

# Enhancing LLM Reliability via Explicit Knowledge Boundary Modeling

Hang Zheng<sup>1\*</sup>, Hongshen Xu<sup>1\*</sup>, Yuncong Liu<sup>1</sup>, Lu Chen<sup>1</sup>, Pascale Fung<sup>2</sup>, Kai Yu<sup>1</sup>

<sup>1</sup>X-LANCE Lab, Department of Computer Science and Engineering,

Shanghai Jiao Tong University

<sup>2</sup>Center for Artificial Intelligence Research (CAiRE),

Hong Kong University of Science and Technology

{azure123, xuhongshen, chenlusz, kai.yu}@sjtu.edu.cn

## Abstract

Large language models (LLMs) frequently hallucinate due to misaligned self-awareness, generating erroneous outputs when addressing queries beyond their knowledge boundaries. While existing approaches mitigate hallucinations via uncertainty estimation or query rejection, they suffer from computational inefficiency or sacrificed helpfulness. To address these issues, we propose the *Explicit Knowledge Boundary Modeling* (EKBM) framework, integrating fast and slow reasoning systems to harmonize reliability and usability. The framework first employs a fast-thinking model to generate confidence-labeled responses, enabling immediate use of high-confidence outputs. For uncertain predictions, a slow refinement model conducts targeted reasoning to improve accuracy. To align model behavior with our proposed object, we propose a hybrid training pipeline, enhancing self-awareness without degrading task performance. Evaluations on dialogue state tracking tasks demonstrate that EKBM achieves superior model reliability over uncertainty-based baselines. Further analysis reveals that refinement substantially boosts accuracy while maintaining low computational overhead. Our work establishes a scalable paradigm for advancing LLM reliability and balancing accuracy and practical utility in error-sensitive applications.

## 1 Introduction

Recently, large language models (LLMs) have exhibited impressive text generation capabilities (Abdullah et al., 2022). However, despite their advancements, LLMs are prone to hallucination, where generated content misaligns with context or factual information (Zhang et al., 2023). Such inaccuracies can be particularly harmful in low error-tolerance applications, eroding user trust in LLM reliability.

Extensive works have been devoted to mitigating hallucinations in LLMs (Tonmoy et al., 2024).

\*Equal contribution.

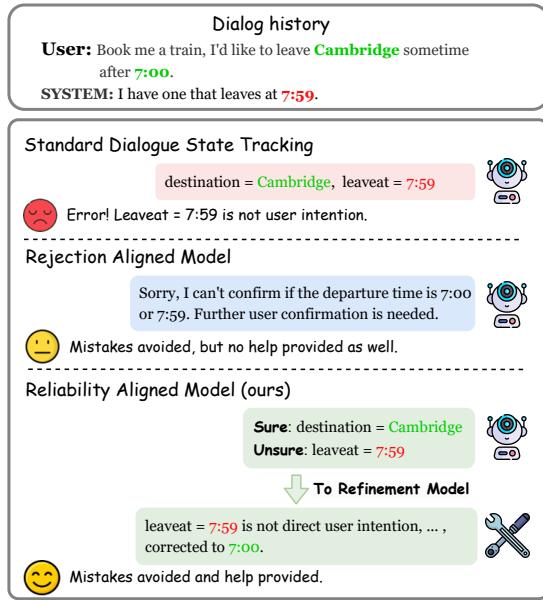


Figure 1: A case study on Dialog State Tracking: comparison of different alignment objectives.

Hallucinations often arise from misalignments between an LLM’s outputs and its intrinsic knowledge boundaries, leading to errors when models attempt to answer questions beyond their expertise(Li et al., 2024b). Improving the model’s self-awareness—its capacity to accurately assess its own knowledge boundaries and outputs correctness, can effectively mitigate this misalignment. Several uncertainty-based methods have been proposed to estimate model confidence and indirectly reflect the model’s self-awareness (Huang et al., 2024). However, these approaches are not inherent to the model’s endogenous capabilities and frequently suffer from high computational costs or instability across thresholds. Alternative strategies emphasize aligning knowledge boundaries with actions by rejection of out-of-scope queries (Xu et al., 2024b), avoiding mistakes but sacrificing helpfulness—a trade-off that undermines model utility particularly in complex, multi-step tasks where partial correctness remains valuable.

