Think Twice Before Trusting: Self-Detection for Large Language Models through Comprehensive Answer Reflection

Anonymous ACL submission

Abstract

Self-detection for Large Language Model (LLM) seeks to evaluate the LLM output trustability by leveraging LLM's own capabilities, alleviating the output hallucination issue. However, existing self-detection approaches only retrospectively evaluate answers generated by LLM, typically leading to the over-trust in incorrectly generated answers. To tackle this limitation, we propose a novel self-detection paradigm that considers the comprehensive answer space beyond LLM-generated answers. It thoroughly compares the trustability of multiple candidate answers to mitigate the over-trust in LLM-generated incorrect answers. Building upon this paradigm, we introduce a two-step framework, which firstly instructs LLM to reflect and provide justifications for each candidate answer, and then aggregates the justifications for comprehensive target answer evaluation. This framework can be seamlessly integrated with existing approaches for superior self-detection. Extensive experiments on six datasets spanning three tasks demonstrate the effectiveness of the proposed framework.

1 Introduction

Large Language Model (LLM) typically suffers from the hallucination issue, (Zhang et al., 2023c; Li et al., 2023a; Golovneva et al., 2022; Bang et al., 2023), which significantly undermines the trustability of LLM's outputs. A promising research direction for evaluating the output trustability and identifying incorrect outputs is self-detection (Zhao et al., 2023c; Miao et al., 2023; Manakul et al., 2023). Given a question, self-detection aims to leverage LLM's own ability to evaluate the trustability of its generated answers, without relying on external knowledge sources or specifically trained detection models. This paper investigates self-detection methods tailored for black-box API LLMs due to their excellent performance and the inherent challenge posed by limited output information (Achiam et al., 2023; OpenAI, 2024).

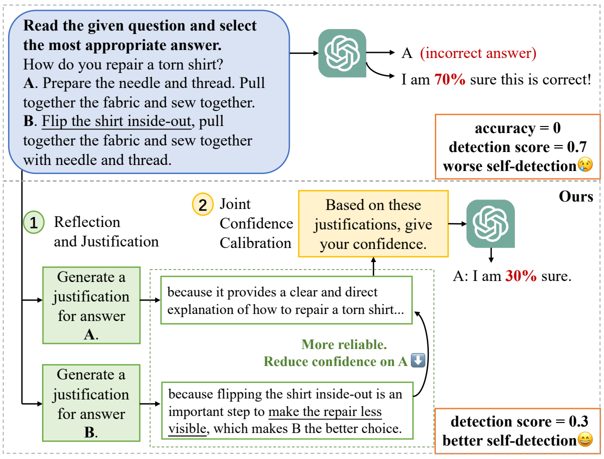

Figure 1: An illustration of *Think Twice before Trusting* framework for mitigating the over-trust issue in LLM self-detection. LLM is instructed to reflect and generate justification on the trustability of each answers before evaluating the trustability of the target answer.

Previous studies in self-detection can be broadly categorized into two paradigms (*cf.* Figure 2). The first paradigm is confidence calibration, aiming to estimate LLM's confidence on the generated answer to align with the actual answer accuracy via multi-answer sampling and aggregation (Xiong et al., 2023; Tian et al., 2023b; Si et al., 2022; Jiang et al., 2023). The second one is self-evaluation, which directly examines the compatibility of question and answer by designing various prompt strategies (Miao et al., 2023; Kadavath et al., 2022; Weng et al., 2023). These two paradigms have also been combined to enhance self-detection capabilities (Chen and Mueller, 2023; Ren et al., 2023a).

However, both self-detection paradigms have shown a significant drawback: an inclination towards over-trusting the incorrect answers generated by LLM (Si et al., 2022; Xiong et al., 2023; Jiang et al., 2023; Kadavath et al., 2022). We argue that one reason may be that both paradigms merely evaluate LLM-generated answers, while LLM contains an inherent bias towards trusting its own generations (Mielke et al., 2022; Lin et al., 2022), leading to serious over-trust in LLM-generated incorrect answers. An ideal self-detection paradigm should consider a more comprehensive answer space beyond LLM's generations. By evaluating on other potentially correct answers in a broader answer space, the strong trustability in these answers can counterbalance the excessive trust in the incorrect LLM answers, thus alleviating the over-trust issue.

In this light, we introduce a new comprehensive answer evaluation paradigm involving the consideration of multiple candidate answers in the answer space to enhance self-detection (*cf.* Figure 2). This paradigm meticulously evaluates each answer's trustability as a correct answer to the question and aggregates these evaluations to enhance the self-detection of the target LLM answer. The biased trust in the LLM-generated incorrect answers can be alleviated through the trustability comparison with other more trustable answers. Our preliminary experiments reveal the efficacy of considering more comprehensive answers to confront over-trust (*cf.* Section 2). To summarize, two key considerations arise to instantiate this new paradigm: 1) resisting the inherent bias of LLM to precisely evaluate the trustability of each question-answer pair, and 2) aggregating these evaluations in the trustability evaluation of the target answer.

To this end, we present a novel self-detection framework to tackle the over-trust issue of LLMs, named Think Twice before Trusting (T^3) (*cf.* Figure 1). Our framework pushes LLM to reflect and justify from different answers' perspectives before arriving at the trustability on the target answer. Firstly, the LLM is instructed to generate justifications regarding the potential correctness of each answer. Subsequently, a prompt-based method is employed to integrate these justifications into joint evaluation for the target answer. Extensive experiments on six datasets across three tasks on three different LLMs show improved performance of T^3 over methods from existing paradigms. Notably, T^3 can be combined with existing methods for superior self-detection. Our analysis also reveals T^3's strong robustness and effective over-trust mitigation. Our contributions are three-fold.

- We introduce a novel self-detection paradigm for mitigating the over-trust issue in LLM, addressing the limitation of existing paradigms by reflection in the broader answer space.
- We present a novel T^3 framework to implement the comprehensive answer evaluation paradigm, which can be seamlessly integrated with existing self-detection methods.
- We conduct extensive experiments on three NLP tasks with six datasets, validating the rationality and effectiveness of the proposed framework.

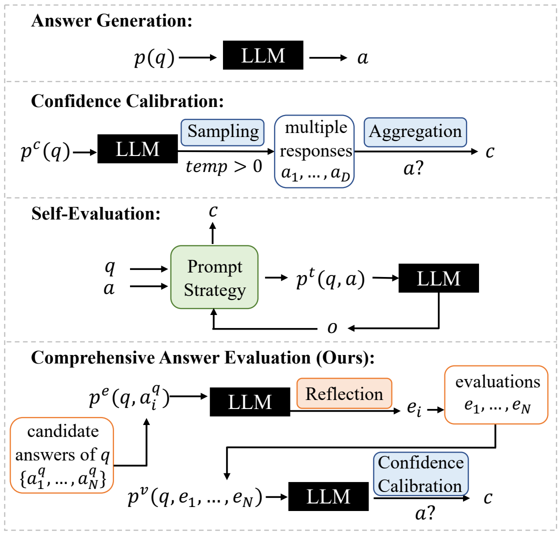

Figure 2: Two existing paradigms of self-detection and our new comprehensive answer evaluation paradigm.

2 Problem Formulation

LLM Self-Detection. We formulate the task of self-detection for LLM as follows. Given the input comprising of question q combined with prompt p, which consists of an instruction and optional in-context examples, LLM can generate the answer a (Brown et al., 2020), denoted as the target answer. Thereafter, self-detection aims to evaluate the trustability of a by LLM's own ability, generally in the form of a detection score $c \in \mathcal{R}$ [1]. The detection score c can be used for indicating the actual accuracy of a, where low c values indicate potential incorrect answers. Denoting the self-detection strategy as a function $SD(\cdot)$, this process can be abstracted as

$$a = LLM(p(q)), \tag{1}$$

$$c = SD(LLM(\cdot), q, a). \tag{2}$$

In the following, we illustrate the existing two paradigms for self-detection, *i.e.,* confidence calibration and self-evaluation, and introduce our proposed comprehensive answer evaluation paradigm.

