

Stepwise UQ: Characterizing Uncertainty in CoT Models

Aman Chandra

The Harker School, 500 Saratoga Avenue, San Jose, CA 95129, USA; aman.p.chandra@gmail.com

ABSTRACT: Existing large language models (LLMs) struggle to solve complex mathematical reasoning problems. To improve model accuracy, LLMs have utilized reasoning in chain-of-thought approaches (CoT) to allow the model to show how it arrives at its conclusion. However, even with CoT reasoning, LLMs still tend to hallucinate (generate incorrect responses). Uncertainty quantification (UQ) methods have been proposed to quantify the certainty of a given response of an LLM, but no UQ method measures the given certainty for distinct CoT steps without consulting the LLM itself. This paper proposes stepwise uncertainty quantification (SUQ), where a UQ will be measured for every single step of the CoT response. This approach is evaluated to see if models can identify where they make mistakes in their CoT reasoning. If the SUQ of a certain step in the answer is less than a threshold amount, the LLM will be prompted to check its work.

KEYWORDS: Artificial Intelligence (AI), Large Language Models (LLMs), Chain-of-Thought (CoT), Uncertainty Quantification (UQ), Stepwise Uncertainty Quantification (SUQ).

■ Introduction

Large Language Models (LLMs) have recently become incredibly powerful in their natural language processing (NLP) tasks, especially after the recent development of sophisticated models such as GPT-4 or Gemini 2.5. However, many of these LLMs tend to be overconfident in their responses: as a result, they can give incorrect facts in their response (hallucinate), which can be dangerous in high-risk fields such as finance and medicine.² To mitigate hallucination problems and increase transparency, the LLMs have been designed to implement chain-of-thought reasoning (CoT) to clearly indicate the steps the model used to arrive at its final answer. CoT incentivizes deeper thinking into the problem and allows the user to be more confident in the generated answer. However, it is still prone to hallucinations during stepwise thinking.¹

To represent model uncertainty numerically, Uncertainty Quantification (UQ) methods were developed.² UQ provides improved transparency, allowing users to see the probability that the model's output is correct. UQ methods can be categorized into black box methods and white box methods. Black box methods generate multiple responses using the same prompt and compare the similarity of the answers, calculating their "distance" and measuring the variance in order to find some quantification of uncertainty. On the other hand, white box methods typically access the logits (the base probability values generated by the LLM) of the given LLM response and aggregate these probabilities to form an overall score.³

In addition, there have been several attempts at calculating the UQ from a CoT process. Most notably, CoT-UQ calculates the UQ during a CoT process by collecting keywords from the steps and ranking their importance, and then finally either aggregating the scaled probabilities or allowing the LLM to self-evaluate its response using a carefully structured prompt.⁴ This model is a huge step in the development of UQ

for CoT processes; however, its calculation relies heavily on inferences from the LLM itself.

This paper proposes a new approach called Stepwise Uncertainty Quantification (SUQ), in which the logits of every step in the CoT process are aggregated to calculate a reliability score based on the epistemic and aleatoric uncertainties to form a SUQ score for every single step of the reasoning process. Then, each SUQ will be compared with an established threshold value to find steps with uncertain results. The model will then be prompted to reevaluate that step, and a new answer may be generated. In contrast to the CoT-UQ approach, SUQ does not use the LLM's importance scoring and mainly focuses on the logits to calculate the UQ for each step.

Overall, this paper:

1. Reviews simple existing techniques used in LLM UQ
2. Proposes a new technique for using CoT in UQ with SUQ and provides a mathematical derivation
3. Conducts experiments testing SUQ.

How does an LLM work?

Large language models (LLMs) have been one of the most important discoveries in this age. They have successfully incorporated the principles of machine learning and artificial intelligence to perform natural language processing (NLP) tasks. An LLM's purpose is to output a given token (a string of characters, typically a word) given a string of input tokens. The LLM will generate a distribution of tokens as its output, and one of these tokens will be selected based on its probability, which is calculated by a series of multi-attention heads and multi-layer perceptrons in the transformer model. Essentially, an LLM can be represented as follows:

$$\mathcal{M}(p) = a \quad (1)$$

where \mathcal{M} represents the LLM, p represents the prompt, and a represents the response in Equation (1). However, a is actually a string of tokens:

$$a = [a_1, a_2, a_3, a_4 \dots, a_N]$$

where N is the number of tokens in the response. Each a_i is chosen from an inherent Dirichlet distribution of tokens α_i , which consists of logits, or the raw unnormalized probabilities for the given tokens in the vocabulary \mathcal{V} . This is represented in Equation (2).

$$\alpha_i^j = \mathcal{M}(a_i = \tau_j) \quad (2)$$

Essentially, the idea behind an LLM is to generate this distribution α of tokens as its output, and it does this repeatedly until it generates a coherent response to a given prompt.⁵

LLM Risk:

- **Hallucinations:**

As Large Language Models (LLMs) are being applied to more practices in different areas such as law, medicine, and finance, the risk due to hallucinations becomes critical.⁶ Hallucinations arise when an LLM provides a response that conflicts with its training data or the prompt itself. These hallucinations can be split into two groups: intrinsic hallucinations vs. extrinsic hallucinations.