To mitigate these limitations, we propose a novel alignment objective: improving model awareness for knowledge boundaries, enabling models to respond maximally while explicitly distinguishing outputs of high and low confidence. We introduce the concepts of "sure" and "unsure" to clarify this distinction. A reliable system should deliver near-perfect accuracy for "sure" predictions and provide helpful information in the "unsure" category, serving as a form of soft rejection, appropriately managing user expectations regarding accuracy.

Building on this concept, we propose the Explicit Knowledge Boundary Modeling (EKBM) framework. EKBM integrates both Fast and Slow systems: a fast-thinking model generates responses annotated with confidence labels, allowing immediate utilization of high-confidence outputs, while a slow-thinking refinement model engages in deliberate reasoning to enhance accuracy of low-confidence predictions. Crucially, EKBM does not merely filter uncertain outputs but leverages them as opportunities for improvement, ensuring both precision and coverage. The collaboration between these two systems strikes a balance between reliability and efficiency, with the self-awareness capability of the fast-thinking model being essential for the system's effective operation.

To operationalize this paradigm, we design a training pipeline that aligns the model's capability boundaries and improves self-awareness. Evaluations on dialogue state tracking (DST) tasks demonstrate that our method effectively enables the model to reliably and accurately categorize its outputs into "sure" and "unsure" categories compared to baseline methods. Additionally, we develop an automatic refinement model for "unsure" outputs, showing that the EKBM system achieves superior performance compared to traditional supervised fine-tuning (SFT) approaches. Our analysis of various training methods and model behaviors provides further valuable insights.

Our contributions are summarized as follows:

- We present the EKBM framework, integrating fast-thinking confidence labeling with slow-thinking refinement, along with metrics for evaluating LLM self-awareness and reliability.
- We develop a comprehensive training pipeline that enhances LLM self-awareness, yielding more reliable models.
- We perform extensive experiments to demonstrate the effectiveness and scalability of the

EKBM framework, providing valuable insights for future research.

## 2 Related Work

The knowledge boundary of LLMs has emerged as a critical topic in recent research, highlighting their limitations in generating reliable outputs (Yin et al., 2024; Li et al., 2024b). Misalignment between model behavior and knowledge boundaries can result in factual hallucinations (Huang et al., 2023) and ambiguous responses (Liu et al., 2023).

Various methods have been proposed to assess these boundaries. Uncertainty-based approaches quantify prediction confidence through token probabilities (Manakul et al., 2023) and consistency (Chen and Mueller, 2024). Calibration strategies, including prompting (Tian et al., 2023) and fine-tuning (Tao et al., 2024) for improved confidence expression, align model confidence with prediction accuracy. Internal State Probing evaluates prediction factuality by analyzing model states, such as activations (Li et al., 2024a). While effective, these methods often rely on external post-hoc techniques, which are computationally expensive and lack integration with the model's endogenous reasoning processes (Huang et al., 2024; Zhou et al., 2024b).

To address this misalignment, researchers have implemented knowledge-enhanced fine-tuning (Zhou et al., 2024a) and retrieval-augmented generation (RAG, Lewis et al., 2020) to bolster model capabilities, thereby reducing hallucinations. Some approaches also guide models in self-reflection to minimize self-contradictions (Wei et al., 2022; Ji et al., 2023), albeit at the cost of increased overhead. For queries beyond the model's capabilities, refusal or rejection strategies can prevent misinformation (Xu et al., 2024b; Chen et al., 2024), though they may inadvertently reduce overall helpfulness.

## 3 Problem Formulation

### 3.1 Alignment for Reliability

To enhance the reliability of LLMs, a prevailing method involves aligning their performance with high-quality training data. This alignment aims to achieve the following objective:

$$s(x, y_c) > s(x, y_w)$$

where  $s$  denotes a scoring function,  $x$  is the input prompt, and  $y_c$  and  $y_w$  denote correct and incorrect responses, respectively, ensuring the score of correct responses exceeds that of incorrect ones.

