# *All Roads Lead to Rome:* Graph-Based Confidence Estimation for Large Language Model Reasoning

**Caiqi Zhang**[1] **Chang Shu**[1] **Ehsan Shareghi**[2] **Nigel Collier**[1]
[1]University of Cambridge [2]Monash University
{cz391, cs2175, nhc30}@cam.ac.uk ehsan.shareghi@monash.edu

## Abstract

Confidence estimation is essential for the reliable deployment of large language models (LLMs). Existing methods are primarily designed for factual QA tasks and often fail to generalize to reasoning tasks. To address this gap, we propose a set of training-free, graph-based confidence estimation methods tailored to reasoning tasks. Our approach models reasoning paths as directed graphs and estimates confidence by exploiting graph properties such as centrality, path convergence, and path weighting. Experiments with two LLMs on three reasoning datasets demonstrate improved confidence estimation and enhanced performance on two downstream tasks.

## 1 Introduction

Confidence estimation quantifies how certain a machine learning model is in its output. Higher confidence typically suggests a greater likelihood that the prediction is correct (Zhang et al., 2024b; Yang et al., 2024, 2025). This capability is critical in real-world applications, particularly in high-risk domains, where over-confidence can have serious consequences (Fadeeva et al., 2023; Zhang et al., 2024a, 2025a,b). When confidence is low, alternative mechanisms (*e.g.*, model self-reflection or information retrieval) can be activated to enhance overall trustworthiness.

Most prior work on confidence and uncertainty estimation focuses on factual QA tasks (Kuhn et al., 2023; Xiong et al., 2024; Fadeeva et al., 2023; Zhang et al., 2024a). However, when applied to **reasoning tasks**, existing methods often fail due to two key differences between factual and reasoning QA: (1) model outputs in the latter are longer and include intermediate steps before the final answer; and (2) these intermediate steps are logically connected, unlike factual texts where facts can be verified independently (Zhang et al., 2024a; Jiang et al., 2024). Existing methods to estimate the confidence in reasoning tasks that focus **solely** on the final answer consistency (Wang et al., 2024a; Lyu et al., 2024) are thus insufficient.

In this paper, we propose a series of novel, training-free, graph-based methods for confidence estimation in reasoning tasks. The graph structure can effectively capture the logical connections among different reasoning paths. Given a question and several independently sampled reasoning steps, we construct a directed graph to represent the reasoning process. Confidence estimation is then formulated using three graph-theoretic concepts: *centrality*, *path convergence*, and *path weighting*. Our approach is model-agnostic and can be applied directly to any language model.

We evaluate our methods on two language models, Llama3.1-8B (Meta, 2024) and Gemma2-9B (Gemma Team et al., 2024), using three reasoning-intensive benchmarks: MATH500 (Lightman et al., 2024a), MMLU-Pro (Wang et al., 2024b), and FOLIO (Han et al., 2022). Our method consistently outperforms baseline approaches. To further demonstrate its effectiveness in downstream tasks, we apply it to selective self-reflection and LLM cascading, achieving improved overall performance with fewer reflection and cascading steps.

## 2 Related Works

**Confidence Estimation in Reasoning.** While most prior work on confidence estimation has focused on factual question answering, there is growing interest in applying similar techniques to reasoning tasks (Razghandi et al., 2025; Li et al., 2025). Wang et al. (2024a) and Lyu et al. (2024) demonstrate that self-consistency can improve calibration in reasoning, though their approaches only consider final answers and ignore intermediate reasoning steps. The most similar work to ours is CoT-UQ (Zhang and Zhang, 2025), which leverages self-



*Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 31814–31824
November 4-9, 2025 

prompt keyword extraction to enhance uncertainty quantification. While both methods address uncertainty in reasoning, our approach is from another angel, focusing on the structural graph properties of multiple reasoning paths rather than keyword extraction from a single path.

**Graph-Based Confidence Estimation.** Graph-based methods have been explored for confidence estimation in short-form QA (Li et al., 2024; Lin et al., 2023). In the context of long-form generation, Jiang et al. (2024) construct semantic entailment graphs and use centrality measures to estimate confidence. Our approach differs fundamentally in both the graph construction methodology and our focus on complex reasoning tasks. Yin et al. (2024) propose token-level uncertainty as a guide for reasoning processes, while Mo and Xin (2024) introduce the Tree of Uncertain Thoughts, which employs Monte Carlo Dropout to estimate uncertainty at intermediate steps. Closest to our work, Da et al. (2025) also apply graph-based modeling to reasoning tasks; however, their graph construction strategy differs significantly from ours.

