einstein was a German-born theoretical physicist.
- Albert Einstein is recognized as one of the greatest and most influential physicists of all time.
- Albert Einstein was best known for developing the theory of relativity.
- Albert Einstein also made important contributions to the development of the theory of quantum mechanics. "
}
],

Question: {question}

Answer: {answer}

Sentences: {sentences}

You need to output only the list of statements with hyphens. There can also be answers that contain only 1 statement. If you cannot find more than 1 statement, then just print the original answer with a hyphen.

D.2 CORRECTNESS VERDICT

Given a set of ground truth statements and a set of answer statements, analyze each statement and classify them in one of the following categories:

- TP (true positive): statements that are present in answer that are also supported by one or more statements in ground truth,

- FP (false positive): statements that are present in answer and that are not supported by any statements in ground truth,

- FN (false negative): statements that are present in the ground truth, but they are not supporting any statements in the answer.

Each statement can only belong to one of the categories. TP and FP are directly related to answer statements and FN directly related to ground truth statements. If a ground truth statement is supporting an answer statement, that statement can NEVER be an FN so avoid classifying it.

Examples: [

{
"question": "What powers the sun and what is its primary function?",
"statements answers":
" - The sun is powered by nuclear fission, similar to nuclear reactors on Earth.
- The primary function of the sun is to provide light to the solar system. ",
"ground_truth":
" - The sun is powered by nuclear fusion, where hydrogen atoms fuse to form helium.
- This fusion process in the sun's core releases a tremendous amount of energy.
- The energy from the sun provides heat and light, which are essential for life on Earth.
- The sun's light plays a critical role in Earth's climate system.
- Sunlight helps to drive the weather and ocean currents. ",

"classification":
" - The primary function of the sun is to provide light to the solar system. This statement is somewhat supported by the ground truth mentioning the sun providing light and its roles, though it focuses more broadly on the sun's energy. VERDICT: TP,
- The sun is powered by nuclear fission, similar to nuclear reactors on Earth. This statement is incorrect and contradicts the ground truth which states that the sun is powered by nuclear fusion. VERDICT: FP,
- The sun is powered by nuclear fusion, where hydrogen atoms fuse to form helium. This accurate description of the sun's power source is not included in the answer. VERDICT: FN
- This fusion process in the sun's core releases a tremendous amount of energy. This process and its significance are not mentioned in the answer. VERDICT: FN
- The energy from the sun provides heat and light, which are essential for life on Earth. The answer only mentions light, omitting the essential aspects of heat and its necessity for life, which the ground truth covers. VERDICT: FN
- The sun's light plays a critical role in Earth's climate system. This broader impact of the sun's light on Earth's climate system is not addressed in the answer. VERDICT: FN
- Sunlight helps to drive the weather and ocean currents. The effect of sunlight on weather patterns and ocean currents is omitted in the answer. VERDICT: FN"
},

{ "question": "What is the boiling point of water?",
"statements answers":
" - The boiling point of water is 100 degrees Celsius at sea level ",
"statements ground_truth":
" - The boiling point of water is 100 degrees Celsius (212 degrees Fahrenheit) at sea level.
- The boiling point of water can change with altitude.",
"classification":
" - The boiling point of water is 100 degrees Celsius at sea level. This statement is directly supported by the ground truth which specifies the boiling point of water as 100 degrees Celsius at sea level. VERDICT: TP,
- The boiling point of water can change with altitude. This additional information about how the boiling point of water can vary with altitude is not mentioned in the answer. VERDICT: FN
- The boiling point of water is 100 degrees Celsius (212 degrees Fahrenheit) at sea level. No need to label it because it is a ground truth statement that supports a statement from the answer.
"
},

{
"question": "Which actor is playing Han Solo in the original Star Wars?",
"statements answers":
" - Han Solo is played by the American actor Harrison Ford.",

"statements ground_truth":
" - Harrison Ford ",
"classification":
" - Han Solo is played by the American actor Harrison Ford. This statement is directly supported by the ground which mentions Harrison Ford. VERDICT: TP
- Harrison Ford. No need to label it because it is a ground truth statement that supports a statement from the answer.
" }

]

Given the question and the statements, evaluate the input from below:

Question: {question}

Statements answer: {statements_answer}

Statements ground_truth: {statements_groundtruth}

FP statements can be found only in the answer and FN statements can be found only in the ground truth. Remeber that if a statement is not present in the ground truth, then that is an FP not an FN! If a ground truth statement is supporting an answer statement, that ground truth statement can NEVER be an FN so you no longer need to classify it. You don't need to look for the exact formulation of a ground truth statement inside an answer statement. Ground truth statements can be present inside an answer statement, but formulated in a different way and using different words (follow the examples I showed you above). Respect the structure of how you can give a verdict: [VERDICT: TP/FP/FN]

D.3 FAITHFULNESS VERDICT

Your task is to judge the faithfulness of a series of statements based on a given context. For each statement you must return verdict as PASSED if the statement can be directly inferred based on the context or FAILED if the statement can not be directly inferred based on the context.