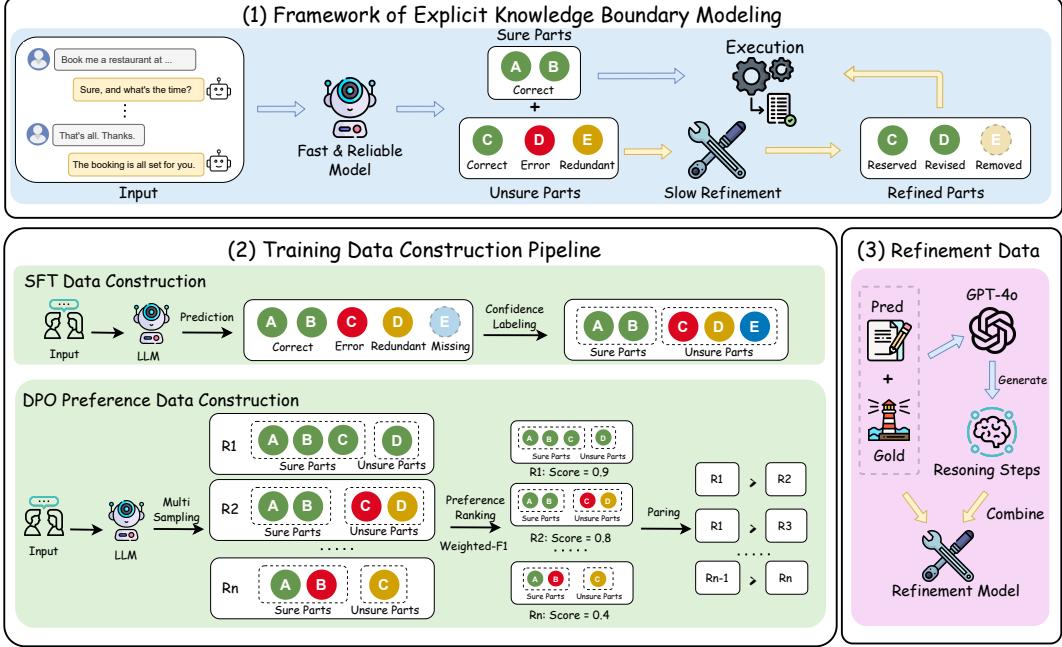


Figure 2: The EKBM framework and the data construction methods.

Recent studies have introduced rejection as a mechanism to help the model differentiate between answerable (in-boundary) and unanswerable (out-of-boundary) queries. For unanswerable queries, the model is encouraged to refrain from responding. The corresponding alignment objective can be articulated as:

$$s(x, y_c) > s(x, y_r) > s(x, y_w)$$

where  $y_r$  denotes a truthful rejection, and  $y_w$  an incorrect response. This suggests that refusal is preferable to providing an incorrect answer.

While this rejection strategy aligns with reliability goals, it may unduly limit the model's helpfulness in practical scenarios. Therefore, we propose a more nuanced objective: the model should aim to provide answers whenever feasible, categorizing outputs into high-confidence ("sure") and low-confidence ("unsure") responses. The model must ensure high accuracy in the "sure" category, while tolerating some errors in the "unsure" category, which can serve as references for users or inputs for further refinement. Our proposed alignment objective is formalized as:

$$s(x, y_c, c_s) > s(x, y_c, c_u) > s(x, y_w, c_u) > s(x, y_w, c_s)$$

Here,  $y_c$  and  $y_w$  represent correct and incorrect responses, respectively, while  $c_s$  and  $c_u$  represent "sure" and "unsure" as levels of confidence.

This objective guarantees that correct "sure" predictions are prioritized, followed by correct "unsure" predictions, which, although less confident,

retain utility. Incorrect predictions in the "unsure" category are discouraged but more acceptable than those in the "sure" category, as the latter significantly undermine reliability.

### 3.2 Reliability Evaluation

In this work, we focus on complex multi-slot problems and have modified several metrics for better evaluation of model reliability.

#### 3.2.1 Weighted-F1

F1-score is usually taken as the evaluation metric for multi-slot problems:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Where TP, FP and FN represent True Positives, False Positives and False Negatives respectively.