**LLM-as-a-Judge and Process Reward Models.** Another line of work focuses on verifying intermediate reasoning steps via *process reward models* (PRMs) and *LLM-as-a-judge* frameworks. These approaches train specialized reward models to assess the correctness of each step in a solution trajectory (Zhang et al., 2025c; Lightman et al., 2024b; Wang et al., 2023; Han et al., 2025). For example, Lightman et al. (2024b) compare outcome vs. process supervision and introduce a PRM trained on 800K human-annotated step-wise feedback labels, significantly improving a model's reliability on math problems. Similarly, Zhang et al. (2025c) integrate an LLM-as-judge to label intermediate steps, finding that naive self-evaluation underperforms compared to using a large language model or human feedback. Wang et al. (2023) present *Math-Shepherd*, a PRM trained with automatically generated step-level supervision signals for math, which is used to rerank solution steps and guide step-by-step reinforcement learning. More recently, Han et al. (2025) introduce a unified verifier, *VerifiAgent*, that performs both meta-level consistency checks and tool-assisted verification, achieving broad improvements with fewer samples and lower cost than prior PRM-based methods. While effective in estimating the likelihood that each reasoning step is correct, these methods typically require heavy training and the use of specialized reward models, making them resource-intensive. Therefore, we do not include them in our comparisons, focusing instead on techniques that do not demand such dedicated reward model training.

## 3 Methodology

**Motivation** Reasoning tasks can be solved through multiple pathways, each potentially leading to the same or different answers. This structure can naturally be modeled as a directed graph. The edges represent the logical flow of reasoning and nodes for each step. Each path from the question to an answer corresponds to a distinct reasoning route. Our central insight is that *an answer supported by numerous, diverse paths is more likely to be correct*, reflecting the idea that "all roads lead to Rome." Based on this intuition, our method generates multiple reasoning chains, merges them into a unified graph, and computes confidence from the resulting structure.

### 3.1 Reasoning Chain Sampling

For a question $Q$, we generate $N$ reasoning chains by prompting the model, denoted as $\mathcal{C} = \{C_1, C_2, \ldots, C_N\}$. Each reasoning chain $C_i$ can be further decomposed into a sequence of steps, *i.e.*, $C_i = \{s_{i1}, s_{i2}, \ldots, s_{iT_i}\}$, where $T_i$ is the number of steps in the $i$-th chain. The final answers produced by the $N$ reasoning chains are denoted as $\mathcal{A} = \{A_1, A_2, \ldots, A_N\}$. Identical answers across different chains are merged into a single node, resulting in a set of distinct answer nodes $\mathcal{A}' \subseteq \mathcal{A}$, with $|\mathcal{A}'|$ ranging from 1 to $N$.

### 3.2 Graph Construction

We then construct a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to represent the multi-path reasoning process. Each node corresponds to either a reasoning step or an answer, and edges capture the structure of reasoning. Starting from the question node $Q$, we iteratively add reasoning steps from each sampled chain. For each chain, we connect consecutive steps using *intra-edges*, which are *directed* edges representing the forward logical progression within a single reasoning path. In contrast, we also add *inter-edges* between nodes from different chains that express equivalent meaning (identified via an auxiliary model; see Appendix C). These *bidirectional* inter-edges capture semantic equivalence across chains. The construction procedure is described in Algorithm 1. An simplified example with only

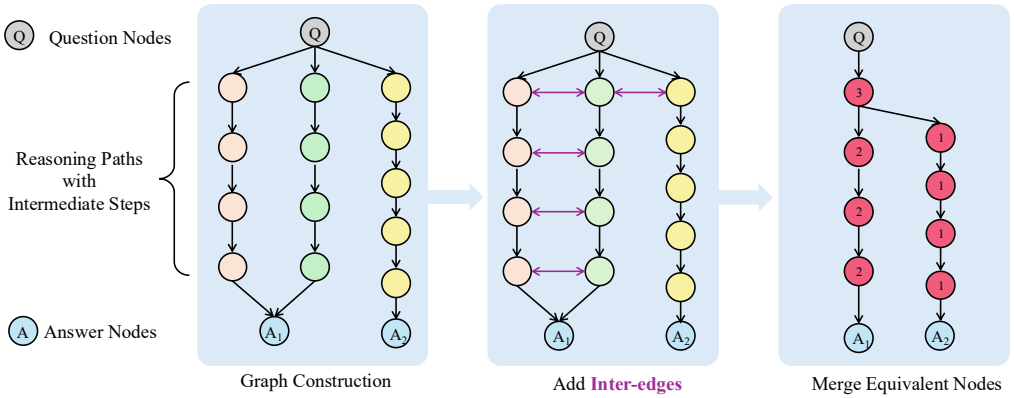


Figure 1: Illustration of the graph construction. The left part shows the graph with only ***intra-edges***. The middle part includes ***inter-edges***. The right part shows the graph after merging equivalent nodes for the PATHWEIGHT method. Node weights are indicated inside the nodes.

