Evaluating the Factuality of Large Language Models using Large-Scale Knowledge Graphs

Xiaoze Liu♣, Feijie Wu♣, Tianyang Xu♣, Zhuo Chen♡,Yichi Zhang♡,
Xiaoqian Wang♣, Jing Gao♣
♣ Purdue University, West Lafayette, IN 47907, USA
♡ Zhejiang University, Hangzhou, China
{xiaoze, wu1977, xu1868, joywang, jinggao}@purdue.edu
{zhuo.chen, zhangyichi2022}@zju.edu.cn

Abstract

The advent of Large Language Models (LLMs) has significantly transformed the AI landscape, enhancing machine learning and AI capabilities. Factuality issue is a critical concern for LLMs, as they may generate factually incorrect responses. In this paper, we propose GraphEval to evaluate an LLM's performance using a substantially large test dataset. Specifically, the test dataset is retrieved from a large knowledge graph with more than 10 million facts without expensive human efforts. Unlike conventional methods that evaluate LLMs based on generated responses, GraphEval streamlines the evaluation process by creating a judge model to estimate the correctness of the answers given by the LLM. Our experiments demonstrate that the judge model's factuality assessment aligns closely with the correctness of the LLM's generated outputs, while also substantially reducing evaluation costs. Besides, our findings offer valuable insights into LLM performance across different metrics and highlight the potential for future improvements in ensuring the factual integrity of LLM outputs. The code is publicly available at `https://github.com/xz-liu/GraphEval`.

1 Introduction

The rapid progress of Large Language Models (LLMs) has markedly boosted artificial intelligence and machine learning due to their strong contextual text generation capabilities. Despite these groundbreaking advancements, recent works [34, 35] have highlighted the significance of LLMs evaluation. LLMs are prone to producing seemingly authentic yet factually inaccurate responses, a phenomenon known as hallucination [19]. Such errors may stem from outdated or incorrect data during training or the model's learned associations, impacting its reliability. The evaluation, therefore, helps identify instances of hallucination and understand the LLM's ability to generate coherent and contextually relevant text, i.e., factuality of LLM outputs.

Recent efforts have been put into the factuality evaluation of LLM. For example, [26] introduce FELM, a benchmark comprising diverse factual samples across various domains. Moreover, [24] and [22] utilize external tools (e.g., search engines and a well-trained factual LLM) to estimate the factuality of the generated texts. Representing structured knowledge related to real-world objects, Knowledge Graphs (KGs) [11, 20, 28, 41] have gained prominence for LLM factuality assessments. They primarily originate from Wikipedia, encapsulate factual information for AI tasks, and form the knowledge base with datasets like Natural Questions [21].

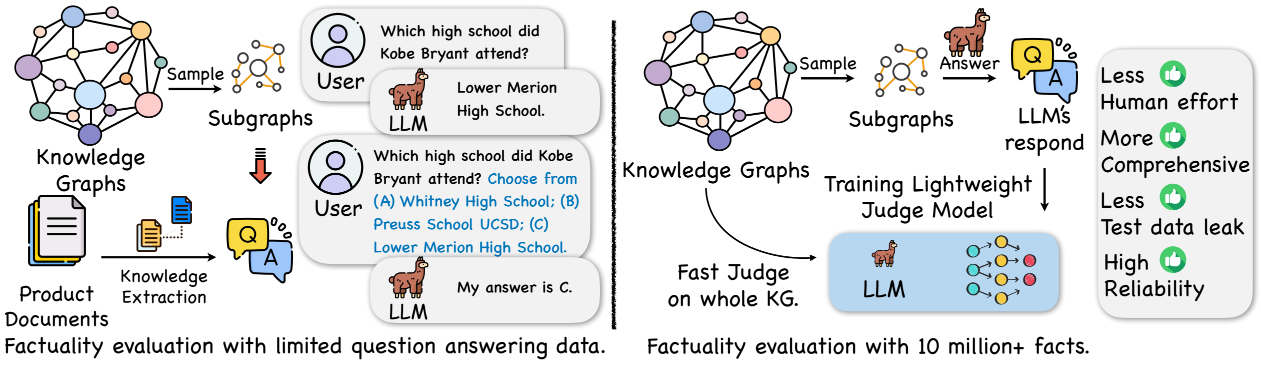

Knowledge Graphs
Sample
Subgraphs
User
Which high school did Kobe Bryant attend?
LLM
Lower Merion High School.
User
Which high school did Kobe Bryant attend? Choose from (A) Whitney High School; (B) Preuss School UCSD; (C) Lower Merion High School.
LLM
My answer is C.
Product Documents
Knowledge Extraction
Factuality evaluation with limited question answering data.
Knowledge Graphs
Sample
Subgraphs
Answer
LLM's respond
Training Lightweight Judge Model
Fast Judge on whole KG.
LLM
Less Human effort
More Comprehensive
Less Test data leak
High Reliability
Factuality evaluation with 10 million+ facts.

Figure 23: Existing works compared to the proposed GraphEval on factuality evaluation.

Several studies [1, 2] have focused on creating benchmarks from knowledge sources by posing factual questions derived from triples in KGs. These methods, as depicted in the left part of Figure 23, either (i) sample subgraphs from large KGs or (ii) extract a subset of knowledge from text documents to construct multiple-choice or text question-answer pairs. Those question pairs are then posed to LLMs to assess their factuality. However, the above-mentioned methods or evaluation strategies face challenges in comprehensively evaluating the factuality of LLMs. *Firstly*, the scope of evaluation data is often limited or incomplete, focusing predominantly on specific domains. This limitation restricts the evaluation's breadth and undermines its applicability across various contexts, failing to cover the wide range of topics LLMs are expected to handle. The specialized nature of these datasets means that the evaluation may not accurately reflect the model's performance in generating factual content across a broader spectrum of subjects. *Secondly*, the process of factuality evaluation itself is inherently time-consuming and costly. It necessitates that an LLM generate full texts, which must then be meticulously assessed for accuracy and reliability. This comprehensive generation and detailed review demand significant computational resources and extensive human effort [35] for validation. As a result, the process becomes less feasible for regular or large-scale applications, limiting the frequency and scope of practical evaluations. *Lastly*, due to the limited size of the evaluation data, there may be biases in the benchmarks [16], or risks of test data leakage [38], which might compromise the validity of the evaluations. Together, these challenges underscore the need for more scalable, efficient, and domain-agnostic approaches to evaluating the factuality of LLM-generated texts.

To this end, we propose GraphEval, which consists of two novel features in terms of the design, as presented on the right side of Figure 23. First, we utilize KGs that encapsulate factual information sourced from verifiable content like Wikipedia. With the KG, millions of prompts can be automatically generated, leading to a significant saving of human efforts in labeling the ground truth. From the data perspective, the KG gives a more diversified and comprehensive evaluation of the LLMs' factuality. Second, the proposed method efficiently reduces the computation costs and speeds up the evaluation process. Specifically, we incorporate a highly reliable and lightweight judge model to decide whether an LLM can generate an accurate response to a designated question. Instead of generating the full text, the judge model returns three options (i.e., True, False, and I don't know) to simulate LLMs' responses to a given prompt. To ensure the reliability of the simulated results, the judge model is trained based on a few question-answer pairs, where the questions are sampled from KGs and the answers are generated by the target LLM. As a result, the judge model can serve as a replacement for the factuality evaluation of the facts extracted from large-scale KGs. In summary, our contributions are as follows:

- We propose GraphEval, a large-scale evaluation framework that assesses the factuality of LLMs using KGs. GraphEval evaluates the factuality of LLMs using the entire KGs, providing a more diversified and comprehensive evaluation of the LLMs' factuality.
- We introduce a judge model to assist with the evaluation process, which reduces the computational

cost and enhances the efficiency of the evaluation. We also give a theoretical analysis of the judge model to demonstrate its validity.

- We conduct extensive experiments on a large-scale KG, i.e., DBpedia, to demonstrate the effectiveness and efficiency of GraphEval in evaluating the factuality of LLMs.
- We provide an in-depth analysis of the LLM's performance on the KGs, including the LLM's performance with respect to relation types, head entity types, tail entity types, and the relation of LLM performance to degree and pageviews.

2 Related Work

Factuality Issue of LLMs Factuality issue [19, 39], is the issue that LLMs may produce content inconsistent with established facts. As outlined in [19], this issue may be due to: ***(i)*** LLMs lacking expertise in specific domains [3, 7]; ***(ii)*** LLMs' unawareness of recent developments or changes [4, 9]; ***(iii)*** LLMs not retaining [21, 35, 40] or forgetting [17, 31, 36, 37, 50] knowledge from its training corpus; and ***(iv)*** LLMs failing to reason with the knowledge they possess [27, 47, 51]. The factuality issue has been addressed by various works, by incorporating Retrieval Augmented Generation (RAG) [12, 14], fine-tuning [24], and knowledge-enhanced models [13, 22]. To summarize, these approaches integrate other knowledge sources into the LLMs' training process or use them to augment the models' knowledge base, thus alleviating the factuality issue. Our work differ from them in that we evaluate the factuality of LLMs, rather than providing factuality enhancement methods.