Confidence Calibration. Confidence calibration aims to estimate LLM's level of certainty on the answer a, *e.g.,* estimating the LLM output probability

[1] If the result of self-detection is a class label (trustable or untrustable), the detection score can be formulated as 1 or 0.

of a, where the obtained confidence score as the detection score c aims to calibrate with the actual answer accuracy. Xiong et al. conclude a general three-step confidence calibration process for LLM as *prompting, answer sampling, aggregation* (*cf.* Figure 2). Denoting the prompt for confidence calibration as $p^c(\cdot)$ and the aggregation function as $Aggr(\cdot)$, this paradigm can be abstracted as,

$$c = Aggr(a, \{a_1, ..., a_D\}), \quad (3)$$
$$\text{where } a_i = LLM(p^c(q)), i \in \{1, ..., D\}.$$

where $D > 1$ refers to the number of sampled answers. For example, self-consistency (Wang et al., 2022; Si et al., 2022) aggregates the probability of a in the sampled outputs of $p(q)$ (*e.g.,* using nucleus sampling (Holtzman et al., 2020)). Formally,

$$c = \frac{\sum_{i=1}^{D} \mathbb{1}(a_i = a)}{D}, \quad (4)$$
$$\text{where } a_i = LLM(p(q)), i \in \{1, ..., D\}.$$

Besides, the Top-K verbalized methods (Lin et al., 2022; Tian et al., 2023b) leverage a well-designed prompt p^b (*cf.* Appendix A.3) to instruct the LLM to sample the K most likely answers and output their corresponding probabilities in one response:

$$[\{a_1, c_1\}, ...\{a_K, c_K\}] = LLM(p^b(q)). \quad (5)$$

where $[\cdot]$ denotes the concatenation of the K most likely answers with their probabilities. The probability of a in the response is utilized as its detection score c ($c = 0$ if a is not in the K answers).

However, confidence calibration methods are observed with severe over-trust issue on LLM, assigning high confidence score in some incorrectly generated answers (Si et al., 2022; Xiong et al., 2023). In fact, LLM has a bias to blindly trust its generated answers, leading to difficulties in distinguishing the correctness of its generated answers (Huang et al., 2023b; Ling et al., 2023; Mielke et al., 2022; Ren et al., 2023b). Although some attempts have been made to reduce high confidence in LLM and achieve better calibration (Jiang et al., 2023; Zhao et al., 2024), the over-trust issue still remains a severe problem towards effective self-detection.

Self-Evaluation. Self-evaluation methods concatenate q and a and leverage various designed prompts to instruct LLM in self-evaluating the correctness of a from different perspectives. The prompt strategy examines the matching of q, a by integrating the self-evaluation output o. Denoting the prompt strategy as a function $ES(\cdot)$, this process can be summarized as

$$c = ES(q, a, o), \quad (6)$$
$$\text{where } o = LLM(p^t(q, a)).$$

where p^t represents one prompt for self-evaluation.

The shortcoming of self-evaluation is that many approaches under this paradigm are specifically designed for the mathematical question answering task, including step-wise checking on Chain-of-Thoughts (CoT) reasoning (Miao et al., 2023), completing masked q using a (Weng et al., 2023), and natural program (Ling et al., 2023), limiting its applicability. The general method P(True) (Kadavath et al., 2022) is straightforward and still demonstrates over-trust to incorrect LLM-generated answers. It directly asks LLM whether a is the true answer to q via the prompt p^r (*cf.* Appendix A.3), and uses the probability of "True" in the sampled responses as c. Formally,

$$c = \frac{\sum_{i=1}^{D} \mathbb{1}(o_i = \mathit{True})}{D}, \quad (7)$$
$$\text{where } o_i = LLM(p^r(q, a)), i \in \{1, ..., D\}.$$

The two paradigms can be combined for better self-detection (Xiong et al., 2023; Chen and Mueller, 2023; Ren et al., 2023a; Agrawal et al., 2023).

A New Comprehensive Answer Evaluation Paradigm. A notable limitation of the existing two paradigms is that their evaluation merely involves LLM-generated answers a_i, in which LLM may exhibit over-trust. We argue that such biased over-trust could be alleviated if LLM had thoroughly compared the trustability of more candidate answers of q beyond LLM-generated answers. We consider the multi-choice question answering setting where a comprehensive answer space is provided. [2] If other answers in q's answer space had a strong tendency to be correct, the high detection score for LLM-generated incorrect a could be diminished, reducing the over-trust issue.

In this light, we propose a novel comprehensive answer evaluation paradigm that considers N potential answers in q's answer space, denoted as $\{a_1^q, a_2^q, ..., a_N^q\}$. First, LLM evaluates the trustability of each (q, a_i^q) pair using the prompt p^e.

$$e_i = LLM(p^e(q, a_i^q)), i \in \{1, ..., N\}. \quad (8)$$

[2] For other settings, the answer space can be obtained via answer retrieval or additional model prediction.

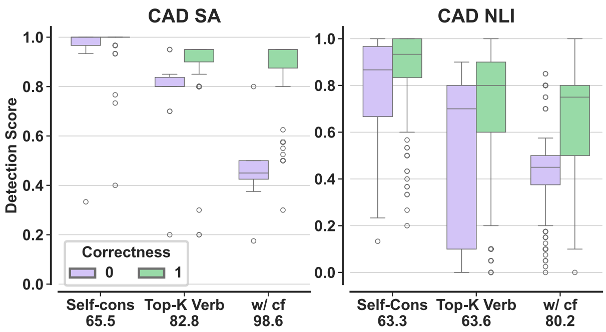

Figure 3: Comparison of self-detection methods on CAD. *w/ cf* denotes our strategy with counterfactual data. The AUROC values are shown in the x-axis. The boxes on the left and right represent the detection scores of incorrect and correct instances, respectively.

Then, the obtained evaluations $e_1, ..., e_N$ can be integrated into existing paradigms to derive a more refined detection score for a by adjusting the prompts. For example, by adjusting the p^c for confidence calibration (*cf.* Eq 3) into a new prompt p^v, the detection score for a can be derived as,

$$c = Aggr(a, \{a_1, ..., a_i\}), \tag{9}$$

where $a_i = LLM(p^v(q, e_1, ..., e_N)), i \in \{1, ..., D\}$.

The evaluations can also be integrated into self-evaluation by adjusting the prompt p^t in Eq 6.

Preliminary Experiments. We conduct a preliminary experiment to validate that considering more answers in the answer space to adjust the detection score is beneficial for self-detection.

Our hypothesis is that the evaluation of other answers can be leveraged to mitigate over-trust in the incorrect a. To demonstrate this, we employ counterfactual questions $\bar{q}$, which is minimally edited from q to have a different label within q's answer space. We aim to utilize the label difference between q and $\bar{q}$ to identify unreliable LLM-generated answer for q and adjust its detection score. Suppose the LLM-generated answers for $\bar{q}$ and q are $\bar{a}$ and a, respectively. If $\bar{a}$ equals a, a and $\bar{a}$ must have at least one wrong answer since $\bar{q}$ and q have different labels. Thus the detection score of a on $p(q)$ (denoted as c_a) should be reduced according to the detection score of $\bar{a}$ on $p(\bar{q})$ (denoted as $c_{\bar{a}}$) because the increasing of $c_{\bar{a}}$ indicates the weakened c_a. Conversely, if $\bar{a}$ differs from a, a and $\bar{a}$ are relatively trustable, and c_a can be an average of itself and $c_{\bar{a}}$. Formally, c_a is re-calculated as

$$c = \begin{cases} \frac{1}{2}(c_a + c_{\bar{a}}) & \text{if } a \neq \bar{a}, \\ \frac{1}{2}(c_a + O(c_{\bar{a}})) & \text{else.} \end{cases} \tag{10}$$

where $O(c_{\bar{a}})$ denotes the detection score that $\bar{q}$'s label is not $\bar{a}$. In a k-classification task, we roughly estimate $O(c_{\bar{a}}) = \frac{1}{k-1}(1 - c_{\bar{a}})$.