**Algorithm 1:** Graph Construction

**Input:** Question $Q$ and reasoning chains $\{C_1, \ldots, C_N\}$ where $C_i = \{s_{i1}, \ldots, s_{iT_i}\}$
**Output:** Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$

Initialize $\mathcal{V} \leftarrow \{Q\}, \mathcal{E} \leftarrow \emptyset$;
**for** $i \leftarrow 1$ **to** $N$ **do**
  **for** $j \leftarrow 1$ **to** $T_i$ **do**
    Add node $s_{ij}$ to $\mathcal{V}$;
    **if** $j = 1$ **then**
      `// First node in the chain`
      Add edge $(Q \rightarrow s_{i1})$ to $\mathcal{E}$;
    **else**
      Add intra-edge $(s_{i(j-1)} \rightarrow s_{ij})$ to $\mathcal{E}$;
**foreach** *pair* $(s_{ij}, s_{kl})$ *from different paths* **do**
  **if** *Equivalent*$(s_{ij}, s_{kl})$ **then**
    Add inter-edge $(s_{ij} \leftrightarrow s_{kl})$ to $\mathcal{E}$;
**for** $i \leftarrow 1$ **to** $N$ **do**
  **if** $A_i \notin \mathcal{V}$ **then**
    `// Only keep unique answers`
    Add answer node $A_i$ to $\mathcal{V}$;
  Add edge $(s_{iT_i} \rightarrow A_i)$ to $\mathcal{E}$;

three paths and a limited number of intermediate steps is shown in Figure 1.

### 3.3 Confidence Calculation

Given a question $x$ and a model output $y$, the confidence is denoted as $\text{Conf}(x, y)$. With the graph $\mathcal{G}$, we formulate the confidence estimation problem in the following ways:

**Centrality-Based Confidence (CENCONF).** Inspired by the self-consistency approach (Wang et al., 2024a; Lyu et al., 2024), which is essentially the *in-degree centrality* of each answer node (Jiang et al., 2024), we adopt *Katz centrality* to capture more graph information. Katz centrality evaluates a node's influence by considering both immediate neighbors and all other nodes that connect to it through these neighbors, with the influence of distant nodes attenuated by a factor $\alpha$. Intuitively, an answer node that is reachable via numerous short, semantically meaningful paths is more likely to be correct. Further discussion on other centrality metrics is provided in Appendix A.

Formally, let $A$ denote the adjacency matrix of the graph $\mathcal{G}$. The Katz centrality score for node $v$ is defined as:

$$\text{Katz}(v) = \alpha \sum_j A_{vj} \, \text{Katz}(j) + \beta$$

where $\alpha$ is the attenuation factor (with $0 < \alpha < 1/\lambda_{\max}$, and $\lambda_{\max}$ being the largest eigenvalue of $A$), and $\beta$ is a constant representing the initial centrality assigned to each node. This recursive formula accounts for the influence of neighboring nodes, with the impact diminishing over longer paths. After computing $\text{Katz}(v)$ for each candidate answer node, we normalize the scores to obtain a confidence measure:

$$\text{Conf}(A_i) = \frac{\text{Katz}(A_i)}{\sum_{A_j \in \mathcal{A}'} \text{Katz}(A_j)}$$

**Path Convergence Confidence (PATHCONV).** In PATHCONV, we leverage the number of distinct reasoning paths from the question node $Q$ to each candidate answer node $A_i \in \mathcal{A}'$. The underlying intuition is that *an answer reached by many distinct (or partially overlapping) paths is more likely to be correct*. Formally, we define: $\text{paths}(Q \rightarrow A_i) = \{\pi \mid \pi = (Q \rightarrow \cdots \rightarrow A_i) \in \mathcal{G}\}$. The total number of paths that reach any answer is:

$$P_{\text{all}} = \sum_{A_j \in \mathcal{A}'} |\text{paths}(Q \rightarrow A_j)|$$

The normalized confidence score is then computed as:

$$\text{Conf}(A_i) = \frac{|\text{paths}(Q \to A_i)|}{P_{\text{all}}}$$

In cases where the graph $\mathcal{G}$ is large and enumerating all paths is computationally intractable, we propose a path sampling strategy to approximate the path counts (Appendix B).

**Path Weighting Confidence (PATHWEIGHT).** In PATHWEIGHT, we merge the nodes with equivalent semantic meaning. Each merged node $v$ is assigned a weight $w(v)$ (default 1) representing the number of original steps combined. For a path $\pi = (Q \to \cdots \to A_i)$ traversing nodes $\{v_1, v_2, \ldots, v_k\}$, we define its score as:

$$\text{pathScore}(\pi) = \prod_{v \in \pi} w(v)$$

The path weighting confidence is:

$$\text{Conf}(A_i) = \frac{\sum_{\pi \in \text{paths}(Q \to A_i)} \text{pathScore}(\pi)}{\sum_{A_j \in \mathcal{A}'} \sum_{\pi' \in \text{paths}(Q \to A_j)} \text{pathScore}(\pi')}$$

PATHWEIGHT emphasizes paths that incorporate commonly shared reasoning steps, thus boosting the confidence of answers supported by converging and repeated logic. Compared to simple path counting, the path weighting approach suppresses the influence of isolated or idiosyncratic steps, resulting in a more robust confidence estimation.