Examples: [
{
"context":
"John is a student at XYZ University. He is pursuing a degree in Computer Science. He is enrolled in several courses this semester, including Data Structures, Algorithms, and Database Management. John is a diligent student and spends a significant amount of time studying and completing assignments. He often stays late in the library to work on his projects.",
"statements":
" - John is majoring in Biology.,
- John is taking a course on Artificial Intelligence.,
- John is a dedicated student.,
- John has a part-time job.",
"answer":
" - John is majoring in Biology. John's major is explicitly mentioned as Computer Science. There is no information suggesting he is majoring in Biology. VERDICT: FAILED
- John is taking a course on Artificial Intelligence.The context mentions the courses John is currently enrolled in, and Artificial Intelligence is not mentioned. Therefore, it cannot be deduced that John is taking a course on AI. VERDICT: FAILED¨
- John is a dedicated student. The context states that he spends a significant amount of time studying and completing assignments. Additionally, it mentions that he often stays late in the library to work on his projects, which implies dedication. VERDICT: PASSED
- John has a part-time job. There is no information given in the context about John having a part-time job. VERDICT: FAILED "

},
{
"context": "Photosynthesis is a process used by plants, algae, and certain bacteria to convert light energy into chemical energy.",
"statements":
"-Albert Einstein was a genius.",
"answer":
" - Albert Einstein was a genius. The context and statement are unrelated. VERDICT: FAILED
"
}

Context: {context}
Statements: {statements}

Include PASSED or FAILED only when you label a statement. Do not count, just label the statements. If something is not mentioned in the provided context, then it should be marked with FAILED. Every statements from the list provided needs to be evaluated. Respect the structure of how you can give a verdict to a statement: [VERDICT: PASSED/FAILED]",

D.4 Constrained Parsing

D.4.1 Correctness

Given a set of statments that were labeled with either TP, FP or FN, append the statement to the correct list. If a statement was labeled with TP tag, the statement should be appended to the TP list. If it was labeled with the FP tag, the statement should be in the FP list. If it was labeled with the FN tag, the statement should be in the FN list. I will provide you an example.

Example:

[
{
"classified_statements":
" - Statement1 VERDICT: TP
- Statement2 VERDICT: TP
- Statement3 VERDICT: TP
- Statement4 VERDICT: FN
- Statement5 VERDICT: FP
- Statement6 VERDICT: FN
",
"output":
" { TP=[Statement1, Statement2, Statement3],
FP=[Statement5],
FN=[Statement4, Statement6] } "
},

{ "classified_statements":
" - Statement1 VERDICT: FP
- Statement2 VERDICT: TP
- Statement3 VERDICT: FN
", "output":
" { TP=[Statement2],
FP=[Statement1],
FN=[Statement3] } " },

{ "classified_statements":
" - Statement1 VERDICT: FN
- Statement2 VERDICT: FP
- Statement3 VERDICT: TP
", "output":
" { TP=[Statement3],
FP=[Statement2],
FN=[Statement1]
" }]

Statements: {statements}

Your task is only to put the number of the statements in the correct list. Sometimes the input you will receive will not be the same like the example, but try as best as you can to fulfill the task correctly.",

D.4.2 FAITHFULNESS

Given a set of statments that were labeled with either PASSED OR FAILED, append the statement to the correct list. If a statement was labeled with PASSED tag, the statement should be appended to the PASSED list. If it was labeled with the FAILED tag, the statement should be in the FAILED list. I will provide you an example.

Examples:

Example 1:
- Statement1 VERDICT: FAILED
- Statement2 VERDICT: PASSED
- Statement3 VERDICT: PASSED
- Statement4 VERDICT: FAILED
Output:
{ PASSED: [Statement2, Statement3]
FAILED: [Statement1, Statement4] }

Example 2:
- Statement1 VERDICT: PASSED
- Statement2 VERDICT: PASSED
- Statement3 VERDICT: FAILED
- Statement4 VERDICT: PASSED
- Statement5 VERDICT: FAILED
Output: {
PASSED: [Statement1, Statement2, Statement4]
FAILED: [Statement3, Statement5]
} "

Statements : {statements}

Your task is only to put the number of the statements in the correct list. Sometimes the input you will receive will not respect strictly the example, but try as best as you can to fulfill the task correctly.,