We experiment with the CAD dataset (Kaushik et al., 2019), which contains human-annotated original and counterfactual data pairs for sentiment analysis (SA) and natural language inference (NLI) tasks. We compare the AUROC with self-consistency and Top-K verbalized methods to evaluate the self-detection performance on GPT-3.5 (see Section 5 and Appendix B for more details).

Figure 3 shows the AUROC and the statistics of detection scores for correct and incorrect q, a instances, respectively. We can observe that 1) the self-consistency and Top-K verbalized methods have notable over-trust. The detection scores for incorrect instances have large overlap with the correct ones, making it challenging to distinguish them. 2) Our strategy, denoted as *w/ cf*, improves AUROC by lowering detection scores on incorrect instances, showing that considering other answers can potentially alleviate the over-trust in incorrect answers. However, human-annotated counterfactual data is not easily available (Li et al., 2023b), motivating us to propose the following framework.

3 Think Twice Before Assure Framework

Implementing the proposed paradigm involves two key considerations. First, given the potential bias of LLM over-trust in the generated answer a, it is essential to develop strategies to resist this bias and thoroughly evaluate the trustability of each answer a_i^q. Secondly, it is crucial to derive strategies to effectively combine these evaluations for effective self-detection of a. To address these concerns, we introduce the following two-step framework.

Step 1: Reflection and Justification. We first instruct LLM to reflect on the trustability of each answer a_i^q and force LLM to seek justification for a_i^q as the correct answer of q, as defined by Eq. 8. The LLM is instructed with the prompt p^e in Table 1 to gather comprehensive evidence e_i from its knowledge, in order to support the rationality of using a_i^q to answer q. The rationality of this step is that p^e instructs LLM to abduct the justification from q and a_i^q, which avoids the LLM bias that lies in the generation direction from $p(q)$ to a. As a minor clue, generating CoT explanations from $p(q)$ before a has been validated to be ineffective for calibration (Zhang et al., 2023a).

p^e	The task is to [task description]. Question: [q]. Answer choices: [$a_1^q, ..., a_N^q$]. The answer is [a_i^q]. Please generate an explanation to try to justify the answer judgment.
p^v	The task is to [task description]. Provide your N best guesses and the probability that each is correct (0.0 to 1.0) for the following question... Question: [q]. Answer choices: [$a_1^q, ..., a_N^q$]. Possible explanation 1: [e^1] ... Possible explanation n: [e^N]

Table 1: Prompts used in our T^3 framework. p^e prompts LLM to reflect and generate justification e_i for each a_i^q, and p^v prompts LLM to estimate confidence according to different e_i.

Step 2: Joint Confidence Calibration. After obtaining the justification e_i for each a_i^q, we choose to integrate these e_i with a confidence calibration method, the Top-K verbalized (*cf.* Eq. 5) to derive the confidence of answer a as the detection score. We choose this method due to its capability to generate a set of K potential answers and their respective probabilities efficiently in a single response, where we set K as the number of answers N. As indicated in the prompt p^v of Table 1, the generated justifications e_i can be seamlessly integrated for confidence calibration of Top-K verbalized.

An alternative approach to determine the final detection score is to put one justification to each p^v, generating N distinct confidence scores for answer a, and then compute the averaged confidence score as the detection score. We do not choose this setting as prompting LLM to estimate from different perspectives via a unified prompt is more efficient and effective than a simple average of the confidence scores (further validated in Section 5.2). Moreover, we find that the detection scores are sensitive to the order of justification in p^v, thus we shuffle the order of e^i in p^v and compute the averaged score. Notably, the T^3 framework can be combined with existing approaches, such as prompt ensemble (Jiang et al., 2023), and Hybrid method which adjust the detection score based on the difference with other methods (Xiong et al., 2023).

4 Related Work

Confidence Calibration of LLM. Confidence calibration has been previously studies in neural networks (Guo et al., 2017) and applied in NLP models (Desai and Durrett, 2020; Dan and Roth, 2021; Hu et al., 2023). After the advent of LLM, many confidence calibration methods utilize the output token probability, such as semantic uncertainty (Kuhn et al., 2023), temperature scaling (Shih et al., 2023), entropy-based (Huang et al., 2023c), semantic significance (Duan et al., 2023), and fine-tuning for calibration (Jiang et al., 2021; Lin et al., 2022). Zhang et al. (2023b) also employ model ensemble for calibration. Our research is orthogonal to them, since we focus on black-box API LLM itself. Other recent work suitable for black-box LLM includes fidelity elicitation (Zhang et al., 2024), fact elicitation (Zhao et al., 2024) and perturbation generation (Gao et al., 2024).

Self-Evaluation of LLM. LLM self-evaluation often focuses on specific domains, *e.g.,* code generation (Zhou et al., 2023), natural language generation (Lin et al., 2023) and fact checking (Manakul et al., 2023). The general methods include P(True) (Kadavath et al., 2022) and directly asking LLM (Li et al., 2024b). Feng et al. (2024) also performs answer reflection and employs model collaboration, yet they still focus on answers generated by LLM. Note that self-detect (Zhao et al., 2023c) is also a general self-evaluation method following the three-step confidence calibration pipeline.

Other works that are related but orthogonal to us include training independent models for LLM evaluation (Wang and Li, 2023; Li et al., 2023c; Khalifa et al., 2023; Zhao et al., 2023b; Li et al., 2024a), and using external tools for LLM verification (Min et al., 2023; Ni et al., 2023). They are usually applied to specific domains, while we aim at LLM self-detection for general tasks. Also, fine-tuning LLM for better trustability (An et al., 2023; Tian et al., 2023a) is orthogonal to us.

Application of LLM Self-Detection. The outcome of self-detection can be applied in many ways to avoid hallucination and erroneous outputs, such as identifying potentially hallucinated generation for knowledge retrieval and verification (Zhao et al., 2023a), guided output decoding (Xie et al., 2023), identifying ambiguous questions (Hou et al., 2023), selective generation (Ren et al., 2023a; Zablotskaia et al., 2023), and LLM self-improve (Huang et al., 2023a). More applications can be found in this survey (Pan et al., 2023).

5 Experiments

Setup. We conduct experiments on six datasets across three tasks. IMDB (Maas et al., 2011) and

Flipkart (Vaghani and Thummar, 2023) for SA, SNLI (Bowman et al., 2015) and HANS (McCoy et al., 2019) for NLI, CommonsenseQA (Talmor et al., 2019) and PIQA (Bisk et al., 2020) for commonsense question answering (CQA). For LLMs, we utilize GPT-3.5 (*gpt-3.5-turbo-1106*) from OpenAI[3], GLM-4 (Du et al., 2022) from ZhipuAI[4], and Gemini (*gemini-1.0-pro-001*) from Google[5]. Dataset statistics and LLM hyperparameters are listed in Appendices A.1 and A.2.

Compared Methods. We utilize the following categories of compared methods. For the first paradigm, we include **Self-cons** (Wang et al., 2022) (*cf.* Eq. 4), **CoT-cons**, an extension of Self-cons by instructing LLM to output the CoT reasoning before the answer, **Top-K Verb** (Tian et al., 2023b) (*cf.* Eq. 5), **Hybrid** (Xiong et al., 2023), an integration of Top-K Verb and Self-cons/CoT-cons, where we show the better results, **Self-detect** (Zhao et al., 2023c), taking the answer entropy of multiple rephrased questions, and **CAPE** (Jiang et al., 2023), a prompt ensemble method that we implement on Top-K Verb. For the second paradigm, we utilize the general **P(True)** (Kadavath et al., 2022). Finally, to show the flexibility of T^3 in combining with existing methods to further improve self-detection, we show the performance of Hybrid T^3 with Top-K Verb (T^3 **+ Top-K Verb**), and T^3 with prompt ensemble following CAPE (T^3 **+ PE**). For a fair comparison, we generate the target answer for each dataset with LLM temperature as 0, and compare all methods based on this target answer (*cf.* Eq 1). More details are in Appendices A.3 and A.4, including a comparison on the number of API calls showing T^3's reasonable cost.