Factuality Evaluation of LLMs The expanding use of LLMs across various domains necessitates the assurance of their output's accuracy and reliability. A range of benchmarks and evaluation methodologies for assessing large language models (LLMs) are proposed. These works primarily focus on evaluating the factuality, truthfulness, reasoning capabilities, and adaptability to new information of LLMs. MMLU [42] and TruthfulQA [5] aim to measure the factuality and truthfulness of LLMs across diverse tasks, while C-Eval [46] focuses on the Chinese context, assessing models' knowledge of Chinese culture and laws. There are also works [1, 2] that propose factuality evaluation using subsets of KGs. However, selecting subsets of KGs to test LLMs can introduce selection bias. For example, random sampling can focus more on a few popular domains or subjects more densely connected with others, thus not showing LLMs' factuality on diversified topics. Our work addresses this limitation by proposing a resource-efficient method to evaluate the factuality of LLMs which allows evaluations on whole KGs instead of subsets, thus providing a more diversified and comprehensive evaluation of the LLMs' factuality, enabling an extensive assessment of LLM's factuality and reasoning abilities in a way that existing individual benchmarks do not as they only focus on specific aspects.

Using KGs in LLMs KGs are structured representations of factual knowledge, typically in the form of (head, relation, tail) triples. There are lots of efforts in constructing [11], and reasoning [43] on KGs. This has made KGs an indispensable resource of factual knowledge for AI tasks. Currently, the most common way of integrating KGs with LLMs is using KGs as an external knowledge source to enhance LLM performance by pre-training, fine-tuning, or in-context learning [8, 23, 25, 33, 45]. Our work is different from these works in that we use KGs to evaluate the factuality of LLMs, rather than enhancing the LLMs with KGs.

3 Method

GraphEval is designed to measure the factuality of a language model in relation to a KG. As presented in Figure 24, the proposed work is divided into three steps:

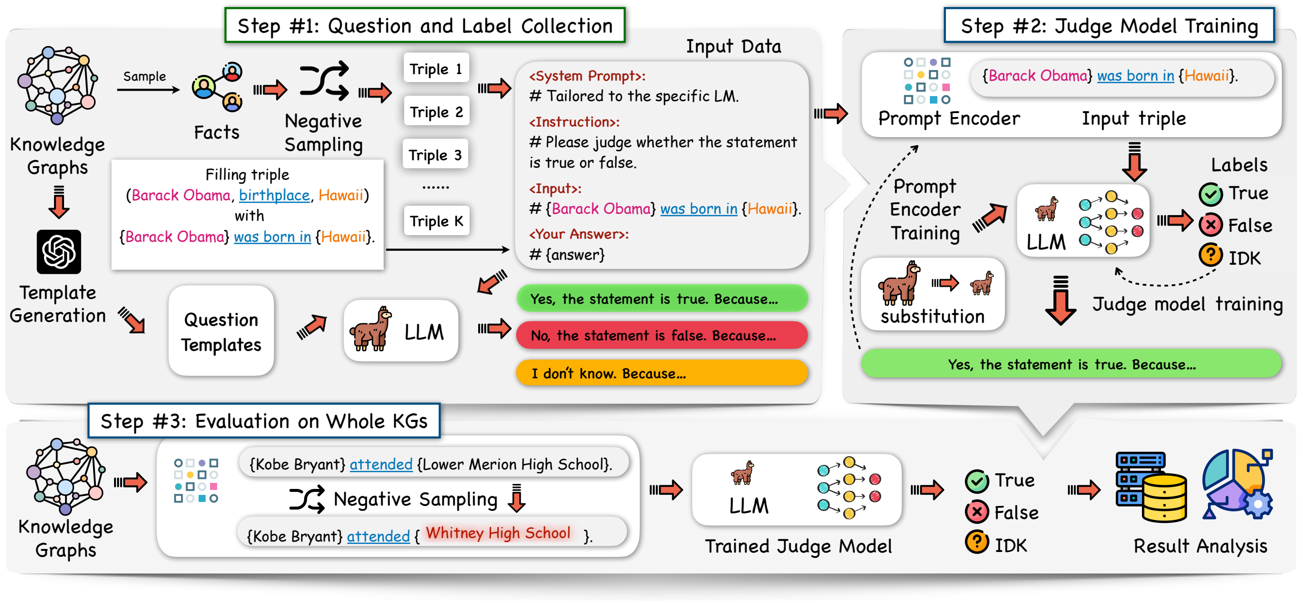

Step #1: Question and Label Collection
Knowledge Graphs
Sample
Facts
Negative Sampling
Triple 1
Triple 2
Triple 3
Triple K
Input Data
<System Prompt>:
# Tailored to the specific LM.
<Instruction>:
# Please judge whether the statement is true or false.
<Input>:
# {Barack Obama} was born in {Hawaii}.
<Your Answer>:
# {answer}
Filling triple (Barack Obama, birthplace, Hawaii) with {Barack Obama} was born in {Hawaii}.
Template Generation
Question Templates
LLM
Yes, the statement is true. Because...
No, the statement is false. Because...
I don't know. Because...
Step #2: Judge Model Training
{Barack Obama} was born in {Hawaii}.
Prompt Encoder
Input triple
Prompt Encoder Training
LLM
Labels
True
False
IDK
substitution
Judge model training
Yes, the statement is true. Because...
Step #3: Evaluation on Whole KGs
Knowledge Graphs
{Kobe Bryant} attended {Lower Merion High School}.
Negative Sampling
{Kobe Bryant} attended { Whitney High School }.
LLM
Trained Judge Model
True
False
IDK
Result Analysis

Figure 24: Overview of the GraphEval framework. **Step #1** retrieves KG statements and collect LLM judgments on them. **Step #2** trains the judge model which classifies LLM hidden states into three categories. **Step #3** evaluates the LLM on all KG statements with the judge model.

- **Step 1: Question and label collection from KGs and LLMs.** The model samples triples from KGs and converts each triple into a declarative statement with GPT-4-crafted templates. To prepare versatile statements, we employ *negative sampling*, where incorrect statements are intentionally generated. Afterward, those statements are posed to collect the labels answered by an LLM (i.e., Yes, No, and I don't know (IDK)).
- **Step 2: Judge model training.** With the triples collected in the first step, we train a judge model to avoid long-generated texts and conserve computational resources. In detail, inspired by [15], we train a classifier with LLMs' hidden states to make a selection within the above three options. We also apply p-tuning [18] to minimize the prompt/instruction size.
- **Step 3: Evaluation on whole KGs.** Similar to the first step, we retrieve all true/false statements from KGs. Subsequently, these statements are fed into the trained judge model to estimate the factuality of LLM. This process enables a thorough and multifaceted analysis of the LLM's performance in terms of factuality, drawing from a wide range of perspectives to provide a more comprehensive and diversified evaluation.

In the following sections, we will discuss the details of each step.

3.1 Question and Label Collection

Question Generation In order to evaluate the language model's ability to identify false statements, we directly construct a declarative sentence for each triple. This addresses the ineffectiveness of multiple-choice questions in our task. Firstly, multiple-choice prompts may cause misalignment with parametric knowledge in LLMs. Since LLMs mainly learn parametric knowledge through text data, in which knowledge facts are mostly represented as declarative sentences [6], employing multiple-choice questions may hinder the evaluation of factuality. Secondly, multiple-choice questions have more complex labels (i.e. A, B, C, D) than declarative sentences (i.e. True, False, IDK), which can complicate the tasks for the judge model, influencing the overall effectiveness. We use an example to illustrate this.

Example 3.1: For the triple `(Barack Obama, birthPlace, Hawaii)`, a multi-choice question can be generated as `Where was Barack Obama born?` with choices `A. Hawaii B. Chicago C. New York D. Los Angeles`. Here, for the same triple, we can also generate another multi-choice question as `Where was Barack Obama born?` with choices `A. China, B. Hawaii, C. Japan, D. Russia`. The two questions represent the same triple, but the choices are different. As mentioned in the last paragraph, this can introduce complexity and potential misalignment with an LLM's training and result in inconsistent responses.

To address the ineffectiveness of multiple-choice questions, we propose to directly ask the LLMs whether a statement is true or not. For instance, considering the triple `(Barack Obama, birthPlace, Hawaii)`, we can formulate a fact `Obama was born in Hawaii` by integrating the entities `Barack Obama` and `Hawaii` into the template `{head} was born in {tail}`. Each template corresponds to the relation of a triple, and they are crafted to be clear and straightforward statements. GPT-4 is employed to generate these templates for all relations in the KG. These generated templates are then manually reviewed and refined to ensure their compatibility with the KG. Then, we can ask a question to the LLMs, such as `Is the statement "Barack Obama was born in Hawaii" true or false?`. Here, the templates are corresponding to the relations of the triples. This is because the number of relations in the KG is limited, while the number of triples is large. Therefore, we can use the relations to categorize the triples, and then use the templates to generate questions for each category. This can significantly reduce human labor, i.e., monitoring less than 1000 templates compared with monitoring more than 10 million triples. See the Appendix A.5 for the detailed settings of the relation templates.

Negative sampling Although the declarative sentences simplify the training of the judge model, they alone are insufficient to evaluate the language model's factual accuracy. LLMs can simply answer true for every question, and still get a high accuracy. To address this, we introduce negative sampling, a technique commonly used in KG completion tasks, to generate false statements. Specifically, we randomly replace one entity or relation in the original triple with another entity or relation sampled from the KG. For example, given the triple `(Barack Obama, birthPlace, Hawaii)`, we can replace the tail entity `Hawaii` with another entity `Chicago` to form the false statement `Barack Obama was born in Chicago`. These false statements are then presented to the LLMs to evaluate their ability to identify falsehoods.

