Lightweight Query Checkpoint: Classifying Faulty User Queries to Mitigate Hallucinations in Large Language Model Question Answering

Anonymous ACL submission

Abstract

Question Answering (QA) with large language models has shown impressive performance, yet hallucinations still persist, particularly when user queries carry incorrect premises, insufficient context, or linguistic ambiguity. To address this issue, we propose **Lightweight Query Checkpoint (LQC)**, a small classification model that detects verification-required queries before the LLM generates a potentially faulty answer. LQC leverages hidden states extracted from intermediate layers of a smaller-scale, non-instruct-tuned LLM to effectively distinguish queries requiring verification from clear queries. We first systematically define categories of queries that need verification, construct a dataset comprising both defective and clear queries, and train a binary contrastive learning model. Through extensive experiments on various QA datasets, we demonstrate that incorporating LQC into QA pipelines reduces hallucinations while preserving strong answer quality.

1 Introduction

Recently, as the performance of large language models has steadily improved, user experience in QA systems has also been greatly enhanced (Jin et al., 2024; Ren et al., 2024; Fu et al., 2020). Through simple interfaces such as web browsers or chatbots, users can easily input queries and receive immediate answers, providing a more intuitive user experience. However, in real-world usage, questions containing incorrect premises, insufficient context, or linguistic ambiguities are often entered (Tanjim et al., 2025; Kim et al., 2024; Vadlapati, 2023), which still poses a high risk of the model producing incorrect answers—so-called "hallucinations."

For instance, as shown in Figure 1, a question asking about the legal age to buy alcohol, such as *"At what age is it legal to buy alcohol?"*, without providing **any national or regional context whatsoever**, can lead the LLM to assume a specific country arbitrarily or present a universal figure without sufficient basis, thereby conveying inaccurate information (Thakur et al., 2024). In particular, because users may not realize their question is ambiguous, they tend to trust any inaccurate information provided by the model. If such situations recur, misinformation can spread indiscriminately, and this risk becomes especially serious in sensitive domains such as medicine or law (Alber et al., 2025; Stevenson and Guo, 2010).

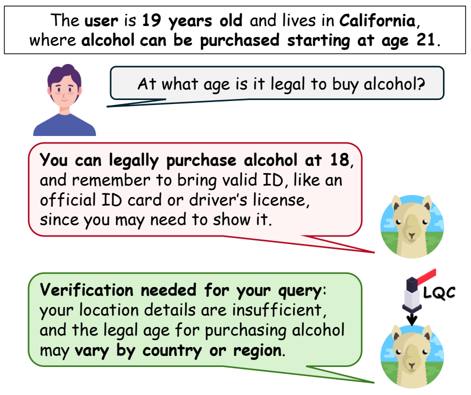

The user is 19 years old and lives in California, where alcohol can be purchased starting at age 21.
At what age is it legal to buy alcohol?
You can legally purchase alcohol at 18, and remember to bring valid ID, like an official ID card or driver's license, since you may need to show it.
LQC
Verification needed for your query: your location details are insufficient, and the legal age for purchasing alcohol may vary by country or region.

Figure 1: LQC before-and-after QA situation

In order to address this issue, this paper proposes a lightweight model called **Lightweight Query Checkpoint (LQC)**, which utilizes internal representations within the LLM to **preemptively identify questions "in need of verification"**. By detecting queries with potential flaws before generating answers and explicitly guiding users about those flaws, LQC reduces hallucinated answers that may arise from ambiguous questions, while still ensuring fast responses for normal questions.

To achieve this, LQC utilizes a binary classification model that takes as input the hidden states extracted from an intermediate layer of a transformer(Vaswani et al., 2023) based LLM. In other words, it pairs a "normal (Clear) query" with a "Defective query (one that requires verification)" and performs contrastive learning, thereby aiming to achieve high classification performance even in resource-constrained environments. Here, a "Defective query" is defined as one that contains a flaw making it impossible to provide a single, definitive answer—specifically, it might include incorrect premises that render the query unanswerable, lack concrete details such as time or location, or be open to multiple interpretations due to grammatical or linguistic ambiguity.

In this study, we constructed a publicly available dataset of such queries organized by category and conducted various experiments to verify that using LQC can reduce hallucinatory answers in real QA pipelines. Furthermore, we compared how both the model size (from which hidden states are extracted) and whether the model had undergone instruct tuning affect classification performance. Through this comparison, we demonstrated that even smaller-scale, non-instruct models can yield clearer signals for defect detection.

- We propose LQC, which determines the presence of query flaws through internal representations of LLM inputs, and confirm that applying LQC to actual QA environments enables the construction of a more reliable QA pipeline.
- We systematically define various types of errors inherent in user queries and build a public QA dataset based on these types, making it available for research purposes.
- We show that hidden states extracted from intermediate layers of smaller-scale or Non-Instruct LLMs provide even clearer signals for error detection, confirming their applicability in resource-limited environments.