# 4 Experiments

## 4.1 Setup

**Models and Datasets.** We use the instruction-tuned Llama3.1-8B (Meta, 2024) and Gemma2-9B (Gemma Team et al., 2024). For evaluation, we select MATH500 (Lightman et al., 2024a), MMLU-Pro STEM (Wang et al., 2024b), and FOLIO (Han et al., 2022), which span arithmetic, STEM, and logical reasoning tasks. We deliberately select tasks that yield a balanced mix of correct and incorrect predictions, as this scenario best showcases the value of confidence estimation. [1]

**Baselines and Metrics.** Given the large number of confidence elicitation methods, we only select representative approaches. We also exclude methods that cannot provide confidence to each response (*e.g.*, semantic uncertainty (Kuhn et al., 2023)). For model self-reported confidence, we include: $p$(true) (Kadavath et al., 2022), and self-verbalized confidence (Self-Verb) (Tian et al., 2023; Xiong et al., 2024). For consistency-based methods, we consider self-consistency (Self-Cons) (Wang et al., 2024a; Lyu et al., 2024), Degree Matrix (Deg) (Lin et al., 2023), and Eccentricity (Ecc) (Lin et al., 2023). Additionally, we include LUQ (Zhang et al., 2024a), a method specifically designed for long-form outputs. To ensure a fair comparison, we exclude PRMs (Wang et al., 2023; Zhang et al., 2025c), as they require extensive training. For evaluation, we use AUROC (Bradley, 1997) as our main metric following (Kuhn et al., 2023; Lin et al., 2023). For comparison, we also include commonly used metrics: Brier Score (BS) (Brier, 1950) and Expected Calibration Error (ECE) (Naeini et al., 2015).

**Experiment Settings.** We use NetworkX (Hagberg et al., 2008) for graph construction and computation. The model is prompted to output in a clear structure, explicitly illustrating each step. Additional experiment details, evaluation strategies and prompts are provided in Appendix C and E.

## 4.2 Main Results

Across all three benchmarks and both foundation models, our proposed graph-based methods consistently outperforms non-graph baselines. For Gemma, PATHWEIGHT raised AUROC from 60.9% to 81.5% on MATH500 while cutting ECE from 35.6% to 15.5%; PATHCONV pushed the Brier Score down from 41.1% to 17.2% with a comparable AUROC of 80.9%. Similar trends hold for Llama, where PATHWEIGHT reaches an AUROC of 84.0% (vs. 64.0%) and an ECE of 10.4%, and PATHCONV attains the lowest Brier Score of 10.3%. CENCONF trails the two graph methods but still outperforms baseline methods. The results confirm that modeling logical connections and leveraging graph properties can significantly improve confidence estimation for reasoning tasks.

# 5 Applications

We demonstrate two downstream uses of the PATH-WEIGHT confidence estimator: *selective self-reflection* and *LLM cascading*. Both interventions are triggered only for the $k \in \{5, 10, 15\}\%$ lowest-confidence instances. For comparison, we also report a naive baseline that applies the same intervention to *all* queries ($k = 100\%$). Unless otherwise specified, the initial response is generated

[1] Datasets like GSM8K (Cobbe et al., 2021), where 7B-scale models already achieve >90% accuracy, are less suitable for evaluating uncertainty estimation methods, since always predicting high confidence can still yield good calibration.

| Method | MATH500 | | | MMLU-Pro | | | FOLIO | | |
|---|---|---|---|---|---|---|---|---|---|
| | AUROC↑ | BS↓ | ECE↓ | AUROC↑ | BS↓ | ECE↓ | AUROC↑ | BS↓ | ECE↓ |
| **Gemma-2-9B-It** | | | | | | | | | |
| p(true) | 60.9 | 41.1 | 35.6 | 59.4 | 40.7 | 35.6 | 62.3 | 39.5 | 34.9 |
| Self-Verb | 61.6 | 36.8 | 33.5 | 60.7 | 37.7 | 32.8 | 64.4 | 35.7 | 31.2 |
| Self-Cons | 75.3 | 24.4 | 21.6 | 69.8 | 25.5 | 26.9 | 76.1 | 24.1 | 19.4 |
| Deg | 63.7 | 36.3 | 30.3 | 61.3 | 36.6 | 33.0 | 64.0 | 35.9 | 30.1 |
| Ecc | 64.9 | 34.6 | 31.0 | 62.9 | 35.7 | 31.7 | 68.1 | 35.1 | 27.6 |
| LUQ | 75.0 | 25.1 | 20.9 | 69.6 | 26.2 | 27.4 | 74.9 | 25.4 | 21.0 |
| CenConf | 76.7 | 18.8 | 17.0 | 77.9 | 18.3 | 19.2 | 78.3 | 16.6 | 16.7 |
| PathConv | 80.9 | 17.2 | 16.6 | 78.5 | 18.7 | 16.6 | 81.2 | 21.0 | 11.2 |
| PathWeight | 81.5 | 17.8 | 15.5 | 80.8 | 19.1 | 13.1 | 82.9 | 15.4 | 13.2 |
| **Llama-3-8B-Instruct** | | | | | | | | | |
| p(true) | 64.0 | 36.3 | 31.2 | 63.3 | 37.4 | 32.5 | 64.8 | 35.1 | 30.6 |
| Self-Verb | 65.5 | 33.3 | 28.4 | 64.7 | 34.2 | 29.1 | 66.5 | 32.3 | 27.0 |
| Self-Cons | 79.3 | 15.1 | 18.2 | 73.1 | 22.5 | 20.2 | 80.0 | 20.5 | 17.4 |
| Deg | 68.2 | 32.2 | 27.1 | 66.3 | 33.1 | 28.3 | 70.2 | 30.3 | 25.2 |
| Ecc | 68.8 | 31.5 | 26.0 | 67.1 | 32.6 | 27.2 | 70.8 | 29.4 | 24.1 |
| LUQ | 79.5 | 17.5 | 17.5 | 73.5 | 23.2 | 19.5 | 80.5 | 20.5 | 16.5 |
| CenConf | 83.0 | 16.1 | 11.7 | 81.0 | 19.7 | 16.3 | 80.5 | 17.2 | 13.2 |
| PathConv | 83.5 | 10.3 | 15.8 | 83.5 | 15.5 | 9.5 | 84.0 | 21.8 | 9.8 |
| PathWeight | 84.0 | 15.3 | 10.4 | 82.0 | 15.1 | 11.1 | 85.5 | 19.4 | 9.5 |