3.2 Judge Model

Normally, to evaluate the factual accuracy of a language model, we would generate questions from a KG and then pose these questions to the language model. However, given the expansive nature of KGs, it's impractical to label every generated question by the LLM. A more efficient approach is to use the last token logits of the LLMs as their answers. However, recent research has highlighted discrepancies between these logits and the model's actual text outputs [29]. Therefore, we introduce a novel judge model to assist with this task. The judge model, initially trained on a subset of labeled questions, is then employed to label the remaining questions. Uniquely, inspired by [15], the judge model utilizes the LLM's hidden state as input, as a replacement of the LLM's last layer with compressed output tokens. Specifically, three output classes are used: *True*, *False*, and *I don't know*. The judge model is a two-layer feed-forward neural network, with a layer normalization and a ReLU activation function. This approach diverges from standard practices where LLMs generate answers, as here we only forward the transformer once. Consequently, this operation is significantly less resource-intensive than full answer generation, allowing the judge model to efficiently process a large number of questions with limited labeled data. With this model, we can glance at the correctness of an LLM, i.e., how likely the model can answer a question relevantly and correctly. We evaluate the performance of the judge model using two metrics: (i) *Truthfulness*, i.e., the likelihood that the judge model prediction matches the LLM correctness under a given question; and (ii) *Informativeness*, i.e., the likelihood that the judge model does not give a prediction of 'I don't know.' Since the evaluation is based on a general KG that spans multiple domains, other metrics such as "Relevance" would typically require a more specific contextual framework. Nonetheless, future research could explore the use of more context-specific metrics tailored to the LLM's domain of application.

Efficiency To further enhance the judge model's efficiency, we include 2 extra components. First, we found that the instruction prefix of the LLMs is too large for the judge model to process efficiently. We thus fine-tune a *prompt encoder* [32] to reduce the large input of the prompt prefix, which would be the same for all questions. Second, we found that our judge model, with the training process on the labeled dataset, is robust to the LLM's hidden states. In experiments, we observed that our judge model can seamlessly utilize hidden states from distinct LLMs without significant differences in performance. For instance, within the LLaMA 2 model family, which contains 3 models with different parameters: 7B, 13B, and 70B, we found that the judge model's performance is consistent regardless of whether the hidden states are from 7B, 13B, or 70B. Therefore, we can use the model with the least parameters, as a *substitute model* when computing the hidden states. This gives us a huge reduction in computational cost.

Analysis of Judge model In this part, we assume there are two datasets; one is for training the judge model, and the other is for evaluation, denoted by $\mathcal{D}_S$ and $\mathcal{D}_T$, respectively. As the proposed judge model leads to a triple classification task, we assume a hypothesis portfolio $h = \{h_t, h_f, h_{idk}\}$, where these three hypotheses separately predict if a sample can be correctly answered by the LLM, i.e., True, False, and IDK. In other words, the hypothesis $\hat{h} \in h$ maps an input $\mathbf{x}$ to $\{0, 1\}$, where 1 means the input satisfies the hypothesis conditions. For a given input $\mathbf{x}$, the equality $h(\mathbf{x}) = h_t(\mathbf{x}) + h_f(\mathbf{x}) + h_{idk}(\mathbf{x}) = 1$ always holds because the judge model provides an only output. Define the convex loss function for a hypothesis $\hat{h} \in h$ to be

$$L_{\mathcal{D}}(\hat{h}) = \sum_{(\mathbf{x},y)\in\mathcal{D}} |\hat{h}(\mathbf{x}) - \mathbf{1}_{\hat{h}}(y)|,$$

where $\mathbf{1}_{\hat{h}}(y)$ indicates if the data indeed satisfies the hypothesis. Since a wrong prediction for data $(\mathbf{x}, y)$ results in $\sum_{\hat{h}\in h} |\hat{h}(\mathbf{x}) - \mathbf{1}_{\hat{h}}(y)| = 2$, we define the misclassification rate as

$$L_{\mathcal{D}}(h) = \frac{1}{2}\left(L_{\mathcal{D}}(h_t) + L_{\mathcal{D}}(h_f) + L_{\mathcal{D}}(h_{idk})\right)$$

Below is a theoretical analysis to understand the bound of the misclassification rate, which is driven by Theorem 2 of [30].

Theorem 3.1: Let $\mathcal{H} = \{\mathcal{H}_t, \mathcal{H}_f, \mathcal{H}_{idk}\}$ be a set of hypothesis spaces of VC dimension d. If $\mathcal{U}_S$,$\mathcal{U}_T$ are the samples of size m each, drawn from $\mathcal{D}_S$ and $\mathcal{D}_T$, respectively, then for any $\delta \in (0, 1)$, with probability at least $1 - \delta$, for every $h \in \mathcal{H}$, we have

$$L_{\mathcal{D}_T}(h) \le L_{\mathcal{D}_S}(h) + \frac{3}{4} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{U}_S, \mathcal{U}_T) + 6\sqrt{\frac{2d\log(2m) + \log(2/\delta)}{m}} + \frac{1}{2}\lambda \quad (3)$$

where $\lambda = \inf_{h\in\mathcal{H}}\left(L_{\mathcal{D}_S}(h) + L_{\mathcal{D}_T}(h)\right)$ is the optimal combined error, $d_{\mathcal{H}\Delta\mathcal{H}}$ measures the distribution discrepancy between two distributions.

The above theorem provides insights for the generalization bound of the judge model. The bound is associated with the discrepancy between training data $\mathcal{D}_S$ and evaluation data $\mathcal{D}_T$, and the discrepancy can be measured by drawing samples from both training and evaluation datasets for an equivalent size. Moreover, the bound is affected by the optimal hypothesis over all the data, i.e., $\mathcal{D}_S \cup \mathcal{D}_T$, where a lower error leads to improved performance of the judge model.

3.3 Evaluation

For evaluating the LLM's performance, we consider *Correctness*, which is defined as the proportion of questions for which the LLM's response matches the true label (or false label if the question is generated from a negative triple). This captures the accuracy of the LLM in identifying correct information and distinguishing it from fabricated (negative) triples. We also adopt the metrics of *Truthfulness* and *Informativeness*, as defined in [5]. *Truthfulness* refers to the likelihood of the language model (LLM) providing an honest response. A response is considered *Truthful* if the LLM either provides the correct answer or opts for 'I don't know'. This criterion assesses the model's ability to be honest about what it knows and to admit uncertainty rather than making false statements. *Informativeness* is the probability of the LLM offering any substantive information, irrespective of its accuracy. An answer is deemed *Informative* if it is anything other than 'I don't know'. This reflects the model's capacity to provide substantial information without resorting to uncertainty or avoidance of an answer.

When considering multiple negative triples sampled, we combine the results for all negative triples sampled from a triple τ, as well as the results for their original positive triple τ, to calculate the overall performance of the LLM. Since correctly detecting a real triple from KG is much simpler than detecting a negative triple, we want to give a max penalty to the LLM's wrong response to the real triple when designing the metric. Therefore, if a real triple is predicted as false, the LLM will score 0 across all metrics. Then, the negative triple results are averaged to give a fine-grained evaluation of the LLM's performance. To achieve this, for each performance metric, we define functions $\mathcal{F}$ which evaluates the LLM's response to τ and $\mathcal{F}'$ to each negative triple τ' sampled from τ. The overall performance metric for τ is then calculated as:

$$\text{Metric}(\tau) = \max\left(0, \mathcal{F}(\tau) - \frac{1}{|\mathcal{N}(\tau)|}\sum_{\tau' \in \mathcal{N}(\tau)} \mathcal{F}'(\tau')\right) \tag{4}$$

Here, $\mathcal{N}(\tau)$ represents the set of all negative triples generated from the positive triple τ. $\mathcal{F}$ and $\mathcal{F}'$ are defined as follows: ***(i)*** *Correctness.* $\mathcal{F}$ is defined such that it is 1 if the judge model predicts that a real (positive) triple is True, and it is 0 otherwise; $\mathcal{F}'$ is 0 if the judge model predicts a negative triple as False, and 1 otherwise. ***(ii)*** *Truthfulness.* When measuring *Truthfulness*, $\mathcal{F}$ is set to 1 if the judge model's prediction for the input τ is either True or IDK, and it is 0 otherwise. Similarly, $\mathcal{F}'$ is set to 1 if the judge model's prediction for the input τ' is True, and 0 otherwise; and ***(iii)*** *Informativeness.* For *Informativeness*, $\mathcal{F}$ is defined as 1 if the judge model's prediction for the input τ is anything other than "I don't know", and it is 0 otherwise. $\mathcal{F}'$ is set to $1 - \mathcal{F}$ on the informativeness metric. By applying this equation, we can systematically compute the *Correctness*, *Truthfulness*, and *Informativeness* of an LLM's responses in a consistent and comprehensive manner, offering a detailed insight into its overall performance.

#Entities	#Relations	#Triples	Avg. degree	Density
4,928,232	633	16,915,848	6.80	7.18×10^{-7}

Table 11: Statistics of the DBpedia knowledge graph.

4 Experiments

4.1 Experiment Setup

Data We use DBpedia [11], a large-scale knowledge graph constructed from Wikipedia. We report the statistics of the DBpedia knowledge graph in Table 11. Note that there are "dummy" entities in DBpedia that represent a fact that is only true on a specific time period. An example is `https://dbpedia.org/page/Kathy_Greenlee__Tenure__1`. For simplicity, we remove these dummy entities and triples related from the knowledge graph. We refer to the remaining triples as the DBpedia knowledge graph. The DBpedia knowledge graph contains 4,928,232 entities, 633 relations, and 16,915,848 triples. The average node degree of the knowledge graph is 6.80, and the density of the knowledge graph is 7.18×10^{-7}.

