Self-reflective Uncertainties: Do LLMs Know Their Internal Answer Distribution?

Michael, Kirchhof [1]   **Luca, Füger** [2]   **Adam, Golinski** [1]   **Eeshan Gunesh, Dhekane** [1]   **Arno, Blaas** [1]
Sinead, Williamson [1]

Abstract

To reveal when a large language model (LLM) is uncertain about a response, uncertainty quantification commonly produces percentage numbers along with the output. But is this all we can do? We argue that in the output space of LLMs, the space of strings, exist strings expressive enough to summarize the *distribution over* output strings the LLM deems possible. We lay a foundation for this new avenue of uncertainty explication and present SelfReflect, a theoretically-motivated metric to assess how faithfully a string summarizes an LLM's internal answer distribution. We show that SelfReflect is able to discriminate even subtle differences of candidate summary strings and that it aligns with human judgement, outperforming alternative metrics such as LLM judges and embedding comparisons. With SelfReflect, we investigate a number of self-summarization methods and find that even state-of-the-art reasoning models struggle to explicate their internal uncertainty. But we find that faithful summarizations can be generated by sampling and summarizing.

1. Introduction

When large language models (LLMs) are uncertain about a response, either because the query is ambiguous or because they are factually unsure, they should indicate it. Consider the example in Figure 1. The LLM's internal distribution comprises a variety of answers, but this variability is ignored if we just output the greedy response. While existing uncertainty quantification approaches augment the greedy response (or any other single sample from the distribution) with a numerical measure of uncertainty (Aichberger et al., 2024; Fadeeva et al., 2023; Fomicheva et al., 2020; Malinin & Gales, 2020) or verbalize the confidence in the response

(Lin et al., 2022; Yona et al., 2024), this offers limited insight into the model's beliefs: we do not see the full range of cities the LLM believes are plausible, nor the variety of supporting information (e.g., that Paris hosts the French government).

We believe we can do better than this. As motivation, consider the following comment on Gödel's proof on the incompleteness of number theory.

> *Gödel had the insight that a statement of number theory could be about a statement of number theory (possibly even itself), if only numbers could somehow stand for statements.*
>
> Hofstadter (1979)

Gödel's key idea was that statements of number theory are expressive of much more than just integers. The same holds for strings: An answer string s generated by an LLM is expressive enough to describe a *distribution over* all answer strings the LLM could generate. We can therefore use a single string s to summarize the LLM's distribution $p_\theta (A \mid q)$ over responses A to a query q. We see this in the "self-reflective uncertainty" example of Figure 1: A single string conveys the relative degrees of belief in different cities, and includes additional details provided in the samples.
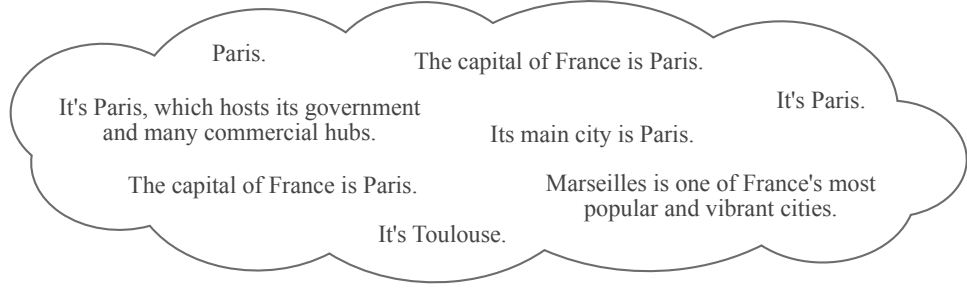
Our paper lays a foundation for this new avenue for uncertainty quantification. We define a metric that evaluates whether a given self-summarization technique faithfully represents an LLM's internal distributions over responses to queries. The underlying challenge here is to define a distance between a string and a distribution over strings that quantifies whether they both *"carry the same information"*, both in terms of facts and in terms of their relative likelihoods. We formulate the SelfReflect metric based on an information-theoretic perspective on predictive sufficiency in the string space. We verify that it discriminates good from bad (and almost-good) summaries of answer distributions on both free-form and closed-form question datasets, and that it agrees with human judgements, in both cases outperforming reasonable baselines such as LM judges and embedding distances.

---
[*]Equal contribution [1]Apple [2]Independent Researcher. Correspondence to: Michael Kirchhof <email see website>.

User query q: What is the main city of France?

LLM's internal distribution $p_\theta(A \mid q)$

> Paris.
>
> The capital of France is Paris.
>
> It's Paris, which hosts its government and many commercial hubs.
>
> Its main city is Paris.
>
> It's Paris.
>
> The capital of France is Paris.
>
> Marseilles is one of France's most popular and vibrant cities.
>
> It's Toulouse.

Normal (greedy) answer: 'The capital of France is Paris.'

Numerical uncertainty: ('The capital of France is Paris.', 75%)

Verbalized uncertainty: 'I'm very sure that the capital of France is Paris.'

Self-reflective uncertainty: 'I'm 75% sure that it's Paris, its capital and commercial hub, but it could also be Toulouse or Marseilles.'

Figure 1. LLMs have internal answer distributions about user queries. Rather than just sampling an output, possibly combined with a percentage, LLMs should generate a string that is self-reflective of their internal distribution, summarizing all possibilities and which they find the most likely.

Having defined the SelfReflect metric, we open up the possibility of evaluating whether LLMs can be made aware of their own internal answer distributions—that is, whether they can generate strings that explain their uncertainties. We find that such self-reflective outputs pose a hard challenge even to modern reasoning models. It is, however, possible to give insights into the internal answer distributions by explicitly sampling and then summarizing them. These findings mark but the start of enabling LLMs to output honest descriptions of their internal uncertainties. We expect that future advances along our SelfReflect metric can unlock more faithful and trustworthy LLM interactions.

2. Related Work

2.1. Uncertainty in LLMs

Most work on uncertainty in LLMs associates a single numerical expression of uncertainty to a specific string like the greedily decoded response. Since LLMs are, in essence, probabilistic next-token classifiers, one can attempt to read their uncertainty off their token logits (Aichberger et al., 2024; Fadeeva et al., 2023; Fomicheva et al., 2020; Malinin & Gales, 2020). These methods can be extended to longer LLM answers for example by searching for fact tokens and extracting their logits (Fadeeva et al., 2024) and made more human-readable by transforming the numeric uncertainty into a string like "I am very sure that..." (Lin et al., 2022; Yona et al., 2024). Still, these approaches quantify the uncertainty of only a single element of the LLM's internal distribution.

So how can the full uncertainty of the LLM's distribution be captured? Farquhar et al. (2024) cluster answers sampled from the LLM's internal distribution semantically and calculate an entropy over the clusters. This considers the full distribution over strings, but it still reduces the uncertainty to a single number and presents this number alongside a single string from the distribution. Moving towards richer uncertainty explications, Xu et al. (2024) generate multiple samples from an LLM, use GPT-4 to summarize the distribution of samples and train the LLM to output such summaries. Similarly, Yang et al. (2024b) train an LLM to output strings that delineate which facts it is uncertain about. This is arguably one of the richest ways to express an LLM's uncertainty. But both papers, focusing on the generation of summaries rather than on evaluation, use simple LM judges to rate the summary strings. As we show in Section 4.1, LM judges can not discern how faithfully a string reflects a distribution over strings beyond relatively simple good vs bad cases. Our SelfReflect gives a better-founded and more precise metric to compare whether a summary string contains the same information as the LLM's internal distribution, enabling to further develop this new avenue of LLM uncertainties.

2.2. Summarization

Testing whether a summary of a long document is *good* has a long history in natural language processing (NLP) (Zhang et al., 2024). Summaries are traditionally rated in terms of faithfulness to the long document, relevance of the chosen information, and fluency and coherence of their sentences (Särkkä & Solin, 2019), as rated by humans or recently by LM judges (Jain et al., 2023). In modern LLM-generated summaries, fluency and coherence are usually granted, so that the focus lays on the faithfulness and relevance of the
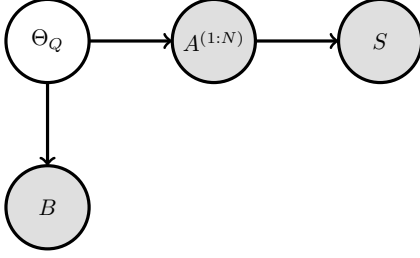
Figure 2. Graphical model for the sufficiency of the summary for the answer distribution that SelfReflect quantifies.

summary, in other words, whether it *contains the same information* as the long document. This fundamental question dates back to the Cloze test (Taylor, 1953). This test, originally designed for human language learners, masks out words from the long document and asks to fill them in. Summarization metrics like BLANC (Vasilyev et al., 2020) run this test twice, once when conditioning an NLP model on the summary and once without. If the summary contains correct information, the NLP model should fill in better words. The masked-out performance can be quantified either as an accuracy gain (Vasilyev et al., 2020) or, more softly, as a pseudo log-likelihood (Shin et al., 2019; Wang & Cho, 2019; Salazar et al., 2020; Kauf & Ivanova, 2023).

Since our SelfReflect metric also quantifies the quality of a summary, we base it off Cloze-like masked-out tasks. But there is a twist: The summary string s does not summarize another string but a *distribution over* strings $p_\theta(A \mid q)$. This means we must go beyond comparing s to a specific string $a \sim p_\theta(A \mid q)$, to quantifying how faithfully s represents the density over the string space that $p_\theta(A \mid q)$ defines, i.e., to all possible answers and how likely they are. To this end, we re-think masked-out tasks from the lens of sufficient statistics in the following section.

3. Distances between summary strings and distributions over strings

Our main challenge is to find a distance that quantifies the extent to which a summary string *carries the same information as* an LLM's internal answer distribution. We build a theoretical foundation for sufficient statistics in string spaces in Section 3.1 and develop the SelfReflect metric in Section 3.2.

3.1. Summaries as predictive sufficient statistics

Suppose we have an LLM (which we denote LLM$_\theta$), prompted with a random query Q. We posit that this puts us in a state Θ_Q, which allows us to sample random responses B. We are interested in summarizing this distribution over responses. Let $A^{(1:N)} := (A^{(1)}, \ldots, A^{(N)}) \in \mathcal{X}^N$ be a set of responses sampled from LLM$_\theta$, where $\mathcal{X}$ is the

space of finite strings.[1] Consider a summarization function $\psi : \mathcal{X}^N \longrightarrow \mathcal{X}$ that, given $A^{(1:N)}$, generates a summary $S := \psi(A^{(1:N)})$. What criteria should ψ satisfy if its summaries are to exactly capture LLM$_\theta$'s distribution over B?

Continuing the example from Figure 1, we can see that an ideal summary of $A^{(1:N)}$ should neither omit important details from the answer distribution nor add extra details. For example, a summary stating "The capital of France is Paris" would ignore the LLM's belief in Marseilles or Toulouse, whereas a summary stating "The capital of France is Paris but for a period in history, it was Orléans" would be adding unfaithful details. The same holds for the relative likelihood of answers: the ideal summary should state that the capital of France is most likely Paris, and not Toulouse or Marseilles, because this answer has a higher probability mass in the LLM's internal distribution. This indicates that an *ideal summary should capture exactly the same information about the answer distribution as that contained in the sampled answers*. We can formalize this in terms of mutual information,

Definition 3.1 (Ideal summary). An ideal summary S of answers $A^{(1:N)}$ of an LLM satisfies

$$\mathcal{I}\left\{A^{(1:N)} ; B\right\} = \mathcal{I}\left\{S ; B\right\} \tag{1}$$

Here, $\mathcal{I}\{Y ; Z\}$ denotes the mutual information between Y and Z. Intuitively, for any subsequent answer B from the LLM, the information about B contained in $A^{(1:N)}$ is exactly captured by S.

This definition is closely tied to the notion of predictive sufficiency (Lauritzen, 1974), whereby a statistic $T(X^{(1:N)})$ of observations $X^{(1:N)}$ is called sufficient if it satisfies $p(X \mid X^{(1:N)}) = p(X \mid T(X^{(1:N)}))$ for any subsequent observation X. In fact, we can reframe Definition 3.1 as follows:

Proposition 3.2 (Connection to predictive sufficiency). *For an ideal summary S of answers $A^{(1:N)}$,*

$$\mathcal{I}\left\{A^{(1:N)} ; B\right\} = \mathcal{I}\{S ; B\} \iff p\left(B \mid A^{(1:N)}\right) = p(B \mid S) \tag{2}$$

Intuitively, the ideal summary S is a predictive-sufficient statistic of the answers $A^{(1:N)}$ for B.

From Definition 3.1 and Proposition 3.2, we see that a measure of how much $p(B \mid A^{(1:N)})$ diverges from $p(B \mid S)$ would be a good metric for measuring how faithfully S reflects the sampled answers $A^{(1:N)}$. Towards this, we formulate a Cloze-task based on masked-token prediction that constitutes a simple yet equivalent characterization of the desired predictive sufficiency. Let B_i denote the ith word of B and let $B_{-i} := (B_j)_{j \neq i}$ denote all other words of the

---

[1]These N samples may be generated independently and identically to B, but we do not require this; for example, the distribution over subsequent answers could depend on the previous answers.

answer. We propose predicting the missing word B_i from the rest of the words B_{-i} with the extra context of either the sampled answers $A^{(1:N)}$ or their summary S. Identical behavior in this masked-token prediction task turns out to be equivalent to predictive sufficiency (and hence, Definition 3.1):

Proposition 3.3 (Informal; towards the SelfReflect metric). *For answers $A^{(1:N)}$ and their summary S, under mild conditions on all involved distributions and support of B, we have:*

$$p\left(B \mid A^{(1:N)}\right) = p\left(B \mid S\right) \Longleftrightarrow \text{for all masking indices } i,$$
$$p\left(B_i \mid A^{(1:N)}, B_{-i}\right) = p\left(B_i \mid S, B_{-i}\right)$$

Full details and proofs of Propositions 3.2 and 3.3 are given in Appendix A. Proposition 3.3 motivates us to measure the divergence between the distributions $p\left(B_i \mid S, B_{-i}\right)$ and $p\left(B_i \mid A^{(1:N)}, B_{-i}\right)$ as a tractable metric for the quality of a summary, forming the basis of the SelfReflect metric.

3.2. The SelfReflect metric

Proposition 3.3 tells us we can use a sequence of masked-out tasks to quantify whether a summary s contains the same information about LLM_θ's distribution $p_\theta\left(B \mid q\right)$ as a sequence of N samples from that distribution. We approximate this task using a second judge LLM, LLM_J, to estimate the conditional distribution over masked-out words. Intuitively, irrespective of whether we show the sampled answers or their ideal summary, a judge LLM should predict the same masked tokens.

Concretely, we sample a new response B at temperature 1 from LLM_θ, mask out one word B_i, and ask LLM_J to predict B_i given the remainder of the answer B_{-i}, the query q, and either the summary s or a sequence $a^{(1:N)}$ of N samples from $p_\theta\left(A^{(1:N)} \mid q\right)$, see Figure 3. This yields two distributions $p_J\left(B_i \mid Q = q, A^{(1:N)} = a^{(1:N)}, B_{-i} = b_{-i}\right)$ and $p_J\left(B_i \mid Q = q, S = s, B_{-i} = b_{-i}\right)$, over the vocabulary space of LLM_J which we compare using the 1-Wasserstein distance.[2] We marginalize over B and index i to satisfy the requirements of Proposition 3.3. To convert this into a general-purpose metric for a summarization strategy ψ, we take the expectation over queries and sampled responses:

$$m_{\text{SelfReflect}}(\psi) = \mathbb{E}_{Q, A^{(1:N)}, B, i}[\mathcal{W}^1(p_J(B_i \mid Q, \psi(Q), B_{-i}),$$
$$p_J(B_i \mid Q, A^{(1:N)}, B_{-i}))].$$

Here, ψ is any method that makes LLM_θ output a sum-

mary of its internal distribution in response to a query.[3] We estimate SelfReflect via Monte Carlo sampling with 1000 queries per dataset, a set of $N = 50$ samples $A^{(1:N)}$ per query, and masked-out tasks over $M = 50$ samples of B for all possible i. In Appendix B we show that these values converge to stable estimates. Literature notes that Cloze-like evaluations are often limited by synonyms (Kauf & Ivanova, 2023), so we post-hoc flatten p_J with $\tau = 5$ to put likelihood on broader synonyms. We quantitatively find this improves discriminability.

We explore different choices of LLM_J and find that SelfReflect is robust to the exact choice, see the quantitative results in Appendix C and the qualitative example in Appendix D. We find that Qwen 2.5 Instruct (Yang et al., 2024a) captures both textual details and the implicit relative certainties in summaries or concatenated samples in its context even when they are subtle. The 7B model provides results almost on par with the 72B model, so we choose it for efficiency. For further efficiency, we exclude stopwords from the masked tasks and use vLLM (Kwon et al., 2023) to parallelize and prefix-cache the computations. Overall, it takes 67 minutes to calculate the SelfReflect score over a dataset of 1000 queries and $N = M = 50$ answers per query on a node of 8 NVIDIA A100 GPUs.

4. Do SelfReflect scores work in practice?

We now verify that the SelfReflect metric works in practice, based on three pillars: An interventional study with known good and bad summaries on free-form questions (Section 4.1), a simplified study with closed-form multiple-choice QA answer distributions (Section 4.2), and a comparison to which summaries humans deem faithful (Section 4.3). In all studies, we compare our SelfReflect metric to several other baseline metrics, both from related fields and from ablating parts of SelfReflect.

Baselines. While developing SelfReflect, we experimented with approaches from various roots for comparing a summary string s to a set of strings $a^{(1:N)}$. First, *Summarization* treats $a^{(1:N)}$ as a single document and assesses the summary s in terms of consistency, fluency, relevance, and coherence (Jain et al., 2023). Second, *LM Judge* prompts LLM_J to rate how well s matches $a^{(1:N)}$, following the chain-of-thoughts prompt of Xu et al. (2024). Third, we turn to the neighboring field of calibration. Wang & Holmes (2024) argue that calibration can be seen as a distance to a centroid. We implement this in *Embedding* by comparing embeddings of s to $a^{(1:N)}$. Finally, for *Opt. transport* (Peyré et al., 2019), we let LLM_J split s into a "distribution" over atomic state-

---

[2]If LLM_J is a black-box model that only returns the top-predicted word, i.e., p_J are one-hot vectors, our 1-Wasserstein comparison simplifies into an accuracy that tests whether the two predicted words are equal.

[3]While the link to sufficiency only holds if ψ depends only on $a^{(1:N)}$, the metric is well-defined whether the summary generation involves taking samples in-between or generating a summary answer for q in other ways.
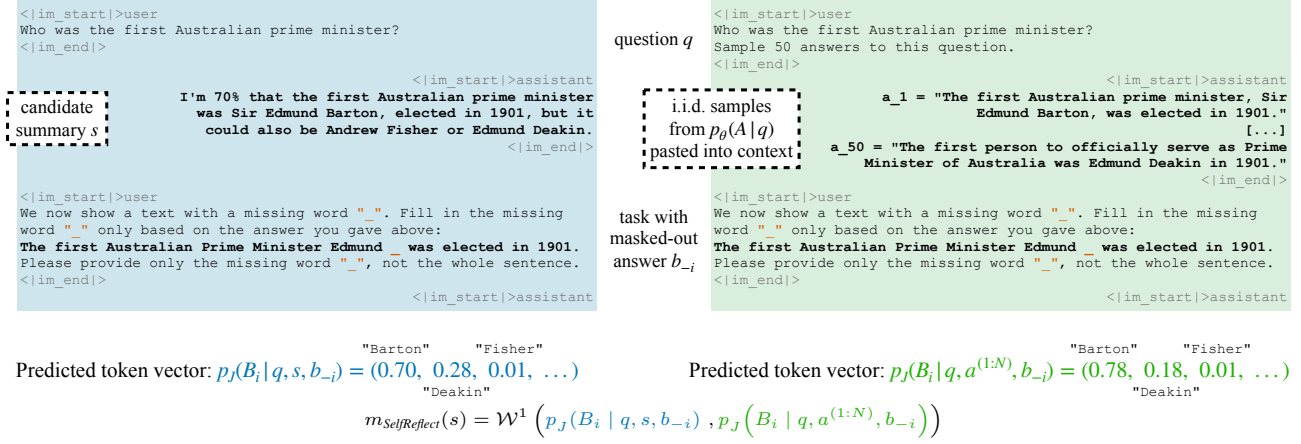
Figure 3. To test whether a summary string s contains the same information as a set of samples $a^{(1:N)}$, SelfReflect prompts an LLM twice. First, it provides the summary as context, then the concatenated samples. It then compares the resulting distributions via a masked-out task.

ments and likelihoods, compute a pairwise entailment matrix and return the Earth Mover's distance to $p_\theta(A \mid q)$.