Table 1: Experiment Results on Gemma and Llama. All values are in percentages. The best three results are highlighted in orange. The best results largely fall into our graph-based methods.

by Llama3.1-8B-Instruct.

**Selective Self-Reflection.** Self-reflection prompts the *same* model a second time to critique and revise its own answer. For each low confidence example, we append a concise *reflect-then-finalise* prompt (see Appendix E), asking the model to (i) identify flaws and (ii) provide a corrected response.

Table 2 shows that reflecting on just the bottom 15% of low-confidence queries yields accuracy improvements of $+3$ to $+5$ points across three benchmarks. Moreover, reflecting *all* the time ($+100\%$) proves suboptimal—and in the case of MATH500, even *reduces* accuracy. This echoes prior findings that excessive self-critique may cause correct answers to degrade, due to sycophancy (Sharma et al., 2024) or over-revision (Laban et al., 2023).

| Dataset | Base | +5% | +10% | +15% | +100% |
|---|---|---|---|---|---|
| MATH500 | 49.8 | 52.1 | 53.4 | 54.3 | 53.0 |
| MMLU-Pro | 47.3 | 50.2 | 51.5 | 52.4 | 50.8 |
| FOLIO | 63.5 | 66.0 | 66.8 | 67.2 | 64.8 |

Table 2: Accuracy (%) after Selective Self-Reflection on the $k\%$ least-confident samples.

**LLM Cascading.** Instead of reflecting, we can *escalate* low-confidence queries to a more capable (larger but slower) model. In our setting, low-confidence cases are routed to Llama3-70B-Instruct with the same prompt, while the rest are handled by the original Llama3-8B-Instruct. As shown in Table 3, cascading just the least-confident 15% of queries yields accuracy improvements of around $+2$ to $+5$ points.

| Dataset | Base | +5% | +10% | +15% | +100% |
|---|---|---|---|---|---|
| MATH500 | 49.8 | 51.2 | 53.0 | 54.6 | 58.1 |
| MMLU-Pro | 47.3 | 49.1 | 51.7 | 52.9 | 56.3 |
| FOLIO | 63.5 | 64.2 | 65.8 | 67.1 | 70.4 |

Table 3: Accuracy (%) when cascading the $k\%$ least-confident queries to Llama3-70B-Instruct.

## 6 Conclusion

We present a suite of graph-based, training-free methods for confidence estimation in reasoning tasks. By modeling reasoning paths and their logical connections as directed graphs, our approach captures deeper structural signals often overlooked by existing methods. Empirical results show consistent improvements over baselines across multiple benchmarks. Our methods also improve downstream applications such as self-reflection and LLM cascading, highlighting the practical benefits of our methods. We hope this work inspires future research into graph-based reasoning and uncertainty modeling in large language models.

## Limitations and Future Work

**Compute/latency overhead.** Our approach increases inference-time cost by sampling multiple reasoning chains and constructing a graph per instance. This cost is shared with consistency-style methods that similarly rely on sampling rather than internal logits. In practice, the overhead can be mitigated with early-exit heuristics (e.g., skipping graph construction when all samples agree) and by trading sample count for speed in settings with tight latency budgets.