LLMs In this paper, we evaluate the Meta LLaMA 2 family [49], including LLaMA-2-7B, LLaMA-2-13B, and LLaMA-2-70B, and Google's Gemma [44] including Gemma-2B and Gemma-7B. For each language model, we first randomly sample 2000 triples, and perform a negative sampling to obtain another 2000 negative triples. For each triple, we ask the LLM 3 times the same question, on whether the triple is true, false, or the LLM doesn't know. We use majority voting to determine the LLM's final answer. When asking, we use huggingface's pipeline with default settings and FP16 precision. This is to form a labeled dataset. We randomly sample 70% for the training set and 30% for the validation set, then train a judge model to classify the LLM's hidden state into 3 classes: LLM correctly answering the question (True), LLM incorrectly answering the question (False), and LLM responding with I don't know (IDK). We refer to Table 12 for the statistics of the labeled dataset.

Metrics For the LLM's performance, we report the estimated factuality of the LLMs on the DBpedia knowledge graph. We report the LLM's performance in terms of *Truthfulness*, *Informativeness*, and *Correctness*. For evaluating the judge model's performance (See Appendix A.1), we seek to maximize the similarity between the judge model's prediction and the LLM's answer. Thus, we use the common metrics Precision (P), Recall (R), and F1 score (F) to evaluate the judge model's accuracy; and the time it takes to predict to evaluate the judge model's efficiency.

Hyperparameter Settings For the judge model classifier training, we train 100 epochs with a batch size of 8. We use the Adam optimizer with a learning rate of 1e-4. We use the same settings for all the evaluated LLMs. For LLaMA 2 7B, 13B, and 70B, we use LLaMA 2 7B as the judge model's hidden state input. For Gemma 2B and 7B, we use Gemma 2B as the judge model's hidden state input. The judge model is trained on a server with NVIDIA A6000 GPUs.

For the training of the prompt encoder, we use the same settings for all the evaluated LLMs. To be specific, we use 20 virtual tokens, 1 transformer submodule, 12 attention heads, 12 layers, MLP as the encoder reparameterization type, 4096 as the encoder hidden size, and 2e-5 as the learning rate. We train the prompt encoder for 5 epochs with a batch size of 8. We use the Adam optimizer with a weight decay of 0.01.

For the evaluation, we use two servers, one with NVIDIA A6000 GPUs and the other with NVIDIA A100 GPUs. For inference, we use Flash Attention 2 [10] as the attention implementation, and use FP16 precision.

Model	True	False	IDK	Truthful	Informative	Correct
LLaMA-2-7B	1901	1545	554	0.965	0.550	0.516
LLaMA-2-13B	2100	1796	104	0.979	0.980	0.959
LLaMA-2-70B	338	126	3536	0.993	0.007	0.006
Gemma-2B	1760	1786	454	0.056	0.867	0.024
Gemma-7B	1509	1751	740	0.206	0.657	0.056

Table 12: Statistics and performance metrics of LLMs. True, False, and IDK denote the number of labels from the LLMs in the labeled dataset. *Truthful*, *Informative*, and *Correct* represent performance metrics.

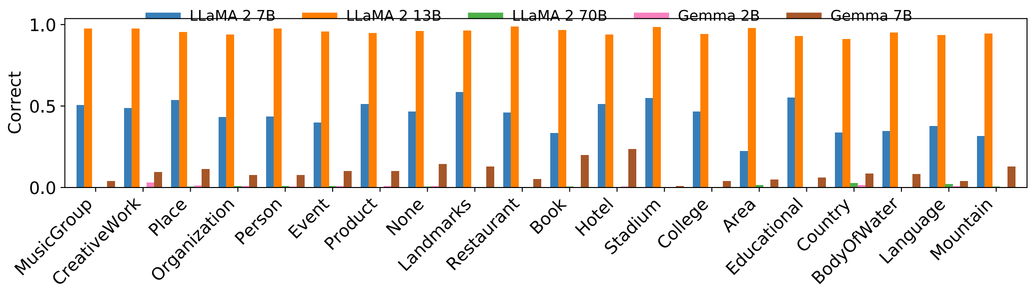
4.2 LLM's Performance Analysis

We report the estimated factuality of the LLMs on DBpedia in Table 12. Overall, the LLaMA-2 series shows an increase in model size up to 13B, particularly in terms of balanced *truthfulness*, *informativeness*, and *correctness*. However, the 70B variant diverges, excelling in *truthfulness* but failing to provide useful or accurate information. We will discuss this phenomenon in the detailed LLaMA analysis. The Gemma series struggles with *truthfulness* and *correctness*, despite being *informative*. This might indicate that these models are better at generating detailed content but need careful consideration for tasks requiring high accuracy or reliability. The performance of these models highlights the complex trade-offs between being *truthful*, *informative*, and *correct*. We further provide a correlation analysis between the LLM's performance and the degree/popularity of the entities in the Appendix A.3.

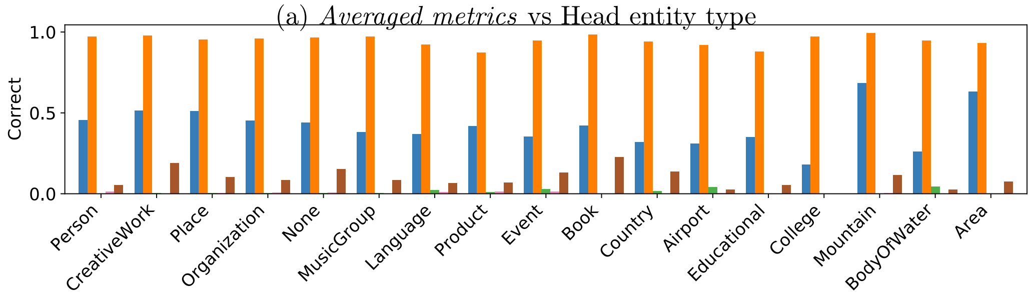
LLaMA-2 Analysis LLaMA-2-7B shows good *truthfulness* (.965) but is moderate in being *informative* (.550) and *correct* (.516). This suggests that while the model is generally reliable in its outputs, it may not always provide highly detailed or accurate information. LLaMA-2-13B significantly improves across all metrics compared to LLaMA-2-7B, with very high scores in *truthfulness* (.979), *informativeness* (.980), and *correctness* (.959). This indicates a strong overall performance, making it a very reliable and accurate model for generating information.

LLaMA-2-70B, despite its high *truthfulness* (.993), scores extremely low in both *informativeness* (.007) and *correctness* (.006), which is puzzling. We hypothesize that the model may have difficulty in making a decision, and thus selecting 'I don't know' as the answer. This may be related to a more clear knowledge boundary of LLMs, as larger LMs tend to give up on more questions [48], meaning they have a better understanding on whether they know the answer or not. A detailed analysis of the knowledge boundary of LLMs can be found in Appendix A.4. This can also be confirmed by the fact that the model has the highest *truthfulness* score among all models, indicating that it is more likely to provide a correct answer when it knows the answer. However, it is still important to note that a high number of 'I don't know' answers may indicate the model's inability to answer factual questions.

Gemma Analysis Gemma-2B has an exceptionally low *truthfulness* score (.056) but is quite high in *informativeness* (.867). Its *correctness* score (.024) is also very low. This suggests that despite providing detailed responses, the model's outputs are often neither *truthful* nor accurate. It might be generating detailed but misleading or incorrect information. Gemma-7B improves on *truthfulness* (.206) compared to Gemma-2B but still falls short of being considered reliable. Its *informativeness* (.657) is respectable, and its *correctness* (.056) remains low. Similar to Gemma-2B, while it can provide detailed responses, those are not often true or correct.



(a) *Averaged metrics* vs Head entity type



(b) *Averaged metrics* vs Tail entity type

Figure 25: The LLM's *averaged metrics* with respect to head entity types and tail entity types

4.3 Relation Type Study

There are more than 600 different relation types in the DBpedia knowledge graph, and each relation type has different characteristics. It is unclear if we directly compare the performance of the LLMs on different relation types. Thus, to gain a better understanding of the LLM's performance, we first analyze the LLM's performance with respect to relation types. In DBpedia, most entities are associated with a `https://schema.org/` type. Thus, we can categorize the relations into different types by the triples they belong to. We denote a relation's head/tail entity type as the most frequent schema type of the head/tail entity of the triples associated with the relation. For example, the relation `birthPlace` is associated with triples like `(Barack Obama, birthPlace, Hawaii)`, and the head entity `Barack Obama` is associated with the schema type `Person`, and the tail entity `Hawaii` is associated with the schema type `Place`. Then, the relation's head entity type is `Person`, and tail entity type is `Place`. We then analyze the LLM's performance with respect to these relation types. We report the performance of the LLMs on different relation types, by taking the average of the 3 metrics, *correctness*, *truthfulness*, and *informativeness*, for each relation type. We present the results in Figure 25. Here, "None" refers to entities not linked to a schema type. We also present a detailed analysis of the LLM's performance with respect to head and tail entity types in Appendix A.2. We can observe variability in model performance across relation types, such as "MusicGroup" and "CreativeWork" achieving high scores while "Area" and "Mountain" face lower performance, highlighting the diverse challenges in modeling different kinds of information. These performance differences suggest that the effectiveness of LLMs in handling structured knowledge heavily depends on the nature of the relations being modeled.