Ablations. We also ablate key characteristics of SelfReflect. *SR-PMI* forgoes the masked-out task and directly compares the log likelihoods of full sample answers given either the summary or the samples in terms of pairwise mutual information; this can be seen as analogous to Proposition 3.2. *SR-sampling-free* uses the masked-out task, but compares the masked-out logits given the summary to predictions of LLM_θ given q itself, without sampling answers. *SR-P(True)* changes from a generative to a discriminative masked-out task, asking LLM_J whether several candidates words fit, given either the summary or the samples. We provide prompts and implementation details in Appendix E.

4.1. Study 1: Distinguishing good from bad and almost-good summaries

We first conduct an interventional study to test whether summaries that we know are good are judged as better (lower SelfReflect score) than summaries that we know are bad. To generate these summaries, we use Qwen2.5 7B Instruct to sample answers $A^{(1:50)}$ for 1,000 open-ended questions from the Natural Questions dataset (Kwiatkowski et al., 2019). We then prompt Gemini 2.0 Flash to generate *good* summaries (containing all possibilities, details, and relative likelihoods), and *bad* summaries (which alter key facts of the good summaries, but keep their remaining style). In Appendix F, we run the same experiments with human-written summaries, yielding similar results.

Table 1 shows that SelfReflect correctly discriminates good from bad in 99.8% of cases. But all other baseline metrics also score over 90%. So we make the task harder by comparing good to *almost-good* summaries, which only contain facts that are faithful to the answer distribution, but leave out

some possibilities and details that the good summary mentions. SelfReflect gives the good summary a better score than the almost-good summaries in 94.2% of all questions. Most other approaches, including the LM judge used in literature, can no longer distinguish these fine-grained quality differences.

To investigate this further, we test the metrics' ability to score the existence of details and the relative likelihoods of possible answers in isolation. First, we subsample questions where all answers in the answer distribution are the same, only varying in level of detail. In 98.06% of the cases, SelfReflect correctly gives a summary that is informative of all details mentioned in the answer distribution a better score than a summary string that is shortened to remove details. The second-best metric is again a SelfReflect ablation, *SR-sampling-free* with 82.58%. Second, we subsample questions whose answer distributions contain multiple possible answers. SelfReflect gives a verbalized summary of the form *"It is most likely... but could also be ..."* a better score than a summary that only mentions the majority answer in 95.04% of questions. Interestingly, *Embedding* consistently prefers the most likely answer, probably because it is on average closer to the centroid. When making this harder by comparing to a summary that does mention all possibilities, but not their relative likelihoods (*"It is ... or ... or ..."*, SelfReflect still gives the verbalized summary a better score in 74.38% of the cases, while all baselines except the $m_{\text{SR-P(True)}}$ ablation perform at or below random chance level. The same holds when using a numerical uncertainty summary (*"I'm X% sure that ..., but it could also be ... (Y% sure)"*) instead of a verbalized one, at 83.47%. These tests verify that SelfReflect can correctly discern whether a summary is faithful to an answer distribution both in terms of textual facts and relative frequencies.

Table 1. How well does SelfReflect, and other metrics, discriminate between good and bad summaries of answer distributions. For each column, the second summary lacks textual details or misrepresents relative probabilities. Mean $\pm$ 95% confidence interval.

Metric	Good summaries vs bad summaries	Good vs almost-good	Detailed vs truncated	Verbalized uncertainty vs only majority answer	Verbalized vs or-concatenated	Percentage vs or-concatenated
Summarization	$97.40\%_{\pm0.99\%}$	$38.70\%_{\pm3.02\%}$	$53.55\%_{\pm7.85\%}$	$11.57\%_{\pm5.70\%}$	$57.02\%_{\pm8.82\%}$	$65.29\%_{\pm8.48\%}$
LM Judge	$99.00\%_{\pm0.62\%}$	$47.90\%_{\pm3.10\%}$	$65.16\%_{\pm7.50\%}$	$24.79\%_{\pm7.69\%}$	$33.88\%_{\pm8.43\%}$	$38.02\%_{\pm8.65\%}$
Opt. Transport	$91.80\%_{\pm1.70\%}$	$53.90\%_{\pm3.09\%}$	$45.16\%_{\pm7.83\%}$	$48.76\%_{\pm8.91\%}$	$47.11\%_{\pm8.89\%}$	$70.25\%_{\pm8.15\%}$
Embedding	$94.40\%_{\pm1.43\%}$	$29.90\%_{\pm2.84\%}$	$47.10\%_{\pm7.86\%}$	$5.79\%_{\pm4.61\%}$	$46.28\%_{\pm8.88\%}$	$38.02\%_{\pm8.65\%}$
SR-PMI	$90.80\%_{\pm1.79\%}$	$45.70\%_{\pm3.09\%}$	$69.68\%_{\pm7.24\%}$	$23.97\%_{\pm7.61\%}$	$9.09\%_{\pm5.12\%}$	$16.53\%_{\pm6.62\%}$
SR-sampling-free	$96.20\%_{\pm1.19\%}$	$71.00\%_{\pm2.81\%}$	$82.58\%_{\pm5.97\%}$	$39.67\%_{\pm8.72\%}$	$29.75\%_{\pm8.15\%}$	$30.58\%_{\pm8.21\%}$
SR-P(True)	$55.80\%_{\pm3.08\%}$	$71.50\%_{\pm2.80\%}$	$62.58\%_{\pm7.62\%}$	$92.56\%_{\pm4.68\%}$	$69.42\%_{\pm8.21\%}$	$\mathbf{85.95\%}_{\pm6.19\%}$
SelfReflect	$\mathbf{99.80\%}_{\pm0.28\%}$	$\mathbf{94.20\%}_{\pm1.45\%}$	$\mathbf{98.06\%}_{\pm2.17\%}$	$\mathbf{95.04\%}_{\pm3.87\%}$	$\mathbf{74.38\%}_{\pm7.78\%}$	$83.47\%_{\pm6.62\%}$

Table 2. Correlations between how SelfReflect, and others, and a metric specialized for MMLU rank summaries. Mean $\pm$ 95% CI.

Metric	Rank Corr. per question	Rank Corr. over avg. of 1k questions
Summarization	$0.49_{\pm0.02}$	$0.80_{\pm0.00}$
LM Judge	$\mathbf{0.80}_{\pm0.02}$	$\mathbf{1.00}_{\pm0.00}$
Opt. Transport	$0.63_{\pm0.02}$	$0.80_{\pm0.00}$
Embedding	$0.29_{\pm0.03}$	$0.18_{\pm0.02}$
SR-PMI	$-0.03_{\pm0.03}$	$-0.20_{\pm0.00}$
SR-sampling-free	$0.57_{\pm0.03}$	$0.83_{\pm0.00}$
SR-P(True)	$0.66_{\pm0.03}$	$\mathbf{1.00}_{\pm0.00}$
SelfReflect	$0.66_{\pm0.03}$	$\mathbf{1.00}_{\pm0.00}$

4.2. Study 2: Distances of multiple-choice distributions

Next, we investigate SelfReflect in a narrower setup. We generate 1,000 answer distributions for MMLU (Hendrycks et al., 2021), a multiple-choice dataset with choices A, B, C, and D for each question. To give a spectrum of different-quality summaries, we create summaries of the form *"The answer is most likely C (54% sure), but it could also be B (32% sure) or A (14% sure)."* that either match the true ratio of answers, mention the most likely answer only, are overconfident, or give random percentages. These simple summary strings allow testing how well SelfReflect and the baseline metrics capture distributional faithfulness. As a reference-metric for faithfulness in this narrow setup, we compute the true Wasserstein distance between the distribution described in the summary and that of the test-set answers. We then calculate the correlation of the ranks that SelfReflect assigns to the summaries of a question and that which the reference metric assigns. In order to broaden our analysis, the answer-distributions are generated with a different LLM than in the previous section. We use Gemma 3 12B (non-Instruct) (Gemma Team et al., 2025), whose majority/greedy answer has a 71% accuracy on MMLU.

Table 2 shows that most metrics have a positive rank correlation with the reference metric. The LM judge metric even outperforms SelfReflect, indicating that SelfReflect may be slightly noisy on individual questions when summaries con-

tain exact probabilities. However, as soon as we compute the average score across all 1000 questions, as it will later be used in the benchmark, SelfReflect, like LM Judge and the P(True) ablation, achieves a perfect rank correlation with the reference metric, i.e., ranks the different types of summaries the same way as the reference metric for this special case would.

4.3. Study 3: Do the ratings align with human ratings?

Finally, we assess whether SelfReflect scores are aligned with human judgements. We conduct a user study using 200 open-ended questions from the TriviaQA dataset (Joshi et al., 2017). For each question, we generate ten sample responses using Phi-4 (Abdin et al., 2024), and four summaries: a *good* summary and a *bad* summary, generated using Gemini 2.0 Flash as in Section 4.1; a *greedy* summary, i.e., the greedy response of Phi-4; and a Chain of Thought (*CoT*) summary, using Phi-4 to reason about possible answers and then summarize its reasoning. Note that the greedy and CoT summaries are not based on the actual samples. All prompts are provided in Appendix E.

Raters were shown the question, the ten sample answers, and two of the summaries, and asked to choose which best summarized the set of samples. Each question/summary combination was evaluated by 5 raters. To assess agreement between human raters, we calculate Krippendorff's α. Alternative agreement metrics such as Cohen's kappa or Fleiss' kappa are not appropriate here since each rater only rates a subset of the combinations. We then calculate Krippendorff's α between the majority human preference and that of SelfReflect and other scores. Further details are in Appendix G.

As we see from Table 3, SelfReflect has the highest overall alignment with the majority human judgement ($\alpha = 0.690$). This is close to the inter-human alignment ($\alpha = 0.723$) and significantly higher than any of the competing methods or ablations. Looking into the individual summary types, we see all metrics other than SR-P(True) have good alignment with humans on the *bad* vs *good*, *bad* vs *greedy*, and *bad*

Table 3. Agreement of metrics with human preference (consensus over five raters) on a pairwise summary preference task, using Krippendorff's α (values in [-1, 1]; positive numbers indicate agreement). Also shown is Krippendorff's α between individual human raters. Mean $\pm$ 95% CI.

	all	bad vs good	bad vs greedy	bad vs CoT	good vs greedy	good vs CoT	greedy vs CoT
Summarization	$0.480_{\pm 0.050}$	$0.950_{\pm 0.046}$	$0.910_{\pm 0.050}$	$\mathbf{0.940}_{\pm 0.046}$	$-0.211_{\pm 0.156}$	$-0.067_{\pm 0.135}$	$0.260_{\pm 0.121}$
LM Judge	$0.517_{\pm 0.046}$	$0.940_{\pm 0.048}$	$\mathbf{0.920}_{\pm 0.058}$	$0.930_{\pm 0.046}$	$-0.063_{\pm 0.152}$	$-0.015_{\pm 0.151}$	$0.267_{\pm 0.128}$
Opt. Transport	$0.487_{\pm 0.047}$	$0.850_{\pm 0.076}$	$0.779_{\pm 0.085}$	$0.679_{\pm 0.104}$	$0.098_{\pm 0.155}$	$0.265_{\pm 0.132}$	$0.191_{\pm 0.146}$
Embeddings	$0.435_{\pm 0.047}$	$0.750_{\pm 0.081}$	$0.799_{\pm 0.087}$	$0.477_{\pm 0.125}$	$-0.363_{\pm 0.136}$	$0.331_{\pm 0.135}$	$\mathbf{0.490}_{\pm 0.121}$
SR-PMI	$0.436_{\pm 0.053}$	$0.820_{\pm 0.081}$	$0.890_{\pm 0.067}$	$0.769_{\pm 0.080}$	$-0.246_{\pm 0.156}$	$0.029_{\pm 0.147}$	$0.246_{\pm 0.114}$
SR-sampling-free	$0.530_{\pm 0.045}$	$0.829_{\pm 0.076}$	$0.870_{\pm 0.071}$	$0.799_{\pm 0.080}$	$0.025_{\pm 0.143}$	$0.241_{\pm 0.131}$	$0.340_{\pm 0.141}$
SR-P(True)	$-0.032_{\pm 0.052}$	$-0.029_{\pm 0.138}$	$-0.335_{\pm 0.124}$	$-0.474_{\pm 0.120}$	$0.311_{\pm 0.147}$	$0.409_{\pm 0.125}$	$-0.024_{\pm 0.143}$
SelfReflect	$\mathbf{0.690}_{\pm 0.036}$	$\mathbf{0.990}_{\pm 0.015}$	$0.850_{\pm 0.066}$	$0.850_{\pm 0.070}$	$\mathbf{0.489}_{\pm 0.131}$	$\mathbf{0.599}_{\pm 0.103}$	$0.329_{\pm 0.125}$
Human vs human	$0.723_{\pm 0.027}$	$0.988_{\pm 0.013}$	$0.906_{\pm 0.035}$	$0.871_{\pm 0.048}$	$0.441_{\pm 0.075}$	$0.636_{\pm 0.064}$	$0.452_{\pm 0.069}$

vs *CoT* comparisons. However, the other metrics show poor agreement with humans on the more nuanced *good* vs *greedy* and *good* vs *CoT*. For all pairs of summary type, SelfReflect is close to inter-human agreement and either the most aligned with the majority human preference, or has overlapping 95% confidence intervals with the most aligned metric.

5. Can LLMs generate self-reflective responses?

Now that we have a metric to judge how well summaries summarize the distribution of LLM answers, we explore the performance of different summarization methods. We distinguish two broad categories of methods: A) *Sample & summarize*: draw multiple independent samples from the model, and then summarize the resulting distribution, B) *Single-decoding*: methods which utilize only one decoding, without explicitly elicitating intermediate samples. Of particular interest is whether any of the single-decoding methods are able to match the performance of the multi-sample methods. We consider three single-decoding methods: a) *Greedy*: a baseline simply using a greedy-decoding answer to the question as the summary; b) *Basic*: a prompt asking the LLM for a summary of all possible answer options; c) *CoT*: a prompt inducing chain-of-thoughts reasoning about the possible answers and then summarizing them. Our intuition behind *Basic* and *CoT* is that, by encouraging responses that include multiple possible answers, we may be able to recover a reasonable approximation to the true distribution. We evaluate these summarization methods on 1000 randomly chosen questions from each of the three datasets: Natural Questions, SimpleQA and TriviaQA. We use the same LLM to sample the answers to the question and generate the summaries in order to assess whether LLMs can access and describe their *own* internal distributions. We provide more details in Appendix H.

As we see in Table 4, *Sample & summarize* is able to con-

sistently create summaries that reflect the model's internal uncertainty better than the *Greedy* answer. In fact, its score matches that of humans asked to summarize samples from an LLM distribution, with humans achieving $90 \cdot 10^{-3}$ when summarizing Qwen 2.5 72B Instruct answer distributions and *Sample & summarize* achieving $88 \cdot 10^{-3}$ on the data-split of Appendix F. However, it is of particular interest if we can generate such self-reflective outputs without needing to sample in-between, which would improve runtime and be more elegant. Single-decoding methods that implement this are, however, not able to consistently out-perform the *Greedy* approach, corroborating that the LLMs are not able to fully verbalize their own uncertainty by themselves, despite our best efforts to optimize the prompts. This is because *Greedy* is in fact a strong baseline: On questions where a model has a unimodal distribution on a specific answer, *Greedy* is in fact the best possible summary of this distribution and in turn achieves a competitive SelfReflect score. Table 4 reveals that this makes *Greedy*'s average score a strong baseline particularly on datasets with high $p_\theta (A \mid q)$ *unimodal* percentages.

5.1. Which answers does Chain-of-Thoughts consider?

We might expect that by considering and summarizing multiple possible options, *CoT* can capture an LLM's distribution better than the single *Greedy* response. However, in the previous section we found this not to be the case. To understand why, we compare *CoT* summaries to samples from the answer distribution, focusing on a single large-scale model, Qwen2.5 72B Instruct. To explore whether *CoT* correctly captures the spread of the answer distribution, i.e., whether it focuses on a single answer when the true distribution is unimodal and includes multiple options when the true distribution is multimodal, we let Gemini 2.0 Flash classify whether the *CoT* summaries and $a^{(1:N)}$ are certain (only mentioning one answer option) or uncertain (mentioning semantically different options).

The results are shown in Figure 4. We can see that *CoT* is

Table 4. SelfReflect score ↓ ($\times 10^{-3}$, rounded for readability) averaged across TriviaQA, NQ, and SimpleQA. Per-dataset results are in Appendix H. The results in small font are relative to *Greedy*. $p_\theta(A|q)$ *unimodal* is the proportion of questions for which the LLM always gives the same answer.

| Model	$p_\theta(A \mid q)$ unimodal	Single-decoding methods			Sample & summarize	
		Greedy	Basic	CoT	$N = 10$	$N = 20$
Qwen2.5 0.5B Instruct (Yang et al., 2024a)	7%	96	95$_{-1}$	94$_{-2}$	96$_{-0}$	96$_{-0}$
Qwen2.5 1.5B Instruct (Yang et al., 2024a)	17%	94	94$_{-0}$	92$_{-2}$	87$_{-7}$	87$_{-7}$
Qwen2.5 3B Instruct (Yang et al., 2024a)	27%	97	99$_{+2}$	99$_{+2}$	91$_{-6}$	89$_{-8}$
Qwen2.5 7B Instruct (Yang et al., 2024a)	36%	96	99$_{+3}$	101$_{+5}$	91$_{-5}$	90$_{-6}$
Qwen2.5 14B Instruct (Yang et al., 2024a)	52%	92	97$_{+5}$	99$_{+7}$	86$_{-6}$	85$_{-7}$
Qwen2.5 32B Instruct (Yang et al., 2024a)	49%	96	102$_{+6}$	105$_{+9}$	91$_{-5}$	91$_{-0}$
Qwen2.5 72B Instruct (Yang et al., 2024a)	50%	91	94$_{+3}$	96$_{+5}$	85$_{-6}$	84$_{-7}$
Phi 4 14B (Abdin et al., 2024)	36%	92	92$_{-0}$	93$_{+1}$	85$_{-7}$	84$_{-8}$
Ministral 8B Instruct 2410 (Jiang et al., 2024)	25%	107	106$_{-1}$	105$_{-2}$	101$_{-6}$	100$_{-7}$
Llama 3.1 70B Instruct (Meta AI, 2024a)	51%	92	92$_{-0}$	95$_{+3}$	87$_{-5}$	87$_{-5}$
Llama 3.3 70B Instruct (Meta AI, 2024b)	63%	94	98$_{+4}$	104$_{+10}$	89$_{-5}$	88$_{-6}$
Llama 4 Scout 17B 16e Instruct (Meta AI, 2025)	53%	91	96$_{+5}$	101$_{+10}$	88$_{-3}$	87$_{-4}$
Gemma 3 1B Instruct (Gemma Team et al., 2025)	26%	116	129$_{+13}$	129$_{+13}$	117$_{+1}$	111$_{-5}$
Gemma 3 4B Instruct (Gemma Team et al., 2025)	52%	108	124$_{+16}$	128$_{+20}$	101$_{-7}$	100$_{-8}$
Gemma 3 12B Instruct (Gemma Team et al., 2025)	59%	105	116$_{+11}$	121$_{+16}$	102$_{-3}$	101$_{-4}$
Gemma 3 27B Instruct (Gemma Team et al., 2025)	71%	100	113$_{+13}$	120$_{+20}$	97$_{-3}$	96$_{-4}$
Generation time (seconds)		1.56	1.59	2.48	3.65	4.50
Length (characters)		104.79	195.12	303.09	174.70	219.22
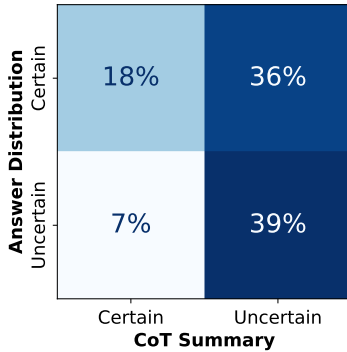
Figure 4. Confusion matrix between the uncertainty of Qwen2.5 72B Instruct answer distributions and *CoT* summaries. Details can be found in Appendix I.

often *underconfident*. In 36% of the questions, its summary is uncertain even when the answer distribution samples are not, meaning it suggests multiple answers options that do not have high probability under the true distribution. The *Greedy* method, by contrast, would give the ideal summary here, mentioning the only real answer option. But *Greedy* is *overconfident* in other cases: For 46% of the questions, there are multiple options in the answer distribution, but *Greedy* still collapses them to a single one (leading to *Greedy* underperforming *Sample & Summarize* in Table 4). A balance is clearly needed, but not trivial.

