On the Scoring Functions for RAG-based Conformal Factuality

Yi Chen [1] **Caitlyn Heqi Yin** [1] **Sukrut Chikodikar** [1] **Ramya Korlakai Vinayak** [1]

Abstract

Large language models (LLMs) frequently generates hallucinated or non-factual outputs. To mitigate this, conformal factuality frameworks utilize scoring functions to filter model-generated claims and provide statistical factuality guarantees. This study systematically evaluates three distinct scoring functions within a retrieval-augmented generation (RAG) context: non-reference model confidence, reference model confidence, and entailment scores. Performance is assessed using empirical factuality, power, and false positive rates. We further investigate the robustness of these scoring functions when the assumption of data exchangeability is mildly violated by introducing deliberately hallucinated claims into the evaluation set. Our findings indicate that reference model confidence scores generally outperform other methods by achieving superior power and robustness. Notably, entailment-based scoring shows the lowest false positive rates under conditions of induced hallucinations. This work highlights the critical importance of scoring function selection in optimizing factuality and robustness for RAG-based conformal frameworks.

1. Introduction

Although large language models (LLMs) excel at tasks like summarization, chatbotting, and even coding (Achiam et al., 2023; Zhang et al., 2024; Nam et al., 2024), they may still generate non-factual content and produce hallucinations (Nadeau et al., 2024; Huang et al., 2025). To alleviate hallucinations, various methods are proposed. One type of method utilizes conformal predictions to remove the non-factual content of the output of an LLM (Vovk et al., 2005; Angelopoulos & Bates, 2021; Mohri & Hashimoto, 2024; Cherian et al., 2024). Besides making the output factual by removing the non-factual content, these methods also provide a statistical guarantee on the factuality of the output. Such a guarantee can be summarized as

$$\mathbb{P}(\text{The output of the LLM is factual}) \geq 1 - \alpha.$$

For example, in (Mohri & Hashimoto, 2024), the authors proposed a method named *conformal factuality*, which filters out the claims made by the LLM whose score is below a certain threshold. How the threshold is set is determined by a user-specified factuality value α and the distribution of scores in a calibration dataset. Therefore, the function that scores these claims plays a vital role in the conformal factuality pipeline.

In this work, we investigate how the scoring function affects the outcome of conformal factuality under a slightly different setting where a reference text, obtained using an RAG system, and a query are given to an LLM. We compare the performance of different scoring functions and also evaluate their robustness when the assumption of conformal factuality is violated.

Our contributions: We evaluate the ***performance*** and ***robustness*** of various scoring functions designed to assess claims generated by a LLM. Our evaluation is situated within a RAG framework, where the LLM conditions its responses on a user query and related reference text provided by the RAG system; this reference text is also utilized by the applicable scoring functions.

2. Preliminaries

Let $\mathcal{S}$ denote the set of all possible sentences. Let $x \in \mathcal{X} \subset \mathcal{S}$ denote a query. For example, the following sentence can be considered as a query:

> Tell me a paragraph bio of Fernando.

Let $r \in \mathcal{R} \subset 2^{\mathcal{S}}$ denote a reference text obtained by plugging some query x into a function called $RAG : \mathcal{X} \to \mathcal{R}$. For example, the following paragraph is considered a reference text to the query above:

> Fernando was born in Alto Paraíso de Goiás. In June 2007, he signed a five-year contract with

[1]University of Wisconsin-Madison, Madison, Wisconsin, USA. Correspondence to: Yi Chen <yi.chen@wisc.edu>.

Published at ICML 2025 Workshop on Reliable and Responsible Foundation Models.

> Porto directly from the Série C, having started his career at Vila Nova.

We use a large language model, denoted as $LLM : \mathcal{X} \times \mathcal{R} \to 2^{\mathcal{S}}$, to generate a response to a query x, given a reference text $RAG(x)$. Usually, a response is composed of various sentences. We assume that the LLM generates a set of sentences. Therefore, the range of this function is the power set of all possible sentences. We will call each sentence generated by the LLM a claim.

Let $f : \mathcal{S} \times \mathcal{X} \times \mathcal{R} \to \mathbb{R}$ denote a scoring function that will evaluate the factuality of a claim given the corresponding query and the reference text of the query. The factuality score will be in the range of real numbers, but usually is within the range of $[0, 1]$ or $\{-1, 0, +1\}$.