**Black-box scope (no logits).** We intentionally design the method to be *black-box* and model-agnostic, requiring only generated text. This broadens applicability to APIs and closed models where token-level logits are unavailable. A natural extension for *white-box* access is to inject logit-derived signals into the graph: for example, (i) weight intra-edges in a chain by token- or step-level probabilities to reflect per-step confidence, and (ii) weight inter-edges by the verifier's (or auxiliary judge's) confidence in semantic equivalence. Such integrations could further sharpen both path aggregation and centrality scores, but would narrow the method's deployment footprint to models exposing internals. We leave this to future work.

**Single-property estimators (no ensembling).** To isolate the contribution of distinct structural signals, we study three graph-based confidence estimators *independently*. Prior work suggests that fusing complementary confidence cues can yield more robust estimators; ensembling these graph properties (e.g., via learned stacking, temperature-free weighted voting, or calibration-aware mixtures) is therefore a promising avenue we deliberately defer to future work.

**Sensitivity to graph construction.** Our estimates depend on faithful step decomposition and reliable cross-path equivalence detection. Noisy step segmentation or spurious equivalence links can perturb the topology (e.g., by creating shortcuts or cycles), which in turn can bias centrality and path counts. Robustness could be improved with stricter agreement thresholds, cycle-aware pruning, or by marginalizing over multiple equivalence graphs rather than committing to a single one.

## Ethics Statement

Our research adheres to strict ethical guidelines. We verified the licenses of all software and datasets used in this study to ensure full compliance with their terms. No privacy concerns have been identified. We have conducted a thorough assessment of the project and do not anticipate any further risks.

## Acknowledgment

We thank Chengzu Li and Ying Xu for their constructive feedback throughout this project. We are also grateful to the anonymous reviewers for their insightful comments and suggestions during the review process.

## References

Andrew P Bradley. 1997. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159.

Glenn W Brier. 1950. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *ArXiv preprint*, abs/2110.14168.

Longchao Da, Xiaoou Liu, Jiaxin Dai, Lu Cheng, Yaqing Wang, and Hua Wei. 2025. Understanding the uncertainty of llm explanations: A perspective based on reasoning topology. *Preprint*, arXiv:2502.17026.

Ekaterina Fadeeva, Roman Vashurin, Akim Tsvigun, Artem Vazhentsev, Sergey Petrakov, Kirill Fedyanin, Daniil Vasilev, Elizaveta Goncharova, Alexander Panchenko, Maxim Panov, Timothy Baldwin, and Artem Shelmanov. 2023. LM-polygraph: Uncertainty estimation for language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 446–461, Singapore. Association for Computational Linguistics.

Gemma Team, Morgane Riviere, and Shreya Pathak et al. 2024. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. 2008. Exploring network structure, dynamics, and function using networkx. In *Proceedings of the 7th Python in Science Conference*, pages 11 – 15, Pasadena, CA USA.

Jiuzhou Han, Wray Buntine, and Ehsan Shareghi. 2025. Verifiagent: a unified verification agent in language model reasoning. *ArXiv preprint*, abs/2504.00406.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, Lucy Sun,

Alex Wardle-Solano, Hannah Szabo, Ekaterina Zubova, Matthew Burtell, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, and 16 others. 2022. Folio: Natural language reasoning with first-order logic. *Preprint*, arXiv:2209.00840.

Mingjian Jiang, Yangjun Ruan, Prasanna Sattigeri, Salim Roukos, and Tatsunori Hashimoto. 2024. Graph-based uncertainty metrics for long-form language model outputs. *Preprint*, arXiv:2410.20783.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, and 17 others. 2022. Language models (mostly) know what they know.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626.

Philippe Laban, Lidiya Murakhovs'ka, Caiming Xiong, and Chien-Sheng Wu. 2023. Are you sure? challenging llms leads to performance drops in the flipflop experiment. *Preprint*, arXiv:2311.08596.

Yinghao Li, Rushi Qiang, Lama Moukheiber, and Chao Zhang. 2025. Language model uncertainty quantification with attention chain. In *Second Conference on Language Modeling*.

Yukun Li, Sijia Wang, Lifu Huang, and Li-Ping Liu. 2024. Graph-based confidence calibration for large language models. *Preprint*, arXiv:2411.02454.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024a. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024b. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. 2023. Generating with confidence: Uncertainty quantification for black-box large language models. *Preprint*, arXiv:2305.19187.

Qing Lyu, Kumar Shridhar, Chaitanya Malaviya, Li Zhang, Yanai Elazar, Niket Tandon, Marianna Apidianaki, Mrinmaya Sachan, and Chris Callison-Burch. 2024. Calibrating large language models with sample consistency. *Preprint*, arXiv:2402.13904.

Meta. 2024. Llama 3 model card.

Shentong Mo and Miao Xin. 2024. Tree of uncertain thoughts reasoning for large language models. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2024, Seoul, Republic of Korea, April 14-19, 2024*, pages 12742–12746. IEEE.

Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. 2015. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 2901–2907. AAAI Press.