To study *CoT*-like summaries further, we turn to reasoning models, trained with reinforcement learning with verifi-

able rewards (RLVR) (DeepSeek-AI et al., 2025). Asking them to output a summary of all possibilities, i.e., the *Basic* prompt, is relatively similar to *CoT* on RLHF models, since it automatically invokes a reasoning block. Table 5 shows that RLVR models do not perform any better than RLHF models in Table 4. Qualitatively, the summaries produced by RLVR models with *Basic* prompts are similar to the ones produced by RLHF models with *CoT* prompts. Generating self-reflective summaries that are faithful to the model's internal uncertainty thus remains challenging.

But what awaits us at the end of this road? With improving faithfulness to the subjective distribution, and LLMs' subjective distributions becoming more aligned with the objective ground-truth (-distributions), we can hope to cover ground-truth answers more often than with greedy answers. Following the best practices of Santilli et al. (2025), we measure the RougeL-Recall on Natural Questions' short answers, i.e., the longest substring of the true answer that appears in a summary, as percentage of the true answer's length. We find that *Greedy* answers have an average overlap of 59.5% with the true answers. *Basic* summaries have 62.0%, *CoT* summaries 64.0%, and *Sample & Summarize* summaries 65.6%. An LM Judge-based evaluation shows the same trend, rating that 71.3%, 72.2%, 74.1%, and 76.0% of the summaries include the true answer. In other words, summaries of the LLM's internal distributions don't just hallucinate extra possibilities but actually cover the true answer more often.

Table 5. SelfReflect score $\downarrow$ ($\times 10^{-3}$) of RLVR models averaged over TriviaQA & SimpleQA. *Greedy* is generated w/o reasoning. *Basic* and *Sample & Summarize* reason and then output a summary.

Model	Single-decoding methods		Sample & Summarize	
	Greedy	Basic	$N=10$	$N=20$
QwQ 32B (Qwen Team, 2025b)	96	106_{+10}	92_{-4}	91_{-5}
DeepSeek R1 Distill Qwen 2.5 32B (DeepSeek-AI et al., 2025)	96	109_{+13}	92_{-4}	91_{-5}
Qwen3 32B (Reasoning enabled) (Qwen Team, 2025a)	93	96_{+3}	86_{-7}	85_{-8}
Qwen3 8B (Reasoning enabled) (Qwen Team, 2025a)	103	106_{+3}	92_{-11}	91_{-12}
Generation time (seconds)	1.96	3.60	6.99	8.57
Length (characters)	107.56	224.98	287.31	350.98

6. Discussion

We present SelfReflect, a metric that judges how faithfully a single string represents a distribution over output strings. SelfReflect is intended to guide the field towards developing methods to make LLMs honestly describe all possible answers to a question. We have seen in our benchmark that this is a hard task, but a solution to this problem would be a fundamental building block in many applications: Describing internal LLM distributions in a string provides a human-interpretable measure of model uncertainty, which can be useful in building appropriate trust in the LLM's outputs. Extracting all output possibilities could also drive development of conformal approaches, which are popular for classification but less explored for LLMs where the output possibilities are not immediately available. Finally, an accurate description of a distribution can also be recast into a numeric uncertainty value.

To outline the limitations of our work, we note that 1-Wasserstein-based SelfReflect scores are not directly interpretable without baselines. A simplified version, like the percentage of equal top-predicted words using either summary or answer samples, would give more standardized values in [0,1]. However, we found that such an approach is less sensitive to differences in good vs *almost*-good summaries. Second, we repeat that the faithfulness we measure is with respect to an LLM's *subjective* uncertainty. We intentionally did not develop SelfReflect to quantify objective truthfulness, with the outlook that larger LLMs approximate their training datasets better and better, such that more faithful summaries of subjective uncertainties will ultimately lead to better objective uncertainties.

References

Abdin, M., Aneja, J., Behl, H., Bubeck, S., Eldan, R., Gunasekar, S., Harrison, M., Hewett, R. J., Javaheripi, M., Kauffmann, P., et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

Aichberger, L., Schweighofer, K., and Hochreiter, S. Re-thinking uncertainty estimation in natural language generation. *arXiv preprint arXiv:2412.15176*, 2024.

Bernardo, J. M. and Smith, A. F. *Bayesian theory*, volume 405. John Wiley & Sons, 2009.

Cover, T. M. *Elements of information theory*. John Wiley & Sons, 1999.

DeepSeek-AI, Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z. F., Gou, Z., Shao, Z., Li, Z., Gao, Z., Liu, A., Xue, B., Wang, B., Wu, B., Feng, B., Lu, C., Zhao, C., Deng, C., Zhang, C., Ruan, C., Dai, D., Chen, D., Ji, D., Li, E., Lin, F., Dai, F., Luo, F., Hao, G., Chen, G., Li, G., Zhang, H., Bao, H., Xu, H., Wang, H., Ding, H., Xin, H., Gao, H., Qu, H., Li, H., Guo, J., Li, J., Wang, J., Chen, J., Yuan, J., Qiu, J., Li, J., Cai, J. L., Ni, J., Liang, J., Chen, J., Dong, K., Hu, K., Gao, K., Guan, K., Huang, K., Yu, K., Wang, L., Zhang, L., Zhao, L., Wang, L., Zhang, L., Xu, L., Xia, L., Zhang, M., Zhang, M., Tang, M., Li, M., Wang, M., Li, M., Tian, N., Huang, P., Zhang, P., Wang, Q., Chen, Q., Du, Q., Ge, R., Zhang, R., Pan, R., Wang, R., Chen, R. J., Jin, R. L., Chen, R., Lu, S., Zhou, S., Chen, S., Ye, S., Wang, S., Yu, S., Zhou, S., Pan, S., Li, S. S., Zhou, S., Wu, S., Ye, S., Yun, T., Pei, T., Sun, T., Wang, T., Zeng, W., Zhao, W., Liu, W., Liang, W., Gao, W., Yu, W., Zhang, W., Xiao, W. L., An, W., Liu, X., Wang, X., Chen, X., Nie, X., Cheng, X., Liu, X., Xie, X., Liu, X., Yang, X., Li, X., Su, X., Lin, X., Li, X. Q., Jin, X., Shen, X., Chen, X., Sun, X., Wang, X., Song, X., Zhou, X., Wang, X., Shan, X., Li, Y. K., Wang, Y. Q., Wei, Y. X., Zhang, Y., Xu, Y., Li, Y., Zhao, Y., Sun, Y., Wang, Y., Yu, Y., Zhang, Y., Shi, Y., Xiong, Y., He, Y., Piao, Y., Wang, Y., Tan, Y., Ma, Y., Liu, Y., Guo, Y., Ou, Y., Wang, Y., Gong, Y., Zou, Y., He, Y., Xiong, Y., Luo, Y., You, Y., Liu, Y., Zhou, Y., Zhu, Y. X., Xu, Y., Huang, Y., Li, Y., Zheng, Y., Zhu, Y., Ma, Y., Tang, Y., Zha, Y., Yan, Y., Ren, Z. Z., Ren, Z., Sha, Z., Fu, Z., Xu, Z., Xie, Z., Zhang, Z., Hao, Z., Ma, Z., Yan, Z., Wu, Z., Gu, Z., Zhu, Z., Liu, Z., Li, Z., Xie, Z., Song, Z., Pan, Z., Huang, Z., Xu, Z., Zhang, Z., and Zhang, Z. Deepseek-r1: Incentivizing rea-

soning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Enevoldsen, K., Chung, I., Kerboua, I., Kardos, M., Mathur, A., Stap, D., Gala, J., Siblini, W., Krzemiński, D., Winata, G. I., Sturua, S., Utpala, S., Ciancone, M., Schaeffer, M., Sequeira, G., Misra, D., Dhakal, S., Rystrøm, J., Solomatin, R., Ömer Çağatan, Kundu, A., Bernstorff, M., Xiao, S., Sukhlecha, A., Pahwa, B., Poświata, R., GV, K. K., Ashraf, S., Auras, D., Plüster, B., Harries, J. P., Magne, L., Mohr, I., Hendriksen, M., Zhu, D., Gisserot-Boukhlef, H., Aarsen, T., Kostkan, J., Wojtasik, K., Lee, T., Šuppa, M., Zhang, C., Rocca, R., Hamdy, M., Michail, A., Yang, J., Faysse, M., Vatolin, A., Thakur, N., Dey, M., Vasani, D., Chitale, P., Tedeschi, S., Tai, N., Snegirev, A., Günther, M., Xia, M., Shi, W., Lù, X. H., Clive, J., Krishnakumar, G., Maksimova, A., Wehrli, S., Tikhonova, M., Panchal, H., Abramov, A., Ostendorff, M., Liu, Z., Clematide, S., Miranda, L. J., Fenogenova, A., Song, G., Safi, R. B., Li, W.-D., Borghini, A., Cassano, F., Su, H., Lin, J., Yen, H., Hansen, L., Hooker, S., Xiao, C., Adlakha, V., Weller, O., Reddy, S., and Muennighoff, N. MMTEB: Massive multilingual text embedding benchmark. *arXiv preprint arXiv:2502.13595*, 2025.

Fadeeva, E., Vashurin, R., Tsvigun, A., Vazhentsev, A., Petrakov, S., Fedyanin, K., Vasilev, D., Goncharova, E., Panchenko, A., Panov, M., Baldwin, T., and Shelmanov, A. LM-polygraph: Uncertainty estimation for language models. In Feng, Y. and Lefever, E. (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, December 2023.

Fadeeva, E., Rubashevskii, A., Shelmanov, A., Petrakov, S., Li, H., Mubarak, H., Tsymbalov, E., Kuzmin, G., Panchenko, A., Baldwin, T., Nakov, P., and Panov, M. Fact-checking the output of large language models via token-level uncertainty quantification. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, August 2024.

Farquhar, S., Kossen, J., Kuhn, L., and Gal, Y. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630, 2024.

Flamary, R., Courty, N., Gramfort, A., Alaya, M. Z., Boisbunon, A., Chambon, S., Chapel, L., Corenflos, A., Fatras, K., Fournier, N., Gautheron, L., Gayraud, N. T., Janati, H., Rakotomamonjy, A., Redko, I., Rolet, A., Schutz, A., Seguy, V., Sutherland, D. J., Tavenard, R., Tong, A., and Vayer, T. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021. URL http://jmlr.org/papers/v22/20-451.html.

Fomicheva, M., Sun, S., Yankovskaya, L., Blain, F., Guzmán, F., Fishel, M., Aletras, N., Chaudhary, V., and Specia, L. Unsupervised quality estimation for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:539–555, 2020.

Gemma Team, Kamath, A., Ferret, J., Pathak, S., Vieillard, N., Merhej, R., Perrin, S., Matejovicova, T., Ramé, A., Rivière, M., et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

Hofstadter, D. R. *Godel, Escher, Bach: An Eternal Golden Braid*. Basic Books, Hassocks, England, 1979.

Jain, S., Keshava, V., Sathyendra, S. M., Fernandes, P., Liu, P., Neubig, G., and Zhou, C. Multi-dimensional evaluation of text summarization with in-context learning. *arXiv preprint arXiv:2306.01200*, 2023.

Jiang, A., Chahine, A. A., Sablayrolles, A., Tacnet, A., Boissonnet, A., Kothari, A., Héliou, A., Lo, A., Peronnin, A., Meunier, A., Roux, A., Faure, A., Paul, A., Darcet, A., Mensch, A., Herblin-Stoop, A., Garreau, A., Birky, A., Sooriyarachchi, A., Rozière, B., Conklin, B., Bouillon, B., de Beauregard, B. S., Rambaud, C., Feldman, C., de Freminville, C., Mauro, C., Yeh, C.-K., Bamford, C., Auguy, C., Heintz, C., Dubois, C., Chaplot, D. S., Casas, D. L., Costa, D., Arcelin, E., Hanna, E. B., Metzger, E., Autran, F. O., Lesage, F., Gourdel, G., Blanchet, G., Vidal, G. D., Lengyel, G. M., Bour, G., Lample, G., Denis, G., Rajaona, H., Jaju, H., Mack, I., Mathew, I., Delignon, J.-M., Facchetti, J., Chudnovsky, J., Studnia, J., Murke, J., Khandelwal, K., Chiu, K., Riera, K., Blier, L., Suslian, L., Deschaseaux, L., Martin, L., Ternon, L., Saulnier, L., Lavaud, L. R., Yang, S., Jennings, M., Pellat, M., Torelli, M., Janiewicz, M., Felardos, M., Darin, M., Hoff, M., Seznec, M., Kenyon, M. J., Derwiche, N., Zaragoza, N. C., Faurie, N., Moreau, N., Schuhl, N., Raghuraman, N., Muhs, N., de Garrigues, O., Rozé, P., Wang, P., von Platen, P., Jacob, P., Buche, P., Muddireddy, P. R., Savas, P., Stock, P., Agrawal, P., de Peretti, R., Sauvestre, R., Sinthe, R., Soletskyi, R., Vaze, S., Subramanian, S., Garg, S., Ghosh, S., Regnier, S., Antoniak, S., Scao, T. L., Gervet, T., Schueller, T., Lavril, T., Wang, T., Lacroix, T., Nemychnikova, V., Shang, W., Sayed, W. E., and Marshall, W. Un ministral, des ministraux. 2024. URL https://mistral.ai/news/ministraux?utm_source=tldrai.

Joshi, M., Choi, E., Weld, D. S., and Zettlemoyer, L. Triviaqa: A large scale distantly supervised challenge dataset

for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, Vancouver, Canada, July 2017. Association for Computational Linguistics.

Kauf, C. and Ivanova, A. A better way to do masked language model scoring. In Rogers, A., Boyd-Graber, J., and Okazaki, N. (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, July 2023.

Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Devlin, J., Lee, K., Toutanova, K., Jones, L., Kelcey, M., Chang, M.-W., Dai, A. M., Uszkoreit, J., Le, Q., and Petrov, S. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7, 2019.

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J. E., Zhang, H., and Stoica, I. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Lauritzen, S. L. Sufficiency, prediction and extreme models. *Scandinavian Journal of Statistics*, pp. 128–134, 1974.

Li, Z., Zhang, X., Zhang, Y., Long, D., Xie, P., and Zhang, M. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*, 2023.

Lin, S., Hilton, J., and Evans, O. Teaching models to express their uncertainty in words. *arXiv preprint arXiv:2205.14334*, 2022.

Malinin, A. and Gales, M. Uncertainty estimation in autoregressive structured prediction. *arXiv preprint arXiv:2002.07650*, 2020.

Meta AI. Llama 3.1 model card. 2024a. URL https://www.llama.com/docs/model-cards-and-prompt-formats/llama3_1/.

Meta AI. Llama 3.3 model card. 2024b. URL https://www.llama.com/docs/model-cards-and-prompt-formats/llama3_3/.

Meta AI. Llama 4 model card. 2025. URL https://www.llama.com/docs/model-cards-and-prompt-formats/llama4/.

Peyré, G., Cuturi, M., et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

Qwen Team. Qwen3, April 2025a. URL https://qwenlm.github.io/blog/qwen3/.

Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025b. URL https://qwenlm.github.io/blog/qwq-32b/.

Sahoo, P., Singh, A. K., Saha, S., Jain, V., Mondal, S., and Chadha, A. A systematic survey of prompt engineering in large language models: Techniques and applications. *arXiv preprint arXiv:2402.07927*, 2024.

Salazar, J., Liang, D., Nguyen, T. Q., and Kirchhoff, K. Masked language model scoring. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, July 2020.

Santilli, A., Golinski, A., Kirchhof, M., Danieli, F., Blaas, A., Xiong, M., Zappella, L., and Williamson, S. Revisiting uncertainty quantification evaluation in language models: Spurious interactions with response length bias results. *arXiv preprint arXiv:2504.13677*, 2025.

Särkkä, S. and Solin, A. *Applied stochastic differential equations*, volume 10. Cambridge University Press, 2019.

Shin, J., Lee, Y., and Jung, K. Effective sentence scoring method using bert for speech recognition. In Lee, W. S. and Suzuki, T. (eds.), *Proceedings of The Eleventh Asian Conference on Machine Learning*, volume 101 of *Proceedings of Machine Learning Research*, pp. 1081–1093. PMLR, 17–19 Nov 2019. URL https://proceedings.mlr.press/v101/shin19a.html.

Taylor, W. L. Cloze procedure: A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433, 1953.

Vasilyev, O., Dharnidharka, V., and Bohannon, J. Fill in the BLANC: Human-free quality estimation of document summaries. In Eger, S., Gao, Y., Peyrard, M., Zhao, W., and Hovy, E. (eds.), *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pp. 11–20, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.eval4nlp-1.2. URL https://aclanthology.org/2020.eval4nlp-1.2/.

Wang, A. and Cho, K. BERT has a mouth, and it must speak: BERT as a Markov random field language model. In Bosselut, A., Celikyilmaz, A., Ghazvininejad, M., Iyer, S., Khandelwal, U., Rashkin, H., and Wolf, T. (eds.), *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, June 2019.

Wang, Z. and Holmes, C. On subjective uncertainty quantification and calibration in natural language generation. *arXiv preprint arXiv:2406.05213*, 2024.

Xu, T., Wu, S., Diao, S., Liu, X., Wang, X., Chen, Y., and Gao, J. Sayself: Teaching llms to express confidence with self-reflective rationales. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 5985–5998, 2024.

Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., Lu, K., Bao, K., Yang, K., Yu, L., Li, M., Xue, M., Zhang, P., Zhu, Q., Men, R., Lin, R., Li, T., Xia, T., Ren, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Wan, Y., Liu, Y., Cui, Z., Zhang, Z., and Qiu, Z. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.

Yang, R., Zhang, C., Zhang, Z., Huang, X., Yang, S., Collier, N., Yu, D., and Yang, D. Logu: Long-form generation with uncertainty expressions. *arXiv preprint arXiv:2410.14309*, 2024b.

Yona, G., Aharoni, R., and Geva, M. Can large language models faithfully express their intrinsic uncertainty in words? *arXiv preprint arXiv:2405.16908*, 2024.