Evaluation Metrics. We mainly use **AUROC** (Boyd et al., 2013) and **PRAUC** (Manning and Schutze, 1999) to evaluate the self-detection ability. They assess the effectiveness of detection scores in distinguishing answer correctness using true positive/false positive and precision/recall curves, respectively. Additionally, we use the Expected Calibration Error (**ECE**) to evaluate the calibration performance for confidence calibration methods.

5.1 Results

Table 2 shows the performance of the compared methods on GPT-3.5. We can observe the findings. 1) T^3 outperforms all compared methods in AUROC and PRAUC on all datasets except HANS and PIQA, and in ECE on all datasets except SNLI, demonstrating its effectiveness. 2) After combining T^3 with other methods *i.e.,* Top-K Verb and PE, our method surpasses all compared methods on all datasets in the three evaluation metrics, showing the potential and flexibility of T^3 in combining with others to further improve self-detection. 3) Hybrid with Top-K Verb usually improves T^3's performance in AUROC and PRAUC, which is in line with the performance improvement from Self-cons/CoT-cons to Hybrid. 4) CAPE is very competitive in AUROC and PRAUC, showing that the self-detection is largely influenced by the prompt. Combining T^3 with PE usually improves T^3 in AUROC and PRAUC except for SNLI and Flipkart, which is in line with the performance decrease from Top-K Verb to CAPE. This is potentially re-

	IMDB			Flipkart		
	AUROC ↑	PRAUC ↑	ECE ↓	AUROC ↑	PRAUC ↑	ECE ↓
Self-cons	65.5	96.8	0.115	71.4	91.4	0.106
CoT-cons	75.6	97.7	0.104	72.8	91.9	0.133
Top-K Verb	82.8	98.5	0.242	79.3	93.7	0.131
P(True)	80.1	98.1	0.104	54.5	86.7	0.243
Hybrid	87.0	98.8	0.183	79.5	94.2	0.176
Self-detect	68.9	97.1	0.320	71.2	91.4	0.146
CAPE	87.7	**98.9**	0.096	76.4	93.9	0.107
T^3	87.9	**98.9**	**0.045**	81.3	94.5	**0.015**
+ Top-K Verb	88.0	**98.9**	0.052	**81.6**	**94.9**	0.019
+ PE	**88.1**	**98.9**	0.056	74.2	92.9	0.157

(a) SA.

	SNLI			HANS		
	AUROC ↑	PRAUC ↑	ECE ↓	AUROC ↑	PRAUC ↑	ECE ↓
Self-cons	63.3	71.4	0.047	56.0	64.8	0.051
CoT-cons	66.7	73.8	0.043	59.4	67.9	0.152
Top-K Verb	63.6	74.0	0.089	53.3	64.9	0.273
P(True)	55.4	67.4	0.117	60.8	70.1	0.067
Hybrid	66.7	78.8	0.029	62.0	71.1	0.193
Self-detect	59.3	68.5	0.142	55.3	64.5	0.063
CAPE	69.0	79.6	0.030	71.9	80.1	0.028
T^3	**77.9**	84.6	0.157	69.9	77.5	**0.022**
+ Top-K Verb	77.1	**84.7**	**0.024**	71.3	79.6	0.030
+ PE	70.8	76.7	0.130	**74.5**	**81.2**	0.034

(b) NLI.

	CommonsenseQA			PIQA		
	AUROC ↑	PRAUC ↑	ECE ↓	AUROC ↑	PRAUC ↑	ECE ↓
Self-cons	70.7	81.7	0.151	78.6	94.0	0.043
CoT-cons	81.8	88.9	0.049	76.7	94.2	0.097
Top-K Verb	69.4	81.5	0.026	76.8	93.3	0.060
P(True)	62.5	78.0	0.097	71.9	93.9	0.176
Hybrid	77.5	89.0	0.015	82.4	95.5	0.088
Self-detect	67.9	81.5	0.261	68.5	91.0	0.161
CAPE	78.7	88.8	0.021	87.9	97.8	0.067
T^3	83.5	90.7	**0.009**	83.4	95.2	0.016
+ Top-K Verb	**85.8**	**93.4**	0.017	85.3	96.2	**0.010**
+ PE	84.4	92.1	0.019	**90.3**	**97.9**	0.034

(c) CQA.

Table 2: Results of the compared methods on GPT-3.5. Bold font and underline indicate the best and second best performance, respectively.

[3] https://openai.com/blog/openai-api.
[4] https://open.bigmodel.cn/.
[5] https://gemini.google.com/app.

	IMDB		Flipkart		SNLI		IMDB		Flipkart		SNLI	
	AUROC↑	PRAUC↑	AUROC↑	PRAUC↑	AUROC ↑	PRAUC ↑	AUROC↑	PRAUC↑	AUROC↑	PRAUC↑	AUROC ↑	PRAUC ↑
T^3	**87.9**	**98.9**	**81.3**	**94.5**	**77.9**	**84.6**	**69.9**	**77.5**	**83.9**	90.9	83.4	95.2
w/ CoT expl	72.4	97.5	76.6	93.4	67.1	75.2	53.7	64.1	78.7	86.8	81.3	94.8
sep expl	86.5	98.8	79.5	94.2	68.5	75.3	54.1	63.8	83.3	**92.0**	**84.0**	**95.8**
w/o shuffle	75.9	98.3	71.7	92.0	70.6	77.6	60.7	67.9	80.3	87.8	80.4	94.3

Table 3: Ablation studies.

lated to the prompt sensitivity of these methods and the specific prompts adopted. 5) For other methods, CoT-cons outperforms Self-cons in AUROC and PRAUC in 5 out of 6 datasets, as many tasks performs better with CoT reasoning. P(True) has ambivalent results which limits its applicability.

5.2 In-depth Analysis

Ablation Studies. We conduct the following ablation studies to further validate the rationality of our framework design. 1) ***w/ CoT expl***: substituting $e_1, ..., e^N$ in p^v with N different CoT reasoning generated from $p(q)$ to reveal the rationality of reflection on various answers. 2) ***sep expl***: placing a single e_i in p^v each time and calculating the averaged detection score to reveal the effectiveness of joint considering all e_i in one p^v. 3) ***w/o shuffle***: ablating the order shuffling of e_i in p^v.

From Table 3, we can observe that: 1) *w/ CoT expl* largely underperforms T^3 on all three tasks, demonstrating the rationality of pushing LLM to reflect and justify from each answer's perspective. 2) *sep expl* underperforms T^3 on both SA and NLI tasks, showing that jointly considering multiple justifications in one prompt is often more beneficial, and thus we choose this setting. It slightly outperforms T^3 on the CQA task, potentially due to the higher independency and objectivity of the answer choices. 3) *w/o shuffle* also underperforms T^3, indicating that there exists order sensitivity for e_i. Order shuffling and score average improve self-detection by mitigating their position bias.

Effect on Bias Mitigation. Since our goal of improving self-detection is to reduce the over-trust on incorrect answers, we show the statistics of the detection scores for each dataset regarding the answer correctness in Figure 4 to reveal the mechanism of T^3. We compare T^3 with Self-cons and Top-K Verb which are witnessed with over-trust. We can observe that T^3 clearly reduces the detection score overlaps between correct and incorrect q, a instances on all datasets, and significantly decreases the detection scores on incorrect instances in IMDB, Flipkart, SNLI and HANS. Thus, the answer accuracy is more separable by the detection score, achieving better self-detection.

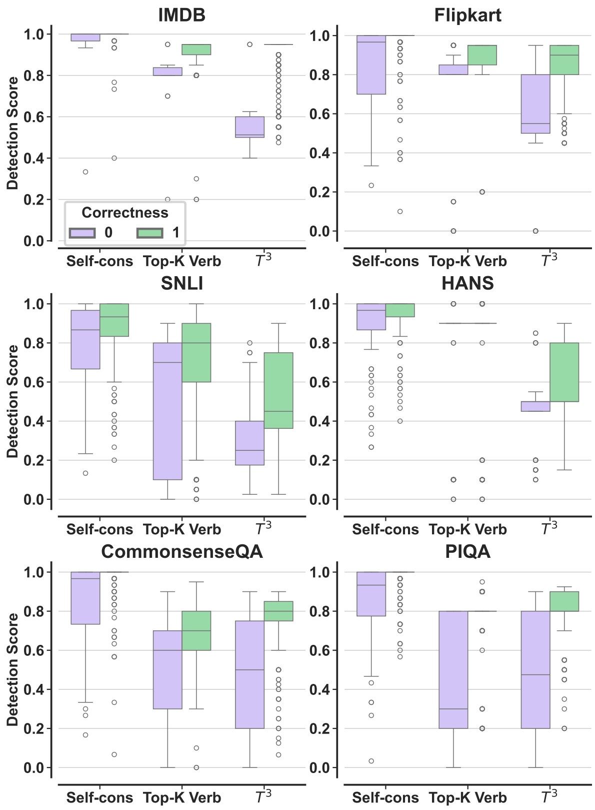

Figure 4: Visualization of bias mitigation effect of T^3 which largely reduces the detection score overlaps between correct (right) and incorrect (left) instances.