2.1. Conformal Factuality

The conformal factuality system requires an input of a calibration dataset $\{x_i\}_{i=1}^n$ together with the following information:

Each query x_i has a corresponding $r_i := RAG(x_i)$. For each x_i, r_i, we obtain a set of claims (sentences) $C^i := \{c_j^i\}_{j=1}^{m_i}$ using a LLM. Each claim c_j^i corresponds with a ground truth a_j^i that indicates if the claim is factual or not:

$$a_j^i = \begin{cases} 1 & \text{if } c_j^i \text{ is factual,} \\ 0 & \text{if } c_j^i \text{ is non-factual.} \end{cases}$$

We use A^i to denote the set of a_j^i's corresponding to claims in C^i.

The conformal factuality system finds a threshold τ_α using the calibration dataset so that for a new query x_{n+1} and its corresponding set of claims C^{n+1}, the set of filtered claims with scores above the threshold is all factually correct with high probability.

More specifically, let $F(C^{n+1}, \tau_\alpha)$ denote the set of claims whose score is greater than the threshold. That is,

$$F(C^{n+1}, \tau_\alpha) := \{c_j^{n+1} \mid f(c_j^{n+1}, x_{n+1}, r_{n+1}) \geq \tau_\alpha\}.$$

Note that this set is a subset of C^{n+1}. The statistical guarantee provided by the conformal factuality framework is

$$\mathbb{P}(\forall c_j^{n+1} \in F(C^{n+1}, \tau_\alpha) \mid a_j^{n+1} = 1) \geq 1 - \alpha,$$

which makes sure that on average, at least $1 - \alpha$ (where α is significance level can be selected manually) fraction of the responses after the conformal factuality pipeline is entirely factual.

Now, we will discuss how the threshold τ_α is chosen. First, for each query x_i in the calibration dataset, we compute a conformity score, which is the smallest threshold that leads to a fully factually correct filtered response:

$$\text{conformity}_i := \inf\{\tau : \forall c_j^i \in F(C^i, \tau), a_j^i = 1\}.$$

The threshold τ_α is then chosen at the $\frac{\lceil (n+1)(1-\alpha) \rceil}{n}$-quantile of the conformity scores $\{\text{conformity}_i\}_{i=1}^n$. Intuitively, the method computes the *smallest safe threshold ensuring all claims above it are factual* with probability at least $1 - \alpha$ over all the queries. This guarantee holds when the new query is ***exchangeable*** with the queries in the calibration dataset.

3. Methodology

To assess the performance and robustness of different scoring functions f, this section first outlines the specific functions being evaluated. It then details the performance metrics adopted for this study, and finally, introduces the experimental setup for evaluating the robustness of these scoring functions.

3.1. Scoring functions

Here are the list of scoring functions that we are going to evaluate:

1. **Non-reference model confidence score.** We ask the LLM to give a factuality score between 0 and 1, inclusive, given the claim and the query. The prompt we used will be presented in Appendix A.1.
2. **Reference model confidence score.** We ask the LLM to give a factuality score between 0 and 1, inclusive, given the claim, the query, and the reference text based on the query. The prompt we used will be presented in Appendix A.2.
3. **Entailment score.** We use an entailment model (Laurer et al., 2022) trained on the DocNLI dataset (Yin et al., 2021), which gives a probability of whether the reference text entails the claim. We use the probability as the entailment score.

3.2. Performance Metrics

1. **Empirical Factuality.** Observed factuality, corresponding to the target factuality $1 - \alpha$.
2. **Power.** Proportion of factual claims retained correctly as factual claims in the filtered responses (equivalent to true positive rate (TPR).
3. **False Positive Rate (FPR).** Proportion of nonfactual claims incorrectly accepted as factual claims in the filtered responses.

3.3. Robustness

A major assumption of the conformal factuality framework is that the test datapoint, i.e., x_{n+1}, is exchangeable with the queries in the calibration dataset. However, in a real-world scenario where we deploy the model, it is possible that this assumption is mildly violated.

To test how robust these scoring functions are under this scenario, for each query x_i, the corresponding reference text $r_i = RAG(x_i)$, and the set of claims C^i, we ask the LLM to modify each $c_j^i \in C^i$, given x_i and r_i such that it would become something that the model would hallucinate. We defer the prompts we used to generate these hallucination claims in Appendix A.5.