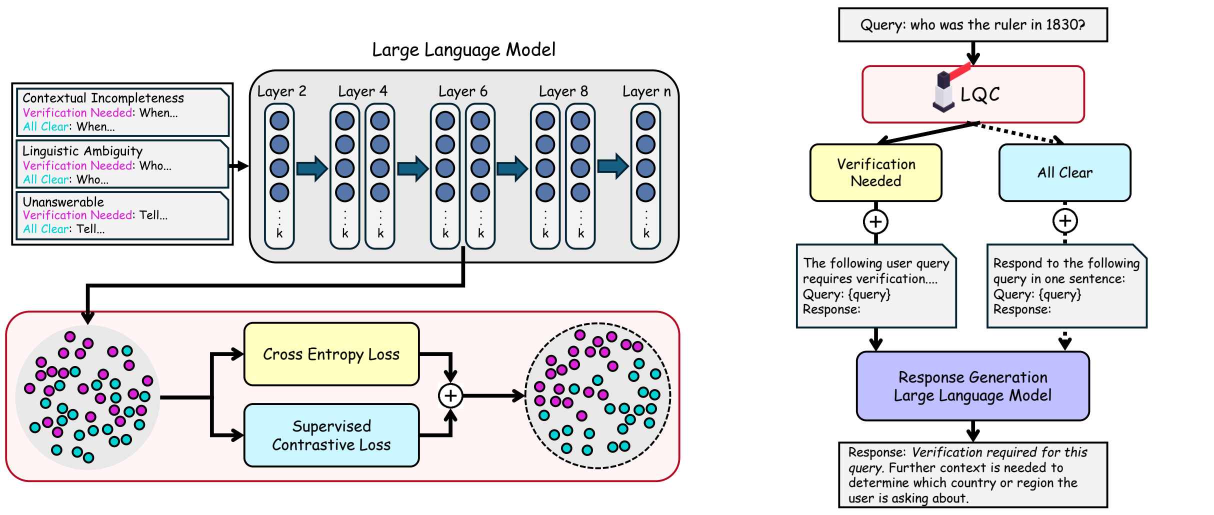
2 Related Work

Internal Representations of LLM Recently, there has been active research on interpreting the internal representations of large language models (LLMs) and leveraging them (Zhang et al., 2024a; Lin et al., 2024b; Muttenthaler et al., 2020). Transformer-based models such as BERT (Devlin et al., 2019) have demonstrated excellent performance (Oren et al., 2024; Chen et al., 2024), leading to the development of probing techniques (He et al., 2024; Li et al., 2025; Chen and Gao, 2022) and visualization analyses (Katz and Belinkov, 2023) for examining their hidden layers. A representative example is the study by (Jawahar et al., 2019), which analyzed BERT's layer-wise representations. They observed a hierarchical structure in which lower layers capture superficial lexical and phrase-level information, intermediate layers capture syntactic features, and upper layers capture increasingly deeper semantic features (Tenney et al., 2019). There is also growing interest in whether these internal representations of LLMs can detect anomalies in queries (Slobodkin et al., 2023). In (Ji et al., 2024), it was reported that solely by looking at the hidden states of an LLM, one can determine whether a question contains content unseen during training and whether the model is uncertain about the answer. Another line of research found that although LLMs tend to produce hallucinatory answers when facing unanswerable questions, the hidden representation of merely the first output token already contains information about whether the question can be answered (Slobodkin et al., 2023).

Verification Needed Queries Detecting erroneous questions or questions without correct answers in the process of utilizing LLMs is a key task for mitigating hallucinatory responses (Cole et al., 2023). For example, in the domain of reading comprehension QA, some methods attach an additional classification head to a Transformer model (such as BERT). This classification head is then trained to predict whether a question is unanswerable (Schmidt et al., 2020; Guan et al., 2022; Jiang et al., 2022). Classifying user query intentions and detecting out-of-domain queries are also important issues for improving QA system efficiency. Recently, contrastive learning approaches have garnered attention for determining question types and domain suitability (Wang and Mine, 2024; Yue et al., 2021). In addition, there have been multiple attempts to improve the detection of incorrect answers by fine-tuning the LLM's parameters themselves (Kim et al., 2024). While these methods are effective at analyzing queries and identifying errors, they often exhibit limitations, such as being biased toward certain error types and thus lacking generality.

3 Methodology

The primary goal of this study is to enable an open-domain, LLM-based question-answering system to recognize the "need for verification" when a user query contains flaws, explaining this need to the user, while still delivering high-quality answers for


Large Language Model
Contextual Incompleteness
Verification Needed: When...
All Clear: When...
Linguistic Ambiguity
Verification Needed: Who...
All Clear: Who...
Unanswerable
Verification Needed: Tell...
All Clear: Tell...
Layer 2
Layer 4
Layer 6
Layer 8
Layer n
Cross Entropy Loss
Supervised Contrastive Loss
Query: who was the ruler in 1830?
LQC
Verification Needed
All Clear
The following user query requires verification.... Query: {query} Response:
Respond to the following query in one sentence: Query: {query} Response:
Response Generation Large Language Model
Response: Verification required for this query. Further context is needed to determine which country or region the user is asking about.

(a) LQC Training Procedure

(b) Workflow when LQC is Integrated into a QA Pipeline

Figure 2: (a) Showing how LQC is trained and (b) the structure illustrating how it operates when applied to an LLM.

queries that do not contain any flaws. To achieve this, we propose an LLM-based classifier called **Lightweight Query Checkpoint (LQC)**, which determines in advance whether the user query requires verification before the LLM generates an answer. The overall pipeline of the LLM-based QA system incorporating LQC is shown in Figure 2. Specifically, Figure 2(a) illustrates the training process for LQC, and Figure 2(b) describes the stage in which different answer templates are generated according to LQC's classification results (whether verification is required). In this chapter, we examine each of these steps in detail.

3.1 Classification with Contrastive Learning

Training Dataset Collection In order to build a contrastive learning-based classifier, we first divided the "queries requiring verification" into three main types. Specifically, these types are: *Contextual Incompleteness*, which lacks the necessary information; *Linguistic Ambiguity*, which involves linguistic ambiguity; and *Unanswerable*, which is difficult to answer because it contains incorrect assumptions or requests information that does not exist. We then collected data from publicly available datasets in which each of these three types includes both defective and non-defective queries. Detailed information on dataset construction can be found in Appendix A.

Hidden State Extraction To obtain a representation of a user query x based on a pretrained LLM, we first structure the user query in a chat template format, convert it into a token sequence, and then feed it into the LLM. The input token sequence passes through internal Transformer layers, generating a hidden state for each token. In this study, we use the hidden states extracted from one of the intermediate layers as the query embeddings. We then apply a Hybrid Pooling strategy that combines Mean Pooling and Last Token Pooling on the extracted hidden states: Mean Pooling reflects the overall context, while the last token supplements key information of the sentence. We implement a weighted sum of the mean vector of all tokens in the sequence ($\mathbf{h}$mean) and the last token vector ($\mathbf{h}$last) as follows:

$$\mathbf{h}_{\text{hybrid}} = \alpha\, \mathbf{h}_{\text{last}} + (1-\alpha)\, \mathbf{h}_{\text{mean}}, \qquad (1)$$

α is a hyperparameter that adjusts the weight between the two vectors. By using it to balance both the overall query distribution and the key token information, it can achieve higher classification performance compared to simple pooling.