Effect on Selective Prediction via Detection Score. To show the utility of the detection score, we conduct experiments in selective prediction. The idea of selective prediction is to abstain the LLM-generated answers with low detection score to maintain better accuracy of the remaining instances. In Figure 5, we show the accuracy of the remaining instances by abstaining 0% - 50% of instances with the lowest detection scores from T^3. We can observe that by increasing the percentage of abstained instances, the accuracy for these datasets gradually improves around 10% - 30%, and IMDB even achieves 100% accuracy. Naturally, the increase for datasets with lower accuracy is generally easier than datasets with higher accuracy. The result shows that T^3 possess strong potential to be applied in selective prediction scenarios.

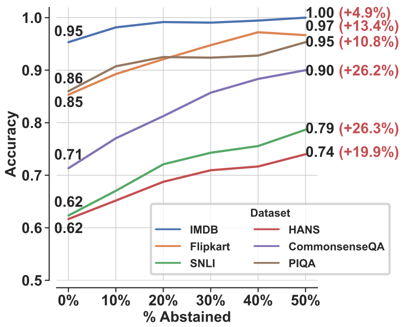

Figure 5: Accuracy improvement of selective prediction on T^3 detection scores.

		Flipkart	HANS	CommonsenseQA
a^{sc}	Self-cons	72.7	52.7	68.2
	CoT-cons	74.4	57.5	80.4
	Top-K Verb	80.4	51.8	69.2
	T^3	**82.2**	**69.5**	**82.7**
a^{cc}	Self-cons	78.3	57.0	68.1
	CoT-cons	79.2	57.8	74.3
	Top-K Verb	83.9	53.3	67.5
	T^3	**84.3**	**69.2**	**75.0**

Table 4: AUROC on two different target answers.

	Flipkart		HANS		CommonsenseQA	
	AUROC↑	PRAUC↑	AUROC↑	PRAUC↑	AUROC ↑	PRAUC ↑
CoT Cons	73.4	88.8	66.4	87.5	83.1	97.0
Top-K Verb	81.1	92.1	65.4	88.0	72.3	95.3
Hybrid	80.4	92.0	69.9	89.4	79.4	97.2
CAPE	82.3	92.7	82.4	94.0	80.0	96.8
T^3	83.3	**93.4**	82.0	93.9	72.5	96.0
+ Top-K Verb	82.7	93.2	80.9	93.9	**81.0**	**97.6**
+ PE	**83.8**	**93.4**	**84.9**	**95.7**	76.9	96.6

Table 5: Performance comparison of Flipkart, HANS and CommonsenseQA on GLM-4.

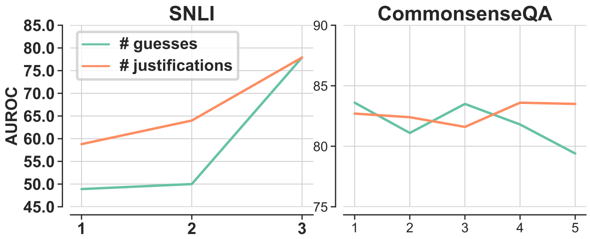

Figure 6: Parameter sensitivity, *i.e.,* changing the number of justifications and number of guesses in p^v.

Analysis on the Robustness of T^3. We evaluate the robustness of T^3 from three aspects: different target answers, different LLMs, and parameter sensitivity. In addition, we examine prompt sensitivity of p^e and p^v in Appendix C.

Firstly, the generation of target answer a may vary under LLM randomness, *e.g.,* setting the temperature greater than 0. We verify the robustness of T^3 by utilizing **different target answers**, *i.e.,* the majority answer of Self-cons (a^{sc}) and CoT-cons (a^{cc}), respectively, as shown in Table 4. We can observe the following. 1) For both sets of target answers, T^3 largely outperforms baselines, showing its effectiveness. 2) Different target answers may have very different self-detection performance. Specifically, a^{cc} on CommonsenseQA has a sharp decrease in AUROC of T^3 and CoT-cons compared with the other target answers, which is potentially due to the majority voting with CoT explanation diminished the the effect of the explanations in T^3.

Secondly, we evaluate T^3 on **different LLMs**. Table 5 shows the performance comparison of Flipkart, HANS and CommonsenseQA on GLM-4. We can observe that across different LLMs, combining T^3 with PE or Top-K Verb outperforms compared methods, validating its effectiveness. Besides, the self-detection ability may vary greatly across LLMs, *e.g.,* T^3's AUROC of HANS on GLM-4 largely outperforms that on GPT-3.5. More results on Gemini can be found in Appendix D.

Thirdly, we evaluate the **parameter sensitivity** of T^3 by changing the number of justifications and number of guesses in p^v. We conduct experiments on CommonsenseQA with five answer choices, and SNLI with three answer choices. From Figure 6, we can observe the followings. 1) A larger number of justifications increases the performance on both datasets, indicating a sufficient number of justifications is vital for better self-detection. 2) Increasing the number of guesses results in a significant performance improvement on the SNLI dataset, revealing that enough number of guesses is demanded for the NLI task. 3) Comparably, the change in the number of guesses has a slight effect on the performance of the CommonsenseQA dataset, which is potentially because the CQA task is more objective than NLI.

6 Conclusion

In this paper, we tackled the over-trust issue of self-detection on black-box API LLMs. We categorized existing methods into two paradigms and pointed out their limitation of merely evaluating on LLM-generated answer with potential LLM over-trust. We proposed a novel paradigm to address this limitation by comprehensively evaluating the trustability of multiple candidate answers in the answer space. Following our paradigm, we presented a two-step framework T^3 by asking LLM to reflect and justify the trustability of each answer for joint confidence calibration. Our framework achieved improved self-detection performance over compared methods and was combined with existing methods for further improvement. In future work, we will explore the combination of T^3 with more methods, and its utility in white-box LLMs.

Limitations

Our work has several limitations. Firstly, our research scope is limited to the self-detection for black-box API LLM. While our framework is suitable for many state-of-the-art LLMs in this form, it might not be optimal for white-box LLMs, which offer access to token probabilities, thus limiting its broader applicability. Secondly, the utility of self-detection is not primarily studies in this work. Although we demonstrate the utility of detection scores in selective prediction scenarios, the challenge still lies in leveraging them to enhance task accuracy or enable LLM self-correction, calling for further exploration. Lastly, our framework lacks consideration in prompt optimization for self-detection, an area where future self-detection methods are expected to consider.

Ethics Statement

Our ethical concerns involve the following. First, our experimental results are mainly obtained in English datasets, where the applicability on other languages are not comprehensively evaluated. Secondly, our research scope is black-box API LLMs, where open-sourced LLMs are more advocated for its reproducibility. Finally, the self-detection of LLM may mislead people to blindly trust LLM and easily accept untrustable answers, causing potential harms.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Ayush Agrawal, Lester Mackey, and Adam Tauman Kalai. 2023. Do language models know when they're hallucinating references? *arXiv preprint arXiv:2305.18248*.

Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, Jian-Guang Lou, and Weizhu Chen. 2023. Learning from mistakes makes llm better reasoner. *arXiv preprint arXiv:2310.20689*.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023*.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Kendrick Boyd, Kevin H Eng, and C David Page. 2013. Area under the precision-recall curve: point estimates and confidence intervals. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III 13*, pages 451–466. Springer.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Jiuhai Chen and Jonas Mueller. 2023. Quantifying uncertainty in answers from any language model via intrinsic and extrinsic confidence assessment. *arXiv preprint arXiv:2308.16175*.