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. SelfReflect and predictive sufficiency: propositions and proofs

In this appendix, we provide details of the propositions from the main text and their proofs. We begin with the definition of predictive sufficiency and provide a proof of its two equivalent characterizations in the context of the SelfReflect metric. We then prove an equivalence between solving the masked-token prediction task of the SelfReflect metric and the desired predictive sufficiency of the summary, providing a theoretical foundation for the design of the SelfReflect metric.
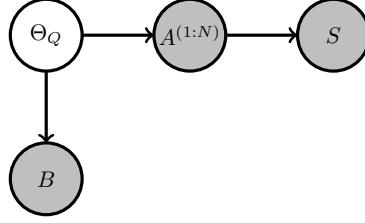

Figure 5. The graphical model for the setting of SelfReflect metric. The figure is reproduced from Figure 2 of the main text for the sake of better readability of the formalization that follows.

A.1. Setup, notations, and assumptions

Recall that prompting a given LLM with question Q puts it in state Θ_Q, from which we sample N answers $A^{(1:N)}$. A summarization mechanism function ψ generates the summary of these answers as $S = \psi\left(A^{(1:N)}\right)$. For developing the SelfReflect metric, we generate another sample B from the same state Θ_Q and require an ideal summary S to capture all the information about B that is captured by the samples $A^{(1:N)}$. Now, we formalize this setup of the SelfReflect metric by setting the notation, listing the assumptions of the setup, and providing their justifications.

SETUP AND NOTATION

1. Firstly, Figure 2 shows the graphical model of this setup, which we also reproduce here in Figure 5 for better readability. In this graphical model, observed variables are shaded gray, which includes the sampled answers $A^{(1:N)}$, their summary S, and a subsequent answer B, whereas unobserved/latent variables are unshaded, which includes the LLM state Θ_Q.

2. We will use upper-case non-boldface letters (like B or S) to represent random variables/vectors and the corresponding lower-case non-boldface letters (like b or s) to represent particular samples from their underlying distributions.

3. For a random variable Y, the sampling of a particular value y will be denoted as $y \sim Y$ or $y \in \text{supp}(Y)$, where $\text{supp}(Y)$ represents the support of the random variable Y.

4. Let $\mathbf{V}$ denote a finite vocabulary of words (or tokens), which is used to generate questions, the corresponding answers, and their summaries.

5. Let Q denote the random variable for a question.

6. Prompting the given LLM with this question Q is assumed to put it in state, which is represented with the random variable Θ_Q. From this state, we can sample multiple answers, which are then used to define the SelfReflect metric.

7. The random variables $A^{(1:N)} := \left(A^{(1)}, \cdots, A^{(N)}\right)$ are used to denote the N answers sampled from the LLM in state Θ_Q. These samples may be sampled in an i.i.d. manner but we do not necessitate this. In fact, one can sample each answer $A^{(n)}$ conditioned on all previous samples $A^{(1:n-1)}$ as well. We allow for this generality because throughout our derivation, we will always consider these answers jointly as $A^{(1:N)}$.

8. A summarization mechanism inputs the sampled answers and generates their summary S.

9. Suppose B denote a subsequent sample from the LLM in the same state Θ_Q. For the SelfReflect metric, we require an idea summary S of sampled answers $A^{(1:N)}$ to capture all information about this subsequent answer B.

ASSUMPTIONS

1. The support of question Q is assumed to be the set of all finite-length sentences generated from $\mathbf{V}$, which we denote by $\mathcal{X}$.

2. The support of each $A^{(i)}$ is also assumed to be $\mathcal{X}$, the set of all finite-length sentences generated from vocabulary $\mathbf{V}$.

3. The summarization mechanism that inputs the sampled answers $A^{(1:N)}$ and generates their summary S is assumed to be a function ψ. Formally, $\psi : \mathcal{X}^N \longrightarrow \mathcal{X}$ inputs any N sampled answers $A^{(1:N)}$ from the LLM and generates their summary S as $S := \psi\left(A^{(1:N)}\right)$. Note that the support of the summary S, will be a subset of the set of all finite-length sentences, i.e., $\text{supp}(S) \subseteq \mathcal{X}$. This condition models our setup sufficiently well, where we have a candidate summary S per set of answers $A^{(1:N)}$. However, we acknowledge that it is a restrictive condition in that it doesn't allow for modeling a conditional distribution over all summaries given the answers. Generalizing our SelfReflect metric for this case or proving its generality in this case is an interesting direction for future work.

4. We define the support of the subsequent new answer B to be the set $\mathcal{X}_L := \mathbf{V}^L$ of all possible sentences from the vocabulary $\mathbf{V}$ that are of length L. Despite being slightly restrictive, this assumption is not unreasonable; all LLMs have a maximum context length, which can be viewed as an upper limit on the length of the answer B. Also, sentences with smaller lengths are usually padded to achieve the maximum context length.

5. Throughout our derivations, we will assume all required marginal and conditional distributions to be strictly positive. This assumption is reasonable for our setting because in practice, we would be implementing corresponding distributions using the given LLM. For instance, $p(W)$ would represent the probability of sentence W under the given LLM. Further, $p(Y \mid Z)$ would represent the probability of sentence Y when the LLM is prompted with the context Z. Since the LLMs generate distribution over the entire vocabulary $\mathbf{V}$, all the conditional distributions will have strictly positive values, albeit extremely small in certain cases.

A.2. Predictive sufficiency and equivalent characterizations

Now, having set the notations and assumptions, we define the notion of sufficiency and connect it with the definition of an ideal summary.

Definition A.1 (Bayesian and Predictive Sufficiency (Bernardo & Smith, 2009))**.** Consider a distribution parameterized in terms of a parameter ϕ. Let $X^{(1:M)}$ denote M (i.i.d.) samples from this distribution. A statistic (function) $T\left(X^{1:M}\right)$ is called a **Bayesian sufficient statistic** of samples $X^{(1:M)}$ for ϕ if and only if we have: $p\left(\phi \mid X^{(1:M)} = x^{(1:M)}\right) = p\left(\phi \mid T\left(X^{(1:M)}\right) = t\left(x^{(1:M)}\right)\right)$. On the other hand, it is called a **predictive sufficient statistic** of samples $X^{(1:M)}$ if and only if we have: $p\left(X = x \mid X^{(1:M)} = x^{(1:M)}\right) = p\left(X = x \mid T\left(X^{(1:M)}\right) = t\left(x^{(1:M)}\right)\right)$ for any subsequent sample X (with concrete value $x \in \text{supp}(X)$) from the same distribution.

Note that our Definition 3.1 of an ideal summary is closely related to predictive sufficiency as defined in Definition A.1. However, it turns out that Bayesian and predictive sufficiency notions are not exactly equivalent. In light of this, our reason for defining an ideal summary to be predictive sufficient, rather than Bayesian sufficient, is as follows. An LLM trained on a huge corpus of data contains information about a wide array of aspects. However, through the summary, we are interested in capturing only those aspects of the state Θ_Q of the LLM that are related to answering the given question Q. For this, requiring the summary to be predictive sufficient serves the purpose precisely.

Now, in the context of the Definition A.1 of predictive sufficiency, Definition 3.1 of ideal summary, and the graphical model of Figure 5, we prove Proposition 3.2, which asserts the equivalence in the information theoretic and conditional distribution based formulations of the ideal summary. We begin by proving a lemma about the graphical model of Figure 5.

Lemma A.2 (Conditioning on $A^{(1:N)}$ and S)**.** *Under the graphical model given in Figure 5, we have:*

$$p\left(B \mid A^{(1:N)}, S\right) = p\left(B \mid A^{(1:N)}\right)$$

Proof. Consider the following manipulations:

$$
\begin{aligned}
p\left(B \mid A^{(1:N)}, S\right) &=^{(1)} \int_\theta \mathrm{d}\theta\, p\left(B, \Theta_Q = \theta \mid A^{(1:N)}, S\right) \\
&=^{(2)} \int_\theta \mathrm{d}\theta\, \frac{p\left(\Theta_Q = \theta, B, A^{(1:N)}, S\right)}{p\left(A^{(1:N)}, S\right)} \\
&=^{(3)} \int_\theta \mathrm{d}\theta\, \frac{p\left(\Theta_Q = \theta\right) \cdot p\left(B \mid \Theta_Q = \theta\right) \cdot p\left(A^{(1:N)} \mid \Theta_Q = \theta\right) \cdot p\left(S \mid A^{(1:N)}\right)}{p\left(A^{(1:N)}\right) \cdot p\left(S \mid A^{(1:N)}\right)} \\
&=^{(4)} \int_\theta \mathrm{d}\theta\, \frac{p\left(\Theta_Q = \theta\right) \cdot p\left(B \mid \Theta_Q = \theta\right) \cdot p\left(A^{(1:N)} \mid \Theta_Q = \theta\right)}{p\left(A^{(1:N)}\right)} \\
&=^{(5)} \int_\theta \mathrm{d}\theta\, \frac{p\left(\Theta_Q = \theta, B, A^{(1:N)}\right)}{p\left(A^{(1:N)}\right)} \\
&=^{(6)} \int_\theta \mathrm{d}\theta\, p\left(B, \Theta_Q = \theta \mid A^{(1:N)}\right) =^{(7)} p\left(B \mid A^{(1:N)}\right)
\end{aligned}
\tag{3}
$$

Here, steps $(2), (5), (6)$ follow from chain rule. Step (4) follows by cancellation of the common terms. Steps $(1), (7)$ follows from integrating out variable Θ_Q. Step (3) follows from the graphical model of Figure 5. Finally, an analogous derivation would follow by replacing integration with summation in the case of Θ_Q being a discrete variable. $\qquad\square$

Now, we prove Proposition 3.2 establishing the equivalence of the information theoretic and conditional distribution based formulations of the desired predictive sufficiency.

Theorem A.3 (Connection of SelfReflect to Predictive Sufficiency)**.** *Consider the graphical model given in Figure 5. Under this graphical model, for ideal summary S of answers $A^{(1:N)}$,*

$$
\mathcal{I}\left\{A^{(1:N)}; B\right\} = \mathcal{I}\left\{S; B\right\} \iff p\left(B \mid A^{(1:N)}\right) = p\left(B \mid S\right)
$$

Proof. Consider following steps:

$$
\begin{aligned}
\mathcal{I}\left\{A^{(1:N)}; B\right\} = \mathcal{I}\left\{S; B\right\} &\iff^{(1)} \mathbb{E}_{A^{(1:N)}, B}\left[\log \frac{p\left(A^{(1:N)}, B\right)}{p\left(A^{(1:N)}\right) \cdot p(B)}\right] = \mathbb{E}_{S, B}\left[\log \frac{p(S, B)}{p(S) \cdot p(B)}\right] \\
&\iff \mathbb{E}_{B, A^{(1:N)}, S}\left[\log \frac{p\left(A^{(1:N)}, B\right) \cdot p(S)}{p(S, B) \cdot p\left(A^{(1:N)}\right)}\right] = 0 \iff^{(2)} \mathbb{E}_{B, A^{(1:N)}, S}\left[\log \frac{p\left(B \mid A^{(1:N)}\right)}{p(B \mid S)}\right] = 0 \\
&\iff^{(3)} \mathbb{E}_{B, A^{(1:N)}, S}\left[\log \frac{p\left(B \mid A^{(1:N)}, S\right)}{p(B \mid S)}\right] = 0 \iff^{(4)} \mathcal{I}\left\{A; A^{(1:N)} \mid S\right\} = 0 \\
&\iff^{(5)} p\left(B, A^{(1:N)} \mid S\right) = p(B \mid S) \cdot p\left(A^{(1:N)} \mid S\right) \\
&\iff^{(6)} p\left(B \mid A^{(1:N)}, S\right) = p(B \mid S) \iff^{(7)} p\left(B \mid A^{(1:N)}\right) = p(B \mid S)
\end{aligned}
\tag{4}
$$

Here, step (1) follows from the definition of mutual information, steps (2) and (6) from chain rule, steps (3) and (7) from Lemma A.2, step (4) from the definition of conditional mutual information, and step (5) from the equality condition of conditional mutual information. For details on mutual information and conditional mutual information, we refer the reader to (Cover, 1999). $\qquad\square$

A.3. SelfReflect metric and equivalence to predictive sufficiency

Now, we demonstrate that the masked-token prediction task of SelfReflect is equivalent to the above notion of predictive sufficiency. For the SelfReflect metric, we consider the random variable B for a new subsequent sample from the LLM in state Θ_Q and dissect it in terms of its words. In particular, we have: $B \equiv (B_1, \cdots, B_L)$, where L is length of the sentence B (which, as we saw, could be chosen to be the maximum context length for the LLM). Here, B_i represents the random

variable for the $i-$th word of the sentence B for each value of $i \in \{1, \cdots, L\}$. For each i, we use the shorthand notation B_{-i} to represent the variable for all the words in the sentence B except for B_i, i.e., $B_{-i} := (B_1, \cdots, B_{i-1}, B_{i+1}, \cdots, B_L) = (B_j)_{j \neq i}$. Note that B_ℓ, which represents the $\ell-$th word of sentence B, is not to be confused with $A^{(k)}$, which represents the $k-$th sampled answer from the LLM. For each B_i, its support is going to be the vocabulary $\mathbf{V}$ and the supports of B_{-i} and B are $\mathbf{V}^{L-1}$ and $\mathbf{V}^L \equiv \mathcal{X}_L$ respectively. With this setup, we can prove Proposition 3.3, which asserts that under assumptions from subsection A.1, SelfReflect metric provides an equivalent formulation of the desired predictive sufficiency of ideal summary S. This is done as follows.

Theorem A.4 (SelfReflect Metric and Predictive Sufficiency). *Suppose all involved conditionals are modeled via the given LLM and hence, are strictly positive. Then, we have:*

$$p\left(B \mid A^{(1:N)}\right) = p\left(B \mid S\right) \iff \text{for all masking indices } i, \ p\left(B_i \mid A^{(1:N)}, B_{-i}\right) = p\left(B_i \mid S, B_{-i}\right) \tag{5}$$

Proof. ($\Longrightarrow$) Suppose we are given that $p\left(B \mid A^{(1:N)}\right) = p\left(B \mid S\right)$. Consider the following steps:

$$p\left(B \mid A^{(1:N)}\right) = p\left(B \mid S\right) \Longrightarrow p\left(B_1, \cdots, B_L \mid A^{(1:N)}\right) = p\left(B_1, \cdots, B_L \mid S\right)$$
$$\Longrightarrow \sum\nolimits_{b_i \in \mathbf{V}} p\left(B_1, \cdots, B_i = b_i, \cdots, B_L \mid A^{(1:N)}\right) = \sum\nolimits_{b_i \in \mathbf{V}} p\left(B_1, \cdots, B_i = b_i, \cdots, B_L \mid S\right)$$
$$\Longrightarrow^{(1)} p\left(B_{-i} \mid A^{(1:N)}\right) = p\left(B_{-i} \mid S\right) \tag{6}$$

Here, step (1) follows from integrating out variable B_i. Combining this result with the premise gives:

$$p\left(B \mid A^{(1:N)}\right) = p\left(B \mid S\right), \ p\left(B_{-i} \mid A^{(1:N)}\right) = p\left(B_{-i} \mid S\right)$$
$$\Longrightarrow \frac{p\left(B \mid A^{(1:N)}\right)}{p\left(B_{-i} \mid A^{(1:N)}\right)} = \frac{p\left(B \mid S\right)}{p\left(B_{-i} \mid S\right)} \Longrightarrow^{(1)} p\left(B_i \mid A^{(1:N)}, B_{-i}\right) = p\left(B_i \mid S, B_{-i}\right) \tag{7}$$

Here, step (1) follows because B is formed of the $i-$th word B_i and the rest of the words B_{-i}. Since we can carry out these steps for any index i, we prove the forward direction of the theorem.

($\Longleftarrow$) Now, to prove the converse, suppose we are given that for all masking indices i, we have: $p\left(B_i \mid A^{(1:N)}, B_{-i}\right) = p\left(B_i \mid S, B_{-i}\right)$ and we have to prove that $p\left(B \mid A^{(1:N)}\right) = p\left(B \mid S\right)$. Since this is an equality of the random variables, we prove the equality of random variables by proving it for any and all choices of the samples of those random variables. Note that this works because of the assumption of summary mechanism S being a function of $A^{(1:N)}$, which allows us to use the given condition as well as prove the desired result by assuming particular instantiations of $A^{(1:N)} = \bar{a}^{(1:N)}$ and using the corresponding summary $S = \bar{s} := \psi\left(\bar{a}^{(1:N)}\right)$. Pick any instantiations of sampled answers from their support as $a^{(1:N)} \sim A^{(1:N)}$. Since the summary mechanism is a function, it gives us a concrete sample $s = \psi\left(a^{(1:N)}\right) \in \mathcal{X}$. Now, suppose we want to prove the desired result for any particular given sample $b \sim B$ with $b := (b_1, \cdots, b_L) \in \mathbf{V}^L$. Consider a fixed sentence $b^* \in \mathbf{V}^L$ with $b^* := (b_1^*, \cdots, b_L^*)$. Now, we define a sequence of sentences as follows:

$$x^{(0)} := (b_1, b_2, \cdots, b_L) = b \in \mathbf{V}^L$$
$$x^{(1)} := (b_1^*, b_2, \cdots, b_L) \in \mathbf{V}^L$$
$$x^{(2)} := (b_1^*, b_2^*, \cdots, b_L) \in \mathbf{V}^L$$
$$\vdots$$
$$x^{(L)} := (b_1^*, b_2^*, \cdots, b_L^*) = b^* \in \mathbf{V}^L \tag{8}$$

Intuitively, we create a sequence of sentences where each subsequent sentence $x^{(i)}$ differs from the previous sentence and the next sentence in exactly one word and as we go from sentence $x^{(0)}$ to $x^{(L)}$, we change the given sentence b to the fixed

sentence b^*. Now, we consider the following manipulations for $p\left(B = b \mid A^{(1:N)} = a^{(1:N)}\right)$:

$$p\left(B = b \mid A^{(1:N)} = a^{(1:N)}\right) = p\left(B = x^{(0)} \mid A^{(1:N)} = a^{(1:N)}\right)$$

$$=^{(1)} p\left(B = x^{(0)} \mid A^{(1:N)} = a^{(1:N)}\right) \cdot \prod_{\ell=1}^{L} \frac{p\left(B = x^{(\ell)} \mid A^{(1:N)} = a^{(1:N)}\right)}{p\left(B = x^{(\ell)} \mid A^{(1:N)} = a^{(1:N)}\right)}$$

$$=^{(2)} \left(\prod_{\ell=1}^{L} \frac{p\left(B = x^{(\ell-1)} \mid A^{(1:N)} = a^{(1:N)}\right)}{p\left(B = x^{(\ell)} \mid A^{(1:N)} = a^{(1:N)}\right)}\right) \cdot p\left(B = b^* \mid A^{(1:N)} = a^{(1:N)}\right) \tag{9}$$

In an exactly analogous way, we get following manipulations for $p\left(B = b \mid S = s\right)$:

$$p\left(B = b \mid S = s\right) = p\left(B = x^{(0)} \mid S = s\right)$$

$$=^{(1)} p\left(B = x^{(0)} \mid S = s\right) \cdot \prod_{\ell=1}^{L} \frac{p\left(B = x^{(\ell)} \mid S = s\right)}{p\left(B = x^{(\ell)} \mid S = s\right)}$$

$$=^{(2)} \left(\prod_{\ell=1}^{L} \frac{p\left(B = x^{(\ell-1)} \mid S = s\right)}{p\left(B = x^{(\ell)} \mid S = s\right)}\right) \cdot p\left(B = b^* \mid S = s\right) \tag{10}$$