Intrinsic hallucinations indicate that the LLM response contradicts the training data.² Essentially, the LLM generates information that is clearly incorrect when compared to the different texts that were used to train it. However, extrinsic hallucinations involve an LLM's response that contains information that is not present in the source documents. The LLM fabricated data that are present in the response.²

- **Transparency:**

As the risk of hallucinations increases in LLM applications, the need for a way to understand how the LLM works rises. It is a mystery how LLMs generate their responses because of their incredibly large and unpredictable models.⁷ LLMs rely on self-attention and MLPs to generate the probability distribution for the next token. However, tracking this progress is incredibly complicated because of how many different complex calculations are conducted. In addition, many LLMs are black-box, which do not allow access to the logits and the probability distribution for the next given token, which further restricts the knowledge we have about the generation of the response.⁷

- **Explainability:**

To address this transparency issue, different explainability techniques have been developed.⁸ Different methods exist to provide an insight into how these LLMs generate their responses. Among these are ways that investigate the sublayers of the transformer models to find out how each token is generated. Other methods look at the different input tokens and identify how relevant they are to the different output tokens. In addition, different probing methods have been utilized to

understand the background of how an LLM gets the information to generate its response.⁸

In addition, there have been different approaches for LLMs to demonstrate how they get to their answer. Rather than looking into the workings of the LLM, different prompting strategies have been used to understand the process the LLM takes to get to its response. For example, Chain-of-Thought (CoT) prompting asks the LLM to show a justification process for its final response. Another example is few-shot prompting, where the LLM is provided with examples of what it should do and will try to recreate that process.⁹ Overall, explainability seeks to address the transparency issue with LLMs as they become larger and larger and are prone to hallucinations.

■ Overview of Uncertainty Quantification (UQ)

The true definition of a LLM's uncertainty is hazy. However, most modern-day researchers agree that it refers to the model's doubt when comparing two similar concepts. This LLM doubt leads to hallucination, or the process of an LLM responding with an incorrect answer to a given prompt, providing wrong and possibly harmful information to the user. Hallucinations have hindered the rapid spread of LLMs to various high-profile industries such as finance, medicine, and law.¹¹ As a result, UQ has become an important area of research to identify and minimize these hallucinations.

Types of Uncertainty:

Uncertainty is classified into two broad types: aleatoric uncertainty (AU) and epistemic uncertainty (EU). These two different uncertainties relate to different types of hallucinations that an LLM can generate.

Aleatoric Uncertainty:

Aleatoric uncertainty relates to the inherent data generation randomness and ambiguity in the LLM. Specifically, it reflects ambiguous prompting because it addresses prompts with different possible answers. For example, when an LLM is prompted, "Which year was this book written?", the generated response will have a high aleatoric uncertainty because the LLM does not know which book the prompt is referring to. In other words, aleatoric uncertainty refers to the doubt an LLM has when developing an answer to an ambiguous problem.¹²

Epistemic Uncertainty:

On the other hand, epistemic uncertainty refers to the inherent contradictions between an LLM's response and its training data. The LLM generates text that provides incorrect information to the user. For example, when a LLM is prompted, "What is 10+10?" high epistemic uncertainty is evident when the model gives an incorrect answer, such as 11, because the query has no inherent ambiguity and there is a definite answer.¹² This form of uncertainty can be dangerous in many scenarios, because it can offer incorrect advice in high-risk areas such as healthcare and finance. High epistemic uncertainty

can lead to intrinsic hallucinations because the LLM will generate an output that openly contradicts its training data.

These uncertainties can lead to less reliability for LLMs in being adopted in different studies. As a result, to promote a more transparent LLM model, Uncertainty Quantification (UQ) has emerged, which provides a good way to quantify the given aleatoric and epistemic uncertainty in a model's response.

Types of UQ:

UQ can be classified into two main categories: UQ for white-box LLMs and UQ for black-box LLMs.

White Box UQ:

White-box UQ relies on accessing the LLM's tokens and logits. As a result, these methods attempt various ways of aggregating the probabilities for the given token distributions to output one score for reliability. However, one of the biggest drawbacks to white-box UQ is that it can report misleading confidence scores because the model can be overconfident and give a high score for an incorrect answer.¹⁰ The following are some of the possible white-box UQ methods.