Soham Dan and Dan Roth. 2021. On the effects of transformer size on in-and out-of-domain calibration. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2096–2101.

Shrey Desai and Greg Durrett. 2020. Calibration of pre-trained transformers. *arXiv preprint arXiv:2003.07892*.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. Glm: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335.

Jinhao Duan, Hao Cheng, Shiqi Wang, Chenan Wang, Alex Zavalny, Renjing Xu, Bhavya Kailkhura, and Kaidi Xu. 2023. Shifting attention to relevance: Towards the uncertainty estimation of large language models. *arXiv preprint arXiv:2307.01379*.

Shangbin Feng, Weijia Shi, Yike Wang, Wenxuan Ding, Vidhisha Balachandran, and Yulia Tsvetkov. 2024. Don't hallucinate, abstain: Identifying llm knowledge gaps via multi-llm collaboration. *arXiv preprint arXiv:2402.00367*.

Xiang Gao, Jiaxin Zhang, Lalla Mouatadid, and Kamalika Das. 2024. Spuq: Perturbation-based uncertainty quantification for large language models. *arXiv preprint arXiv:2403.02509*.

Olga Golovneva, Moya Peng Chen, Spencer Poff, Martin Corredor, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. 2022. Roscoe: A suite of metrics for scoring step-by-step reasoning. In *The Eleventh International Conference on Learning Representations*.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Bairu Hou, Yujian Liu, Kaizhi Qian, Jacob Andreas, Shiyu Chang, and Yang Zhang. 2023. Decomposing uncertainty for large language models through input clarification ensembling. *arXiv preprint arXiv:2311.08718*.

Mengting Hu, Zhen Zhang, Shiwan Zhao, Minlie Huang, and Bingzhe Wu. 2023. Uncertainty in natural language processing: Sources, quantification, and applications. *arXiv preprint arXiv:2306.04459*.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023a. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068, Singapore. Association for Computational Linguistics.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023b. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*.

Yuheng Huang, Jiayang Song, Zhijie Wang, Huaming Chen, and Lei Ma. 2023c. Look before you leap: An exploratory study of uncertainty measurement for large language models. *arXiv preprint arXiv:2307.10236*.

Mingjian Jiang, Yangjun Ruan, Sicong Huang, Saifei Liao, Silviu Pitis, Roger Baker Grosse, and Jimmy Ba. 2023. Calibrating language models via augmented prompt ensembles.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. 2021. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. 2022. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*.

Divyansh Kaushik, Eduard Hovy, and Zachary Lipton. 2019. Learning the difference that makes a difference with counterfactually-augmented data. In *International Conference on Learning Representations*.

Muhammad Khalifa, Lajanugen Logeswaran, Moontae Lee, Honglak Lee, and Lu Wang. 2023. Grace: Discriminator-guided chain-of-thought reasoning.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations*.

Junyi Li, Xiaoxue Cheng, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2023a. Halueval: A large-scale hallucination evaluation benchmark for large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6449–6464.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2024a. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36.

Loka Li, Guangyi Chen, Yusheng Su, Zhenhao Chen, Yixuan Zhang, Eric Xing, and Kun Zhang. 2024b. Confidence matters: Revisiting intrinsic self-correction capabilities of large language models. *arXiv preprint arXiv:2402.12563*.

Moxin Li, Wenjie Wang, Fuli Feng, Hanwang Zhang, Qifan Wang, and Tat-Seng Chua. 2023b. Hypothetical training for robust machine reading comprehension of tabular context. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1220–1236, Toronto, Canada. Association for Computational Linguistics.

Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. 2023c. Making language models better reasoners with step-aware verifier. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5315–5333.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Teaching models to express their uncertainty in words. *Transactions on Machine Learning Research*.

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

Zhan Ling, Yunhao Fang, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, and Hao Su. 2023. Deductive verification of chain-of-thought reasoning. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Potsawee Manakul, Adian Liusie, and Mark Gales. 2023. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9004–9017, Singapore. Association for Computational Linguistics.

Christopher Manning and Hinrich Schutze. 1999. *Foundations of statistical natural language processing*. MIT press.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Ning Miao, Yee Whye Teh, and Tom Rainforth. 2023. Selfcheck: Using llms to zero-shot check their own step-by-step reasoning. *arXiv preprint arXiv:2308.00436*.

Sabrina J Mielke, Arthur Szlam, Emily Dinan, and Y-Lan Boureau. 2022. Reducing conversational agents' overconfidence through linguistic calibration. *Transactions of the Association for Computational Linguistics*, 10:857–872.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100, Singapore. Association for Computational Linguistics.

Ansong Ni, Srini Iyer, Dragomir Radev, Veselin Stoyanov, Wen-tau Yih, Sida Wang, and Xi Victoria Lin. 2023. Lever: Learning to verify language-to-code generation with execution. In *International Conference on Machine Learning*, pages 26106–26128. PMLR.

OpenAI. 2024. Chatgpt.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2023. Automatically correcting large language models: Surveying the landscape of diverse self-correction strategies. *CoRR*, abs/2308.03188.

Jie Ren, Yao Zhao, Tu Vu, Peter J Liu, and Balaji Lakshminarayanan. 2023a. Self-evaluation improves selective generation in large language models. *arXiv preprint arXiv:2312.09300*.

Ruiyang Ren, Yuhao Wang, Yingqi Qu, Wayne Xin Zhao, Jing Liu, Hao Tian, Hua Wu, Ji-Rong Wen, and Haifeng Wang. 2023b. Investigating the factual knowledge boundary of large language models with retrieval augmentation. *arXiv preprint arXiv:2307.11019*.

Andy Shih, Dorsa Sadigh, and Stefano Ermon. 2023. Long horizon temperature scaling. *arXiv preprint arXiv:2302.03686*.

Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Lee Boyd-Graber, and Lijuan Wang. 2022. Prompting gpt-3 to be reliable. In *The Eleventh International Conference on Learning Representations*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158.

Katherine Tian, Eric Mitchell, Huaxiu Yao, Christopher Manning, and Chelsea Finn. 2023a. Fine-tuning language models for factuality. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D Manning. 2023b. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. *arXiv preprint arXiv:2305.14975*.

Nirali Vaghani and Mansi Thummar. 2023. BFlipkart product reviews with sentiment dataset. `https://www.kaggle.com/dsv/4940809`.

Danqing Wang and Lei Li. 2023. Learning from mistakes via cooperative study assistant for large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10667–10685.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. 2023. Large language models are better reasoners with self-verification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2550–2575, Singapore. Association for Computational Linguistics.

Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, Xu Zhao, Min-Yen Kan, Junxian He, and Qizhe Xie. 2023. Self-evaluation guided beam search for reasoning. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. 2023. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. *arXiv preprint arXiv:2306.13063*.

Polina Zablotskaia, Du Phan, Joshua Maynez, Shashi Narayan, Jie Ren, and Jeremiah Liu. 2023. On uncertainty calibration and selective generation in probabilistic neural summarization: A benchmark study. *arXiv preprint arXiv:2304.08653*.

Hanlin Zhang, Yi-Fan Zhang, Yaodong Yu, Dhruv Madeka, Dean Foster, Eric Xing, Hima Lakkaraju, and Sham Kakade. 2023a. A study on the calibration of in-context learning. *arXiv preprint arXiv:2312.04021*.

Jiaxin Zhang, Zhuohang Li, Kamalika Das, Bradley A. Malin, and Sricharan Kumar. 2023b. SAC^3: Reliable hallucination detection in black-box language models via semantic-aware cross-check consistency. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Mozhi Zhang, Mianqiu Huang, Rundong Shi, Linsen Guo, Chong Peng, Peng Yan, Yaqian Zhou, and Xipeng Qiu. 2024. Calibrating the confidence of large language models by eliciting fidelity. *arXiv preprint arXiv:2404.02655*.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. 2023c. Siren's song in the ai ocean: A survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*.