Note that in both Equation 9 and Equation 10 above, step (1) follows from multiplying and dividing by the same terms and step (2) follows from rearranging the terms and recognizing $x^{(L)} = b^*$ by definition. Now, we consider the $\ell-$th term from the Equation 9 and simplify it as follows:

$$\frac{p\left(B = x^{(\ell-1)} \mid A^{(1:N)} = a^{(1:N)}\right)}{p\left(B = x^{(\ell)} \mid A^{(1:N)} = a^{(1:N)}\right)}$$

$$=^{(1)} \frac{p\left(B_1 = b_1^*, \cdots, B_{\ell-1} = b_{\ell-1}^*, B_\ell = b_\ell, B_{\ell+1} = b_{\ell+1}, \cdots, B_L = b_L \mid A^{(1:N)} = a^{(1:N)}\right)}{p\left(B_1 = b_1^*, \cdots, B_{\ell-1} = b_{\ell-1}^*, B_\ell = b_\ell^*, B_{\ell+1} = b_{\ell+1}, \cdots, B_L = b_L \mid A^{(1:N)} = a^{(1:N)}\right)}$$

$$=^{(2)} \frac{p\left(B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right) \mid A^{(1:N)} = a^{(1:N)}\right)}{p\left(B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right) \mid A^{(1:N)} = a^{(1:N)}\right)}$$

$$\times \frac{p\left(B_\ell = b_\ell \mid A^{(1:N)} = a^{(1:N)}, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right)}{p\left(B_\ell = b_\ell^* \mid A^{(1:N)} = a^{(1:N)}, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right)}$$

$$=^{(3)} \frac{p\left(B_\ell = b_\ell \mid A^{(1:N)} = a^{(1:N)}, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right)}{p\left(B_\ell = b_\ell^* \mid A^{(1:N)} = a^{(1:N)}, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right)} \tag{11}$$

Again, in an exactly analogous way, we simplify the $\ell-$th terms of Equation 10 as follows:

$$\frac{p\left(B = x^{(\ell-1)} \mid S = s\right)}{p\left(B = x^{(\ell)} \mid S = s\right)}$$

$$=^{(1)} \frac{p\left(B_1 = b_1^*, \cdots, B_{\ell-1} = b_{\ell-1}^*, B_\ell = b_\ell, B_{\ell+1} = b_{\ell+1}, \cdots, B_L = b_L \mid S = s\right)}{p\left(B_1 = b_1^*, \cdots, B_{\ell-1} = b_{\ell-1}^*, B_\ell = b_\ell^*, B_{\ell+1} = b_{\ell+1}, \cdots, B_L = b_L \mid S = s\right)}$$

$$=^{(2)} \frac{p\left(B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right) \mid S = s\right)}{p\left(B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right) \mid S = s\right)}$$

$$\times \frac{p\left(B_\ell = b_\ell \mid S = s, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right)}{p\left(B_\ell = b_\ell^* \mid S = s, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right)}$$

$$=^{(3)} \frac{p\left(B_\ell = b_\ell \mid S = s, A_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right)}{p\left(B_\ell = b_\ell^* \mid S = s, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right)} \tag{12}$$

In both these simplifications, step (1) follows from the definition of the sentences $x^{(\ell-1)}, x^{(\ell)}$, step (2) follows from chain rule, and step (3) follows from canceling the common terms. However, given equality $p\left(B_i \mid A^{(1:N)}, B_{-i}\right) =$

$p\left(B_i \mid S, B_{-i}\right)$ for all masking locations i implies that for all ℓ:

$$p\left(B_\ell = b_\ell \mid A^{(1:N)} = a^{(1:N)}, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right)$$

$$= p\left(B_\ell = b_\ell \mid S = s, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right) \text{ and} \tag{13}$$

$$p\left(B_\ell = b_\ell^* \mid A^{(1:N)} = a^{(1:N)}, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right)$$

$$= p\left(B_\ell = b_\ell^* \mid S = s, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right) \tag{14}$$

$$\implies \frac{p\left(B_\ell = b_\ell \mid A^{(1:N)} = a^{(1:N)}, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right)}{p\left(B_\ell = b_\ell^* \mid A^{(1:N)} = a^{(1:N)}, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right)}$$

$$= \frac{p\left(B_\ell = b_\ell \mid S = s, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right)}{p\left(B_\ell = b_\ell^* \mid S = s, B_{-\ell} = \left(b_1^*, \cdots, b_{\ell-1}^*, b_{\ell+1}, \cdots, b_L\right)\right)}$$

$$\implies \frac{p\left(B = x^{(\ell-1)} \mid A^{(1:N)} = a^{(1:N)}\right)}{p\left(B = x^{(\ell)} \mid A^{(1:N)} = a^{(1:N)}\right)} = \frac{p\left(B = x^{(\ell-1)} \mid S = s\right)}{p\left(B = x^{(\ell)} \mid S = s\right)} \text{ for all } \ell \in \{1, \cdots, L\}. \tag{15}$$

Combining this with Equation 9 and Equation 10, we get an interesting result:

$$\frac{p\left(B = b \mid A^{(1:N)} = a^{(1:N)}\right)}{p\left(B = b \mid S = s\right)}$$

$$= \frac{\left(\prod_{\ell=1}^L \frac{p\left(B = x^{(\ell-1)} \mid A^{(1:N)} = a^{(1:N)}\right)}{p\left(B = x^{(\ell)} \mid A^{(1:N)} = a^{(1:N)}\right)}\right) \cdot p\left(B = b^* \mid A^{(1:N)} = a^{(1:N)}\right)}{\left(\prod_{\ell=1}^L \frac{p\left(B = x^{(\ell-1)} \mid S = s\right)}{p\left(B = x^{(\ell)} \mid S = s\right)}\right) \cdot p\left(B = b^* \mid S = s\right)}$$

$$=^{(1)} \frac{p\left(B = b^* \mid A^{(1:N)} = a^{(1:N)}\right)}{p\left(B = b^* \mid S = s\right)} \tag{16}$$

Here, step (1) follows from canceling equal terms in both the numerator and the denominator. What Equation 16 implies is that given $A^{(1:N)} = a^{(1:N)}$, thereby giving $S = s := \psi\left(a^{(1:N)}\right)$, the ratio $\frac{p\left(B=b \mid A^{(1:N)}=a^{(1:N)}\right)}{p(B=b \mid S=s)}$ equals the ratio $\frac{p\left(B=b^* \mid A^{(1:N)}=a^{(1:N)}\right)}{p(B=b^* \mid S=s)}$ for any and all values of $b \in \mathbf{V}^L$, thereby making it a constant $c := c\left(a^{(1:N)}\right)$ (a constant that dependents on $a^{(1:N)}$). Now, we can integrate out B and obtain the value of this constant as follows:

$$\text{For all } b \in \mathbf{V}^L, \ \frac{p\left(B = b \mid A^{(1:N)} = a^{(1:N)}\right)}{p\left(B = b \mid S = s\right)} = c\left(a^{(1:N)}\right)$$

$$\implies 1 = \sum_{b \in \mathbf{V}^L} p\left(B = b \mid A^{(1:N)} = a^{(1:N)}\right) = \sum_{b \in \mathbf{V}^L} c\left(a^{(1:N)}\right) \cdot p\left(B = b \mid S = s\right)$$

$$= c\left(a^{(1:N)}\right) \cdot \sum_{b \in \mathbf{V}^L} p\left(B = b \mid S = s\right) = c\left(a^{(1:N)}\right) \cdot 1 = c\left(a^{(1:N)}\right) \tag{17}$$

This proves that in fact $c\left(a^{(1:N)}\right) = 1$, which gives that for all $b \sim B$, we have: $p\left(B = b \mid A^{(1:N)} = a^{(1:N)}\right) = p\left(B = b \mid S = s\right)$. Since this result holds for all $b \sim B$, we can write the corresponding result with the underlying random variable as: $p\left(B \mid A^{(1:N)} = a^{(1:N)}\right) = p\left(B \mid S = s\right)$. However, since this result holds for any sample choice of $A^{(1:N)} = a^{(1:N)}$ (and corresponding $S = s := \psi\left(a^{(1:N)}\right)$), we get the desired results involving all underlying random variables: $p\left(B \mid A^{(1:N)}\right) = p\left(B \mid S\right)$. This proves the reverse direction of the equivalence. $\square$

A.4. Modeling with LLM: From derivation to implementation

Now, having proved the equivalence of the basis of the SelfReflect metric and the desired predictive sufficiency of summary, we show the connection with the exact definition of the SelfReflect metric. Suppose we are given with a question $Q = q \in \mathcal{X}$, which is shown to an LLM labeled LLM_θ. This puts LLM_θ in a state $\Theta_Q = \theta_q$, from which we sample answers $A^{(1:N)} = a^{(1:N)}$, and a subsequent sample $B = b \in \mathbf{V}^L$. Now, to calculate the SelfReflect metric, the core idea is that conditional distributions of the form $p\left(Y \mid Z\right)$ involved in the theoretical considerations above are modeled by prompting

the judge LLM_J with context Z and checking the probability of Y. In our implementation, this LLM_J will be temperature-scaled with temperature $\tau = 5$ as mentioned in the main text in order to flatten its distribution and make it consider more synonyms. Then, we build the prompt of LLM_J by including the question $Q = q$ and either the samples $A^{(1:N)} = a^{(1:N)}$ or their summary $S = s := \psi\left(a^{(1:N)}\right)$, along with a description t of the masked-token prediction task to tell the LLM_J judge what it needs to do. We then mask each word of $B = b$ one by one to obtain the masked word $B_m = b_m \in \mathbf{V}$ and the rest of the sentence $B_{-m} = b_{-m} \in \mathbf{V}^{L-1}$. Then, we model the required conditional distributions that appear in the derivation using the LLM_J judge as follows:

$$
\begin{aligned}
&p\left(B_m = b_m \mid A^{(1:N)} = a^{(1:N)}, B_{-m} = b_{-m}\right) \\
&\quad := p_{\text{LLM}_J}\left(B_m = b_m \mid Q = q, A^{(1:N)} = a^{(1:N)}, t, B_{-m} = b_{-m}\right), \text{ and} \\
&p\left(B_m = b_m \mid S = s, B_{-m} = b_{-m}\right) \\
&\quad := p_{\text{LLM}_J}\left(B_m = b_m \mid Q = q, S = s, t, B_{-m} = b_{-m}\right)
\end{aligned}
\tag{18}
$$

This modeling along with Theorem A.4 demonstrates the efficacy of SelfReflect metric:

Corollary A.5 (Efficacy of SelfReflect Metric). *For any question Q, for all masking indices m,*

$$
\begin{aligned}
&\mathcal{W}^1\left(p_{LLM_J}\left(B_m \mid Q, A^{(1:N)}, t, B_{-m}\right), p_{LLM_J}\left(B_m \mid Q, S, t, B_{-m}\right)\right) = 0 \\
&\Longleftrightarrow^{(1)} p_{LLM_J}\left(B_m \mid Q, A^{(1:N)}, t, B_{-m}\right) = p_{LLM_J}\left(B_m \mid Q, S, t, B_{-m}\right) \\
&\Longleftrightarrow^{(2)} p\left(B_m \mid A^{(1:N)}, B_{-m}\right) = p\left(B_m \mid S, B_{-m}\right) \\
&\Longleftrightarrow^{(3)} p\left(B \mid A^{(1:N)}\right) = p\left(B \mid S\right) \\
&\Longleftrightarrow^{(4)} \mathcal{I}\left\{A^{(1:N)}; B\right\} = \mathcal{I}\left\{S; B\right\}
\end{aligned}
\tag{19}
$$

Proof. Step (4) follows from Theorem A.3, step (3) follows from Theorem A.4, step (2) follows from modeling in Equation 18, and step (1) follows from the fact that the $\mathcal{W}^1$ $(1-\text{Wasserstein})$ distance between two distributions is 0 if and only if the distributions are identical. $\square$

DISCUSSION

We conclude this section by discussing two important points about our derivation.

1. Firstly, LLMs are known to behave significantly better with careful design of prompts (Sahoo et al., 2024). Thus, in our modeling of Equation 18, one may try to optimize the prompting template and the task description t in order to further obtain sharper versions of the SelfReflect metric. In this aspect, note that our derivation does not provide a mechanism for optimizing for the prompt template or task description t. In fact, irrespective of this detail, the derivation holds true.

2. Secondly, we state the assumptions required for the derivation, as stated in Appendix A.1, are needed for establishing the connection of SelfReflect metric with the notion of predictive sufficiency. However, these are not needed for defining, implementing, or using the SelfReflect metric. Users may find our SelfReflect metric useful even in cases where one or more of the assumptions are loosened. Also, further generalizing the SelfReflect metric in cases where the assumptions are loosened or proving that the current formulation holds in those scenarios remains an interesting direction for future theoretical work.

B. Convergence of the SelfReflect metric

In the main paper, we evaluate SelfReflect on 1000 questions per dataset with $N = M = 50$ conditioning and masked-out answers. This is based on a convergence analysis that we present in this section. We use Qwen 2.5 72B Instruct and Natural Questions as an example and calculate the average SelfReflect score across an increasing number of questions and conditioning and masked-out answers in Figures 6 to 10. The question is how many questions are needed to arrive at a stable average score.

It can be seen in Figure 6 that at $N = M = 50$, the SelfReflect score converges at 1000 questions, our setup for the paper. One can of course reduce N and M, which will roughly linearly reduce the runtime required to compute the score. However, when for example reducing to $N = M = 20$ questions in Figure 7, convergence to the final value sets in only at about 2500 questions, which linearly increases the runtime, so that the runtime advantage vanishes. If one allows the score to be a bit less converged, for example in development rather than in reporting test results, we suggest to use $N = M = 10$ and 500 questions. This reduces the runtime to calculate SelfReflect to 9 minutes on a node with 8 A100 GPUs, compared to the 67 minutes of $N = M = 50$ and 1000 questions.

The only real outlier to these trends is $N = M = 1$. Here, it is especially important that $N = 1$, i.e., in the context of the answer distribution prompt, there is only a single response. In this case, the ideal summary is actually to return exactly this response rather than a summary of the distribution. Hence, in Figure 10, *Greedy* obtains a better SelfReflect score than *Sample & Summarize*. This underlines the importance of why SelfReflect uses *multiple* samples from the answer distribution.
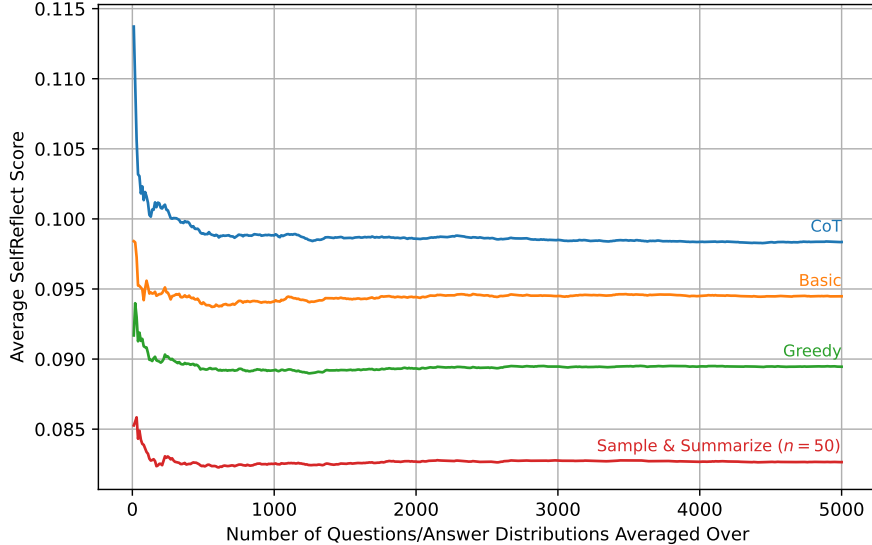
Figure 6. Convergence of the SelfReflect score with $N = M = 50$ and an increasing number of queries we evaluate on. Answer Distributions of Qwen 2.5 72B Instruct on Natural Questions.
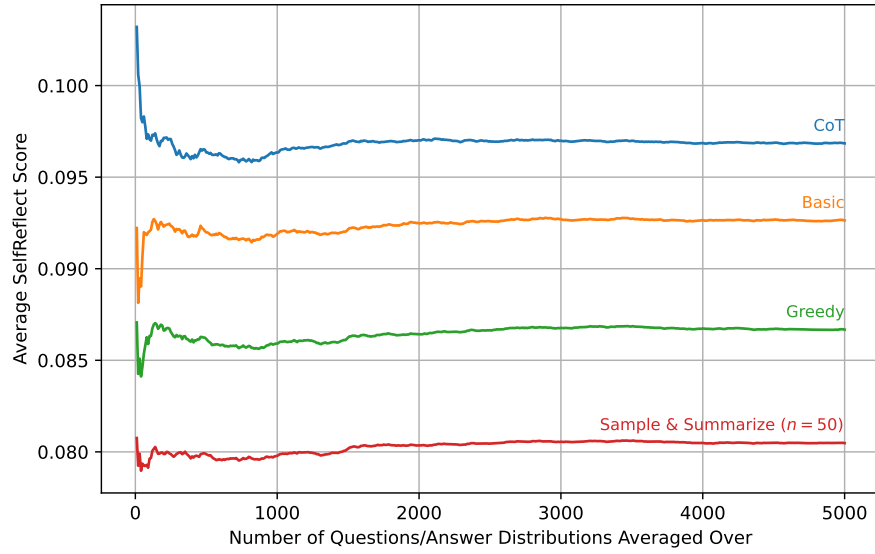
Figure 7. Convergence of the SelfReflect score with $N = M = 20$ and an increasing number of queries we evaluate on. Answer Distributions of Qwen 2.5 72B Instruct on Natural Questions.
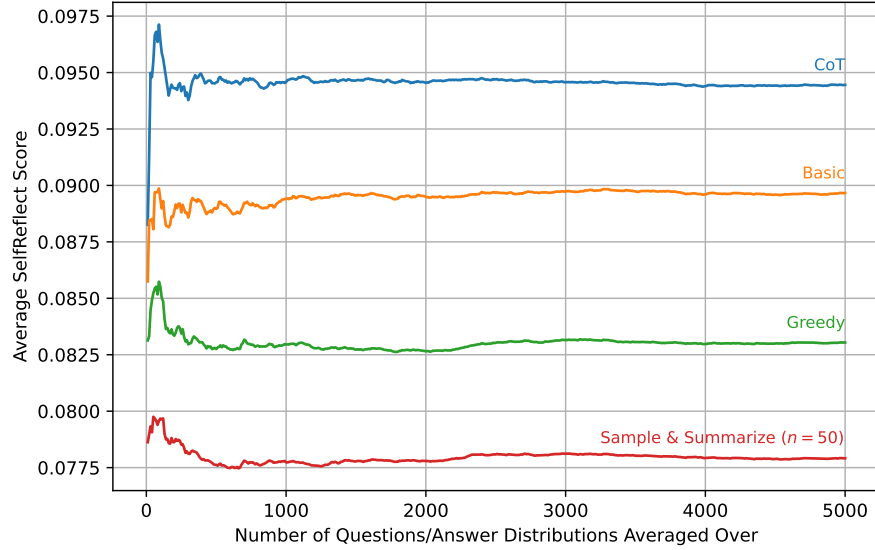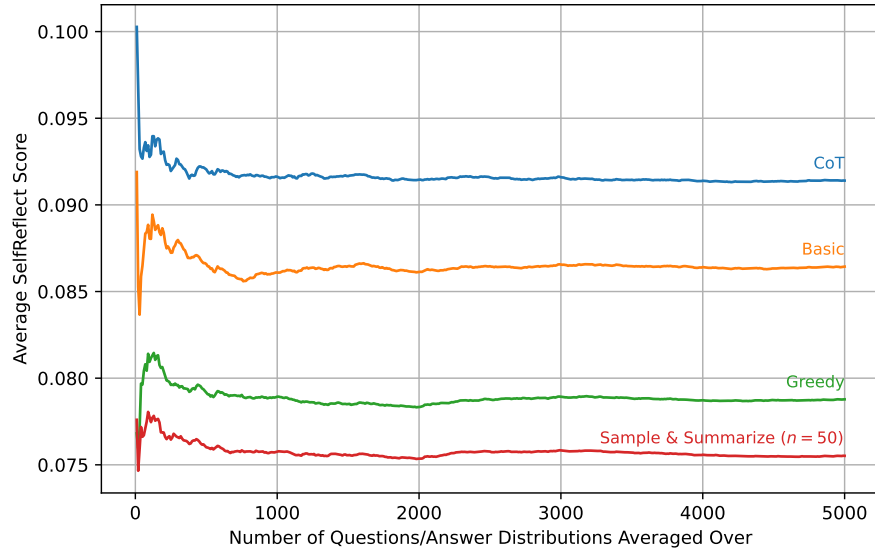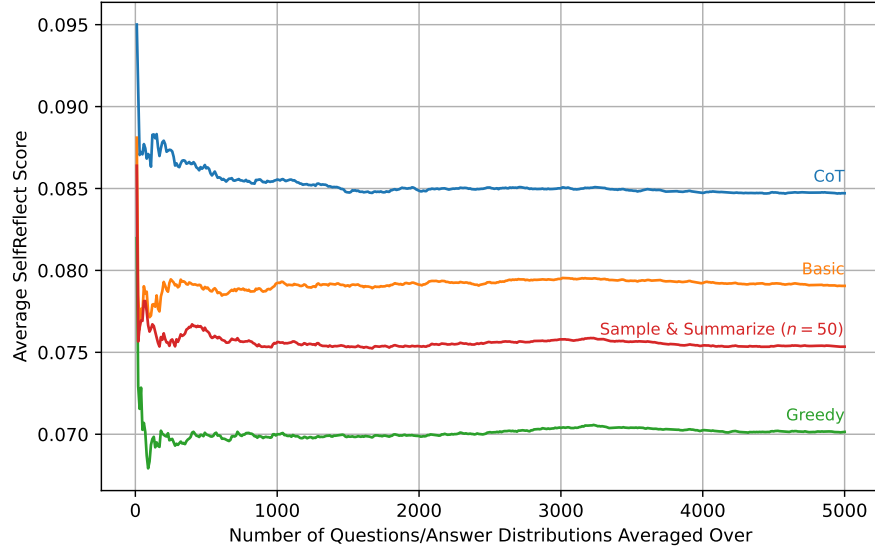

Figure 8. Convergence of the SelfReflect score with $N = M = 10$ and an increasing number of queries we evaluate on. Answer Distributions of Qwen 2.5 72B Instruct on Natural Questions.