Our goal in creating these hallucination claims is that we want these claims to confuse the model so that the model would think that these claims are actually generated by them. To achieve this goal, after we generate a hallucinated claim, we ask the LLM to check if it thinks that the claim might be generated or hallucinated by itself (prompt in Appendix A.6), given the same x_i, r_i. If the hallucination claim can cause the model to think that it is the one who generates it, given x_i and r_i, then we keep this hallucination claim. Otherwise, we repeat this process and generate a new hallucination claim.

These hallucinations would of course have a label $a_j^i = 0$, meaning that they are all false. We ask the three scoring functions to score each of the hallucinated claims. We obtain the threshold on a calibration dataset without any hallucination claims and test the performance of these scoring functions on the test set that mixes with these hallucination claims.

4. Experimental Setup

4.1. Datasets

Following (Mohri & Hashimoto, 2024), we use the FActScore dataset (Min et al., 2023), which consists of 50 queries that ask an LLM to generate a paragraph biography of people that are not very well-known (therefore, their information may not be stored in the parametric knowledge of the LLM). Conveniently, these people in the FActScore dataset have their own Wikipedia page. We use the Python package `wikipediaapi` [1] to obtain the Wikipedia page content of these people and use the text therein as $RAG(\cdot)$.

4.2. Models

The LLM model we target in this work is `gemini-flash-1.5` (Team et al., 2023). Note that after the LLM generates a paragraph, instead of directly splitting the paragraph into sentences, we use the same LLM to parse the paragraph into claims, following the prior work (Appendix A.3) (Mohri & Hashimoto, 2024).

4.3. Ground Truth Labels

Instead of using a human annotator to label whether a claim is factual or not, we prompt the same LLM so that it will evaluate if a claim is factually true or not, given a reference text. The prompt we used is deferred to Appendix A.4.

5. Results and Discussions

5.1. Distribution of Scores

We begin by examining the distributions of scores produced by the evaluated functions. Ideally, factually true claims should receive higher scores, while factually false claims should receive lower scores. Figure 1 illustrates the performance of each scoring function in this regard. Notably, for scoring functions that operate without access to the reference text (e.g., non-reference model confidence), approximately half of the factually true claims are assigned a score of 0, indicating poor discriminative ability in this scenario. Conversely, when scoring functions use the reference text (e.g., reference model confidence and entailment-based models), a significant improvement is observed, with the majority of true claims achieving high scores. It is important to note that the dataset of LLM-generated claims is mainly factual (840 true out of 862 claims), which can be attributed to RAG.

5.2. Performance of the Scoring Functions

We vary the significance level α across the set $\{0.25, 0.2, 0.15, 0.1, 0.05\}$. To ensure robust statistical analysis, error bars for our results are established by conducting 1000 trials for each experimental configuration. Figure 2 illustrates the empirical factuality and its upper and lower bounds derived in (Mohri & Hashimoto, 2024). Notably, when test data are exchangeable with calibration data (a key assumption for conformal methods) the empirical factuality for all three evaluated scoring functions is consistently well-contained within these theoretical limits.

Figure 3 shows the power and the false positive rate (FPR) of the three scoring functions. All three scoring functions lead to a similar false positive rate. However, on the power side, the reference model confidence score has the highest power. Then, it is the entailment score. This indicates that using the reference text with the LLM yields a more effective scoring function than using the entailment model.

[1] https://pypi.org/project/Wikipedia-API/

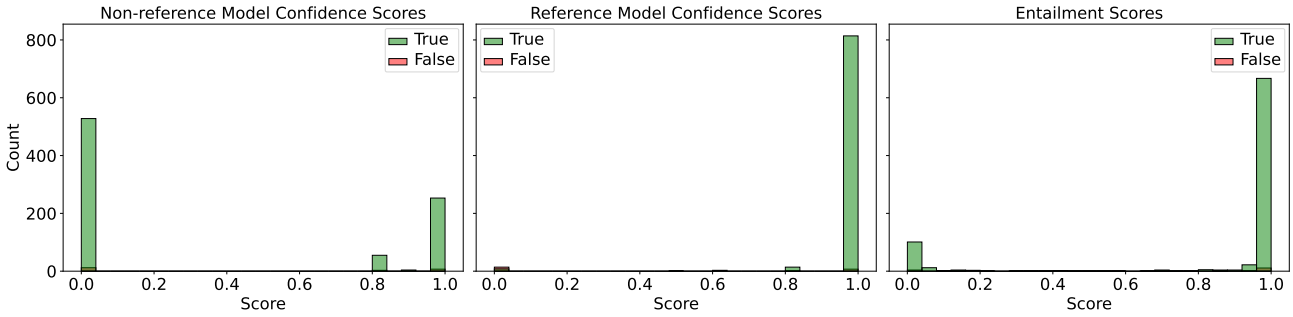

Figure 1. Distribution of the scores of each scoring function.