5 Conclusions

We introduce GraphEval, an innovative approach for appraising the efficacy of LLMs against a voluminous test dataset derived from an extensive knowledge graph containing over 10 million facts, significantly mitigating the necessity for costly human intervention. GraphEval, by embedding a judge module within the LLM itself, not only refines the evaluation process but also establishes a new benchmark for assessing the veracity of the information presented by these models. The empirical evidence from our experiments substantiates the judge model's proficiency in fact-checking, exhibiting a high degree of concordance with the accuracy of the LLM's outputs, and simultaneously diminishing the resources required for evaluation. The insights gleaned from our study shed light on the multifaceted performance of LLMs and lay the groundwork for future endeavors aimed at enhancing the reliability of their generated content. Moreover, we consider extending this work to cross-lingual KGs to evaluate the performance of various LLMs in different languages.

References

[1] Sun, K. et al.. Head-to-tail: How knowledgeable are large language models (llm)? AKA will llms replace knowledge graphs?. arXiv preprint arXiv:2308.10168, , 2023.

[2] Liang, P. et al.. Holistic evaluation of language models. arXiv preprint arXiv:2211.09110, , 2022.

[3] Lu, P. et al.. Learn to Explain: Multimodal Reasoning via Thought Chains for Science Question Answering. , 2022.

[4] Yao, Y. et al.. Editing large language models: Problems, methods, and opportunities. arXiv preprint arXiv:2305.13172, , 2023.

[5] Lin, S. et al.. TruthfulQA: Measuring How Models Mimic Human Falsehoods. :3214–3252, 2022.

[6] Weller, O. et al.. " According to..." Prompting Language Models Improves Quoting from Pre-Training Data. arXiv preprint arXiv:2305.13252, , 2023.

[7] Elliot Bolton, et al.. BioMedLM: A 2.7B Parameter Language Model Trained On Biomedical Text. , 2024.

[8] Yasunaga, M. et al.. Deep bidirectional language-knowledge graph pretraining. Advances in Neural Information Processing Systems, 35:37309–37323, 2022.

[9] Jia, Z. et al.. Tempquestions: A benchmark for temporal question answering. :1057–1062, 2018.

[10] Dao, T.. Flashattention-2: Faster attention with better parallelism and work partitioning. arXiv preprint arXiv:2307.08691, , 2023.

[11] Auer, S. et al.. Dbpedia: A nucleus for a web of open data. :722–735, 2007.

[12] Haoyu Wang, et al.. BlendFilter: Advancing Retrieval-Augmented Large Language Models via Query Generation Blending and Knowledge Filtering. , 2024.

[13] Shizhe Diao, et al.. Mixture-of-Domain-Adapters: Decoupling and Injecting Domain Knowledge to Pre-trained Language Models Memories. , 2023.

[14] Lewis, P. et al.. Retrieval-augmented generation for knowledge-intensive nlp tasks. Advances in Neural Information Processing Systems, 33:9459–9474, 2020.

[15] Azaria, A., Mitchell, T.. The Internal State of an LLM Knows When It's Lying. :967–976, 2023.

[16] Gallegos, I.O. et al.. Bias and Fairness in Large Language Models: A Survey. arXiv preprint arXiv:2309.00770, , 2023.

[17] Kotha, S. et al.. Understanding catastrophic forgetting in language models via implicit inference. arXiv preprint arXiv:2309.10105, , 2023.

[18] Liu, X. et al.. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. arXiv preprint arXiv:2110.07602, , 2021.

[19] Wang, C. et al.. Survey on factuality in large language models: Knowledge, retrieval and domain-specificity. arXiv preprint arXiv:2310.07521, , 2023.

[20] Carlson, A. et al.. Toward an architecture for never-ending language learning. 24:1306–1313, 2010.

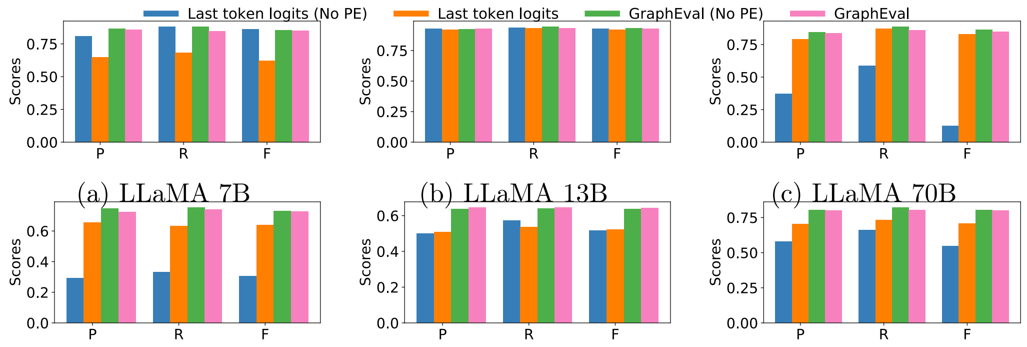
[21] Kwiatkowski, T. et al.. Natural Questions: A Benchmark for Question Answering Research. Transactions of the Association for Computational Linguistics, 7:453–466, 2019. MIT Press-Journals.

[22] Feng, S. et al.. FactKB: Generalizable Factuality Evaluation using Language Models Enhanced with Factual Knowledge. :933–952, 2023.

[23] Kim, J. et al.. KG-GPT: A general framework for reasoning on knowledge graphs using large language models. arXiv preprint arXiv:2310.11220, , 2023.

[24] Tian, K. et al.. Fine-tuning language models for factuality. arXiv preprint arXiv:2311.08401, , 2023.
[25] Luo, L. et al.. Reasoning on graphs: Faithful and interpretable large language model reasoning. arXiv preprint arXiv:2310.01061, , 2023.
[26] Shiqi Chen, et al.. FELM: Benchmarking Factuality Evaluation of Large Language Models. , 2023.
[27] Berglund, L. et al.. The Reversal Curse: LLMs trained on" A is B" fail to learn" B is A". arXiv preprint arXiv:2309.12288, , 2023.
[28] Bollacker, K. et al.. Freebase: a collaboratively created graph database for structuring human knowledge. :1247–1250, 2008.
[29] Wang, X. et al.. " My Answer is C": First-Token Probabilities Do Not Match Text Answers in Instruction-Tuned Language Models. arXiv preprint arXiv:2402.14499, , 2024.
[30] Ben-David, S. et al.. A theory of learning from different domains. Machine learning, 79:151–175, 2010. Springer.
[31] Wang, Y. et al.. Preserving In-Context Learning ability in Large Language Model Fine-tuning. arXiv preprint arXiv:2211.00635, , 2022.
[32] Liu, X. et al.. P-Tuning: Prompt Tuning Can Be Comparable to Fine-tuning Across Scales and Tasks. :61–68, 2022.
[33] Jiang, J. et al.. ReasoningLM: Enabling Structural Subgraph Reasoning in Pre-trained Language Models for Question Answering over Knowledge Graph. :3721–3735, 2023.
[34] Chang, Y. et al.. A survey on evaluation of large language models. ACM Transactions on Intelligent Systems and Technology, , 2023. ACM New York, NY.
[35] Cunxiang Wang, et al.. Evaluating Open-QA Evaluation. , 2023.
[36] Goodfellow, I.J. et al.. An empirical investigation of catastrophic forgetting in gradient-based neural networks. arXiv preprint arXiv:1312.6211, , 2013.
[37] Zhai, Y. et al.. Investigating the Catastrophic Forgetting in Multimodal Large Language Models. arXiv preprint arXiv:2309.10313, , 2023.
[38] Zhou, K. et al.. Don't Make Your LLM an Evaluation Benchmark Cheater. arXiv preprint arXiv:2311.01964, , 2023.
[39] Zhang, Y. et al.. Siren's song in the AI ocean: a survey on hallucination in large language models. arXiv preprint arXiv:2309.01219, , 2023.
[40] Joshi, M. et al.. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. :1601–1611, 2017.
[41] Suchanek, F.M. et al.. Yago: a core of semantic knowledge. :697–706, 2007.
[42] Dan Hendrycks, et al.. Measuring Massive Multitask Language Understanding. Proceedings of the International Conference on Learning Representations (ICLR), , 2021.
[43] Bordes, A. et al.. Translating embeddings for modeling multi-relational data. Advances in neural information processing systems, 26, 2013.
[44] Team, G. et al.. Gemma: Open models based on gemini research and technology. arXiv preprint arXiv:2403.08295, , 2024.
[45] Zhang, M. et al.. Knowledge Graph Enhanced Large Language Model Editing. arXiv preprint arXiv:2402.13593, , 2024.
[46] Huang, Y. et al.. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. Advances in Neural Information Processing Systems, 36, 2024.
[47] Liu, A. et al.. We're Afraid Language Models Aren't Modeling Ambiguity. arXiv preprint arXiv:2304.14399, , 2023.
[48] Ren, R. et al.. Investigating the factual knowledge boundary of large language models with retrieval augmentation. arXiv preprint arXiv:2307.11019, , 2023.
[49] Touvron, H. et al.. Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288, , 2023.
[50] Chen, S. et al.. Recall and Learn: Fine-tuning Deep Pretrained Language Models with Less Forgetting. :7870–7881, 2020.
[51] Tan, Y. et al.. Can ChatGPT replace traditional KBQA models? An in-depth analysis of the question answering performance of the GPT LLM family. :348–367, 2023.

[52] Talmor, A., Herzig, J., Lourie, N., and Berant, J. CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge. arXiv preprint arXiv:1811.00937, 2019. https://arxiv.org/abs/1811.00937.