Figure 9. Convergence of the SelfReflect score with $N = M = 5$ and an increasing number of queries we evaluate on. Answer Distributions of Qwen 2.5 72B Instruct on Natural Questions.

Figure 10. Convergence of the SelfReflect score with $N = M = 1$ and an increasing number of queries we evaluate on. Answer Distributions of Qwen 2.5 72B Instruct on Natural Questions.

C. Which LLM$_J$ judge to use to generate SelfReflect logits

Table 6. To find out which LLM judge produces the best logits, we test how often SelfReflect correctly distinguishes a good (top) from a bad (bottom) summarywith different possible judges LLM$_J$ that calculate the SelfReflect metric, across different LLM's LLM$_\theta$ whose answer distributions are being summarized. Automatically generated summaries on Natural Questions, following Table 1. Results for Phi 4 14B as a judge for Llama 3.1 8B Instruct are pending and will be added.

LLM$_\theta$	LLM$_J$	Good summaries vs bad summaries	Good vs almost-good	Detailed vs truncated	Verbalized uncertainty vs only majority answer	Verbalized vs or-concatenated	Percentage vs or-concatenated
Llama 3.1 8B Instruct	Llama 3.1 8B Instruct	99.73%$_{\pm0.37\%}$	96.13%$_{\pm1.38\%}$	94.92%$_{\pm3.96\%}$	97.39%$_{\pm2.91\%}$	80.00%$_{\pm7.31\%}$	87.83%$_{\pm5.98\%}$
Phi 4 14B	Llama 3.1 8B Instruct	99.75%$_{\pm0.49\%}$	97.50%$_{\pm1.53\%}$	100.00%$_{\pm0.00\%}$	96.30%$_{\pm5.03\%}$	51.85%$_{\pm13.33\%}$	66.67%$_{\pm12.57\%}$
Qwen2.5 7B Instruct	Llama 3.1 8B Instruct	99.70%$_{\pm0.34\%}$	94.10%$_{\pm1.46\%}$	100.00%$_{\pm0.00\%}$	97.52%$_{\pm2.77\%}$	47.93%$_{\pm8.90\%}$	80.99%$_{\pm6.99\%}$
Llama 3.1 8B Instruct	Phi 4 14B	%$_{\pm}$%	%$_{\pm}$%	94.92%$_{\pm3.96\%}$	99.13%$_{\pm1.70\%}$	87.83%$_{\pm5.98\%}$	86.09%$_{\pm6.32\%}$
Phi 4 14B	Phi 4 14B	100.00%$_{\pm0.00\%}$	94.25%$_{\pm2.28\%}$	94.44%$_{\pm5.33\%}$	48.15%$_{\pm13.33\%}$	59.26%$_{\pm13.11\%}$	59.26%$_{\pm13.11\%}$
Qwen2.5 7B Instruct	Phi 4 14B	99.70%$_{\pm0.34\%}$	93.10%$_{\pm1.57\%}$	98.71%$_{\pm1.78\%}$	95.04%$_{\pm3.87\%}$	59.50%$_{\pm8.75\%}$	75.21%$_{\pm7.69\%}$
Llama 3.1 8B Instruct	Qwen2.5 7B Instruct	100.00%$_{\pm0.00\%}$	95.73%$_{\pm1.45\%}$	95.76%$_{\pm3.64\%}$	95.65%$_{\pm3.73\%}$	80.87%$_{\pm7.19\%}$	85.22%$_{\pm6.49\%}$
Phi 4 14B	Qwen2.5 7B Instruct	99.25%$_{\pm0.85\%}$	96.75%$_{\pm1.74\%}$	98.59%$_{\pm2.74\%}$	94.44%$_{\pm6.11\%}$	70.37%$_{\pm12.18\%}$	77.78%$_{\pm11.09\%}$
Qwen2.5 7B Instruct	Qwen2.5 7B Instruct	99.80%$_{\pm0.28\%}$	94.20%$_{\pm1.45\%}$	98.06%$_{\pm2.17\%}$	95.04%$_{\pm3.87\%}$	74.38%$_{\pm7.78\%}$	83.47%$_{\pm6.62\%}$
Llama 3.1 8B Instruct	Qwen2.5 72B Instruct	99.87%$_{\pm0.26\%}$	96.13%$_{\pm1.38\%}$	97.46%$_{\pm2.84\%}$	99.13%$_{\pm1.70\%}$	86.96%$_{\pm6.15\%}$	78.26%$_{\pm7.54\%}$
Phi 4 14B	Qwen2.5 72B Instruct	98.75%$_{\pm1.09\%}$	97.50%$_{\pm1.53\%}$	98.59%$_{\pm2.74\%}$	96.30%$_{\pm5.03\%}$	72.22%$_{\pm11.95\%}$	55.56%$_{\pm13.25\%}$
Qwen2.5 7B Instruct	Qwen2.5 72B Instruct	99.80%$_{\pm0.28\%}$	94.40%$_{\pm1.43\%}$	99.35%$_{\pm1.27\%}$	99.17%$_{\pm1.62\%}$	75.21%$_{\pm7.69\%}$	66.94%$_{\pm8.38\%}$

A mandatory component to calculate the SelfReflect metric is a judge LLM$_J$ that predicts which masked-out words are possible, given either a summary or a concatenation of samples. This judge needs to be able to "understand" both the details of the answer and the probabilistic aspect of this task, all the while not overwriting its context information with its own world knowledge when making the prediction. The choice of the judge can thus be seen as a hyperparameter to be optimized to produce SelfReflect scores that are as discriminative as possible between good and bad and almost-good summaries. We test four different judges in this section, Llama 3.1 8B Instruct, Phi 4 14B, Qwen 2.5 7B Instruct (which we ultimately use in the paper), and Qwen 2.5 72B Instruct. We generate answer distributions on Natural Questions for different LLM$_\theta$ (Llama 3.1 8B Instruct, Phi 4 14B, and Qwen 2.5 7B Instruct), then use Gemini 2.0 to generate summaries like in Section 4.1, and calculate how often SelfReflect correctly tells apart good from bad (or almost-good) summaries.

Table 6 shows that SelfReflect is very robust to the choice of the judge LLM: All judges can tell apart good from bad summaries in almost all cases. In particular, there is also no indication of a "home-bias", i.e., that a judge would perform better in judging answer distributions that it sampled itself. This, along with the fact that especially bad summaries, which explicitly introduce statements that are wrong and go against the judge's world knowledge, are almost always judged as worse than good summaries, shows that there is no world-knowledge leakage. We attribute this to LLMs' abilities to predict from their context, and to the fact that SelfReflect runs its prediction both conditional on the summary and conditional on the answer distribution, so that should there be any world knowledge leakage, it would likely be equal and removed.

To make the choice of which LLM judge to use, we pay particular attention to the last three columns of Table 6: Comparing a verbalized or percentage uncertainty answer to an or-concatenated answer is among the most subtle challenges and tests whether the judge correctly infers the relative probabilities in both the answer distributions and the summaries, even when they are not explicit. Here we see that the Qwen family sets itself slightly off Phi 4 and Llama 3.1. Within the Qwen family, the 7B model is within the confidence interval of the 72B model (with a mean result better for percentage vs or-concatenated, and worse for the other two), so we use it in the main paper due to its lower inference cost. We note that we also tried using a Qwen 2.5 0.5B Instruct judge, however, this small model was not able to tell apart good from bad summaries. Finally, we note that there exists a research opportunity in developing an LLM judge specialized to perform the SelfReflect judging, either to compress the 7B model into a smaller and faster one, or to improve the last bits of performance on challenging cases. However, we decide against this in this paper, since a specialized model would increase the complexity of our method and add a dependency on a particular model (-checkpoint), which is likely to be outdated soon in the fast-moving field of LLMs.

D. Example of SelfReflect scores per masked-out word

To deepen the understanding of how the SelfReflect score judges summaries, we provide a worked example. We break down the SelfReflect score to the penalty it gives to each masked-out word. To simplify this educational example, we use only $N = M = 7$ samples and make the answers in the conditioning of the prompt equal to the masked-out test answers.

The question posed to the LLM is *"Who received the first Nobel Prize in physics?"*. As can be seen below, the LLM's answer distribution includes Wilhelm Conrad Röntgen as most likely answer, as well as Hendrik Antoon Lorentz and Pieter Zeeman or Henri Becquerel as additional possibilities, and details on their work. Let us now first look at how SelfReflect judges a relatively bad summary of this distribution which just returns the greedy answer *"Wilhelm Conrad Röntgen received the first Nobel Prize in Physics."*. Overall, SelfReflect assigns this bad summary a distance of 0.102 (or taken ×1000 like in Table 4: 102). This score is due to SelfReflect detecting that Hendrik Antoon Lorentz and Pieter Zeeman or Henri Becquerel are not predictable from the summary, and neither the details of the works, as we can see in the per-word penalties below (darker red = higher penalty).

Figure 11. SelfReflect per-word penalties on how far the prediction of each masked-out word based on the summary *"Wilhelm Conrad Röntgen received the first Nobel Prize in Physics."* differs from the prediction based on the samples from the internal distribution. Total penalty: 0.102.

We can now improve this summary by adding the two other possibilities, namely *"It's most likely that Wilhelm Conrad Röntgen received the first Nobel Prize in Physics. But the laureates could also have been Hendrik Antoon Lorentz and Pieter Zeeman or Henri Becquerel."*. With this better summary, SelfReflect correctly removes the penalty on Hendrik Antoon Lorentz, Pieter Zeeman, and Henri Becquerel. But it correctly still penalizes the summary for not mentioning the details of any of the works. This results in an overall score of 0.084 (or 84).

Figure 12. SelfReflect per-word penalties on how far the prediction of each masked-out word based on the summary *"It's most likely that Wilhelm Conrad Röntgen received the first Nobel Prize in Physics. But the laureates could also have been Hendrik Antoon Lorentz and Pieter Zeeman or Henri Becquerel."* differs from the prediction based on the samples from the internal distribution. Total penalty: 0.084.

Having added all answer possibilities, we can now add details mentioned in the individual answers. As a good summary, we give *"It's most likely that Wilhelm Conrad Röntgen received the first Nobel Prize in Physics in recognition of his discovery of X-rays which are now named after him. But the laureates could also have been Hendrik Antoon Lorentz and Pieter*

Zeeman or Henri Becquerel.". SelfReflect removes the penalty on X-rays, which the summary mentions. The remaining penalty of 0.078 (or 78) is due to the summary still not mentioning the details on the Zeeman effect, plus some remaining noise mostly on the names.

Figure 13. SelfReflect per-word penalties on how far the prediction of each masked-out word based on the summary *"It's most likely that Wilhelm Conrad Röntgen received the first Nobel Prize in Physics in recognition of his discovery of X-rays which are now named after him. But the laureates could also have been Hendrik Antoon Lorentz and Pieter Zeeman or Henri Becquerel."* differs from the prediction based on the samples from the internal distribution. Total penalty: 0.078.

These examples demonstrate that SelfReflect punishes summaries for the correct reasons: Either when they don't mention all possibilities or all details of the actual internal answer distribution. We have seen in Sections 4.1 and 4.2 that SelfReflect also correctly punishes deviations from the relative frequencies. To this end, let us modify the second summary which previously had a score of 0.084 (or 84) and state that Henri Becquerel was the most likely first Nobel laureate, which is in conflict with the LLM's internal answer distribution: *"It's most likely that Henri Becquerel received the first Nobel Prize in Physics. But the laureates could also have been Hendrik Antoon Lorentz and Pieter Zeeman or Wilhelm Conrad Röntgen."*. This correctly leads to higher penalties on Wilhelm Conrad Röntgen and Henri Becquerel because both of their implied probabilities are off (while keeping the same penalties on Hendrik Antoon Lorentz and Pieter Zeeman, as well as the in both cases unmentioned details on their works) and worsens the SelfReflect score to 0.092 (or 92).

Figure 14. SelfReflect per-word penalties on how far the prediction of each masked-out word based on the summary *"It's most likely that Henri Becquerel received the first Nobel Prize in Physics. But the laureates could also have been Hendrik Antoon Lorentz and Pieter Zeeman or Wilhelm Conrad Röntgen."* (note that Henri Becquerel is in fact not the most likely; it is Wilhelm Conrad Röntgen) differs from the prediction based on the samples from the internal distribution. Total penalty: 0.092.

This demonstrates that SelfReflect's score works as intended, not only on the dataset or question level as studied in the main paper, but also on a word-level granularity. This example is a regular case, one of the 95%+ (see Table 1) where SelfReflect correctly scores the summaries. We note, however, that there are around 5% of questions where it does not score correctly. In most of these cases, the scores of a good and a slightly worse summary are very close to one another and the mis-decision is mostly due to noise. We thus recommend to run SelfReflect over 1000 questions per dataset, as noted in Appendix B and the main paper, in order to smoothen out some of the remaining noise.

E. Implementation details

E.1. SelfReflect score

To calculate the SelfReflect score, in every masked-out task we run the two prompts in Figure 3 through a judge LLM, which is by default Qwen 2.5 7B-Instruct. This makes the judge predict the logits over the vocabulary size for the current token of the fill-in word. If a fill-in word consists of multiple tokens, where we add the tokens of the true fill-in word one after another into the autoregressive context of the assistant answer. Given the two fill-in token vectors conditioned either on the summary or on the concatenated answers, we apply a temperature of 5 to flatten it. This is in order to give some weight to synonyms, since instruct-tuned LM judges otherwise would give nearly probability 1 to only one possible token (in which case SelfReflect would still be valid, but simplify into comparing whether the two contexts lead to predicting the exactly same word). We found that a temperature of $\tau = 5$ improves the SelfReflect score, making it able to discern good from almost-good summaries more often on a validation dataset. We then softmax the flattened logit vectors and calculate the 1-Wasserstein distance between the log probability vectors. Since these are categorical vectors, the 1-Wasserstein distance simplifies into the L_1 distance, times 0.5. We repeat this over all tokens of a masked-out word, then over all masked-out words of each of the $M = 50$ answers (that are not stopwords), then across all 1000 questions of a dataset. The global average gives the SelfReflect score.

E.2. SR sampling-free score

The sampling-free ablation of the SelfReflect metric also gives two prompts to a judge LLM to calculate the masked-in task. The difference is that the prompt which in SelfReflect contains the sampled answers does not contain sampled answers. Instead it just gives the question and then the masked-out task.

E.3. SR-PMI score

The PMI ablation of the SelfReflect metric uses no masked out task. Instead it poses the question, gives either the summary or the sampled answers as background information in context, and then measures the logit vectors assigned to each token of each of the $M = 50$ answers. In other words, the answers are not given word-by-word with masked-out tasks, but measured as one full answer. As for SelfReflect, we then calculate the 1-Wasserstein distance between the flattened logit vectors and average.

E.4. SR-P(True) score

In the P(True) ablation of SelfReflect, we turn the generative masked-out task into a discriminative one. We first generate three candidate words to fill in the masked word: One is the true masked word, one is a word sampled from the masked-out task prompt given the summary and the last is a word sampled from the masked-out task prompt given the distribution samples. With these candidate fill-in words, we then run two prompts, one conditional on the summary and one on the answers, to let the judge LLM predict how likely they fit in, see Figure 15. As in the normal SelfReflect, this gives a distribution over the vocabulary size, concentrated on "True" and "False" tokens. We then compare the two flattened logit vectors via the 1-Wasserstein distance and average as in the original SelfReflect.

E.5. Embedding score

We compare the gte-Qwen2-7B-instruct (Li et al., 2023) embedding of the summary to the embedding of the samples of the distribution, normalize them and take the inner product to form cosine distances. We average over all samples. The reason why we select this particular embedding model is that at the time of submission it was the best-performing open-source model on the MTEB benchmark (Enevoldsen et al., 2025).

E.6. Summarization score

We follow the prompts of Jain et al. (2023) that prompt an LLM to judge a summary in terms of consistency, fluency, relevance, and coherence with a few-shot example. We then normalize all scores to [0, 1] and average them to get the summarization score.