1. Average Token Log-Probability:

$$\text{Average}(r) = -\frac{1}{L} \sum_{j=1}^L \log \left(P(x_{<j}) \right) \quad (3)$$

where L denotes the length of the response, x_j denotes the j th token in the given response, and r denotes the response in Equation (3). This measurement directly relates to the probability that the given response was generated (the product of all probabilities). However, it fails to consider the individual token distributions and can be skewed by insignificant words in the response due to random low probabilities. Because its value is a log probability times -1 , it can range from 0 to infinity.

Average token Log-Probability provides the simplest way of combining the token's probabilities in each response. As a result, it can be used for a quick and easy calculation of the UQ score. However, one of its potential drawbacks is that it does not consider the given probability distribution for each token: it does not show the likelihood that one token was chosen over another, but rather just combines all the scores.¹⁰

2. Perplexity:

$$\text{Perplexity}(r) = \exp \left(-\frac{1}{L} \sum_{j=1}^L \log \left(P(x_{<j}) \right) \right) = \exp(\text{Average}(r)) \quad (4)$$

Perplexity is the exponentiated form of the average token log-probability as seen in Equation (4). It is also directly related to the actual multiplied probability of the given tokens in the response.¹⁰ As a result, it shares the same drawbacks as using the average token log-probabilities.¹³ The perplexity score represents a log probability exponentiated, so a smaller perplexity indicates a better probability distribution and less uncertainty.¹⁰

3. Maximum Probability:

$$\text{MP}(r) = 1 - \prod_{j=1}^L P(x_{<j}) \quad (5)$$

Maximum probability considers the given response generation and treats it as a confidence score of the given LLM.¹³ Essentially, it calculates the probability that the given response was generated and takes its complement as seen in Equation (5). As a result, a lower maximum probability indicates less uncertainty, and this score ranges from 0 to 1.¹⁰

4. Maximum Entropy:

$$\text{Entropy}(p) = \left(x_{<j} \right) \quad (6)$$

where

$$H(x_{<j}) = - \sum_{x \in V} P(x_{<j}) \log P(x_{<j}) \quad (7)$$

using Shannon entropy. V represents the vocabulary of the LLM. This white-box model calculates the entropy of a given token in Equation (7) and then finds the token for which this entropy is the maximum in Equation (6) as a way of displaying the maximum uncertainty that the generated text has. Entropy represents the "peakiness" of the given distribution, which essentially means that the more uniform the probability distribution for a given token is, the greater the Shannon entropy of the system, and therefore the greater the uncertainty of the given LLM prediction. Maximum entropy finds the greatest entropy score for the given response generation as its score. This model is so far the best because it provides a way to investigate the given probability distribution for individual tokens, but it still does not consider every single token.¹³

5. P(True):

$$P(\text{True}) = 1 - P(y_1 = \text{True}) \quad (8)$$

After generating a response, the model is prompted whether it thinks that its response is true or not. The probability of the first token with the word "True" is taken as the uncertainty of the model's response in Equation (8).¹⁴ As the models increase in power, their ability to self-evaluate increases as well at a faster rate. While $P(\text{True})$ may seem trivial due to model overconfidence, studies show that when the LLM is given a series of its responses, it can correctly self-evaluate.¹⁴ This method provides an easy and computationally efficient method of quantifying uncertainty for LLMs.

Currently, none of these methods is being used in LLM UQ because they are rather basic or take up too much time and memory to calculate. However, they have been used to quantify uncertainty in Bayesian Neural Networks instead, providing much more value.¹⁰

Black Box UQ:

Not every LLM allows for access to the logits or the log probabilities. As a result, new innovative ways to calculate the UQ have been developed. Most of them involve generating multiple responses with a given LLM prompt and finding the

similarity between these responses. Some of these similarity measurements are given below.

1. Natural Language Inference (NLI) Scores:

NLI scores classify two responses as an entailment, contradiction, or neutral relation, as seen in Equations (9) and (10).⁶ NLI models, such as RoBERTa and DeBERTa, have been developed to quantify the probabilities for each of these categories. As a result, the similarity can be measured as follows.¹³

$$\text{sim}(t_1, t_2) = p_{\text{entail}} \quad (9)$$

or

$$\text{sim}(t_1, t_2) = 1 - p_{\text{contradict}} \quad (10)$$

2. Jaccard Index:

The Jaccard index calculates the ratio between the intersection of two sets and their union.¹³ When generating two responses, it can be expressed as seen in Equation (11):

$$J(t_1, t_2) = \frac{|t_1 \cap t_2|}{|t_1 \cup t_2|} \quad (11)$$

These similarity calculation methods are combined to find the uncertainty over a large number of response generations. There are many ways to combine these values, a few of which are given below:

1. Number of Semantic Sets:

This model considers the NLI semantic similarity. If an LLM generated K responses, then the entailment of every pair of two different answers can be calculated. If two responses have a high entailment score, then they will be put together in the same group. A count of the total groups provides a reasonable uncertainty estimation, and the more groups that exist, the higher the uncertainty. For example, if the prompt "Who was the first president of the USA?" was given to a LLM and three responses were generated: "George Washington", "George Washington was the first president", and "Thomas Jefferson", the number of semantic sets would be two because the first two responses have a high entailment because they give the same information.¹⁵

2. Sum of Eigenvalues of the Graph Laplacian:

After generating K responses, a graph can be used to visualize the interconnectedness of the different answers to a given prompt, with the edges highlighting how similar any two given responses are. Then, using a matrix known as the graph Laplacian, these edges can be quantified using the sum of the eigenvalues. As a result, this quantification directly shows how interconnected the different generated responses are, giving a valuable uncertainty quantification estimate.¹⁵

More Advanced Forms of UQ:

Using AU and EU for Calculating UQ:

A new form of UQ for white-box LLM, LogU, incorporates both the AU and EU of a given response generation.¹⁶ They argue that the softmax normalization of the logits outputted by an LLM inherently erases the magnitude of the certainty. As

a result, they calculate the AU and EU based on the raw logits and combine them to form a reliance score for the response. This model has the benefit of identifying different types of uncertainty in the model, such as having multiple responses or having too little information to make a definite claim.

Using UQ for CoT Processes:

Zhang *et al.* propose CoT-UQ, a novel method for calculating the UQ for every CoT step. They also employ a white-box LLM technique, but for each step, they use the LLM to find the top K keywords and rank them according to importance.⁴ Then, they aggregate these values multiplied by their probability values from the logit distribution generated by the LLM. CoT-UQ provides the important groundwork for SUQ with the idea that different steps can be evaluated for a more direct uncertainty location and quantification.

■ Derivation and Testing of SUQ

Introduction to SUQ:

Stepwise Uncertainty Quantification (SUQ) is a way to quantify uncertainty for every distinct step in a CoT process, thereby isolating the step where the given LLM goes awry and makes a mistake. This method draws from both the CoT-UQ method and the LogU method, where the AU and EU are accurately quantified based on the token distribution and then combined to form a reliability score.¹⁶ This reliability score will be calculated for every single step, and then they will be compared to a given threshold value to determine which steps have lower reliability scores than required for a relatively certain generation. Then, a new prompt will be given to the LLM to reconsider its CoT process and amend the prompted steps, if necessary. This method seeks to isolate errors and fix them in a CoT generation for different math problems.

Derivation of SUQ:

The SUQ model draws from both the CoT-UQ model and the LogU probability predictions method. It essentially extends the LogU probability prediction to the chain-of-thought and tests its capability of calculating that metric for multiple steps. Then, it proposes a novel prompt re-engineering technique to elicit a more accurate response from the LLM.

First, the prompt p will be modified to prompt p_c so that it prompts the CoT reasoning of the LLM \mathcal{M} as given in Equation (12):

$$M(p_c) = s + \hat{y} \quad (12)$$

where $s = [s_1, s_2, \dots, s_N]$, which represent the N steps in the CoT reasoning approach. Let $s_i = [s_i^1, s_i^2, \dots, s_i^K]$, where K is the number of tokens in step s_i . For each token s_i^j , there is a probability distribution given by Equation (13), where, given the possible token τ_k in the distribution for the token s_i^j :

$$M(q, s_1, \dots, s_{i-1}, s_i^1, \dots, s_i^{j-1}) = \sigma_{i,k}^j \quad (13)$$

Essentially, $\sigma_{i,k}^j$ refers to the Dirichlet distribution representing that the token τ_k , which is the token that has the k th highest chance of being picked, is chosen. A Dirichlet distribution essentially is a distribution of probability vectors that

are used to represent uncertainty. The distribution of the logits that the LLM outputs, therefore, can be represented as a Dirichlet distribution in Equation (14).¹⁶

$$\sigma_{i,0}^j = \sum_{k=1}^K \sigma_{i,k}^j \quad (14)$$

where K represents the top K logits for the given token s_i^j . $\sigma_{i,0}^j$ represents the sum of the logits for the top K probable tokens.