Ali Razghandi, Seyed Mohammad Hadi Hosseini, and Mahdieh Soleymani Baghshah. 2025. CER: Confidence enhanced reasoning in LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7918–7938, Vienna, Austria. Association for Computational Linguistics.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Esin Durmus, Zac Hatfield-Dodds, Scott R. Johnston, Shauna Kravec, Timothy Maxwell, Sam McCandlish, Kamal Ndousse, Oliver Rausch, Nicholas Schiefer, Da Yan, Miranda Zhang, and Ethan Perez. 2024. Towards understanding sycophancy in language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher Manning. 2023. 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*, pages 5433–5442, Singapore. Association for Computational Linguistics.

Ante Wang, Linfeng Song, Ye Tian, Baolin Peng, Lifeng Jin, Haitao Mi, Jinsong Su, and Dong Yu. 2024a. Self-consistency boosts calibration for math reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6023–6029, Miami, Florida, USA. Association for Computational Linguistics.

Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui.

2023. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *ArXiv preprint*, abs/2312.08935.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024b. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. 2024. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Ruihan Yang, Caiqi Zhang, Zhisong Zhang, Xinting Huang, Sen Yang, Nigel Collier, Dong Yu, and Deqing Yang. 2024. Logu: Long-form generation with uncertainty expressions.

Ruihan Yang, Caiqi Zhang, Zhisong Zhang, Xinting Huang, Dong Yu, Nigel Collier, and Deqing Yang. 2025. Uncle: Uncertainty expressions in long-form generation. *ArXiv preprint*, abs/2505.16922.

Zhangyue Yin, Qiushi Sun, Qipeng Guo, Zhiyuan Zeng, Xiaonan Li, Junqi Dai, Qinyuan Cheng, Xuanjing Huang, and Xipeng Qiu. 2024. Reasoning in flux: Enhancing large language models reasoning through uncertainty-aware adaptive guidance. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2401–2416, Bangkok, Thailand. Association for Computational Linguistics.

Boxuan Zhang and Ruqi Zhang. 2025. CoT-UQ: Improving response-wise uncertainty quantification in LLMs with chain-of-thought. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 26114–26133, Vienna, Austria. Association for Computational Linguistics.

Caiqi Zhang, Fangyu Liu, Marco Basaldella, and Nigel Collier. 2024a. LUQ: Long-text uncertainty quantification for LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5244–5262, Miami, Florida, USA. Association for Computational Linguistics.

Caiqi Zhang, Ruihan Yang, Zhisong Zhang, Xinting Huang, Sen Yang, Dong Yu, and Nigel Collier. 2024b. Atomic calibration of llms in long-form generations.

Caiqi Zhang, Xiaochen Zhu, Chengzu Li, Nigel Collier, and Andreas Vlachos. 2025a. Reinforcement learning for better verbalized confidence in long-form generation. *ArXiv preprint*, abs/2505.23912.

Zhaohan Zhang, Ziquan Liu, and Ioannis Patras. 2025b. Grace: A generative approach to better confidence elicitation in large language models.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025c. The lessons of developing process reward models in mathematical reasoning. *ArXiv preprint*, abs/2501.07301.

# Appendix

## A More on Centrality-Based Methods

During our experiment, we tested other centrality metrics such as closeness, pagerank, and Laplacian; however, they showed inferior performance compared to Katz centrality, which we utilize in CenConf. Katz centrality consistently yielded higher AUROC and significantly lower calibration errors, demonstrating superior effectiveness for this task.

| Method | AUROC↑ | BS↓ | ECE↓ |
|---|---|---|---|
| Katz | 83.0 | 16.1 | 11.7 |
| Closeness | 69.2 | 22.5 | 15.6 |
| Pagerank | 65.3 | 24.6 | 18.2 |
| Laplacian | 64.3 | 27.3 | 19.2 |

Table 4: Performance of Llama-3-8B-Instruct on MATH500.

## B More on PATHCONV

Algorithm 2 outlines a randomized method that traverses the graph from $Q$ up to a maximum path length $L$, sampling $M$ paths and accumulating weighted counts for each candidate answer $A_i$.

### B.1 Path Sampling Strategy

**Algorithm 2:** Path Sampling for Approximating Weighted Path Counts

**Input:** Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, question node $Q$, candidate answer $A_i$, sample size $M$, maximum path length $L$, attenuation factor $\gamma \in (0, 1)$

**Output:** Estimated weighted path count $\widehat{P}(Q \to A_i)$

Initialize accumulator $C \leftarrow 0$;
**for** $m \leftarrow 1$ **to** $M$ **do**
  Set current node $v \leftarrow Q$;
  Set path weight $w \leftarrow 1$;
  **for** $\ell \leftarrow 1$ **to** $L$ **do**
    **if** $v = A_i$ **then**
      $C \leftarrow C + w$;
      **break** the current sample;
    Obtain successor set $\mathcal{N}(v)$ from $\mathcal{G}$;
    **if** $\mathcal{N}(v)$ is empty **then break**;
    Sample a node $v'$ uniformly from $\mathcal{N}(v)$;
    Update path weight: $w \leftarrow w \times \gamma$;
    Set $v \leftarrow v'$;
**return** $\widehat{P}(Q \to A_i) \leftarrow \frac{C}{M}$;

The estimated weighted confidence for each answer $A_i$ is then incorporated into the overall normalization:

$$\text{Conf}(A_i) = \frac{\widehat{P}(Q \to A_i)}{\sum_{A_j \in \mathcal{A}'} \widehat{P}(Q \to A_j)}.$$

### B.2 Avoiding Loops

In our experiments, we observe that loops occasionally appear in the graphs (less than 5% of cases). This is mainly due to misjudgments in equivalence checking. In such cases, we remove the loops by discarding certain inter-edges in the loops.