(a) LLaMA 7B (b) LLaMA 13B (c) LLaMA 70B

(d) Gemma 2B (e) Gemma 7B (f) Average

Figure 26: Evaluation scores on the judge model's performance on the labeled validation set. P, R, and F are Precision, Recall, and F1 Score.

Substitute Model	**LLaMA 2 7B**			**LLaMA 2 13B**			**LLaMA 2 70B**		
	P	R	F	P	R	F	P	R	F
LLaMA 2 7B	.858	.845	.850	.928	.934	.930	.837	.861	.848
LLaMA 2 13B	.850	.868	.855	.930	.940	.932	.837	.851	.844
LLaMA 2 70B	.868	.883	.871	.924	.942	.931	.858	.876	.866

Table 13: Ablation on the LLaMA models as substitute models. The i-th row and j-th column denote the result of using i-th LLM as the substitute hidden state input for training on j-th model's labels. P, R, and F are Precision, Recall, and F1 Score.

A Appendix

A.1 Judge Model Analysis

We analyze the judge model's performance on the labeled validation set. We compare GraphEval's judge model by using the last token logit as the judge model. This is a common practice in evaluating LLMs, as the last token logit is the most common way to extract the hidden state of the LLMs. We also analyze the judge model with or without the prompt encoder (PE), as it may have a negative impact on the judge model's performance. We refer to Figure 26 for the judge model's performance on the labeled validation set.

Accuracy Analysis The GraphEval model, both with and without Prompt Encoder (PE), consistently outperforms the score of using Last token logits in almost all configurations and metrics. This indicates the effectiveness of the GraphEval approach in capturing the nuances of the evaluation task.

Ablation Study *On Prompt Encoder:* As Figure 26 shows, the comparison between models with and without PE indicates a slight performance variation. For GraphEval, the presence of PE does not significantly alter the performance, suggesting that our method of evaluating LLMs is robust to the inclusion or exclusion of PE. For the Last token logits method, removing PE generally results in a perturbation in performance. However, the GraphEval approach's consistency suggests a potentially different or more advanced mechanism of evaluation that is less dependent on PE. *On Substitute Models:* We also evaluate the judge model's performance on different LLMs as hidden state input. We refer to Table 13 for the judge model's performance on different LLMs as hidden state input. We can see that,

Models	LLaMA 2 7B		LLaMA 2 13B		LLaMA 2 70B		Gemma 2B		Gemma7B	
	Speed	#GPUs	Speed	#GPUs	Speed	#GPUs	Speed	#GPUs	Speed	#GPUs
TG (A6000)	2.26	1	1.07	2	0.09	4	2.06 (1.82)	1	2.18 (1.28)	1
GraphEval (A6000)	121.34	1	120.10	1	117.90	1	388.61	1	389.04	1
TG (A100)	2.80	1	1.48	1	0.21	2	2.47	1	2.42	1
GraphEval (A100)	210.59	1	213.05	1	210.30	1	731.98	1	735.62	1

Table 14: Efficiency evaluation. Speed denotes the average number of triple facts on which a conclusion can be given in one second. #GPUs denotes the least number of GPUs to run without OOM. TG denotes text generation. The numbers in parentheses are the speed without Flash Attention 2.

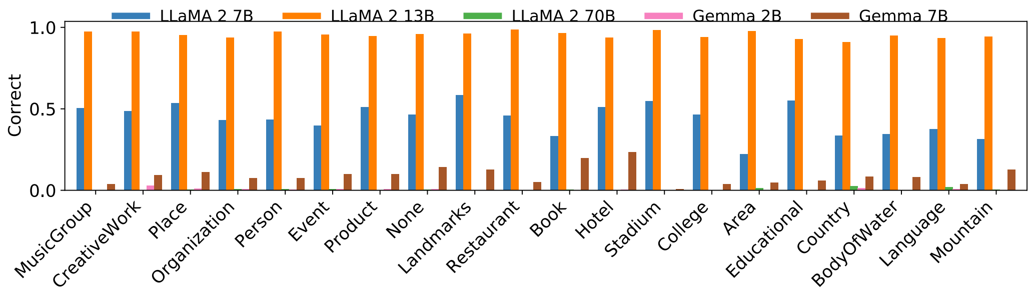
generally, when larger models are applied for feeding the hidden states, there is a slight increase in the fitting accuracy of the judge model. However, there is no significant difference in the judge model's performance.

Efficiency study We also analyze the judge model's efficiency by measuring the time it takes to make a prediction on one triple. The speed of text generation refers to the average rate at which the LLM completes generating a response consisting of one sentence derived from a triple. It's important to recognize that the pace of text generation can vary with different prompts because the LLM may produce responses of varying lengths. Therefore, for a more consistent measure of text generation speeds, it's advisable to consider the rate of token generation. Despite this, our evaluation framework, GraphEval, does not depend on text generation and operates on a triple-based unit. Consequently, we continue to use the triple as the unit of measurement for time. We use the same hardware and software environment for all the experiments. We compare the average speed of the judge model with text generation. We report the time it takes to make a prediction in Table 14. The attention implementation and precision are the same for text generation and for the judge model's input model. We can see that the judge model is significantly faster than text generation. This indicates that the judge model is efficient in evaluating the LLMs. Also, benefiting from the substitute model, our evaluation speed and GPU requirement does not grow with the LLM size, which is an advantage for evaluating large LLMs. We also observe that, paradoxically, the Gemma 2B model operates slower than the Gemma 7B model, despite its smaller size. This counterintuitive result could be attributed to the implementation of Flash Attention 2. To draw a fair comparison, we documented the text generation speed on A6000 GPUs excluding Flash Attention 2, which is indicated within parentheses. The comparative data reveals that Gemma 2B is faster than Gemma 7B when Flash Attention 2 is not utilized. Notwithstanding this, Gemma 2B demonstrates enhanced performance when Flash Attention 2 is active. Therefore, for the sake of consistency, we have decided to maintain the results acquired with Flash Attention 2.

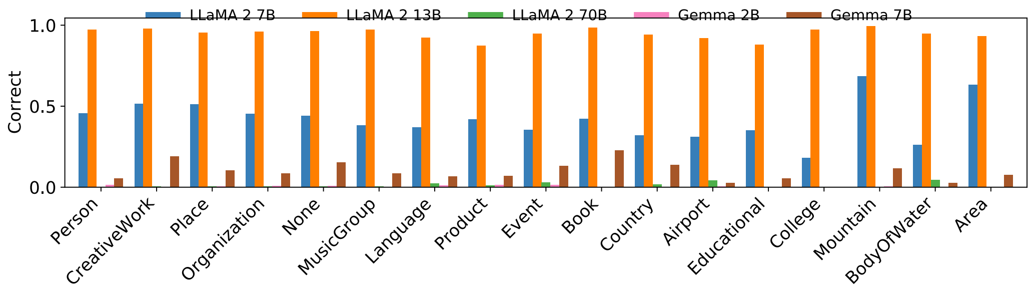
A.2 Detailed Relation Type Analysis

Llama Family Analysis Across the LLaMA family, a progressive improvement in performance is observed from 7b to 13b. The 7b model shows decent performance across categories with a particular strength in the *truthfulness*. However, its *informativeness* and *correctness* metrics show room for improvement, particularly in categories like Book, Hotel, and College, indicating a struggle to accurately provide informative and correct classifications in more nuanced or specific domains.

The LLaMA 13b model demonstrates a significant leap in performance, especially in *informativeness* and *correctness*, nearly reaching perfection across most categories. This jump can be attributed to the model's increased capacity, enabling it to understand and process the nuances of various entities



(a) *Correctness* vs Head entity type



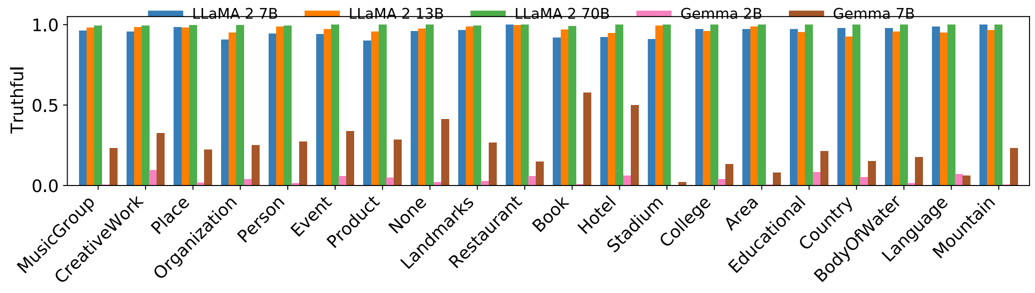
(b) *Correctness* vs Tail entity type

Figure 27: The LLM's *correctness* with respect to head entity types and tail entity types

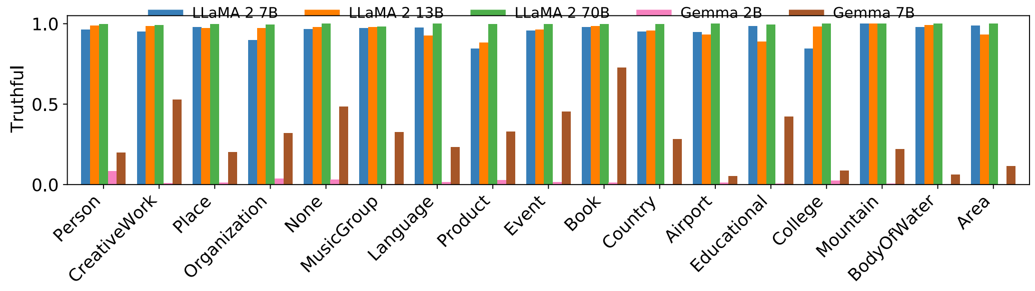
better, resulting in remarkably high scores in nearly all categories, especially noticeable in MusicGroup, CreativeWork, and Place.