To calculate the aleatoric uncertainty, the entropy of the distribution of the token s_i^j , which represents how distributed the logits actually are, is calculated using Equation (15):

$$AU(s_i^j) = - \sum_{k=1}^K \frac{\sigma_{i,k}^j}{\sigma_{i,0}^j} (\psi(\sigma_{i,k}^j + 1) - \psi(\sigma_{i,0}^j + 1)) \quad (15)$$

where ψ represents the digamma function in Equation (16).¹⁶:

$$\psi(x) = \frac{d}{dx} \log \Gamma(x) \quad (16)$$

The $\frac{\sigma_{i,k}^j}{\sigma_{i,0}^j}$ represents the probability between 0 and 1 of the given token represented by $\sigma_{i,k}^j$.

The digamma function comes from the closed-form solution of the entropy of a Dirichlet distribution. Because the entropy is an apt measurement of the nonuniformity of the Dirichlet distribution for the next generated token, it serves as a fitting quantification for the inherent randomness of generating a single token, which is the AU.¹⁶

Now, the EU is calculated using Equation (17):

$$EU(s_i^j) = \frac{K}{\sum_{k=1}^K (\sigma_{i,k}^j + 1)} \quad (17)$$

The reasoning for this is that the higher the logit value for $\sigma_{i,k}^j$, the more certainty the model has with this token. As a result, the higher this value is, the lower the logits values are, the less information that this model has, indicating it has a higher EU.¹⁶

Now, we define the reliability of the given step to be R_i for step s_i , which represents the average of the reliability for the K least reliable keywords as seen in Equation (18):

$$R_i = \frac{1}{K} \sum_{t \in TK} R(t) \quad (18)$$

where TK is the set of the K least reliable keywords.

This reliability score provides a way to combine the aleatoric and epistemic uncertainties through Equation (19) to form one quantification.¹⁶

$$R(s_i^j) = \frac{1}{AU(s_i^j) \times EU(s_i^j)} \quad (19)$$

Finally, once this reliability score has been calculated for every step, it will be compared with a threshold value θ . If $R_i < \theta$ for any step, then this step will be kept track of in the change C . This conditional can be represented through the piecewise Equation (20):

$$\begin{aligned} C(i) &= p_i \text{ if } R_i \geq \theta; \\ &\text{None if } R_i < \theta \end{aligned} \quad (20)$$

The prompt addition p_i indicates that the LLM should check step i again. These changes will be aggregated to form a revised prompt pr as per Equation (21).

$$pr = p + \sum_{a=1}^N C(a) \quad (21)$$

The LLM will be prompted with this new pr to generate a more confident and accurate response.

Results and Discussion

The following prompts were used to test the SUQ approach with the Llama 3.1 8B model. The model generated a CoT process for the given prompts, and the SUQ for each step was calculated using the logits.

Table 1: This table shows the SUQ scores for the eight different steps the LLM generated for the given math question. Step 7 has a noticeable low score compared to the other scores.

What is the integral from 0 to 3x for (sin t)^2? Show your chain of thought, starting with Step 1		
Step	Description	Reliance Score
1	We need to evaluate the integral from 0 to 3x for (sin t)^2.	23.229088
2	Since the integral is from 0 to 3x, we need to substitute 3x for t in the upper bound and 0 for the lower bound. However, since the variable of integration is t, we will leave the bounds as is and focus on integrating with respect to t.	25.661079
3	The integral to be evaluated is the integral of (sin t)^2 from 0 to 3x.	12.082955
4	We can use the trigonometric identity (sin t)^2 = (1 - cos(2t))/2 to rewrite the integral.	14.135010
5	The integral becomes the integral from 0 to 3x of (1 - cos(2t))/2.	15.275674
6	The integral becomes 1/2 * integral from 0 to 3x of 1 - 1/2 * integral from 0 to 3x of cos(2t).	18.352483
7	The integral of 1 with respect to t is t. We need to evaluate this from 0 to 3x.	2.703384
8	The integral of cos(2t) with respect to t is (1/2) * sin(2t). We need to evaluate this from 0 to 3x.	16.975747

Table 2: This table shows the SUQ scores for the eight different steps the LLM generated for the given math question. All the scores are around the same values.