Predicted token vector: $p_J(X \mid q, s, a_{-i}, c_j) = (0.69, \ 0.30, \ \dots)$ "True" "False"

Predicted token vector: $p_J(X \mid q, a^{(1:N)}, a_{-i}, c_j) = (0.82, \ 0.16, \ \dots)$ "True" "False"

$$m_{\text{SR-P(True)}}(s) = \mathcal{W}^1\left(p_J\left(X \mid q, s, a_{-i}, c_j\right)^{1/\tau}, p_J\left(X \mid q, a^{(1:N)}, a_{-i}, c_j\right)^{1/\tau}\right)$$
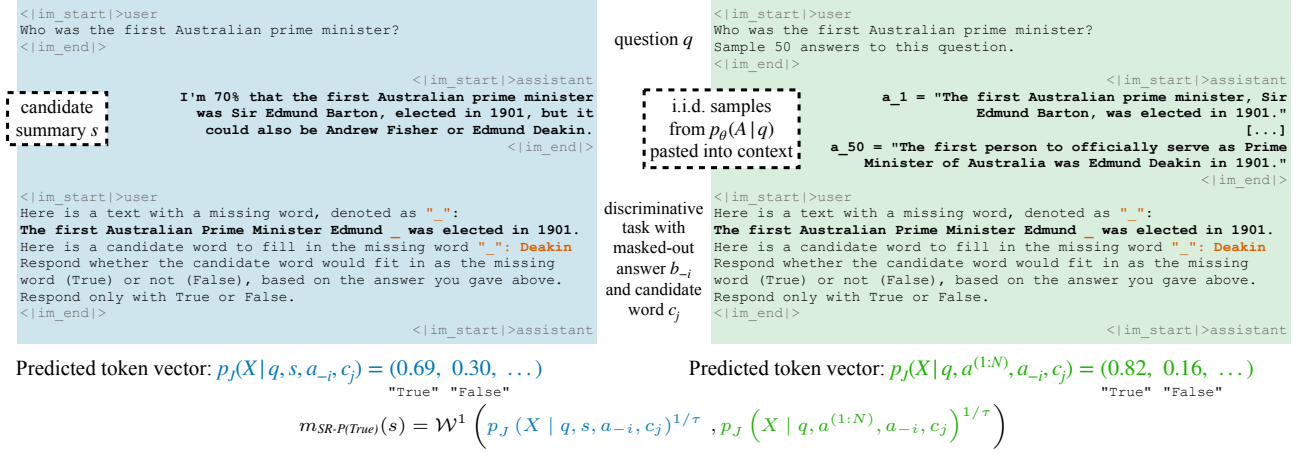
Figure 15. The P(True) ablation of SelfReflect adds a candidate word c_j into the context and asks the judge LLM to classify whether this word fits as masked-out word or not. It compares the probability vectors predicted given either the summary or the concatenated samples.

Prompt for the 'Summarization' metric in Table 1 to judge fluency.

```
Fluency measures the quality of individual sentences, and whether they are
well-written and grammatically correct.  Rate the summary of a given text on a scale
of 0 to 1 on fluency.

Here are some examples:  <4 few-shot examples>

Now here is the summary whose fluency you are supposed to rate:
Summary:  {summary}

Fluency:
```

Prompt for the 'Summarization' metric in Table 1 to judge coherence.

```
Rate the following summaries on a scale from 0 to 1 on coherence, with a higher
value corresponding to higher coherence.  Coherence is a collective quality of
all sentences.  To score highly on it, the summary should be well-structured and
well-organized.  It should not just be a heap of related information, but should
build from sentence to sentence to form a coherent body of information about the
topic.

Here are some examples:  <4 few-shot examples>

Now here is the summary whose coherence you are supposed to rate:
Summary:  {summary}

Coherence:
```

Prompt for the 'Summarization' metric in Table 1 to judge consistency.

```
Consistency measures whether the details in the summary reproduce the facts present
in the text accurately.  Rate the summary of given text on a scale from 0 to 1 on
consistency.

Here are some examples:  <4 few-shot examples>

Now here is the text and summary whose consistency you are supposed to rate:
Text:  We received many answers to our question '{question}'.  Here they are:
```

```
x_1 = '{answer}'
...
x_{n_answers} = '{answer}'

Summary:  {summary}

Consistency:
```

Prompt for the 'Summarization' metric in Table 1 to judge relevance.

```
Relevance is the quality of a summary to capture important information from a
reference text.  Rate the summary on a scale from 0 to 1 on relevance.

Here are some examples:  <4 few-shot examples>

Now here is the text and summary whose relevance you are supposed to rate:
Text:  We received many answers to our question '{question}'.  Here they are:
x_1 = '{answer}'
...
x_{n_answers} = '{answer}'

Summary:  {summary}

Relevance:
```

E.7. LM Judge score

We follow Xu et al. (2024) to build a metric that asks an LM judge to chain-of-thoughts think and rate how well a summary matches a distribution of answers, including a few-shot example. The prompt is shown below.

Prompt for the 'LM Judge' metric in Table 1.

```
Your task is to analyze whether a summarized answer correctly contains all the
possibilities that len(answers) individual answers to a question mention.
Note that some individual answers occur more often than other individual answers.
You should output a score from 0 to 10, indicating whether the summarized answer
mentions all possibilities and whether it correctly outlines which are the most
often appearing individual answers and which appear less often.  A higher score is
means the summarized answer matches the distribution of individual answers better.
Also note that some individual answers may be factually wrong.  Do not correct those,
just report how good the summarized answer matches the individual answers.
You should first provide your reasoning for how well the summarized answer matches
the distribution over individual answers, and then assign a score based on this
reasoning.  The output should be in the following format:

Reason:  [REASON]
Score:  [SCORE]

Here is an example:
Question:  <Example question>
Individual answers:
<Example answer samples>
Summarized answer:  <Example summary>

Then your output can be:
Reason:  The summarized answer mentions the most likely possibility, and it also
correctly mentions that this is the most likely one.  For other possibilities, it
mentions Wilhelm Conrad Röntgen, but does not mention that he got the award for
his discovery of x-rays, which the individual answers do mention.  It also does
not mention the possibility of Hendrik Antoon Lorentz and Pieter Zeeman, which the
individual answers mention.
```

```
Score:  8

Now consider the following case:
{question}
Individual answers:
x_1 = '{answer}'
...
x_{n_answers} = '{answer}'

Summarized answer:  '{summary}'

Please provide the reason and the score of how good the summarized answer matches
the distribution of individual answers.
```

E.8. Optimal Transport score

The optimal transport metric consists of two steps. First, we break down a summary into a distribution of statements and their probabilities. This is done with the following prompt.

Prompt for the 'Optimal transport' metric in Table 1 to split a summary into core statements and their probabilities.

```
Question:  {question}

Here is some background information.  This background information defines a
distribution of possible answers you can later sample from:
{summary}

Now, split this distribution up into its mutually fundamental statements and the
explicitly or implicitly connected probabilities.
Split it up such that each statement is mutually exclusive and the probabilities sum
to 1.
Include an 'I don't know' statement with the remaining percentage if the background
information explicitly mentions not being certain.
Return a json file with a list of dictionaries, where in every dictionary the first
key is called 'prob' and includes the numerical probability and the second key is
'statement' and includes a string of the fundamental statement.
```

In the second step, we use an NLI model to calculate an entailment probability in [0, 1] between how much each sample answer entails each statement and vice versa. We multiply $(1 - \text{entailment probability})$ of both directions to get an distance score for each sample answer and statement. This defines a distance matrix with the statements as rows and the sample answers as columns. Besides a distance matrix, optimal transport also requires marginals for both rows and columns. For the rows, we use the probabilities assigned to the statements in the above prompt. For the columns, we assign each individual sample answer a uniform probability. We then compute the earth movers distance using Flamary et al. (2021). This matches sample answers to summary statements in such a manner that the marginals are preserved and that overall all pairs in sum have the smallest possible distance. The resulting overall distance then tells how far the answer samples are from the summary.

E.9. Licensing information

Table 7 contains licensing information for models used in this paper.

Table 7. Licencing information for models used in this work

LLM	License	Reference
DeepSeek R1 Distill Qwen 2.5 32B	MIT	DeepSeek-AI et al. (2025)
Gemma 3 family	`https://ai.google.dev/gemma/terms`	Gemma Team et al. (2025)
Gemini 2.0 Flash	Apache 2.0	
Ministral 8B Instruct 2410	`https://mistral.ai/static/licenses/MRL-0.1.md`	Jiang et al. (2024)
Llama 3.1 70B Instruct	`https://www.llama.com/llama3_1/license/`	Meta AI (2024a)
Llama 3.3 70B Instruct	`https://www.llama.com/llama3_3/license/`	Meta AI (2024b)
Llama 4 Scout 17b 16e Instruct	`https://www.llama.com/llama4/license/`	Meta AI (2025)
Phi-4	MIT	Abdin et al. (2024)
gte Qwen 2 7B Instruct	Apache-2.0	Li et al. (2023)
Qwen 2.5 family	Apache-2.0	Yang et al. (2024a)
Qwen 3 family	Apache-2.0	Qwen Team (2025a)
QwQ 32B	Apache-2.0	Qwen Team (2025b)

F. Rating good and bad summaries written by humans

Table 8. Mirroring Table 1, we compare how often our SelfReflect metric, and other possible metrics, discriminates good from bad summaries of answer distributions. In this version, the summaries are written by humans rather than by Gemini, on a disjoint set of questions. Mean $\pm$ 95% interval. Confidence intervals are larger than in Table 1 because we have less manually written summaries of answer distributions than the automated ones in Table 1.

Metric	Good summaries vs bad summaries	Good vs almost-good	Detailed vs truncated	Verbalized uncertainty vs only majority answer	Verbalized vs or-concatenated	Percentage vs or-concatenated
Summarization	98.20%$_{\pm1.43\%}$	60.18%$_{\pm5.25\%}$	70.00%$_{\pm20.08\%}$	24.14%$_{\pm7.93\%}$	55.17%$_{\pm18.10\%}$	51.72%$_{\pm18.19\%}$
LM Judge	98.50%$_{\pm1.30\%}$	54.19%$_{\pm5.34\%}$	55.00%$_{\pm21.80\%}$	34.48%$_{\pm17.30\%}$	37.93%$_{\pm17.66\%}$	24.14%$_{\pm15.58\%}$
Opt. Transport	78.14%$_{\pm4.43\%}$	57.19%$_{\pm5.31\%}$	10.00%$_{\pm13.15\%}$	58.62%$_{\pm17.93\%}$	79.31%$_{\pm14.74\%}$	72.41%$_{\pm16.27\%}$
Embedding	74.85%$_{\pm4.65\%}$	44.01%$_{\pm5.32\%}$	60.00%$_{\pm21.47\%}$	20.69%$_{\pm14.74\%}$	41.38%$_{\pm17.93\%}$	13.79%$_{\pm12.55\%}$
SR-PMI	85.33%$_{\pm3.79\%}$	60.18%$_{\pm5.25\%}$	75.00%$_{\pm18.98\%}$	3.45%$_{\pm6.64\%}$	24.14%$_{\pm15.58\%}$	31.03%$_{\pm16.84\%}$
SR-sampling-free	92.22%$_{\pm2.87\%}$	80.24%$_{\pm4.27\%}$	75.00%$_{\pm18.98\%}$	62.07%$_{\pm17.66\%}$	62.07%$_{\pm17.66\%}$	58.62%$_{\pm17.93\%}$
SR-P(True)	47.90%$_{\pm5.36\%}$	58.38%$_{\pm5.29\%}$	35.00%$_{\pm20.90\%}$	**96.55%**$_{\pm6.64\%}$	82.76%$_{\pm13.75\%}$	79.31%$_{\pm14.74\%}$
SelfReflect	**99.70%**$_{\pm0.59\%}$	**94.61%**$_{\pm2.42\%}$	**95.00%**$_{\pm9.55\%}$	86.21%$_{\pm12.55\%}$	**93.10%**$_{\pm9.22\%}$	**82.76%**$_{\pm13.75\%}$

In Table 1 in the main paper, we use Gemini 2.0 Flash to generate various types of good and bad summaries from sampled answers. We choose an automated LLM approach because it is more scalable (with accordingly preciser 95% intervals) and reproducible than manual annotation. For reproducibility, we also report the prompts below. To ensure the quality of the results, we have, however, also replicated the experiments where we wrote the summaries manually for 334 questions of the Natural Questions dataset, on a disjoint split from those in Table 1. Table 8 reports the results on how often SelfReflect, and other metrics, rated good summaries as better than their worse counterparts. The results are analogous to those in the main paper in that SelfReflect scores the highest on all metrics except on one where the P(True) ablation achieves a slightly better result.

Gemini 2.0 Flash prompt to generate 'good' summaries in Table 1.

```
Below, you are given {n_answers} individual answers to the question '{question}'.
```

```
Your goal is to summarize the {n_answers} answers into one answer.

  • The summarized answer should mention the main possibilities mentioned by the
    {n_answers} answers.  If a possibility is mentioned only once, it can be skipped
    so that the summary remains concise.

  • If some possibilities are mentioned much more often than others, delineate
    which possibilities are more often found in the others by using words like "most
    likely" and "could also be".

  • The format of the summarized answer should be the same as each individual answer.
    Provide only the answer, as if it were part of the {n_answers} answers, without
    statements like "The answers include...".

  • Similarly, the summarized answer should use the same wording as the original
    answers.  If the original answer always uses "is situated", then use "is
    situated" and not "is located".

  • The summarized answer should reflect what the {n_answers} answers deem possible.
    They can contain factually wrong options.  Do not correct those, just report the
    possibilities as they are given in the answers.

Here are the {n_answers} answers:
x_1 = '{answer}'
...
x_{n_answers} = '{answer}'

Please provide the summarized answer.
```

Gemini 2.0 Flash prompt to generate an 'almost-good' summary from a 'good' summary in Table 1. Also used to generate 'truncated' from 'detailed' summaries.

```
Below, you are given an answer to the question '{question}'.

Your goal is to shorten the answer.

  • If the answer mentions multiple possibilities, only return the main possibility.

  • If the answer includes a main answer and details, remove the details.

  • The shortened answer should have the same format as the original answer.  If the
    original answer uses full sentences, the shortened answer should also use a full
    sentence.

  • The shortened answer should use the same wording as the original answers.  If
    the original answer always uses "is situated", then use "is situated" and not
    "is located".

  • The answer can contain factually wrong options.  Do not correct those, just
    shorten what the answer says, even if it is factually wrong.

Original answer:  {good_summary}

Please provide the shortened answer.
```

Gemini 2.0 Flash prompt to generate a 'bad' summary from a 'good' summary in Table 1.

```
Below, you are given a response to the question '{question}'.

Your goal is to change the answer.

  • The answer should generally stay close to the original answer, with only some
    key factual terms changed.
```

```
  • The answer might already be factually wrong.  But the goal is still to change
    the key facts, so that the changed answer is different from the original one.

  • The changed answer should have the same format as the original answer.  The
    structure should remain the same, only keywords should be exchanged.

  • The changed answer should also use the same wording as the original answers for
    any non-factual words.  If the original answer always uses "is situated", then
    use "is situated" and not "is located".

Original answer:  {good_summary}

Please provide the changed answer.
```

For verbalized, percentage, or-concatenated and majority answers, we first use a prompt to cluster the answer distribution into clusters of statements and which answers belong to which cluster statement:

Gemini 2.0 Flash prompt to cluster samples from an answer distribution into a list of cluster representatives and cluster memberships.

```
Below, you are given {n_answers} individual answers to the question '{question}'.
These {n_answers} answers can be seen as samples from an answer distribution.  Your
goal is to cluster the distribution in two steps:

First step:  Find the clusters and their representatives.

  • Each cluster contains a set of answers that are essentially the same.  This
    means they may vary in the level of detail, but their primary answer should be
    the same.

  • Different clusters should be mutually exclusive answers.

  • There are at least two clusters.

  • The answer can contain factually wrong options.  Do not correct them, just
    cluster the answers as they are.

  • Output a json file with each entry giving the "cluster_id" (cluster_1, cluster_2,
    ...), and a "representative_answer", copy-pasted from the answers below.

Second step:  Match the answers to their clusters.

  • Match each of the {n_answers} individual answers to one cluster representative.

  • Output a json file with each entry giving the "cluster_id" (cluster_1, cluster_2,
    ...)  and the "cluster_members", a list of [x_1, x_26, ...].

Here are the {n_answers} answers:
x_1 = '{answer}'
...
x_{n_answers} = '{answer}'

Please output the two json files, one after another.  Each json file should start
with ```json
```

We then count how many member each cluster has (manually in code as opposed to asking the LLM since this increases accuracy), and provide lists of the representative answers of each cluster and their relative frequencies to build the percentage and or-concatenated summaries. We sort the resulting list frequency. For the majority answer, we directly return the representative answer of the highest-likely cluster. The verbalized uncertainty summary is built by removing the percentages in their brackets from the percentage summary.

Gemini 2.0 Flash prompt to generate 'percentage-uncertainty' summaries in Table 1.

```
Below, you are given list of answers with their probabilities to the question
'{question}'.

Your goal is to stitch these answers together into one sentence.

  • The sentence should have the structure 'It is most likely that <Answer A>
    (<probability of Answer A>% sure), but it could also be <Answer B> (<probability
    of Answer B>% sure) or <Answer C> (<probability of Answer C>% sure) or ...'

  • Stick to the original wording of the answers as much as possible, but you can
    add small words so that the sentence becomes a grammatically coherent sentence.

  • The answer can contain factually wrong options.  Do not correct those, just
    stitch together the answer options, even if it is factually wrong.

List of answers:
[
  {
  'prob':  0.72,
  'statement':  ...
  },
  {
  'prob':  0.22,
  'statement':  ...
  }
]

Please provide the coherent sentence.
```

Gemini 2.0 Flash prompt to generate 'or-concatenated' summaries in Table 1.

```
Below, you are given list of answers with their probabilities to the question
'{question}'.

Your goal is to stitch these answers together into one sentence.

  • The sentence should have the structure 'Either <Answer A> or <Answer B> or
    <Answer C> or ...'

  • The sentence should be grammatically coherent.

  • Stick to the original wording of the answers as much as possible.

  • The answer can contain factually wrong options.  Do not correct those, just
    stitch together the answer options, even if it is factually wrong.

List of answers:
[
  {
  'prob':  0.72,
  'statement':  ...
  },
  {
  'prob':  0.22,
  'statement':  ...
  }
]

Please provide the coherent sentence.
```

G. User study details

User studies were carried out using TryRating, with five raters per task. Raters were allowed to rate as many tasks as they wanted. All raters were US-based English speakers, and were paid $18/hr.

The users were presented with the instructions shown in Figure 16, which included two examples with hand-crafted summaries (for space reasons, we include only one summary here).

Task Guidelines

You will be shown a question, and ten survey responses to that question.

You will then be shown two possible summaries of the survey responses. Your task is to pick the summary that best summarizes the ten individual responses.

Some things to consider:

- Does the summary reflect all viewpoints in the responses? Is it clear which viewpoints are more or less prevalent? A great summary reflects all viewpoints, and makes clear which are more or less common viewpoints. A good summary reflects all viewpoints, but might not indicate which are more or less common. An OK summary only reflects some viewpoints. A bad summary does not reflect any of the viewpoints.
- Does the summary accurately capture the viewpoints? A great summary accurately captures the information in each respose. A poor summary either excludes key information, or adds information that is not mentioned in the survey responses.
- Is the summary easy to read? If two summaries are equally informative about the ten survey answers, the better summary will be the one that is easier to read and understand.

Do **not** consider any prior knowledge you have about the question! Your task is not to assess which summary is a correct answer to the question, it is to assess which summary best represents the sampled responses (which might not correctly answer the question).

Here are a few examples:

Question: where do peaches come from in the us?

- Peaches come from several states in the US, primarily Georgia, California, and South Carolina.
- Peaches in the U.S. primarily come from California, Georgia, and South Carolina.
- Peaches primarily come from California, followed by South Carolina and Georgia.
- Peaches in the U.S. primarily come from California, Georgia, and South Carolina.
- Peaches are primarily grown in states like Georgia, California, and South Carolina in the US.
- Peaches primarily come from California and Georgia in the USA.
- Peaches in the U.S. primarily come from Georgia, California, and South Carolina.
- Peaches in the US primarily come from California, Georgia, and South Carolina.
- Peaches are primarily produced in California, which is the leading producer in the U.S.
- Peaches primarily come from California, which is the leading producer in the U.S.

Summary 1	Summary 2
Peaches primarily come from California, followed by Georgia and South Carolina.	Peaches primarily come from California.
Summary 1	Summary 2

In the example above, Summary 1 is the better summary. It identifies all states mentioned in the responses, while indicating that California is the primary choice. In contrast, Summary 2 does not mention South Carolina or Georgia.

Figure 16. Instructions for user study (truncated; actual instructions contained a second example, which we have cut here for space).