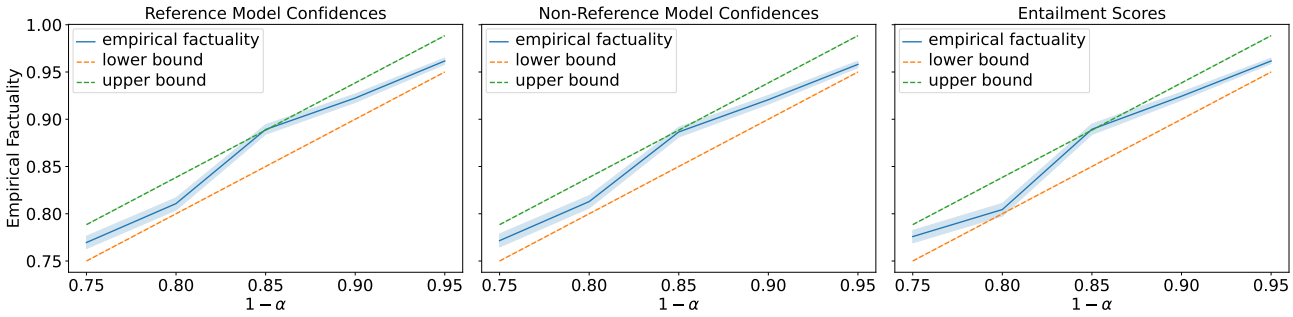

Figure 2. Empirical factuality of each scoring function.

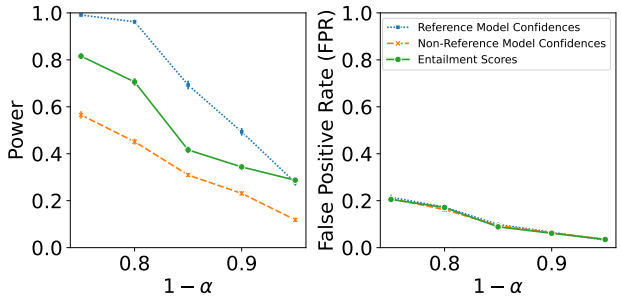

Figure 3. Power and false positive rate (FPR) of each scoring function.

5.3. Robustness of the Scoring Functions

To test the robustness of the scoring function. We replace 10 randomly selected claims for each query in the test set with the corresponding hallucination claims. Figure 4 augments Figure 1 with the hallucination claims. The reference model confidence score performs the best, as most of the false claims (including the hallucinations, as they are factually false) receive a lower score. Entailment scores perform slightly worse. We notice that when the model is not given any reference, it also scores many factually true claims with a lower score. This might be because when the model's parametric knowledge is not enough to score a claim, it tends to be conservative and gives a lower score for all.

Figure 5 shows the empirical factuality of the three scoring functions when hallucination claims are presented in the test set. That is, when the exchangeable assumption is violated. The empirical factuality for all three scoring functions falls outside the theoretical bounds. Moreover, another interesting observation is that as $1 - \alpha$ increases, the empirical factuality tends to get closer to the lower bound of empirical factuality.

Figure 6 illustrates the power and FPR of the three scoring functions when hallucination claims are presented in the test set. Similar to the case when there are no hallucination claims, reference model confidence scores yield the highest power. The difference appears in the FPR plot, where we see that entailment scores yield the lowest false positive rate.