The LLaMA 70b results appear anomalous with extremely high *truthfulness* scores but negligible *informativeness* and *correctness* across all categories. We suspect this discrepancy might be due to the model's knowledge awareness [48], where the model might be less confident in its responses when the parameters are increased, leading to a higher proportion of "I don't know" responses. This could explain the high *truthfulness* scores but low *informativeness* and *correctness* metrics, as the model might be too cautious to provide definitive answers.

Gemma Family Analysis The Gemma models present an interesting contrast. The Gemma 2b model shows a tendency towards high *informativeness* in certain categories like MusicGroup and Book but lacks behind significantly in *truthfulness* and *correctness* metrics. This suggests that while the model might be picking up on relevant information, it struggles to accurately validate the truth behind that information or its applicability to the queried entities. The Gemma 7b model shows improvement in the *truthfulness* metric compared to Gemma 2b, particularly noticeable in categories like Book and Hotel, and even surpasses LLaMA 7b in certain areas like None and Restaurant. However, it still significantly lags behind the LLaMA models, particularly LLaMA 13b, in both *informativeness* and *correctness*. The improved but still limited performance suggests that while Gemma 7b has a better grasp over the veracity of information compared to Gemma 2b, it still struggles with providing highly informative and correct outputs consistently across various entities.



(a) *Truthfulness* vs Head entity type



(b) *Truthfulness* vs Tail entity type

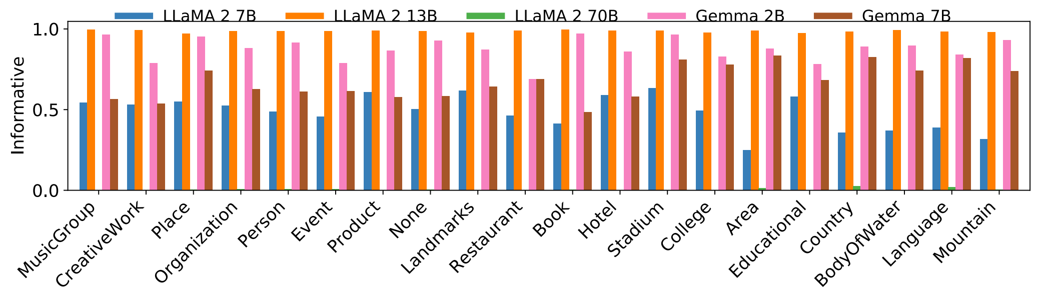
Figure 28: The LLM's *truthfulness* with respect to head entity types and tail entity types

A.3 Correlation Analysis

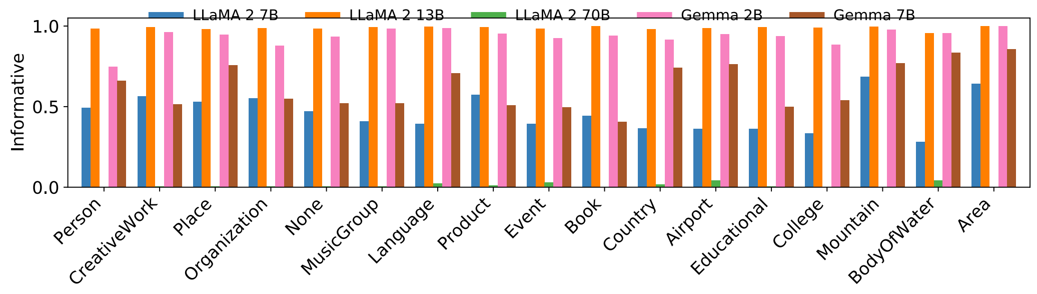
As the current language models are all exposed to Wikipedia knowledge during training, we are interested in how the LLM performance is correlated with the attributes of the triples in the knowledge graphs. As an example, if an entity has a higher degree, it may be linked to more documents, and the LLM may have more chances to learn about the entity during training. Another example is the popularity of the entity. If the entity is more popular, it may be linked to more external documents because it summarizes the relevant knowledge and provides high-level ideas to the general public, and the LLM may have more chances to learn about the entity during training. This raises the question of whether the LLM's performance is correlated with the attributes of the triples in the knowledge graphs. For the entities in a knowledge graph, the degree of an entity is the number of edges connected to the entity. We also collect the *pageviews* of the entities in the knowledge graph from Wikimedia[1], which is the number of pageviews of the Wikipedia page of the entity. This can be seen as a measure of the popularity of the entity because a popular page should appeal to the significant attention of the readers. We collect the pageviews, in the time period of the entities in the knowledge graph from the Wikipedia page of the entity. After collecting the degree and pageviews of the entities in the knowledge graph, we can aggregate the degree and pageviews of the entities to the triples, by simply taking the average of the degree and pageviews of the head and tail entities of the triples.

Here, we analyze whether the LLM's performance is correlated with the attributes of the triples in the knowledge graphs, such as the entity's degree, and page views. We refer to Figure 30 for the correlation heatmap of the LLMs' hidden states and the judge model's predictions. Here, 'T' stands for *Truthful*, 'I' stands for *Informativeness*, 'C' stands for *Correctness*, 'P' stands for Pageviews, and 'D' stands for Degree. We can see that the LLM's performance does not show a strong correlation with the attributes of

[1]https://wikimedia.org/api/rest_v1/metrics/pageviews/per-article/en.wikipedia.org/all-access/all-agents



(a) *Informativeness* vs Head entity type



(b) *Informativeness* vs Tail entity type

Figure 29: The LLM's *informativeness* with respect to head entity types and tail entity types

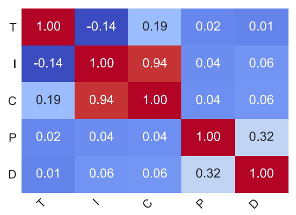
Models	**Llama 3 1B**	**Llama 3 3B**	**Llama 3 8B**	**Llama 3 70B**
CommonsenseQA	54.65	37.73	30.36	**22.28**
TruthfulQA	58.35	52.32	49.03	**23.45**

Table 15: Knowledge boundary analysis results for the Llama 3 series models on CommonsenseQA and TruthfulQA datasets. The table reports the Expected Calibration Error (ECE) values (%) measuring the alignment between model confidence and correctness. Lower is better. The best results are in bold.

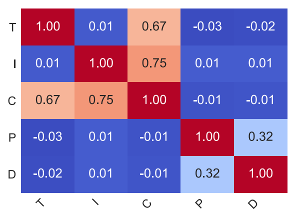
the triples in the knowledge graphs. This indicates that the LLM's performance is not directly correlated with the attributes of the triples in the knowledge graphs. However, the different metrics of LLMs may correlate with each other, such as *Truthful* and *Informativeness*, which is expected. This can be explained by the fact that certain attributes, like the degree of an entity in the knowledge graph, can be misleading. For example, degree is often correlated with popularity, but the popularity metric is 0 for many entities, particularly those in the long tail. This uneven distribution limits the usefulness of popularity as a reliable metric for evaluating LLM performance. In other words, while high-degree or popular entities may influence LLM performance to some extent, the vast majority of entities are long-tail, and their sparse or zero popularity values do not strongly correlate with performance outcomes. This highlights the need for more nuanced or domain-specific metrics to assess LLM performance effectively.

A.4 Knowledge Boundary Analysis

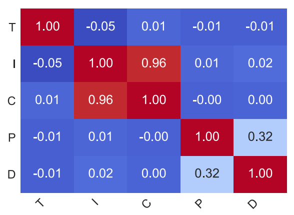
We analyze the knowledge boundaries of large language models (LLMs), as discussed in Section 4.2 and [48], which suggest that larger models have a better understanding whether they know an answer or not. To investigate this hypothesis, we conduct experiments using the Llama 3 model series on two



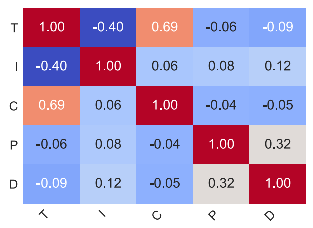
(a) LLaMA-2-7B



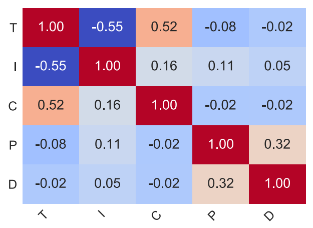
(b) LLaMA-2-13B



(c) LLaMA-2-70B



(d) Gemma-2B



(e) Gemma-7B

Figure 30: Correlation heatmap of the LLMs' hidden states and the judge model's predictions.

question-answering datasets: CommonsenseQA [52] and TruthfulQA [5]. To assess whether an LLM understands its own knowledge boundaries, we directly elicit confidence scores for each answer through prompting, then we calculate the Expected Calibration Error (ECE). ECE measures the misalignment between the correctness of answers and the models' confidence. Mathematically, for LLM responses $\mathcal{A}$, ECE is defined as:

$$\text{ECE} = \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} |\mathbb{I}(a) - \text{conf}(a)|\,, \tag{5}$$

where $\mathbb{I}(a)$ is an indicator function that outputs 1 if a is correct and 0 otherwise, and conf(a) denotes the confidence score assigned by the model. The experimental results are presented in Table 15.