Ruochen Zhao, Xingxuan Li, Shafiq Joty, Chengwei Qin, and Lidong Bing. 2023a. Verify-and-edit: A knowledge-enhanced chain-of-thought framework. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5823–5840, Toronto, Canada. Association for Computational Linguistics.

Theodore Zhao, Mu Wei, J Samuel Preston, and Hoifung Poon. 2023b. Automatic calibration and error correction for large language models via pareto optimal self-supervision. *arXiv preprint arXiv:2306.16564*.

Xinran Zhao, Hongming Zhang, Xiaoman Pan, Wenlin Yao, Dong Yu, Tongshuang Wu, and Jianshu Chen. 2024. Fact-and-reflection (far) improves confidence calibration of large language models. *arXiv preprint arXiv:2402.17124*.

Yukun Zhao, Lingyong Yan, Weiwei Sun, Guoliang Xing, Chong Meng, Shuaiqiang Wang, Zhicong Cheng, Zhaochun Ren, and Dawei Yin. 2023c. Knowing what llms do not know: A simple yet effective self-detection method. *arXiv preprint arXiv:2310.17918*.

Aojun Zhou, Ke Wang, Zimu Lu, Weikang Shi, Sichun Luo, Zipeng Qin, Shaoqing Lu, Anya Jia, Linqi Song, Mingjie Zhan, et al. 2023. Solving challenging math word problems using gpt-4 code interpreter with code-based self-verification. *arXiv preprint arXiv:2308.07921*.

	N	examples
IMDB	2	positive negative
Flipkart	2	positive negative
SNLI	3	entailment, neutral, contradiction
HANS	2	entailment, non entailment
CommonsenseQA	5	(a) yard, (b) basement, (c) kitchen, (d) living room, (e) garden
PIQA	2	(a) pour it onto a plate, (b) pour it into a jar

Table 6: The number (N) and examples of candidate answers for each dataset.

A Details for compared methods.

A.1 LLM Hyperparameters.

For all LLMs, we set the maximum token as 200. For GPT-3.5 and Gemini, if sampling a single response ($N = 1$), we set the temperature as 0, and other hyperparameters as default. If sampling multiple responses, we sample $N = 30$ ($N = 5$ for Gemini due to API call limitation) responses with temperature as 1, which is only for Self-cons, CoT-cons, and P(True). Specially, for Self-detect we sample 15 rephrasing for each question with temperature as 1, and one answer for each rephrased question with temperature as 0, following the original paper. For GLM-4, if sampling a single response, we set the do_sample as False. If sampling a variety of responses, we set temperature as 0.9, top p as 0.9, and $N = 5$. Note that these LLM hyperparameters are not carefully tuned.

A.2 Dataset Detail.

Due to the cost limitation, we randomly sample 300 training data for each dataset in our experiments. For IMDB and SNLI datasets, we use the same randomly sampled 300 data sets as the CAD SA and NLI in the preliminary experiments. We will release the dataset splits. Table 6 shows the number and examples of candidate answers for each dataset.

A.3 Prompts

The basic instructions for different datasets are shown as below, where [] refers to specific task inputs.

- **IMDB**:
  Given a piece of movie review, classify the attitude to the movie as Positive or Negative. [text]

- **Flipkart**:
  Given a piece of text, classify the sentiment as Positive or Negative. [text]

- **SNLI**:
  Determine whether the hypothesis is an entailment (can be logically inferred from the premise), a contradiction (cannot be true given the premise), or neutral (does not have enough information to determine its truth value). Premise: [premise] Hypothesis: [hypopthesis].

- **HANS**:
  Determine whether the second sentence in each pair logically follows from the first sentence. The output is either "entailment" if the second sentence logically follows from the first, or "not entailment" if it does not. Sentence 1: [sentence1]. Sentence 2: [sentence2].

- **CommonsenseQA**:
  Read the given question and select the most appropriate answer by indicating the associated letter. Question: [question]. Answer choices: (a) a_1^q (b) a_2^q (c) a_3^q (d) a_4^q (e) a_5^q.

- **PIQA**:
  Read the given question and select the most appropriate answer by indicating the associated letter. Question: [question]. Answer choices: (a) a_1^q (b) a_2^q.

The prompts for compared methods are shown below, where *[instruction]* denotes the task instruction with the task input, and *[instruction_only]* denotes the instruction without task input.

- **Self-cons**: *[instruction].*

- **CoT-cons**:
  [instruction]. Please output strictly following this format: Explanation: [reasons for the sentiment label] Answer: [Positive or Negative]

- **Top-K Verb** p^b:
  The task is to [instruction_only]. Provide your n best guesses and the probability that each is correct (0.0 to 1.0) for the following question. Give ONLY the guesses and probabilities, no other words or explanation. For example:
  G1: <first most likely guess, as short as possible; not a complete sentence, just the guess!>
  P1: <the probability between 0.0 and 1.0 that G1 is correct, without any extra commentary whatsoever; just the probability!> ... GN: <N-th most likely guess, as short as possible; not a complete sentence, just the guess!>
  PN: <the probability between 0.0 and 1.0 that GN is correct, without any extra commentary whatsoever; just the probability!> [question] [answer choices].

- **P(True)** p^t:
  The task is to [instruction]. Label: [label]. Is the label correct or incorrect?

- **Self-detect**:
  For question rephrasing: *Paraphrase the given sentence. Please make sure the paraphrased sentence has exactly the same meaning as the original sentence. [question]*
  For inference: *[instruction].*

- **CAPE**:
  Provide your 2 best guesses and the probability that each is correct (0.0 to 1.0) for the following task. Give ONLY the guesses and probabilities, no other words or explanation. For example:
  G1: <first most likely guess, as short as possible; not a complete sentence, just the guess!>
  P1: <the probability between 0.0 and 1.0 that G1 is correct, without any extra commentary whatsoever; just the probability!> ... GN: <N-th most likely guess, as short as possible; not a complete sentence, just the guess!>
  PN: <the probability between 0.0 and 1.0 that GN is correct, without any extra commentary whatsoever; just the probability!> Instruction: [instruction_only] [question]
  - a_1^q (or A. a_1^q)
  ...
  - a_N^q (or N. a_N^q)
  Possible explanation 1: [e^1]
  ...
  Possible explanation N: [e^N]

Correct Choice:

- T^3 p^v:
  The task is to [instruction_only]. Provide your n best guesses and the probability that each is correct (0.0 to 1.0) for the following question. Give ONLY the guesses and probabilities, no other words or explanation. For example:
  G1: <first most likely guess, as short as possible; not a complete sentence, just the guess!>
  P1: <the probability between 0.0 and 1.0 that G1 is correct, without any extra commentary whatsoever; just the probability!> ... GN: <N-th most likely guess, as short as possible; not a complete sentence, just the guess!>
  PN: <the probability between 0.0 and 1.0 that GN is correct, without any extra commentary whatsoever; just the probability!>
  [question] [answer choices].
  Possible explanation 1: [explanation 1].
  ...
  Possible explanation N: [explanation N].

A.4 Additional Implementation Detail.

For T^3 and Top-K Verb, the N is set to the number of candidate answers for each dataset as in Table 6.

For the shuffling of the justification order in p^v, we use one original and one reversed order for T^3 on all datasets. For datasets with more than two justifications (SNLI and CommonsenseQA), we set the original justification order for SNLI as "entailment, neutral, contradiction" and follow the given answer choice order for CommonsenseQA in the dataset.

CAPE is prompt ensemble for Top-K Verb. We follow the original paper to adopt two multi-choice template with alphabetic or itemized labels in addition to the original Top-K Verb prompt (See Section A.3). For each multi-choice template, we use the original and the reversed label orders. In total, the confidence score is an average of five prompts.

For T^3 + PE, we put T^3 into the multi-choice template with alphabetic labels, and use two reversed label orders and 2 reversed justification orders, in total four prompts.

The number of API calls for different methods are shown in Table 7. We can observe that compared with other methods T^3 does not incur large increase in number of calls. In our experiments, the maximum value of N is 5. Considering its effectiveness, the cost of T^3 is reasonable.