6. Conclusion and Future Work

In this study, we systematically evaluated three types of scoring functions—non-reference model confidence, reference model confidence, and entailment scores—for their efficacy in conformal factuality frameworks within a RAG context. Our results indicate that the reference model confidence score offers the highest effectiveness in terms of empirical factuality and power. However, entailment scores exhibit greater robustness in maintaining low false positive rates when challenged with non-exchangeability and deliberately

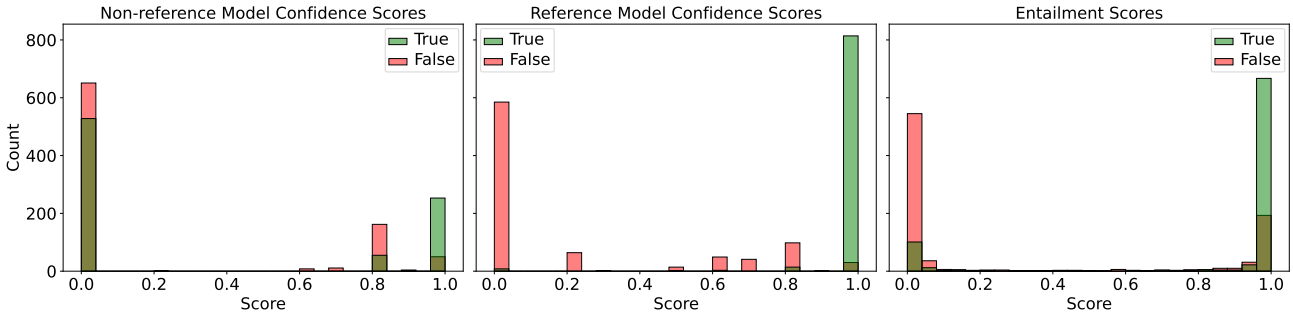

Figure 4. Distribution of the scores of each scoring function, including the hallucination claims.

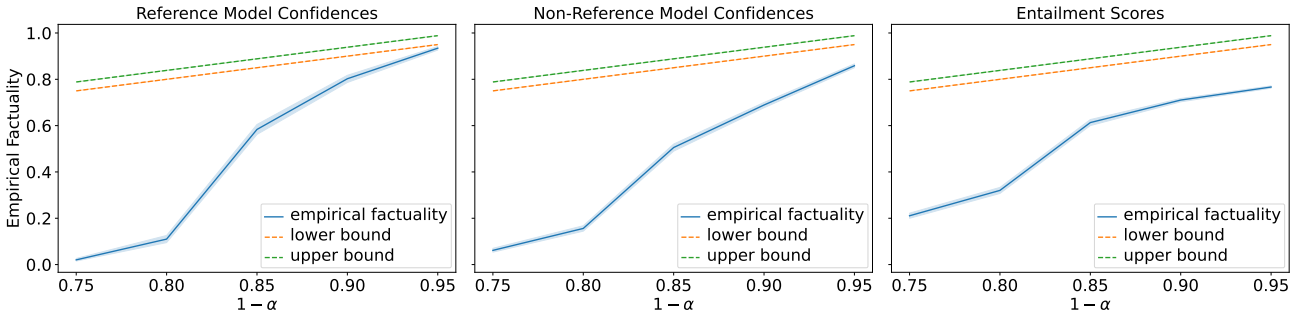

Figure 5. Empirical factuality of each scoring function when hallucination claims are presented in the test set.

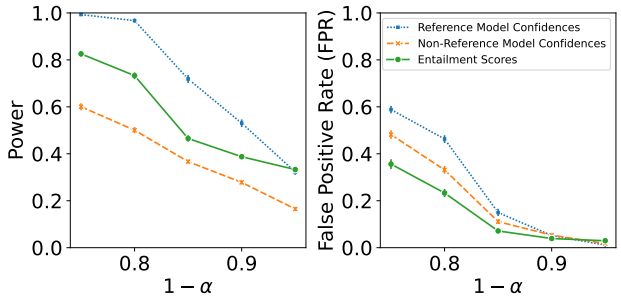

Figure 6. Power and false positive rate (FPR) of each scoring function when hallucination claims are presented in the test set.

hallucinated claims, highlighting a trade-off between overall performance and robustness to specific types of factual errors. The choice of scoring function is therefore a critical design decision, depending on the specific requirements for factuality and robustness in RAG applications. Future work could explore hybrid scoring functions that combine the strengths of different approaches or adaptive methods that adjust scoring strategies based on the nature of the query or the generated content.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Angelopoulos, A. N. and Bates, S. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv preprint arXiv:2107.07511*, 2021.

Cherian, J., Gibbs, I., and Candes, E. Large language model validity via enhanced conformal prediction methods. *Advances in Neural Information Processing Systems*, 37: 114812–114842, 2024.

Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B., et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55, 2025.