Binary Classifier Training Based on Contrastive Learning We feed the embedding $\mathbf{h}_{\text{hybrid}}$ obtained through Hybrid Pooling into an MLP (Multi-Layer Perceptron)–based classifier. At this point, we apply contrastive learning (Gao et al., 2021) to enhance the model's ability to distinguish among queries. Specifically, we project pairs of queries—those requiring verification versus normal queries—into the latent space in such a way that embeddings of the same class remain close to each other, whereas embeddings of different classes are spaced far apart. In the latent space, for a vector h, let $\mathcal{H}^+$ be the set of embeddings belonging to the same class, and let $\mathcal{H}^-$ be the set of embeddings belonging to a different class. The model is trained so that h stays close to $\mathcal{H}^+$ and far from $\mathcal{H}^-$. The following equation represents the training objective of this Contrastive Learning approach.

$$\mathrm{CL}(h, \mathcal{H}^+, \mathcal{H}^-) = \\ -\log \frac{\sum_{h' \in \mathcal{H}^+} \exp\{\mathrm{sim}(h, h')/\tau\}}{\sum_{h' \in (\mathcal{H}^+ \cup \mathcal{H}^-)} \exp\{\mathrm{sim}(h, h')/\tau\}}$$

where sim refers to the cosine similarity and τ is the temperature value. Let $\mathcal{H}c$ be the set of normal queries and $\mathcal{H}v$ be the set of queries requiring verification. We define the contrastive loss as follows:

$$\mathcal{L}_{\mathrm{cont}} = \mathrm{CL}(h_{\mathrm{c}}, \mathcal{H}_{\mathrm{c}}, \mathcal{H}_{\mathrm{v}}) + \mathrm{CL}(h_{\mathrm{v}}, \mathcal{H}_{\mathrm{v}}, \mathcal{H}_{\mathrm{c}}) \quad (3)$$

Finally, the total loss of LQC is obtained by combining the cross-entropy loss $\mathcal{L}_{\mathrm{ce}}$ for classification as follows.

$$\mathcal{L} = \mathcal{L}_{\mathrm{ce}} + \mathcal{L}_{\mathrm{cont}} \quad (4)$$

3.2 Response Generation

Depending on the classification model's prediction regarding whether the query requires verification, we provide one of two types of input to the LLM for answer generation. If the classification model predicts that a query needs verification, we supply an additional verification template (for example, "This query requires additional verification, please provide the reason why verification is necessary.") together with the query. This encourages the LLM to first explain any errors or ambiguities to the user and then ask for additional information. Conversely, if the model determines that the query is clear, we use a simple template similar to existing open-domain QA methods as the LLM input. Figure 2 provides an overview of the entire system's scenario, where QA proceeds in the order of verification classification $\rightarrow$ template branching $\rightarrow$ final answer generation.

4 Experiments

In this chapter, we evaluate the proposed LQC framework, which classifies queries and then generates responses, using various datasets and baseline methods.

4.1 Datasets

We collected both clear queries and queries requiring verification from four different datasets. This setup is well-suited for evaluating how the proposed classification model performs under diverse conditions. **SituatedQA-Geo** and **SituatedQA-Temp**(Zhang and Choi, 2021) are datasets where queries are highly dependent on a specific location (Geo) or a specific time (Temp), and models must leverage contextual understanding to answer accurately. **CLAMBER-Linguistic Ambiguity**(Zhang et al., 2024b) presents queries with polysemy or syntactic ambiguity, and **CoCoNot-False-Presuppositions**(Brahman et al., 2024) includes queries containing false premises that the model must identify and address appropriately.

4.2 Baselines

To compare the performance of our proposed model, we generate answers using two different approaches. First, the INSTANT approach immediately feeds the user query to the LLM without any additional steps, to see if the LLM can inherently recognize queries requiring verification without the help of a classification model. The second approach, REFLECT, uses a prompt that instructs the model itself to determine whether the query "needs verification." This also relies entirely on the model's own reasoning capabilities, without a classification model.

4.3 Evaluation Metrics

In this study, we categorize the model's responses into five categories, as shown in Figure 3. Based on these categories, we compute the Accuracy, F1, $\mathrm{F1}_{\mathrm{c}}$, and $\mathrm{F1}_{\mathrm{v}}$ metrics (Kim et al., 2024) for evaluation.

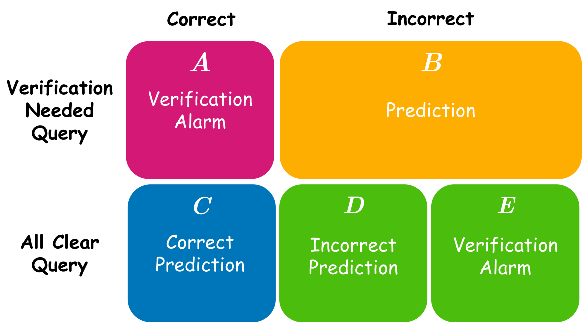


Figure 3: Metric

By comprehensively evaluating these five categories, we can ascertain whether the model appropriately identifies queries that require verification (comparing A vs. B) and still provides accurate answers for queries that are clear (comparing C vs. D, E). We conducted an automated evaluation using a GPT-4o model to classify the generated responses into categories A through E.

Accuracy & F1 Among all responses, **Accuracy** measures the proportion of cases where the model correctly requests verification for queries that need it (A) and provides correct answers for clear queries (C). It is calculated as follows: $\text{Accuracy} = \frac{A+C}{A+B+C+D+E}$. **F1** focuses on how well the model correctly verifies queries that require verification (A), using the harmonic mean of Precision and Recall: $\text{Precision} = \frac{A}{A+B}$, $\text{Recall} = \frac{A}{A+C}$. This indicates how accurately the model identifies situations in which verification is needed.