To ensure quality responses, we constructed twenty "golden answer" tasks, where the summaries were manually constructed to either fit the definition of "good", "nearly good", or "bad" summaries, as described in Section 4.1. Ten of these questions were given as an entrance exam, with raters required to answer the golden answer in 80% of tasks to proceed. The remaining ten questions were periodically included as verification checks. A total of 215 raters passed the entrance exam and contributed ratings.

Confidence intervals were calculated using 100 bootstrapped samples.

H. Automatic summary generation

H.1. Experimental details

In this section, we denote sampling parameters as {T=1, topp=1, topk=None, minp=0}.

Non-reasoning/RLHF models   For the models in Table 4, the same model is used for both generating the answers, and generating the summaries. We sample answers with {T=1, topp=1, topk=None, minp=0}. For summaries generation, we use greedy decoding, i.e., {T=0, topp=1, topk=None, minp=0}.

Reasoning/RLVR models   For the models in Table 5, we also want to sample the answers without reasoning and make use of the reasoning only for the summaries generation. For Qwen3, it is possible to suppress the reasoning with tokenizer.apply_chat_template(..., enable_thinking=False). For QwQ-32B and DeepSeek-R1-Distill-Qwen2.5-32B, we did not find a way to suppress reasoning and hence we sample the answers from Qwen2.5-32B-Instruct, which is the RLHF model which served as a base model for the RLVR training. We sample answers with {T=1, topp=1, topk=None, minp=0}.

For the *Greedy* summary generation, we use non-reasoning greedy decoding with the same model as for the answers generation, i.e., for rows QwQ-32B and DeepSeek-R1-Distill-Qwen2.5-32B, we use Qwen2.5-32B-Instruct.

For the *Basic* and *Sample & Summarize* summaries generation, we use the reasoning mode of each respective model. Unlike for RLHF models, we stray away from using greedy decoding for summary generation, because the creators of the reasoning models we use warn that the use of greedy decoding with reasoning "as it can lead to performance degradation and endless repetitions". Hence, for each model we use the respective recommended sampling parameters available on their respective HuggingFace model card.

H.2. Prompts used

Prompt for the *basic* summary generation method in Section 5.

```
Please respond to the following question '{question}'.

Your goal is to summarize all possible answers to this question:

  • If there are multiple possible answers, the summarized answer should mention the
    main possible answers.  However, you do not have to list possibilities that are
    too unlikely.

  • If some possibilities are more likely than others, delineate which possibilities
    are more more likely by using words like "most likely" and "could also be".

  • The format of the summarized answer should be the same as a normal answer.

  • If there is only clear answer to the question, just provide that answer, without
    hedging across possibilities.

Please provide the summarized answer.
```

Prompt for the *CoT* summary generation method in Section 5.

```
Please respond to the following question '{question}'.

Your goal is to first reason about all possible answers to this question and then
summarize them into a final answer:

  • Reflect on whether there are multiple possible answers to this question.

  • If there are multiple possible answers, the summarized answer should mention the
    main possible answers.  However, you do not have to list possibilities that are
    too unlikely.
```

```
    • If some possibilities are more likely than others, delineate which possibilities
      are more more likely by using words like "most likely" and "could also be".

    • The format of the summarized answer should be the same as a typical answer and
      be stand-alone.

    • If there is only clear answer to the question, just provide that answer, without
      hedging across possibilities.

The output should be in the following format:
Reasoning:  [REASONING ABOUT WHICH POSSIBLITIES THERE ARE AND HOW LIKELY THEY ARE]
Summary:  [SUMMARIZED ANSWER]

Please provide the reasoning and then the summarized answer.
```

H.3. Results per dataset

Table 9. SelfReflect score $\downarrow$ ($\times 10^{-3}$, rounded for readability) for the SimpleQA dataset, averaged across 1000 questions. The results in small font are relative to *Greedy*.

Model	$p_\theta(A \mid q)$ unimodal	Single-decoding methods			Sample & summarize	
		Greedy	Basic	CoT	$N = 10$	$N = 20$
Qwen2.5 0.5B Instruct (Yang et al., 2024a)	1%	98	97_{-1}	95_{-3}	99_{+1}	99_{+1}
Qwen2.5 1.5B Instruct (Yang et al., 2024a)	2%	98	97_{--0}	93_{-5}	90_{-8}	89_{-8}
Qwen2.5 3B Instruct (Yang et al., 2024a)	5%	101	101_{-0}	99_{-1}	93_{-8}	91_{-9}
Qwen2.5 7B Instruct (Yang et al., 2024a)	15%	98	102_{+4}	102_{+3}	93_{-5}	92_{-6}
Qwen2.5 14B Instruct (Yang et al., 2024a)	23%	96	101_{+5}	102_{+7}	88_{-8}	87_{-9}
Qwen2.5 32B Instruct (Yang et al., 2024a)	17%	103	108_{+5}	110_{+8}	95_{-8}	94_{-8}
Qwen2.5 72B Instruct (Yang et al., 2024a)	18%	95	99_{+3}	100_{+5}	88_{-8}	87_{-9}
Phi 4 (Abdin et al., 2024)	3%	99	99_{--0}	97_{-2}	89_{-10}	87_{-12}
Ministral 8B Instruct 2410 (Jiang et al., 2024)	1%	117	116_{--0}	114_{-2}	109_{-7}	107_{-9}
Llama 3.1 70B Instruct (Meta AI, 2024a)	16%	97	97_{-0}	97_{+1}	90_{-6}	90_{-7}
Llama 3.3 70B Instruct (Meta AI, 2024b)	29%	100	103_{+3}	113_{+13}	92_{-8}	91_{-9}
Llama 4 Scout 17B 16e Instruct (Meta AI, 2025)	20%	95	100_{+5}	103_{+8}	89_{-6}	87_{-7}
Gemma 3 1B Instruct (Gemma Team et al., 2025)	17%	123	134_{+11}	135_{+11}	124_{-0}	118_{-6}
Gemma 3 4B Instruct (Gemma Team et al., 2025)	25%	118	135_{+16}	138_{+20}	108_{-10}	106_{-12}
Gemma 3 12B Instruct (Gemma Team et al., 2025)	35%	117	129_{+12}	135_{+18}	112_{-5}	112_{-5}
Gemma 3 27B Instruct (Gemma Team et al., 2025)	49%	109	124_{+16}	134_{+25}	103_{-5}	101_{-7}

Table 10. SelfReflect score ↓ ($\times 10^{-3}$, rounded for readability) for the Natural Questions dataset, averaged across 1000 questions. The results in small font are relative to *Greedy*.

Model	Single-decoding methods			Sample & summarize	
	Greedy	Basic	CoT	$N = 10$	$N = 20$
Qwen2.5 0.5B Instruct (Yang et al., 2024a)	92	90_{-2}	90_{-2}	92_{-0}	92_{-0}
Qwen2.5 1.5B Instruct (Yang et al., 2024a)	90	91_{+1}	89_{-1}	85_{-5}	84_{-6}
Qwen2.5 3B Instruct (Yang et al., 2024a)	95	96_{+1}	97_{+2}	88_{-7}	87_{-8}
Qwen2.5 7B Instruct (Yang et al., 2024a)	94	98_{+4}	101_{+7}	90_{-4}	89_{-5}
Qwen2.5 14B Instruct (Yang et al., 2024a)	91	97_{+6}	100_{+9}	86_{-5}	85_{-6}
Qwen2.5 32B Instruct (Yang et al., 2024a)	94	100_{+6}	104_{+10}	89_{-5}	89_{-5}
Qwen2.5 72B Instruct (Yang et al., 2024a)	89	94_{+5}	98_{+9}	84_{-5}	83_{-6}
Phi 4 (Abdin et al., 2024)	89	88_{-1}	92_{+3}	83_{-6}	82_{-7}
Ministral 8B Instruct 2410 (Jiang et al., 2024)	101	99_{-2}	99_{-2}	95_{-6}	94_{-7}
Llama 3.3 70B Instruct (Meta AI, 2024b)	91	96_{+5}	104_{+13}	86_{-5}	85_{-6}
Llama 4 Scout 17B 16e Instruct (Meta AI, 2025)	90	96_{+6}	104_{+14}	88_{-2}	87_{-3}
Gemma 3 1B Instruct (Gemma Team et al., 2025)	113	126_{+13}	127_{+14}	113_{-0}	108_{-5}
Gemma 3 4B Instruct (Gemma Team et al., 2025)	106	123_{+17}	128_{+22}	100_{-6}	99_{-7}
Gemma 3 12B Instruct (Gemma Team et al., 2025)	103	118_{+15}	121_{+18}	99_{-4}	99_{-4}
Gemma 3 27B Instruct (Gemma Team et al., 2025)	100	116_{+16}	121_{+21}	98_{-2}	97_{-3}

Table 11. SelfReflect score ↓ ($\times 10^{-3}$, rounded for readability) for the TriviaQA dataset, averaged across 1000 questions. The results in small font are relative to *Greedy*.

Model	Single-decoding methods			Sample & summarize	
	Greedy	Basic	CoT	$N = 10$	$N = 20$
Qwen2.5 0.5B Instruct (Yang et al., 2024a)	97	96_{-1}	96_{-1}	98_{+1}	98_{+1}
Qwen2.5 1.5B Instruct (Yang et al., 2024a)	93	94_{+1}	93_{-0}	87_{-6}	87_{-6}
Qwen2.5 3B Instruct (Yang et al., 2024a)	97	99_{+2}	100_{+3}	91_{-6}	90_{-7}
Qwen2.5 7B Instruct (Yang et al., 2024a)	95	98_{+3}	100_{+5}	91_{-4}	91_{-4}
Qwen2.5 14B Instruct (Yang et al., 2024a)	88	93_{+5}	94_{+6}	85_{-3}	84_{-4}
Qwen2.5 32B Instruct (Yang et al., 2024a)	92	98_{+6}	101_{+9}	89_{-3}	89_{-3}
Qwen2.5 72B Instruct (Yang et al., 2024a)	87	89_{+2}	90_{+3}	84_{-3}	83_{-4}
Phi 4 (Abdin et al., 2024)	89	88_{-1}	89_{-0}	84_{-5}	83_{-6}
Ministral 8B Instruct 2410 (Jiang et al., 2024)	104	103_{-1}	103_{-1}	99_{-5}	98_{-6}
Llama 3.1 70B Instruct (Meta AI, 2024a)	89	88_{-1}	89_{-0}	85_{-4}	85_{-4}
Llama 3.3 70B Instruct (Meta AI, 2024b)	91	93_{+2}	95_{+4}	89_{-2}	88_{-3}
Llama 4 Scout 17B 16e Instruct (Meta AI, 2025)	89	92_{+3}	96_{+7}	87_{-2}	86_{-3}
Gemma 3 1B Instruct (Gemma Team et al., 2025)	112	127_{+15}	125_{+13}	113_{+1}	108_{-4}
Gemma 3 4B Instruct (Gemma Team et al., 2025)	101	114_{+13}	118_{+17}	96_{-5}	94_{-7}
Gemma 3 12B Instruct (Gemma Team et al., 2025)	95	103_{+8}	107_{+12}	94_{-1}	93_{-2}
Gemma 3 27B Instruct (Gemma Team et al., 2025)	91	99_{+8}	104_{+13}	91_{-0}	91_{-0}

Table 12. Runtime of generating different summaries in seconds per prompt per GPU, averaged across all three datasets with 1000 questions each. Note that some models are sharded across multiple GPUs – in this case, to represent their total computational requirements, we report the summed runtime of all GPUs, i.e., although a prompt may run through in one second using four GPUs, we will count it as four seconds total. In small font, relative comparisons w.r.t. *Greedy*.

Model	Single-decoding methods			Sample & summarize
	Greedy	Basic	CoT	$N = 10$
Qwen2.5 0.5B Instruct (Yang et al., 2024a)	0.93	$1.26_{\times 1.35}$	$1.26_{\times 1.36}$	$1.79_{\times 1.93}$
Qwen2.5 1.5B Instruct (Yang et al., 2024a)	0.94	$0.91_{\times 0.96}$	$1.29_{\times 1.37}$	$1.89_{\times 2.01}$
Qwen2.5 3B Instruct (Yang et al., 2024a)	0.84	$0.85_{\times 1.02}$	$1.00_{\times 1.20}$	$1.82_{\times 2.18}$
Qwen2.5 7B Instruct (Yang et al., 2024a)	0.80	$0.84_{\times 1.05}$	$1.09_{\times 1.36}$	$1.91_{\times 2.38}$
Qwen2.5 14B Instruct (Yang et al., 2024a)	0.96	$1.01_{\times 1.05}$	$1.33_{\times 1.38}$	$2.44_{\times 2.53}$
Qwen2.5 32B Instruct (Yang et al., 2024a)	1.11	$1.22_{\times 1.09}$	$1.72_{\times 1.54}$	$3.42_{\times 3.07}$
Qwen2.5 72B Instruct (Yang et al., 2024a)	1.68	$1.63_{\times 0.97}$	$3.48_{\times 2.07}$	$4.41_{\times 2.62}$
Phi 4 14B (Abdin et al., 2024)	1.09	$1.02_{\times 0.94}$	$1.43_{\times 1.31}$	$2.96_{\times 2.71}$
Ministral 8B Instruct 2410 (Jiang et al., 2024)	0.91	$0.91_{\times 1.00}$	$1.04_{\times 1.15}$	$1.82_{\times 2.00}$
Llama 3.1 70B Instruct (Meta AI, 2024a)	4.16	$4.45_{\times 1.07}$	$10.58_{\times 2.54}$	$10.59_{\times 2.55}$
Llama 3.3 70B Instruct (Meta AI, 2024b)	3.54	$3.17_{\times 0.89}$	$4.25_{\times 1.20}$	$7.23_{\times 2.04}$
Llama 4 Scout 17B 16e Instruct (Meta AI, 2025)	3.45	$3.92_{\times 1.13}$	$5.67_{\times 1.64}$	$9.05_{\times 2.62}$
Gemma 3 1B Instruct (Gemma Team et al., 2025)	0.89	$0.86_{\times 0.97}$	$0.93_{\times 1.04}$	$1.74_{\times 1.96}$
Gemma 3 4B Instruct (Gemma Team et al., 2025)	0.95	$0.93_{\times 0.98}$	$1.09_{\times 1.15}$	$1.96_{\times 2.06}$
Gemma 3 12B Instruct (Gemma Team et al., 2025)	1.12	$1.14_{\times 1.02}$	$1.54_{\times 1.38}$	$2.43_{\times 2.17}$
Gemma 3 27B Instruct (Gemma Team et al., 2025)	1.32	$1.36_{\times 1.03}$	$2.26_{\times 1.71}$	$3.09_{\times 2.34}$

Table 13. Character length of different summaries in seconds per prompt per GPU, averaged across all three datasets with 1000 questions each. In small font, relative comparisons w.r.t. *Greedy*.

Model	Single-decoding methods			Sample & summarize
	Greedy	Basic	CoT	$N = 10$
Qwen2.5 0.5B Instruct (Yang et al., 2024a)	130.10	$699.93_{\times 5.38}$	$896.85_{\times 6.89}$	$155.39_{\times 1.19}$
Qwen2.5 1.5B Instruct (Yang et al., 2024a)	152.44	$151.44_{\times 0.99}$	$528.68_{\times 3.47}$	$310.16_{\times 2.03}$
Qwen2.5 3B Instruct (Yang et al., 2024a)	81.55	$168.60_{\times 2.07}$	$254.03_{\times 3.11}$	$274.03_{\times 3.36}$
Qwen2.5 7B Instruct (Yang et al., 2024a)	93.16	$149.61_{\times 1.61}$	$198.33_{\times 2.13}$	$178.72_{\times 1.92}$
Qwen2.5 14B Instruct (Yang et al., 2024a)	92.93	$170.49_{\times 1.83}$	$245.97_{\times 2.65}$	$203.53_{\times 2.19}$
Qwen2.5 32B Instruct (Yang et al., 2024a)	64.53	$145.30_{\times 2.25}$	$186.80_{\times 2.89}$	$138.33_{\times 2.14}$
Qwen2.5 72B Instruct (Yang et al., 2024a)	97.30	$157.04_{\times 1.61}$	$247.99_{\times 2.55}$	$177.61_{\times 1.83}$
Phi 4 14B (Abdin et al., 2024)	124.85	$203.35_{\times 1.63}$	$281.96_{\times 2.26}$	$273.05_{\times 2.19}$
Ministral 8B Instruct 2410 (Jiang et al., 2024)	49.15	$100.54_{\times 2.05}$	$179.57_{\times 3.65}$	$130.08_{\times 2.65}$
Llama 3.1 70B Instruct (Meta AI, 2024a)	168.64	$267.63_{\times 1.59}$	$499.54_{\times 2.96}$	$225.05_{\times 1.33}$
Llama 3.3 70B Instruct (Meta AI, 2024b)	113.47	$229.45_{\times 2.02}$	$277.31_{\times 2.44}$	$152.70_{\times 1.35}$
Llama 4 Scout 17B 16e Instruct (Meta AI, 2025)	132.81	$275.57_{\times 2.07}$	$220.68_{\times 1.66}$	$214.80_{\times 1.62}$
Gemma 3 1B Instruct (Gemma Team et al., 2025)	22.66	$38.91_{\times 1.72}$	$183.15_{\times 8.08}$	$61.71_{\times 2.72}$
Gemma 3 4B Instruct (Gemma Team et al., 2025)	26.45	$80.66_{\times 3.05}$	$113.46_{\times 4.29}$	$101.16_{\times 3.83}$
Gemma 3 12B Instruct (Gemma Team et al., 2025)	28.51	$102.76_{\times 3.60}$	$174.70_{\times 6.13}$	$70.67_{\times 2.48}$
Gemma 3 27B Instruct (Gemma Team et al., 2025)	37.77	$137.62_{\times 3.64}$	$192.39_{\times 5.09}$	$70.97_{\times 1.88}$

I. Experiment details of CoT deep dive

We used Gemini 2.0 Flash to judge if a summary mentions multiple semantically different answers or not, and likewise if samples from the distributions do so. In particular, we provide it with the corresponding context, and ask it to classify into three classes: A) Fully certain, B) not fully certain, C) Completely uncertain. We then aggregate B and C to form the general "uncertain" category (we found this to give better results than to directly give a binary task to Gemini, because it allows us to explicitly tell how to handle the "not fully certain" edge case). We manually verified that the classification is qualitatively correct. We show the prompts below.

Gemini 2.0 Flash prompt to classify *CoT* summary in Figure 4. We call A. 'Certain' and group B. and C. into 'Uncertain'.

```
Below, you are given an answer to the question '{question}'.

Your goal is to classify which type of answer this is:
A. The answer is certain, it only mentions one answer option.
B. The answer is not fully certain.  It might mention one or two further answer
options but judges them as less likely.
C. The answer is very uncertain.  It mentions many mutually exclusive answer options,
without a clear single most likely answer.

Ignore differences in form and style.  You are only supposed to judge the answer
semantically.

Here is the answer:  {CoT summary}

Please respond with the category of what type of this answer this is.  Respond only
with A, B, or C.
```

Gemini 2.0 Flash prompt to classify answer distribution samples in Figure 4. We call A. 'Certain' and group B. and C. into 'Uncertain'

```
Below, you are given {n_answers} individual answers to the question '{question}'.
These {n_answers} answers can be seen as samples from an answer distribution.

Your goal is to classify which type of distribution this is:
A. The answers all do not contradict each other, up to one or two that differ from
the majority answer.
B. The answers give multiple mutually exclusive answer options, but there is one
answer option that is given in the majority of cases.
C. The answers give multiple mutually exclusive answer options, and they are almost
all different, without a clear majority answer.

The answers will have some natural variability.  Ignore differences in form and
style.  You are only supposed to judge if answer options are semantically different.

Here are the {n_answers} answers:
x_1 = '{answer}'
...
x_{n_answers} = '{answer}'

Please respond with the category of what type of this distribution this is.  Respond
only with A, B, or C.
```