Laurer, M., van Atteveldt, W., Salleras Casas, A., and Welbers, K. Less annotating, more classifying – addressing the data scarcity issue of supervised machine learning with deep transfer learning and BERT-NLI, June 2022. URL https://osf.io/74b8k. Preprint, Open Science Framework.

Min, S., Krishna, K., Lyu, X., Lewis, M., Yih, W.-t., Koh, P. W., Iyyer, M., Zettlemoyer, L., and Hajishirzi, H. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *arXiv preprint arXiv:2305.14251*, 2023.

Mohri, C. and Hashimoto, T. Language models with conformal factuality guarantees. *arXiv preprint arXiv:2402.10978*, 2024.

Nadeau, D., Kroutikov, M., McNeil, K., and Baribeau, S. Benchmarking llama2, mistral, gemma and gpt for factuality, toxicity, bias and propensity for hallucinations. *arXiv preprint arXiv:2404.09785*, 2024.

Nam, D., Macvean, A., Hellendoorn, V., Vasilescu, B., and Myers, B. Using an llm to help with code understanding. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, pp. 1–13, 2024.

Team, G., Anil, R., Borgeaud, S., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., Millican, K., et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Vovk, V., Gammerman, A., and Shafer, G. *Algorithmic learning in a random world*, volume 29. Springer, 2005.

Yin, W., Radev, D., and Xiong, C. Docnli: A large-scale dataset for document-level natural language inference. *arXiv preprint arXiv:2106.09449*, 2021.

Zhang, Y., Jin, H., Meng, D., Wang, J., and Tan, J. A comprehensive survey on process-oriented automatic text summarization with exploration of llm-based methods. *arXiv preprint arXiv:2403.02901*, 2024.

A. Prmopts

A.1. Non-reference Model Confidence Score

```
You are an AI assistant tasked with assigning a confidence score to a claim based
    on its factuality.
Instructions:
1. You are given a query and a claim made for that query.
2. Rate the factuality of the claim as a score (a float between 0 and 1), where a
    score of 0.0 means that the claim is false, and a score of 1.0 means that
    the claim is factual.

Output Requirements:
1. Return a single **float** between 0 and 1.
2. Do **not** include any explanations, comments, or additional text only the
    score.
3. Box the final score using the angle brackets <>.

Example Input:

Query: Tell me a paragraph biography about Donald Trump.
Claim: Donald Trump is the 45th \& 47th President of the United States.

Example Output:
<1.0>

Input:\n
```

A.2. Reference Model Confidence Score

```
You are an AI assistant tasked with assigning a confidence score to a claim based
    on its factuality.
Instructions:
1. You are given a query and a claim made for that query.
2. You are given a reference text for that query.
3. Rate the factuality of the claim as a score (a float between 0 and 1), where a
    score of 0.0 means that the claim is false or contradicts the reference text
    , and a score of 1.0 means that the claim is factual and well-supported by
    the reference text.

Output Requirements:
1. Return a single **float** between 0 and 1.
2. Do **not** include any explanations, comments, or additional text only the
    score.
3. Box the final score using the angle brackets <>.

Example Input:

Reference Text: Jeremy is a software engineer. He works at a tech company.
Query: Tell me a paragraph biography about Jeremy.
Claim: Jeremy is a software engineer.

Example Output:
<1.0>
```

```
Input:\n
```

A.3. Parsing Paragraph into Claims

```
You are an AI assistant tasked with breaking down input text into small, self-
    contained claims for easy human verification.

Instructions:
1. Parse the provided text into concise, independent, and non-overlapping
    subclaims.
2. Ensure each subclaim is:
  - As small and specific as possible.
  - Independent and self-contained (do not use pronouns or ambiguous references;
      explicitly mention subjects).
  - Factually complete without relying on context from other subclaims.

Output Requirements:
1. The result must be a VALID and COMPLETE JSON list of dictionaries.
2. Each dictionary must have the following structure:

  {
      "subclaim": "Subclaim text"
  }

Example Input:

Jeremy is a software engineer. He works at a tech company.

Example Output:

[
    {
        "subclaim": "Jeremy is a software engineer."
    },
    {
        "subclaim": "Jeremy works at a tech company."
    },
    ...
]

JSON Rules:
- Ensure the JSON is STRICTLY VALID:
  - Use DOUBLE QUOTES ("") for all keys and string values.
  - DO NOT include trailing commas after the LAST item in arrays or objects.
  - Ensure ALL dictionaries are enclosed in curly braces {}.
  - Ensure the JSON list is ENCLOSED in square brackets [].
  - CLOSE the JSON list properly with a closing square bracket ].
- DO NOT include any code block delimiters (e.g., ` ```json `).
- DO NOT include any additional text, explanations, or comments in the
    output return ONLY the JSON list.

Critical Requirement:
- Validate and ensure the JSON is complete, properly CLOSED, and parseable. The
    final JSON MUST pass validation without errors.
```

```
Input:\n
```