F1$_\text{c}$ (Clear Prediction F1) evaluates the accuracy (C) for clear queries while minimizing unnecessary verification (E). It is the harmonic mean of Precision and Recall defined as: $\text{Precision} = \frac{C}{B+C+D}$, $\text{Recall} = \frac{C}{C+D+E}$.

F1$_\text{v}$ (Verification Needed Detection F1) measures how well the model requests verification (A) in queries that actually require it. It is calculated as the harmonic mean of Precision and Recall defined as: $\text{Precision} = \frac{A}{A+E}$, $\text{Recall} = \frac{A}{A+B}$.

4.4 Implementation Details

Hidden State Extraction & Classifier

To extract the hidden states of queries in LQC, we conducted three experiments each on the LLAMA family and the QWEN(Yang et al., 2024) family of LLMs, then selected the model with the highest average performance within each family. As a result, the LLAMA-3.2-1B and QWEN2.5-0.5B models were chosen. For the Llama model, we extracted the hidden state from the 12th layer and set $\alpha = 0.5$ for hybrid pooling, whereas for the Qwen model, we extracted the hidden state from the 18th layer and set $\alpha = 0.25$.

Response Generation Models For the model generating answers based on LQC's classification results, we used **LLAMA-8B-Instruct**, **QWEN2.5-7B-Instruct**, and **QWEN2.5-14B-Instruct**. Each model was tested using two random seeds, and we computed the average results. This setup allows us to comprehensively evaluate both classification accuracy and final answer quality, thereby confirming the model's superior ability to handle ambiguity and provide accurate answers compared to the baselines (, REFLECT).

4.5 Main Results

Overall Performance (Accuracy, F1) Table 1 represents the main results. In each of the four datasets, we measured the four metrics of Accuracy, F1, F1$_\text{c}$, and F1$_\text{v}$. Out of the total of 16 experimental items, LQC$_\text{llama}$ and LQC$_\text{qwen}$ achieved the highest scores on 14 metrics compared to the baselines (INSTANT, REFLECT), showing superiority in most cases other than F1$_\text{c}$ and F1$_\text{v}$ in the False Presuppositions setting. This suggests that the strategy of classifying whether verification is required in advance, then branching to a suitable template before feeding the query into the answer-generating LLM, is effective for reducing incorrect information and minimizing unnecessary verification.

Performance in Identifying Verification-Required Queries (F1$_\text{v}$) Because the LQC models are designed to prevent missing queries requiring verification (suppressing the transition from category A to B), they show a significant improvement in F1$_\text{v}$ compared to INSTANT and REFLECT. Instead of passing queries directly to the LLM, queries classified as "requiring verification" are handled using a verification-focused template. This explicit approach to ambiguity or flaws helps prevent missed verifications and is viewed as a major structural advantage.

Performance in Handling Clear Queries (F1$_\text{c}$) As for clear queries, the improvement in F1$_\text{c}$ indicates that the approach effectively avoids unnecessary verification and yields correct answers promptly (suppressing the transitions from category C to D or E). When LQC classifies a query as clear, it immediately generates an answer using a *simple QA template*, thereby reducing incorrect answers (D) and unnecessary verification (E), which leads to better overall performance. Moreover, there was no significant increase in excessive refusal to answer. Unlike INSTANT or REFLECT—where the model itself must fully decide about ambiguity—LQC **pre-classifies** a question as "clear," making it possible to respond without unnecessary warnings or refusals. Consequently, the system more consistently and efficiently handles clear queries compared to INSTANT and REFLECT.

5 Analysis

In this section, we conduct an in-depth analysis of LQC, a classifier that learns from the hidden states of an LLM in order to determine whether a user's query requires verification. Unless otherwise

Methods	SituatedQA-Geo				SituatedQA-Temp				CLAMBER-Linguistic Ambiguity				CoCoNot-False Presuppositions			
	Acc.	F1	F1$_c$	F1$_v$	Acc.	F1	F1$_c$	F1$_v$	Acc.	F1	F1$_c$	F1$_v$	Acc.	F1	F1$_c$	F1$_v$
LLAMA-3.1-8B-Instruct																
INSTANT	52.56	37.19	51.25	55.04	38.68	35.96	33.96	49.10	43.33	29.46	43.41	43.08	84.25	81.83	62.84	96.33
REFLECT	62.74	51.67	56.32	72.24	41.32	45.21	32.39	58.81	48.33	41.44	43.52	56.87	86.46	81.01	**69.26**	95.93
LQC$_{llama}$	**80.98**	72.33	**67.28**	**94.67**	**52.17**	**57.16**	**39.20**	72.53	**68.75**	**79.30**	**46.87**	**87.60**	**86.46**	84.28	65.01	**97.49**
LQC$_{qwen}$	79.65	**73.07**	66.05	92.68	53.03	60.77	38.29	**74.61**	63.75	78.39	40.73	84.26	83.70	**84.35**	59.92	96.02
QWEN2.5-7B-Instruct																
INSTANT	56.91	36.06	57.76	55.31	44.87	25.31	47.36	38.94	47.92	26.40	51.85	39.74	88.67	78.79	76.05	95.88
REFLECT	56.08	34.79	57.38	53.60	43.36	23.99	46.03	36.90	50.83	27.24	55.63	41.21	90.33	79.72	78.32	96.99
LQC$_{llama}$	**73.28**	55.18	**68.72**	**79.34**	**56.69**	**40.20**	**54.91**	**60.06**	**68.07**	**55.71**	**61.54**	**76.85**	**91.71**	79.37	**81.71**	**97.22**
LQC$_{qwen}$	69.23	**55.56**	62.87	77.82	54.64	36.28	54.64	54.65	59.58	49.43	53.98	68.06	85.28	**81.97**	70.50	93.68
QWEN2.5-14B-Instruct																
INSTANT	63.32	41.77	63.05	63.77	43.55	25.10	45.75	38.28	53.75	28.88	58.86	43.81	86.74	77.26	74.63	93.86
REFLECT	68.32	52.03	64.08	74.32	48.36	36.31	47.37	50.24	54.17	40.00	54.98	52.90	89.50	80.49	75.79	**97.00**
LQC$_{llama}$	**87.47**	70.30	**79.51**	**95.19**	**69.01**	60.56	**59.60**	**81.53**	**77.50**	**71.90**	**66.09**	**88.00**	**90.61**	80.28	80.00	96.20
LQC$_{qwen}$	85.77	**71.42**	78.60	92.25	68.49	**62.32**	58.32	81.40	70.83	68.98	59.59	81.63	90.33	**81.13**	**80.20**	95.43

Table 1: Main results. The shaded portion denotes the QA system incorporating the proposed LQC from this study. Among the four experimental results for each answer generation model, the highest performance is emphasized in **bold**.