	Self-cons	CoT-cons	Top-K Verb	P(True)	Hybrid
# call	30	30	1	30	31
	Self-detect	CAPE	T^3	T^3 + Top-K Verb	T^3 + PE
# call	30	5	N+2	N+3	N+4

Table 7: Comparison on the number of API calls of compared methods, where N denotes the number of choices for different datasets.

B Implementation Detail for Preliminary Experiments.

For the preliminary experiments, we randomly sample 300 instances from the training set of CAD SA and NLI, respectively. For those original questions with more than one counterfactual questions, we randomly select one counterfactual question for experiment. The prompts can be viewed in Section A.3. CAD SA is annotated from IMDB, and CAD NLI is annotated from SNLI. The *w/ cf* is based on Top-K Verb, which is better calibrated than Self-cons. For *w/ cf*, we obtain the Top-K Verb outputs for counterfactual and original questions, respectively. We use the guess with the largest probability in the response as the answer to $\bar{q}$, and the probability as its confidence score. The LLM is GPT-3.5 (*gpt-3.5-1106*). See Section A.1 for LLM hyperparameters.

	PIQA	HANS	Flipkart
p^e	84.2 ± 2.0	62.7 ± 4.3	78.0 ± 2.2
p^v	83.0 ± 0.5	68.3 ± 1.7	81.2 ± 0.3

Table 8: The average and standard deviation of AUROC for T^3 with different rephrasing of prompts on GPT-3.5.

C Prompt Sensitivity

We examine the prompt sensitivity of p^e and p^v by rephrasing each of them three times with ChatGPT[6] and compute the average and standard deviation of AUROC, as shown in Table 8. We can observe the followings. 1) The variation of prompts has a mild effect on the performance of T^3. Across the three datasets, HANS is the most sensitive to prompt rephrasing, potentially related to its lower AUROC performance. 2) The change of p^e has larger impact on the detection performance than p^v. This is probably because the justifications generated by p^e have a larger space of variation than the outputs of p^v, *i.e.,* guesses and probabilities.

[6] https://chat.openai.com/.

	Flipkart		PIQA		CommonsenseQA	
	AUROC	PRAUC	AUROC	PRAUC	AUROC	PRAUC
CoT-cons	67.6	89.1	85.8	96.4	72.2	86.0
Top-K Verb	84.0	94.9	64.7	91.8	67.0	83.9
Hybrid	**85.0**	**95.5**	77.0	96.1	74.5	87.9
CAPE	81.5	94.5	86.5	97.8	77.3	89.5
T^3	82.0	93.9	82.6	96.5	73.4	89.3
+ Top-K Verb	83.1	94.6	77.2	96.1	76.3	**91.2**
+ CAPE	74.8	92.1	**90.5**	**98.3**	**77.5**	91.1

Table 9: Performance comparison of Gemini on Flipkart, PIQA and CommonsenseQA.

D Additional Results on Different LLMs

In addition to GPT-3.5 and GLM-4, we show the results of Gemini on three datasets. From Table 9, we can observe that although T^3 outperforms all compared methods in PIQA and CommonsenseQA, it does not outperform all compared methods on Flipkart. By analyzing the outputs, we discover that Gemini cannot always follow the instruction to perform reflection and generated justification for the designated answer. Instead, it tends to perform answer prediction and followed by an explanation on its predicted answer. Without effective reflection and justification from different answers' perspectives, the effectiveness of T^3 is diminished. Therefore, the effectiveness of T^3 depends on the ability of the specific LLM in following the instructions in Table 1.

E Case study

We present two case study of PIQA. From Table 10, we can observe that the detection score of the incorrect answer (a) is lowered by T^3 ($0.7 \rightarrow 0.45$). The justification (b) points out the reason why (b) is the preferred answer, which increased the confidence in (b) and in turn decreases (a)'s detection score. From Table 11, we can observe that the LLM is not sure whether the sponge should be dampened, thus having an ambivalent confidence score of 0.5 is better than being 0.7 confident on (b).

Input: The task is to read the given question and select the most appropriate answer by indicating the associated letter. Provide your 2 best guesses and the probability that each is correct (0.0 to 1.0) for the following question. Give ONLY the guesses and probabilities, no other words or explanation. For example:
G1: <first most likely guess, as short as possible; not a complete sentence, just the guess!>
P1: <the probability between 0.0 and 1.0 that G1 is correct, without any extra commentary whatsoever; just the probability!> ... G2: <2-th most likely guess, as short as possible; not a complete sentence, just the guess!>
P2: <the probability between 0.0 and 1.0 that G2 is correct, without any extra commentary whatsoever; just the probability!>
Question: How do you repair a torn shirt?
Answer Choices:
(a) To repair a torn shirt, prepare the needle with a piece of thread tied to it. Pull together the fabric where the shirt is torn and sew together.
(b) Prepare the needle with a piece of thread tied to it. Flip the shirt inside-out, pull together the fabric where the shirt is torn and sew together.
Prediction: (a). Ground truth: (b).

Top-K Verb Output: G1: A P1: 0.7 G2: B P2: 0.3.
Confidence: 0.7

Justification for (a): The most appropriate answer is (a) because it provides a clear and direct explanation of how to repair a torn shirt. It instructs to prepare the needle with a piece of thread tied to it and then pull together the fabric where the shirt is torn and sew it together. This answer is straightforward and does not include unnecessary steps, making it the most appropriate choice.
p^v **output 1**: G1: A P1: 0.6 G2: B P2: 0.4
Justification for (b): Option (b) is the most appropriate answer because it provides a more detailed and accurate description of the process of repairing a torn shirt. It includes the step of flipping the shirt inside-out, which is an important step in order to hide the stitches and make the repair less visible. This additional detail makes option (b) the better choice for repairing a torn shirt.
p^v **output 2**: G1: B P1: 0.7 G2: A P2: 0.3
T^3 **Detection Score**: (0.6 + 0.3)/2 = 0.45

Table 10: Case study for PIQA. p^v output 1 refers to p^v with explanation (a) before explanation (b), and p^v output 2 refers to the reversed order.

Input: The task is to read the given question and select the most appropriate answer by indicating the associated letter. Provide your 2 best guesses and the probability that each is correct (0.0 to 1.0) for the following question. Give ONLY the guesses and probabilities, no other words or explanation. For example: G1: <first most likely guess, as short as possible; not a complete sentence, just the guess!> P1: <the probability between 0.0 and 1.0 that G1 is correct, without any extra commentary whatsoever; just the probability!> ... G2: <2-th most likely guess, as short as possible; not a complete sentence, just the guess!> P2: <the probability between 0.0 and 1.0 that G2 is correct, without any extra commentary whatsoever; just the probability!> Question: How do I keep my favorite couch fur free? Answer Choices: (a) Purchase a dry sponge at a pet supply store and use it to lightly wipe dog hair in a sweeping motion from upholstery. (b) Purchase a dry sponge at a pet supply store and dampen it and use it to lightly wipe dog hair in a sweeping motion from upholstery. **Prediction: (b). Ground truth: (a).**
Top-K Verb Output: G1: B P1: 0.7 G2: A P2: 0.3. **Confidence**: 0.7
Justification for (a): The most appropriate answer is (a) because using a dry sponge to wipe dog hair in a sweeping motion from upholstery will help to remove the fur without getting the couch wet. Dampening the sponge, as suggested in option (b), may lead to the couch getting wet and potentially causing damage to the upholstery. Therefore, option (a) is the best choice for keeping the favorite couch fur free. p^v **output 1**: G1: A P1: 0.7 G2: B P2: 0.3 **Justification for (b)**: The most appropriate answer is (b) because dampening the dry sponge will help to better pick up and remove the dog hair from the upholstery. Using a dry sponge alone may not be as effective in removing the fur. Therefore, dampening the sponge will provide better results in keeping the favorite couch fur free. p^v **output 2**: G1: B P1: 0.7 G2: A P2: 0.3 T^3 **Detection Score**: (0.3 + 0.7)/2 = 0.5

Table 11: Case study for PIQA. p^v output 1 refers to p^v with justification (a) before justification (b), and p^v output 2 refers to the reversed order.