A.4. Ground Truth Labels

```
You are a helpful AI assistant tasked with annotating a claim made for a query:

Tasks:
1. For a given query, provide your full chain-of-thought reasoning, showing all
    steps and calculations needed.
2. For a given claim, determine if each claim is correct ("True") or false ("
    False").
3. Incorporate the provided solution into your reasoning and verify its
    correctness.
4. At the very end of your response, on a new line, output the final answer
    wrapped by brackets <>. This line must start with the marker "Final Answer:"
    followed immediately by the answer.
4. Do not include any extra text with the final answer, so that it can be easily
    extracted.
5. Ensure that the chain-of-thought reasoning is clearly separated from the final
     answer.
6. If the reference text clearly indicates the claim is true or false, you can
    use that information to make your decision.

Example Input:

Reference Text: Tony has 3 cookies. Lily has 5 cookies.
Query: How many cookies do Tony and Lily have in total?
Claim: Tony has 3 cookies.

Example Output:

[Your detailed chain-of-thought reasoning...]
Final Answer: <True>

Example Input:
Reference Text: Jeremy was a software engineer. He worked at a tech company. He
    was 81 years old when he passed away.
Query: Tell me a paragraph biography about Jeremy.
Claim: Jeremy passed away at the age of 82.

Example Output:

[Your detailed chain-of-thought reasoning...]
Final Answer: <False>

Input:\n
```

A.5. Generating Hallucination Claims

```
Suppose you are a model that hallucinates easily. You are given a reference text,
     a question, and a claim that answers the question based on the reference
    text. Your task is to modify and hallucinate the given claim based on the
    reference text. This means that the claim you generate should be a false
```

```
    claim, but it should be confusing enough that you would think it is true
    given the reference text and the question. You will also be given a list of
    modified claims that do not work, i.e. they do not confuse another AI model.
    Do not repeat these claims. You will also be given a list of modified claims
    that do work, i.e. they confuse another AI model. You should not repeat these
     claims either.

Example Input:
Modified claims that do not work: []
Modified claims that do work: []
Reference: Jim is a fictional character in the TV show The Office. He works as a
    salesman at Dunder Mifflin's Scranton branch. He is known for his pranks on
    his deskmate Dwight Schrute.
Question: Tell me a paragraph bio of Jim.
Claim: Jim is the deskmate of Dwight Schrute.

Example Output:
Chain-of-thought reasoning:
[Your detailed chain-of-thought reasoning...]
Final Answer:
{"subclaim": "Jim is the supervisor of Dwight Schrute."}

Again, make sure that the claim you generate is false, but it should be confusing
     enough that another AI model would think it is true given the reference text
     and the question. Also, make sure the final snwer is wrapped in a JSON
    object with the key "subclaim". The final answer should be on a new line and
    start with "Final Answer:" followed by the JSON object. Do not repeat the
    claims that do not work or the claims that do work. Do not generate a claim
    that is simliar or related to the claims that do not work or the claims that
    do work. The claim should be a single sentence.

Input:\n
```

A.6. Hallucination Verification

```
You are a helpful AI assistant tasked with determining if a claim sounds like
    something you would say and hallucinate based on the provided reference text
    and question. Wrap your final answer in brackets <>. If the claim sounds like
     something you would say return <True>. Otherwise, return <False>.

Example Input:
Reference Text: Jim is a fictional character in the TV show The Office. He works
    as a salesman at Dunder Mifflin's Scranton branch. He is known for his pranks
     on his deskmate Dwight Schrute.
Question: Tell me a paragraph bio of Jim.
Claim: Jim is the supervisor of Dwight Schrute.

Example Output:
Chain-of-thought reasoning:
Jim is able to play pranks on Dwight. It seems like Jim can get over with it.
    Therefore, it is possible that Jim is the supervisor of Dwight Schrute. I
    could halluciante this claim based on the reference text and the question.
Final Answer: <True>
```

```
Input:\n
```