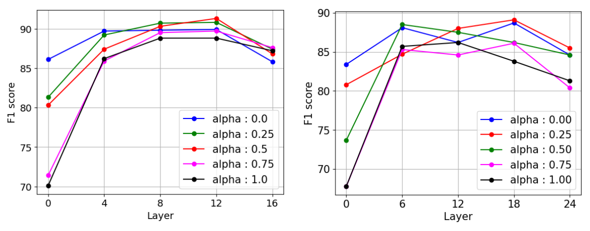
specified, the settings are the same as in the main experiment.

5.1 Ablation Studies

Table 2 compares performance when the query hidden states extracted from the LLM are fed into an MLP classifier, with and without the application of **Contrastive Learning**. According to the experimental results, the model that applies Contrastive Learning consistently achieves higher accuracy and F1 scores than the control group. This is because Verification Needed Query and All Clear Query may exhibit superficially similar word distributions or sentence structures, but it is necessary to correctly understand actual meaning in order to classify them accurately. Through contrastive learning, once the boundaries between classes become distinctly delineated in the embedding space, we confirmed in this experiment that the model can easily distinguish each class during the classifier stage.

5.2 Effectiveness of Using intermediate Layers

In this section, we explore which layer should be used when extracting the query feature vector in LQC to maximize the performance of the verification query classifier, LQC. According to the results presented in Figure 4, the F1 score starts low in the early layers, rises toward the intermediate layers, and then declines again in the final layer. The LLAMA-3.2-1B model achieves its highest score at the 12th layer, while the QWEN2.5-0.5B model attains its peak at the 18th layer. This is because the early layers primarily process the lexical and grammatical information of the input sentence, whereas the intermediate layers more effectively capture contextual and logical relationships. Therefore, the features needed to detect errors such as premise conflicts or linguistic ambiguity inherent in the query appear to be most effectively extracted in the intermediate layers. On the other hand, since

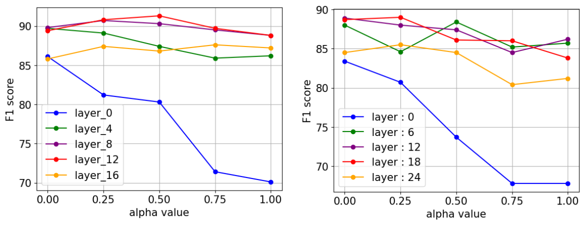


(a) LLAMA-3.2-1B (b) QWEN2.5-0.5B

Figure 4: Comparison of LQC performance by layer. The red graph indicates the setting used in the main experiment.

Model$_{\text{LQC}}$	w/ CL		w/o CL	
	Acc.	F1	Acc.	F1
LLAMA-3.2-1B	**91.3**	**91.3**	89.1	89.7
QWEN2.5-0.5B	**88.7**	**89.1**	85.5	87.2

Table 2: Comparison of LQC performance with or without contrastive learning.



(a) LLAMA-3.2-1B (b) QWEN2.5-0.5B

Figure 5: Comparison of LQC performance by pooling method(α). The red graph indicates the setting used in the main experiment.

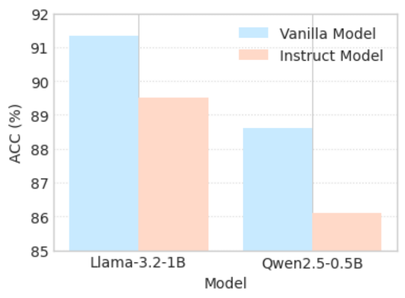
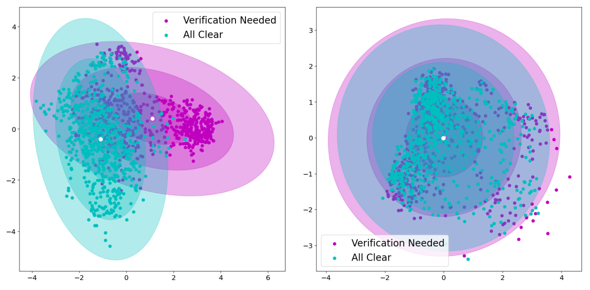


Figure 6: Comparison of LQC performance according to whether Instruction Tuning is applied.



(a) LLAMA-3.2-1B (b) LLAMA-3.2-1B-Instruct

Figure 7: The result of inputting a set of queries requiring verification and normal queries into the LLM, extracting the hidden states, and then performing PCA dimensionality reduction.

the final layer is specialized for language modeling that focuses on generating natural responses, it can be interpreted that the features necessary for response generation overshadow those required for verification. This result is in line with previous research(Tenney et al., 2019) that classifies the stability of large language models (LLMs) by leveraging their internal representations.

5.3 Effectiveness of Hybrid Pooling

In this section, we compare various pooling methods used to extract input sequences for LQC training. We measure the performance of Mean pooling ($\alpha = 0.0$), Last token pooling ($\alpha = 1.0$), and the Hybrid pooling method proposed in this study. As shown in Figure 5, both models exhibit superior performance when Hybrid pooling is applied. This is because Hybrid pooling simultaneously captures the benefits of a global summary effect and the emphasis on key tokens, thus producing richer and more fine-grained embeddings.

5.4 LLMs for Hidden State Extraction

We analyze how the type of LLM used to extract hidden states affects LQC's performance.