## C Experiment Details

We use vLLM (Kwon et al., 2023) for all model inference. To find equivalent steps, we prompt a Llama-3-8B-Instruct with temperature 0.

For path generation, we use the temperature 1 with 3-shot in-context learning. We generate 10 additional samples for graph construction or consistency calculation. For fairness comparison, for p(ture) and Self-Verb, we directly ask the model whether each sampled answer is correct. For consistency-based methods, if the answer does not appear in the samples, it will be given a confidence 0. For all methods, the output scores are normalized to the range $[0, 1]$ before computing ECE and Brier Score.

## D Hyperparameters

**Setup.** Here we report the ablations conducted with **Llama-3-8B** on **MATH500**. Our default settings are $\alpha$=0.1, $N$=10 sampled traces, and $L$=12 maximum path length.

### Attenuation Factor $\alpha$ (for CENCONF)

Table 5 shows that $\alpha$=0.1 yields the best AUROC with a favorable ECE, validating our default.

Table 5: Ablation on attenuation factor $\alpha$ (CENCONF).

| $\alpha$ | AUROC (%) | ECE (%) |
|---|---|---|
| 0.01 | 79.5 | 14.1 |
| **0.1 (Default)** | **83.0** | **11.7** |
| 0.2 | 82.2 | 12.9 |
| 0.5 | 78.9 | 15.4 |

### Number of Sampled Traces $N$ (for PATHWEIGHT)

Increasing $N$ improves performance with diminishing returns (Table 6). $N$=10 provides a strong cost–performance trade-off and is used as default.

### Maximum Path Length $L$ (for PATHCONV)

We observed average reasoning chains of roughly 5–6 steps. Table 7 shows that $L$=12 avoids truncation errors seen at $L$=5; larger caps ($L \geq 15$) yield no meaningful AUROC gains and only modest ECE

Table 6: Sensitivity to the number of sampled traces $N$ (PATHWEIGHT).

| $N$ | **AUROC (%)** | **ECE (%)** |
|---|---|---|
| 3 | 79.1 | 16.2 |
| 5 | 81.7 | 14.0 |
| **10 (Default)** | **84.0** | **10.4** |
| 20 | 84.5 | 10.1 |

improvements, so $L$=12 remains our default for efficiency.

Table 7: Impact of maximum path length $L$ (PATHCONV).

| $L$ | **AUROC (%)** | **ECE (%)** |
|---|---|---|
| 5 | 81.2 | 21.5 |
| **12 (Default)** | **83.5** | 15.8 |
| 15 | 83.6 | 13.4 |
| 20 | 83.6 | 12.6 |

## E Prompt

**PROMPT FOR PATH GENERATION**

Answer the following question. Break down your reasoning process into the smallest possible steps. Each step should represent a single, minimal reasoning action, and each step must logically follow the previous one. Use the following format for each step:
Step N: Thought: [Provide a detailed explanation of your reasoning for this step.]
Present your entire reasoning process in one cohesive response.
After completing all the steps, conclude with:
Final Answer: \boxed{[Your final numerical answer here without the unit or any additional text]}
Ensure that your response strictly follows this format to maintain clarity and consistency.
Question: {question}

Table 8: Prompt for Path Generation.

**SELECTIVE SELF-REFLECTION PROMPT**

Question: {question}
Previous Answer: {model's earlier response}
Please review your previous answer. Identify any errors or flaws, and revise your answer if necessary. For you final answer, output it in the following format:
Final Answer: \boxed{[Your corrected final answer here]}

Table 9: Prompt for Self-Reflection.

**EQUIVALENCE CHECKING PROMPT**

You are tasked with identifying the equivalent reasoning step in Path B for a specific reasoning step in Path A.
**Context:** Path A and Path B are sequences of reasoning steps.
**Inputs:**
Target Step: {A Step in Path A}
Steps in Path B: {All Steps in Path B}
**Your Task:** Identify the single step number in Path B that is equivalent to the given step in Path A. The equivalent step must:

- Contain the same reasoning as the given step in Path A.
- Not contain additional or conflicting information.

If no such step exists, respond with "none".
**Output Format:** Provide only the step number (e.g., "5") or "none".

Table 10: Prompt for Equivalence Checking.