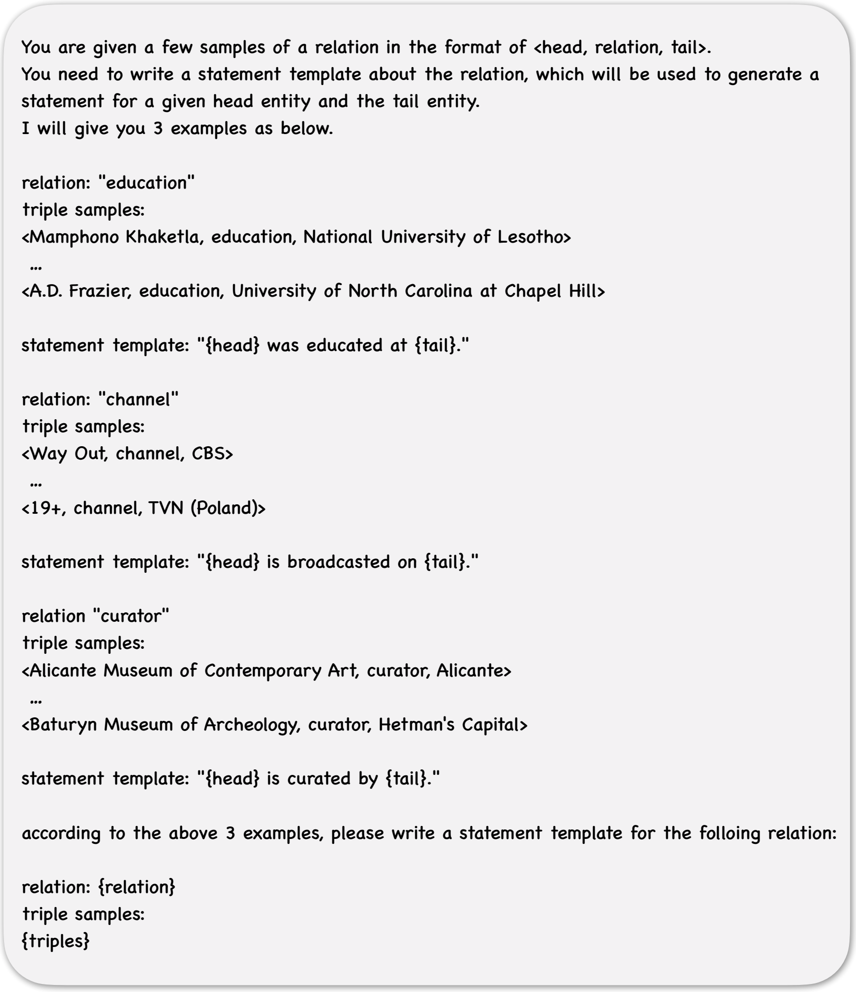
The results in Table 15 align with our previous hypothesis across different Llama 3 model sizes. For both CommonsenseQA and TruthfulQA, the ECE values decrease as model size increases, indicating better alignment between confidence and correctness in larger models. Specifically, for CommonsenseQA, the Llama 3 70B model achieves the lowest ECE (22.28%), demonstrating superior calibration compared to smaller models like Llama 3 1B (54.65%). Similarly, on TruthfulQA, the Llama 3 70B model achieves an ECE of 23.45%, significantly outperforming the smaller Llama 3 1B model with an ECE of 58.35%.

These findings align with the hypothesis that larger models are better calibrated in estimating their confidence, which can partially explain why larger models are more likely to answer "I don't know" when asked about a question as shown in Section 4.2.

Overall, GraphEval aligns well with the results on TruthfulQA and CommonsenseQA, demonstrating that it effectively captures the model's factuality and informativeness across diverse knowledge domains. This alignment validates the robustness of our evaluation framework and confirms its consistency with established benchmarks for assessing LLM reasoning and truthfulness.

A.5 Detailed Settings

Relation templates We use the relation templates to create queries for evaluating the models. These templates are first generated by GPT with Web API in a few-shot manner, then manually curated to ensure the quality of the templates. We refer to Figure 31 for the prompt used for generating the



```
You are given a few samples of a relation in the format of <head, relation, tail>.
You need to write a statement template about the relation, which will be used to generate a
statement for a given head entity and the tail entity.
I will give you 3 examples as below.

relation: "education"
triple samples:
<Mamphono Khaketla, education, National University of Lesotho>
 ...
<A.D. Frazier, education, University of North Carolina at Chapel Hill>

statement template: "{head} was educated at {tail}."

relation: "channel"
triple samples:
<Way Out, channel, CBS>
 ...
<19+, channel, TVN (Poland)>

statement template: "{head} is broadcasted on {tail}."

relation "curator"
triple samples:
<Alicante Museum of Contemporary Art, curator, Alicante>
 ...
<Baturyn Museum of Archeology, curator, Hetman's Capital>

statement template: "{head} is curated by {tail}."

according to the above 3 examples, please write a statement template for the folloing relation:

relation: {relation}
triple samples:
{triples}
```

Figure 31: The prompt to generate the relation template.

relation templates. The prompt is designed to ask the model to generate a query for a given relation type. The model is asked to generate a query that can be used to judge the factuality of the relation type. We then manually curate the generated templates to ensure the quality of the templates. We refer to Table 16 for the curated relation templates. Due to the large number of relation types in the DBpedia knowledge graph, we only showcase a few relation templates in the table, these templates are the most common relation types in the knowledge graph, sorted by the number of triples associated with the relation type. We can see that the relation templates are comprehensive and cover a wide range of topics. This can be seen as a source of multiple-domain knowledge for evaluating the LLMs.

Data and Model We download the DBpedia data dump from https://www.dbpedia.org/. We use the turtle format of the DBpedia knowledge graph. We directly use the LLaMA 2 and Gemma from the Hugging Face model hub. The model cards are meta-llama/Llama-2-7b-chat-hf, meta-llama/Llama-2-13b-chat-hf, meta-llama/Llama-2-70b-chat-hf, gemma-team/gemma-2b-chat-hf, and gemma-team/gemma-7b-chat-hf.

Instruction used for the LLaMA and Gemma models We report the instructions used for creating queries for the LLaMA and Gemma models. The instruction is designed to ask the model to

Relation	**Template**	**Count**
birthPlace	The birthplace of {head} is {tail}.	1,465,157
team	{head} is a part of the {tail} team.	1,265,483
subdivision	The subdivision of {head} is {tail}.	1,070,387
country	{head} is from the country {tail}.	766,844
starring	{head} is a character in a movie or play {tail}".	540,937
location	The location of {head} is {tail}.	523,283
type	The type of {head} is {tail}.	480,274
deathPlace	{head} passed away in {tail}.	435,869
timeZone	The time zone of {head} is {tail}.	433,915
genre	The genre of {head} is {tail}.	415,336
homepage	The homepage of {head} is {tail}.	366,745
position	The position of {head} is {tail}.	319,196
seeAlso	The related item to {head} under the ↪label 'seeAlso' is {tail}.	296,615
writer	The writer of {head} is {tail}.	249,017
almaMater	The alma mater of {head} is {tail}.	217,533
occupation	The occupation of {head} is {tail}.	200,615
award	The award won by {head} is {tail}.	181,521
recordLabel	The record label associated with {head} is {tail}.	178,657
party	The party that {head} is affiliated with is {tail}.	170,931
producer	The producer of {head} is {tail}.	169,628
formerTeam	{head} used to play for {tail} team.	151,374
family	{head} belongs to the {tail} family.	148,818
currentMember	The current member of {head} is {tail}.	148,739
battle	{head} participated in the following battles: {tail}.	148,188
nationality	The nationality of head is tail.	147,525
director	The director of {head} is {tail}.	145,621
associatedBand	The band associated with {head} is {tail}.	135,597
associatedMusical ↪ Artist	The musical artist associated with {head} ↪ in the music industry is {tail}.	135,582
class	The class of {head} is {tail}.	127,837
order	The order of {head} is {tail}.	123,626

Table 16: Relations templates.

judge whether the statement is true or false. We refer to Table 17 for the instruction used for creating queries. We use the same instruction for both the LLaMA and Gemma models with little modification to adjust the model's instruction format. With this instruction, the most frequent responses of LLMs are *Yes, the statement is true*, *No, the statement is false*, and *I don't know*, with some variations on the suffix, mainly explaining the reason for the answer. This is what we expect from the LLMs when using a judge model (or the first-token logit as well), since the judge model doesn't use the LLM's response, but the hidden state of the LLM, which makes the consistency of the response format important.

A.6 Language Setting

As a framework, GraphEval is not constrained by language, as long as the input is in the form of a knowledge graph. However, we did not conduct experiments on multilingual or cross-lingual datasets in the current work. Current experiments are conducted on English knowledge graphs.

Model	Instruction
LLaMA 2	Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.\n\n ### Instruction:\n You are given a statement. You are asked to judge whether the statement is true or false. Answer 'Yes, the statement is true.' if you know the statement is true. Answer 'No, the statement is false.' if you know the statement is false. Otherwise, answer 'I don't know.'\n\n### Input: **Input** \n\n### Response:\n\n
Gemma	start_of_turn>user Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.\n\n### Instruction:\n You are given a statement. You are asked to judge whether the statement is true or false. Answer 'Yes, the statement is true.' if you know the statement is true. Answer 'No, the statement is false.' if you know the statement is false. Otherwise, answer 'I don't know.'\n\n### Input: **Input** <end_of_turn><start_of_turn>model\n\n The answer is "

Table 17: Instruction used for creating queries.