Vanilla Model vs. Instruct Tuned Model As shown in Figure 6, using hidden states from a Vanilla model yields higher performance in capturing query defects (e.g., errors, ambiguities) than using an Instruct-tuned model. This can be interpreted as part of the alignment tax (Lin et al., 2024a; Thakkar et al., 2024; Lou et al., 2024), where greater "helpfulness" training diminishes a model's ability to extract defect-related information. Consequently, for identifying query flaws, an untuned model may be more suitable.

Furthermore, as shown in Figure 7, the embedding space of the Vanilla model more distinctly separates Verification needed queries from All clear queries, whereas the Instruct model's embeddings do not clearly distinguish between these two types of queries. This indicates that the hidden states from a Vanilla model can provide a clearer representation for detecting query flaws.

Parameter size of LLM According to the results shown in Figure 8, increasing the number of parameters in the LLM model used to extract the query's hidden states actually leads to lower classification performance. Although larger-parameter models are expected to provide richer hidden states, they can dilute the features needed to identify query defects, causing the embeddings to become overly dispersed. On the other hand, smaller models may not fully capture complex knowledge or reasoning, but they more distinctly capture query defects, often yielding higher binary classification

Type	Responses
Verification Needed Query	Who served the longest term as president?
INSTANT	Franklin D. Roosevelt served the longest term as President of the United States...
REFLECT	Franklin D. Roosevelt served the longest term as President of the United States...
LQC$_{\text{llama}}$	Verification required for this query. The user's question does not specify a country...
LQC$_{\text{qwen}}$	Verification required for this query. The user's request does not specify a particular country...
All Clear Query	Who became the king of england in 1830?
INSTANT	In 1830, King William IV became the King of England.
REFLECT	Victoria became the Queen of England in 1837, not 1830. Verification is necessary because...
LQC$_{\text{llama}}$	King William IV became the King of England in 1830...
LQC$_{\text{qwen}}$	King William IV became the King of England in 1830...

Table 3: An example comparing how INSTANT, REFLECT, LQC$_{\text{llama}}$, and LQC$_{\text{qwen}}$ actually generate responses for Verification Needed Query (results of QWEN2.5-14B-Instruct) and All Clear Query (results of QWEN2.5-7B-Instruct). Because the LQC approach selectively identifies ambiguity in the query and undergoes a verification process only when necessary, it visually demonstrates that it can suppress incorrect information generation while still providing immediate responses to clear queries.

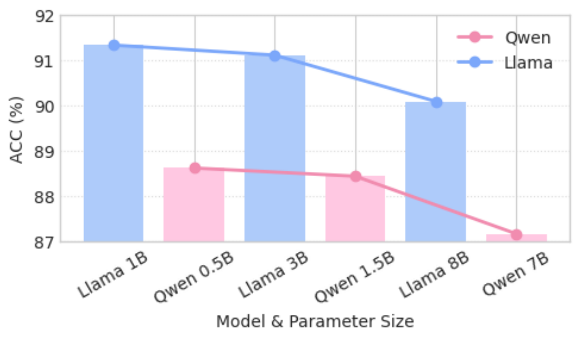


Figure 8: Comparison of LQC performance according to model parameter size.

performance. They also offer lower resource consumption. In this study, we use LLAMA-3.2-1B and QWEN2.5-0.5B models for classification purposes(LQC), confirming that both can be run on a single 24GB RTX3090 GPU without difficulty.

5.5 Qualitative Evaluation

As shown in Table 3, the Verification Needed Query ("who served the longest term as president") lacks essential details (such as a country or time), yet INSTANT and REFLECT both respond immediately without any checks. In contrast, LQCllama and LQCqwen first detect this ambiguity and request verification, thereby avoiding the generation of incorrect information.

The All Clear Query ("Who became the king of england in 1830") explicitly includes relevant information, enabling INSTANT to provide an immediate and accurate answer. REFLECT, however, performs an unnecessary verification step, while LQCllama and LQCqwen respond swiftly and precisely, underscoring their ability to deliver correct answers for well-defined queries.

These findings align with the quantitative evaluation (Section 4.5), indicating that the LQC framework—which selectively applies verification only when needed—confers a clear advantage in actual conversational contexts.

6 Conclusion

In this study, we propose a question-verification model called LQC to prevent hallucination issues in LLMs that can arise from users' incorrect queries, and we present a complete pipeline that integrates LQC into a QA system. LQC extracts the hidden states from intermediate transformer layers using a newly proposed "Hybrid Pooling" technique, then explicitly notifies users of queries that require verification—demonstrating superior performance compared to other baseline. Furthermore, the model can be easily trained by extracting hidden states from a lightweight LLM and employing a simple MLP-based binary classification approach.Above all, our results show that it is possible to clearly identify and address defects in user queries without modifying the LLM's weights or input prompts in any way. This lightweight approach functions as an initial safeguard that complements the input and output stages of LLMs, helping prevent the generation of incorrect information in QA systems and ultimately enhancing user satisfaction.

Limitations

The proposed approach focuses on reducing the frequency of hallucinations by filtering user questions before the LLM generates a hallucinated answer. However, it does not directly control or modify the potential for hallucination inherent in the LLM itself. The question data used in this study consists of relatively simple questions. When applied to a real QA system, the difficulty of determining whether a question needs verification may vary depending on the domain, and there is a possibility of encountering a wider variety of question types. In future research, it will be necessary to obtain a dataset that includes more complex and diverse types and domains of questions to verify the generalizability of the proposed methodology.

References

D.A. Alber, Z. Yang, A. Alyakin, and et al. 2025. Medical large language models are vulnerable to data-poisoning attacks. In *Nature Medicine*.

Faeze Brahman, Sachin Kumar, Vidhisha Balachandran, Pradeep Dasigi, Valentina Pyatkin, Abhilasha Ravichander, Sarah Wiegreffe, Nouha Dziri, Khyathi Chandu, Jack Hessel, Yulia Tsvetkov, Noah A. Smith, Yejin Choi, and Hannaneh Hajishirzi. 2024. The art of saying no: Contextual noncompliance in language models. *Preprint*, arXiv:2407.12043.

Huixin Chen, Jan Büssing, David Rügamer, and Ercong Nie. 2024. Team MGTD4ADL at SemEval-2024 task 8: Leveraging (sentence) transformer models with contrastive learning for identifying machine-generated text. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 1711–1718, Mexico City, Mexico. Association for Computational Linguistics.