What is $(\sqrt{(81) * (2/3)}^2 + (15 - 3) / (2^2))^2$? First provide the answer and then derive the solution. Show the chain of thought starting with Step 1		
Step	Description	Reliance Score
1	First, we need to solve the expressions inside the parentheses. For the first expression, we need to calculate the square root of 81 and multiply it by 2/3. For the second expression, we need to subtract 3 from 15 and divide the result by 2^2.	13.026343
2	We know that the square root of 81 is 9. Therefore, the first expression becomes $(9 * (2/3))^2$.	13.894805
3	We can simplify the expression $(9 * (2/3))^2$ by first multiplying 9 and 2/3. This gives us $(6/1)^2$.	12.898972
4	Now we can simplify $(6/1)^2$. To do this, we need to square the numerator and the denominator. This gives us 36/1.	13.353616
5	Next, we need to calculate the second expression $(15 - 3) / (2^2)$. First, we need to subtract 3 from 15. This gives us 12. Then we need to divide 12 by 2^2.	14.223129
6	We know that 2^2 is equal to 4. Therefore, we need to divide 12 by 4. This gives us 3.	12.862355
7	Now we have the expressions $(\sqrt{(81) * (2/3)}^2 = 36$ and $(15 - 3) / (2^2) = 3$. We can now add these two expressions together.	12.332344
8	Adding 36 and 3 gives us 39.	12.745626

For Table 1, all the SUQ values are reasonably high, indicating low uncertainty, except for step 7, which has a lower value of 2.703384. Such a low SUQ reliance score indicates that there is either a problem in mathematical logic or just text for that step. In this scenario, the LLM forgets the $\frac{1}{2}$ coefficient it had in the previous step to keep in the integral. As a result,

this seems like the most likely reason that the LLM had a high enough reliance score. However, while the model highlights an error in this specific area, it is not guaranteed that a mathematical error is the cause of the low SUQ score: the actual stepwise solution to that problem provides the idea that the low SUQ score is explained by this error in the generated stepwise solution. The new prompt would therefore address this problem, and in the rerun of this code, the model would understand it needs to be clearer where the $\frac{1}{2}$ went in its calculations.

Now, in Table 2, the value calculated for the given expression is correct. All the SUQ reliance scores are close together and above 10, indicating that the uncertainty is low for all steps. Following the progression of logic in this example, the model provides correct steps for every single part of the logic. As a result, SUQ correctly identified that the answer and reasoning behind this problem were correct.

Overall, through the experiments, low confidence scores are typically scores below 10, which indicate either an error in mathematical reasoning or possibly just grammatical syntax for writing as well.

■ Conclusion

The paper proposes a Stepwise Uncertainty Quantification (SUQ) method to quantify uncertainty for every distinct step in a CoT process. This method draws from both the CoT-UQ method and the LogU method, where the AU and EU are accurately quantified based on the token distribution and then combined to form a reliability score, which can be compared to a threshold to identify the steps where the model has low confidence.

The SUQ method provides an accurate method to calculate UQ for different steps in a CoT process. It can be applied to various areas, such as math, logic, and fact selection, due to its ability to calculate both the AU and EU, which gives it the ability to identify different types of limitations of the LLM.

One limitation of this idea is that it requires accessing the logits, which in turn requires a white-box LLM. SUQ is not compatible with other black-box LLMs such as Gemini and ChatGPT. Running the white-box LLMs requires access to a computer with GPUs and a large amount of memory and storage. In addition, many open-source models tested on HuggingFace either do not provide access to the logits or are incapable of chain-of-thought reasoning, which essentially makes it impossible to split up the model response into steps without using some sort of natural language processor.

Possible angles of further research include:

1. Not all LLMs provide access to the actual logit values. For black-box LLMs, different forms of UQ would need to be tested to figure out the best way to calculate a metric similar to SUQ. For example, the semantic similarity of different steps of different generated responses could be compared, with the greater semantic similarity indicating a lower uncertainty score. However, this would be complicated because of the nature that different questions can be solved in different ways.

2. Different LLMs could incorporate UQ into generating responses. If the given UQ score is lower than normal, then the

model could give the user a warning that its response might not be fully accurate. This would be a huge step in the right direction for LLM explainability.

3. With access to various GPUs, this experiment can be scaled to a much larger level. Hundreds of prompts can be evaluated over different LLMs to determine if the pattern observed in this paper holds over most LLMs. Chain of thought is becoming increasingly used as a method to understand the logic of LLMs, and utilizing this method could help understand where LLMs make mistakes and the more complicated parts of different mathematical and logical reasoning steps.

4. SUQ can be trained to dynamically calculate scores based on different problem types. For example, SUQMath can be calibrated to calculate SUQ for different math problems, whereas SUQLogic can be calibrated to calculate SUQ for different logic puzzles and riddles. Other types of SUQ can be calibrated to quantify uncertainty for Chain of thought in writing and literary or historical arguments as well. This research can also include testing SUQ's generalizability to other types of problems.

5. Comparing SUQ to other metrics in this field is another area of further research. Many of the more advanced UQ methods require multiple GPUs to run calculations, and testing SUQ against other processes like CoT-UQ and LogU could be incredibly valuable to see which metric performs the best.