Zeming Chen and Qiyue Gao. 2022. Probing linguistic information for logical inference in pre-trained language models. *Preprint*, arXiv:2112.01753.

Jeremy Cole, Michael Zhang, Daniel Gillick, Julian Eisenschlos, Bhuwan Dhingra, and Jacob Eisenstein. 2023. Selectively answering ambiguous questions. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 530–543, Singapore. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bin Fu, Yunqi Qiu, Chengguang Tang, Yang Li, Haiyang Yu, and Jian Sun. 2020. A survey on complex question answering over knowledge base: Recent advances and challenges. *Preprint*, arXiv:2007.13069.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yue Guan, Zhengyi Li, Jingwen Leng, Zhouhan Lin, Minyi Guo, and Yuhao Zhu. 2022. Block-skim: Efficient question answering for transformer. *Preprint*, arXiv:2112.08560.

Linyang He, Peili Chen, Ercong Nie, Yuanning Li, and Jonathan R. Brennan. 2024. Decoding probing: Revealing internal linguistic structures in neural language models using minimal pairs. *Preprint*, arXiv:2403.17299.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Ziwei Ji, Delong Chen, Etsuko Ishii, Samuel Cahyawijaya, Yejin Bang, Bryan Wilie, and Pascale Fung. 2024. LLM internal states reveal hallucination risk faced with a query. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 88–104, Miami, Florida, US. Association for Computational Linguistics.

Siduo Jiang, Cristopher Benge, and William Casey King. 2022. Bertvision – a parameter-efficient approach for question answering. *Preprint*, arXiv:2202.12210.

Jing Jin, Houfeng Wang, Hao Zhang, Xiaoguang Li, and Zhijiang Guo. 2024. DVD: Dynamic contrastive decoding for knowledge amplification in multi-document question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4624–4637, Miami, Florida, USA. Association for Computational Linguistics.

Shahar Katz and Yonatan Belinkov. 2023. Visit: Visualizing and interpreting the semantic information flow of transformers. *Preprint*, arXiv:2305.13417.

Hyuhng Joon Kim, Youna Kim, Cheonbok Park, Junyeob Kim, Choonghyun Park, Kang Min Yoo, Sang-goo Lee, and Taeuk Kim. 2024. Aligning language models to explicitly handle ambiguity. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1989–2007, Miami, Florida, USA. Association for Computational Linguistics.

Daoyang Li, Haiyan Zhao, Qingcheng Zeng, and Mengnan Du. 2025. Exploring multilingual probing in large language models: A cross-language analysis. *Preprint*, arXiv:2409.14459.

Yong Lin, Hangyu Lin, Wei Xiong, Shizhe Diao, Jianmeng Liu, Jipeng Zhang, Rui Pan, Haoxiang Wang, Wenbin Hu, Hanning Zhang, Hanze Dong, Renjie Pi, Han Zhao, Nan Jiang, Heng Ji, Yuan Yao, and Tong Zhang. 2024a. Mitigating the alignment tax of rlhf. *Preprint*, arXiv:2309.06256.

Yuping Lin, Pengfei He, Han Xu, Yue Xing, Makoto Yamada, Hui Liu, and Jiliang Tang. 2024b. Towards understanding jailbreak attacks in LLMs: A representation space analysis. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7067–7085, Miami, Florida, USA. Association for Computational Linguistics.

Renze Lou, Kai Zhang, and Wenpeng Yin. 2024. Large language model instruction following: A survey of progresses and challenges. *Preprint*, arXiv:2303.10475.

Lukas Muttenthaler, Isabelle Augenstein, and Johannes Bjerva. 2020. Unsupervised evaluation for question answering with transformers. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 83–90, Online. Association for Computational Linguistics.

Matanel Oren, Michael Hassid, Nir Yarden, Yossi Adi, and Roy Schwartz. 2024. Transformers are multi-state RNNs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 18724–18741, Miami, Florida, USA. Association for Computational Linguistics.

Xuan Ren, Biao Wu, and Lingqiao Liu. 2024. I learn better if you speak my language: Understanding the superior performance of fine-tuning large language models with LLM-generated responses. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10225–10245, Miami, Florida, USA. Association for Computational Linguistics.

Lena Schmidt, Julie Weeds, and Julian P. T. Higgins. 2020. Data mining in clinical trial text: Transformers for classification and question answering tasks. *Preprint*, arXiv:2001.11268.

Aviv Slobodkin, Omer Goldman, Avi Caciularu, Ido Dagan, and Shauli Ravfogel. 2023. The curious case of hallucinatory (un)answerability: Finding truths in the hidden states of over-confident large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3607–3625, Singapore. Association for Computational Linguistics.

Mark Stevenson and Yikun Guo. 2010. Disambiguation in the biomedical domain: The role of ambiguity type. *Journal of Biomedical Informatics*, 43(6):972–981.

Md Mehrab Tanjim, Xiang Chen, Victor S. Bursztyn, Uttaran Bhattacharya, Tung Mai, Vaishnavi Muppala, Akash Maharaj, Saayan Mitra, Eunyee Koh, Yunyao Li, and Ken Russell. 2025. Detecting ambiguities to guide query rewrite for robust conversations in enterprise ai assistants. *Preprint*, arXiv:2502.00537.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Megh Thakkar, Quentin Fournier, Matthew D Riemer, Pin-Yu Chen, Amal Zouaq, Payel Das, and Sarath Chandar. 2024. A deep dive into the trade-offs of parameter-efficient preference alignment techniques. *Preprint*, arXiv:2406.04879.

Nandan Thakur, Luiz Bonifacio, Crystina Zhang, Odunayo Ogundepo, Ehsan Kamalloo, David Alfonso-Hermelo, Xiaoguang Li, Qun Liu, Boxing Chen, Mehdi Rezagholizadeh, and Jimmy Lin. 2024. “knowing when you don‘t know”: A multilingual relevance assessment dataset for robust retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 12508–12526, Miami, Florida, USA. Association for Computational Linguistics.

Praneeth Vadlapati. 2023. Investigating the impact of linguistic errors of prompts on llm accuracy. *ESP Journal of Engineering Technology Advancements*, 3:150–153.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need. *Preprint*, arXiv:1706.03762.

Bo Wang and Tsunenori Mine. 2024. One stone, four birds: A comprehensive solution for qa system using supervised contrastive learning. *Preprint*, arXiv:2407.09011.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. 2024. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

Zhenrui Yue, Bernhard Kratzwald, and Stefan Feuerriegel. 2021. Contrastive domain adaptation for question answering using limited text corpora. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9575–9593, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michael J. Q. Zhang and Eunsol Choi. 2021. Situatedqa: Incorporating extra-linguistic contexts into qa. *Preprint*, arXiv:2109.06157.

Shaolei Zhang, Tian Yu, and Yang Feng. 2024a. Truthx: Alleviating hallucinations by editing large language models in truthful space. *Preprint*, arXiv:2402.17811.

Tong Zhang, Peixin Qin, Yang Deng, Chen Huang, Wenqiang Lei, Junhong Liu, Dingnan Jin, Hongru Liang, and Tat-Seng Chua. 2024b. CLAMBER: A benchmark of identifying and clarifying ambiguous information needs in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10746–10766, Bangkok, Thailand. Association for Computational Linguistics.

A Dataset Details

In this section, we provide a detailed overview of the datasets used to train LQC. We outline the dataset categories and offer example sentences—namely, those requiring verification and those deemed clear. For more information, please refer to Table 4 and Table 5.

B Baseline Details

This section describes the implementation details of the baseline.

INSTANT Using the template provided below, we perform inference directly without any additional instructions. In doing so, the generation temperature is set to 0.5, and the generated results are used for evaluation.

```
Respond to the following query in one sentence:
Query: {query}
Response:
```

REFLECT Using the template presented below, the model first determines whether verification is required for a given query and then provides an answer. In this process as well, the generation temperature is set to 0.5, and the generated results are used for evaluation.

```
The following user query requires verification.
Analyze the query and explain in one sentence
why further verification is necessary.
Begin your response with "Verification required
for this query.
Query: {query}
Response:
```

C Prompt for Evaluation

In this section, we present the prompt used to classify the answers generated by the answer-generation model into the five categories shown in Figure 3 for metric calculation.

Verification needed query evaluation prompt
You need to classify the responses of an LLM into two categories.
Query: {query}
Response: {response}
Review the query and response. Classify the response into one of the two categories based on the following guidelines:
If the response mentions that the query is unanswerable due to a wrong premise, contextual incompleteness, linguistic ambiguity, known unknowns, or any other reason, output '1'. Otherwise, output '2'. Only output the corresponding number based on the guidelines.
Classification result:

All clear query evaluation prompt
You need to classify the responses of an LLM into two categories.
Query: {query}
Response: {response}
Review the query and response. Classify the response into one of the two categories based on the following guidelines:
If the response mentions that the query is unanswerable due to a wrong premise, contextual incompleteness, linguistic ambiguity, known unknowns, or any other reason, output '1'. Otherwise, output '2'. Only output the corresponding number based on the guidelines.
Classification result:

D Classifier Results

We compare the performance of LLAMA-3.2-1B and QWEN2.5-0.5B, which are primarily used for LQC implementation. For each model, we present comparison tables for different α values, focusing on the layers from which hidden states are extracted and the pooling methods. The performance comparison of LLAMA-3.2-1B is given in Table 6, and that of QWEN2.5-0.5B is presented in Table 7.

Category	Examples	
	Verification Needed	All Clear
Contextual Incompletenss	when was the decimal currency system introduced	when was the decimal currency system introduced in india
Linguistic Ambiguity	What are some popular spirits at the bar?	Who is the artist that performed the song "Doomed" in the context of the album "That's the Spirit"?
Unanswerable	What does the pancreas do in the urinary system?	What does the urethra do in the urinary system?

Table 4: Details of datset

Category	Definition	Source		Size	
				Verification Needed	All Clear
Contextual Incompletenss	requests that lacks crucial situational or background information needed to provide a definitive answer.	SituatedQA	Geo	3,234	3,423
			Temp	2004	2297
Linguistic Amibuity	requests that can be interpreted in multiple ways due to imprecise or unclear wording or sentence structure.	CLAMBER		340	340
Unanswerable	Requests that either contain a false or unverifiable premise, or refer to concepts or knowledge that cannot be answered because it is universally unknown.	CoCoNot	False Presuppositions	680	340
			Universal Unknowns	313	0

Table 5: Definition, source and size of datasets.

		Alpha									
		0		0.25		0.5		0.75		1	
Layer	0	85.21	86.2	82.03	81.27	78.72	80.32	69.91	71.43	67.5	70.09
	4	89.42	89.78	88.99	89.17	87.06	87.41	85.51	85.93	85.94	86.17
	8	89.81	89.89	90.93	90.74	90.41	90.33	89.51	89.51	88.61	88.83
	12	89.42	89.98	90.84	90.75	**91.27**	**91.32**	89.77	89.72	88.52	88.8
	16	84.74	85.81	86.24	87.41	86.46	86.77	87.4	87.58	87.32	87.18

Table 6: Performance comparison table of LQC for LLAMA-3.2-1B with different α values and hidden-state extraction layers

		Alpha									
		0		0.25		0.5		0.75		1	
Layer	0	82.72	83.44	81.73	80.76	76.7	73.74	51.33	67.84	51.33	67.84
	6	87.83	88.05	85.81	84.68	88.48	88.48	84.78	85.25	86.16	85.71
	12	88.74	88.99	88.39	88	86.93	87.46	85.55	84.57	86.46	86.23
	18	87.83	88.72	**88.69**	**89.06**	86.8	86.15	85.6	86.06	84.18	83.83
	24	83.83	84.56	84.78	85.52	84.09	84.56	81.47	80.4	82.29	81.29

Table 7: Performance comparison table of LQC for QWEN2.5-0.5B with different α values and hidden-state extraction layers