Overall, SUQ is only brushing the surface of UQ, and many different angles of future research can all lead to promising results. UQ provides a way for us to understand the calculations behind LLMs, which can, in turn, increase our confidence in the ideas they generate.

■ Acknowledgments

Thank you to my mentors, Dr. Siddharth Krishnan and Samuel Lefourt, for guiding me through this research process and instilling in me the key aspects of writing an effective research paper. I would not have gotten this far in the research and writing process without their invaluable help. I attest that the ideas, graphics, and writing in this paper are entirely my own.

■ References

1. Islam, S.; Elmekki, H.; Elsebai, A.; Bentahar, J.; Drawel, N.; Rjoub, G.; Pedrycz, W. A Comprehensive Survey on Applications of Transformers for Deep Learning Tasks. *Expert Systems with Applications* **2024**, *241*, 122666. <https://doi.org/10.1016/j.eswa.2023.122666>.
2. Huang, H.-Y.; Yang, Y.; Zhang, Z.; Lee, S.; Wu, Y. A Survey of Uncertainty Estimation in LLMs: Theory Meets Practice. arXiv October 20, 2024. <https://doi.org/10.48550/arXiv.2410.15326>.
3. Vashurin, R.; Fadeeva, E.; Vazhentsev, A.; Rvanova, L.; Vasilev, D.; Tsvigun, A.; Petrakov, S.; Xing, R.; Sadallah, A.; Grishchenkov, K.; Panchenko, A.; Baldwin, T.; Nakov, P.; Panov, M.; Shelmanov, A. Benchmarking Uncertainty Quantification Methods for Large Language Models with LM-Polygraph. *Transactions of the Association for Computational Linguistics* **2025**, *13*, 220–248. https://doi.org/10.1162/tacl_a_00737.
4. Zhang, B.; Zhang, R. CoT-UQ: Improving Response-Wise Uncertainty Quantification in LLMs with Chain-of-Thought. arXiv June 3, 2025. <https://doi.org/10.48550/arXiv.2502.17214>.

5. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. arXiv August 2, 2023. <https://doi.org/10.48550/arXiv.1706.03762>.
6. Chen, J.; Mueller, J. Quantifying Uncertainty in Answers from Any Language Model and Enhancing Their Trustworthiness. arXiv October 4, 2023. <https://doi.org/10.48550/arXiv.2308.16175>.
7. Liao, Q. V.; Vaughan, J. W. AI Transparency in the Age of LLMs: A Human-Centered Research Roadmap. arXiv August 8, 2023. <https://doi.org/10.48550/arXiv.2306.01941>.
8. Luo, H.; Specia, L. From Understanding to Utilization: A Survey on Explainability for Large Language Models. arXiv February 22, 2024. <https://doi.org/10.48550/arXiv.2401.12874>.
9. Amatriain, X. Prompt Design and Engineering: Introduction and Advanced Methods. arXiv May 5, 2024. <https://doi.org/10.48550/arXiv.2401.14423>.
10. Shorinwa, O.; Mei, Z.; Lidard, J.; Ren, A. Z.; Majumdar, A. A Survey on Uncertainty Quantification of Large Language Models: Taxonomy, Open Research Challenges, and Future Directions. arXiv July 1, 2025. <https://doi.org/10.48550/arXiv.2412.05563>.
11. *A Survey on Large Language Models for Critical Societal Domains-Finance, Healthcare, and Law.pdf*. Google Docs. https://drive.google.com/file/d/17E-IEAy4IGwgz-7hBrNIE6zUnRaWBWom/view?usp=drive_link&usp=embed_facebook (accessed 2025-08-01).
12. Hou, B.; Liu, Y.; Qian, K.; Andreas, J.; Chang, S.; Zhang, Y. Decomposing Uncertainty for Large Language Models through Input Clarification Ensembling. arXiv June 10, 2024. <https://doi.org/10.48550/arXiv.2311.08718>.
13. Fadeeva, E.; Rubashevskii, A.; Shelmanov, A.; Petrakov, S.; Li, H.; Mubarak, H.; Tsymbalov, E.; Kuzmin, G.; Panchenko, A.; Baldwin, T.; Nakov, P.; Panov, M. Fact-Checking the Output of Large Language Models via Token-Level Uncertainty Quantification. arXiv June 6, 2024. <https://doi.org/10.48550/arXiv.2403.04696>.
14. Kadavath, S.; Conerly, T.; Askell, A.; Henighan, T.; Drain, D.; Perez, E.; Schiefer, N.; Hatfield-Dodds, Z.; DasSarma, N.; Tran-Johnson, E.; Johnston, S.; El-Showk, S.; Jones, A.; Elhage, N.; Hume, T.; Chen, A.; Bai, Y.; Bowman, S.; Fort, S.; Ganguli, D.; Hernandez, D.; Jacobson, J.; Kernion, J.; Kravec, S.; Lovitt, L.; Ndousse, K.; Olsson, C.; Ringer, S.; Amodei, D.; Brown, T.; Clark, J.; Joseph, N.; Mann, B.; McCandlish, S.; Olah, C.; Kaplan, J. Language Models (Mostly) Know What They Know. arXiv November 21, 2022. <https://doi.org/10.48550/arXiv.2207.05221>.
15. Lin, Z.; Trivedi, S.; Sun, J. Generating with Confidence: Uncertainty Quantification for Black-Box Large Language Models. arXiv May 20, 2024. <https://doi.org/10.48550/arXiv.2305.19187>.
16. Ma, H.; Chen, J.; Wang, G.; Zhang, C. Estimating LLM Uncertainty with Logits. arXiv February 11, 2025. <https://arxiv.org/pdf/2502.00290v1>.
17. *Cycles of Thought: Measuring LLM Confidence through Stable Explanations*. <https://arxiv.org/html/2406.03441v1> (accessed 2025-07-23).
18. Wen, Y.; Wu, J.; Jain, A.; Goldstein, T.; Panda, A. Analysis of Attention in Video Diffusion Transformers. arXiv April 14, 2025. <https://doi.org/10.48550/arXiv.2504.10317>.
19. Jie, Z.; Luong, T. Q.; Zhang, X.; Jin, X.; Li, H. Design of Chain-of-Thought in Math Problem Solving. arXiv September 30, 2023. <https://doi.org/10.48550/arXiv.2309.11054>.
20. Tian, K.; Mitchell, E.; Zhou, A.; Sharma, A.; Rafailov, R.; Yao, H.; Finn, C.; Manning, C. Just Ask for Calibration: Strategies for Eliciting Calibrated Confidence Scores from Language Models Fine-Tuned with Human Feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*; Bouamor, H., Pino, J., Bali, K., Eds.; Association for Computational Linguistics: Singapore, 2023; pp 5433–5442. <https://doi.org/10.18653/v1/2023.emnlp-main.330>.
21. Huang, L.; Yu, W.; Ma, W.; Zhong, W.; Feng, Z.; Wang, H.; Chen, Q.; Peng, W.; Feng, X.; Qin, B.; Liu, T. A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions. *ACM Trans. Inf. Syst.* **2025**, *43* (2), 1–55. <https://doi.org/10.1145/3703155>.
22. [2506.08243] *Temporalizing Confidence: Evaluation of Chain-of-Thought Reasoning with Signal Temporal Logic*. <https://arxiv.org/abs/2506.08243>.
23. Ye, F.; Yang, M.; Pang, J.; Wang, L.; Wong, D. F.; Yilmaz, E.; Shi, S.; Tu, Z. Benchmarking LLMs via Uncertainty Quantification. arXiv October 31, 2024. <https://doi.org/10.48550/arXiv.2401.12794>.
24. Smith, F. B.; Kossen, J.; Trollope, E.; Wilk, M. van der; Foster, A.; Rainforth, T. Rethinking Aleatoric and Epistemic Uncertainty. arXiv August 18, 2025. <https://doi.org/10.48550/arXiv.2412.20892>.
25. Ross, B. L.; Vouitsis, N.; Ghomi, A. A.; Hosseinzadeh, R.; Xin, J.; Liu, Z.; Sui, Y.; Hou, S.; Leung, K. K.; Loaiza-Ganem, G.; Cresswell, J. C. Textual Bayes: Quantifying Uncertainty in LLM-Based Systems. arXiv June 11, 2025. <https://doi.org/10.48550/arXiv.2506.10060>.
26. Zhu, Y.; Li, G.; Jiang, X.; Li, J.; Mei, H.; Jin, Z.; Dong, Y. Uncertainty-Guided Chain-of-Thought for Code Generation with LLMs. arXiv March 19, 2025. <https://doi.org/10.48550/arXiv.2503.15341>.
27. Amini, A.; Gabriel, S.; Lin, P.; Koncel-Kedziorski, R.; Choi, Y.; Hajishirzi, H. MathQA: Towards Interpretable Math Word Problem Solving with Operation-Based Formalisms. arXiv May 30, 2019. <https://doi.org/10.48550/arXiv.1905.13319>.

■ Author

Aman is a senior at The Harker School. He enjoys learning about STEM, especially hardware and software engineering. He has 10 years of coding experience and has taken up an interest in theoretical AI and model architectures. He plays Water Polo and Tennis for his high school team and also enjoys building Legos and playing basketball for fun.