We modify the F1 score to adapt to the alignment objective of our EKBM framework. Since predictions labeled as "unsure" represent an intermediate state of uncertainty, we assign a weighted contribution (denoted by  $\alpha_1$  and  $\alpha_2$ ) to predictions in this category. The weights reflect the partial utility of "unsure" predictions as potential candidates for refinement. The modified precision and recall are defined as:

$$\text{Precision}(\alpha_1) = \frac{\text{STP} + \alpha_1 \cdot \text{UTP}}{\text{STP} + \text{SFP} + \alpha_1 \cdot (\text{UTP} + \text{UFP})}$$

$$\text{Recall}(\alpha_2) = \frac{\text{STP} + \alpha_2 \cdot \text{UTP}}{\text{STP} + \text{UTP} + \text{FN}}$$

Where STP and SFP represent Sure True Positives and Sure False Positives, while UTP and UFP denote the corresponding cases of Unsure. For UTP in Recall calculation, we only require that a prediction is partially correct (able to be refined). The modified **Weighted-F1** score is defined as:

$$\text{Weighted-F1}(\alpha_1, \alpha_2) = 2 \cdot \frac{\text{Precision}(\alpha_1) \cdot \text{Recall}(\alpha_2)}{\text{Precision}(\alpha_1) + \text{Recall}(\alpha_2)}$$

The design allows flexible handling of "unsafe" predictions, effectively managing their intermediate utility.  $\alpha_1$  influences the calculation of Precision, reflecting the model's tolerance for errors among "unsafe" predictions; as  $\alpha_1$  increases, tolerance for such errors decreases. Conversely,  $\alpha_2$  affects Recall, indicating the degree of reward for providing additional reference information for "unsafe" predictions. As  $\alpha_2$  increases, the model is encouraged to enhance helpfulness by generating more "unsafe" predictions as a basis for refinement.

We discuss a few corner cases: (1)  $\alpha_1 = 0, \alpha_2 = 0$ : Weighted-F1 disregards all "unsafe" predictions. (2)  $\alpha_1 = 0, \alpha_2 = 1$ : Errors in the "unsafe" category are ignored, rewarding effective recall, representing the theoretical upper limit post-perfect refinement. (3)  $\alpha_1 = 1, \alpha_2 = 0$ : Lacks significant interpretation. (4)  $\alpha_1 = 1, \alpha_2 = 1$ : Eliminates confidence labeling, reverting to traditional methods.

### 3.3 Reliability Metrics

We employ two specific Weighted-F1 values as performance metrics:

**Optimal-F1** Defined as  $\text{Weighted-F1}(0, 1)$ , representing the theoretical upper limit after perfect refinement of "unsafe" predictions.

**Quality-F1** Defined as  $\text{Weighted-F1}(0.5, 0.5)$ , it assigns "unsafe" predictions half the weight of "sure" ones, encouraging the model to select "sure" when confident while minimizing penalties for "unsafe" outputs. Quality-F1 measures the model's self-awareness in accurately classifying both types.

Among these, Quality-F1 is the core metric for assessing model reliability, while Optimal-F1 serving as a supplementary reference.

## 4 Method

### 4.1 Explicit Knowledge Boundary Modeling

We formalize our proposed framework, Explicit Knowledge Boundary Modeling (EKBM), which aims to improve the reliability and self-awareness of the model by explicitly modeling its knowledge boundaries and improve overall performance by

incorporating a refinement model to improve the "unsafe" predictions. As shown in Figure 2 (1), the framework divides the model's decision-making process into two distinct stages.

**Fast Prediction with Confidence Labeling** In the first stage, the model performs the primary task while simultaneously assigning a confidence label ("sure" or "unsafe") to each prediction. Predictions labeled as "sure" are directly accepted as final outputs, ensuring high precision.

**Slow Refinement for Unsafe Prediction** In the second stage, the "unsafe" predictions are further refined. This refinement process can involve various strategies, such as user confirmation, automated multi-step reasoning, or other post-processing techniques. This additional stage incurs extra cost but increases the overall accuracy and reliability of the system.

Our two-stage framework balances immediate usability with comprehensive coverage, making the model both reliable and helpful.

### 4.2 Reliability Alignment

To align the model's behavior with our proposed alignment objective, we explore various training approaches to enhance its intrinsic self-awareness (Xu et al., 2025). To minimize interference from task performance improvements, we introduce minimal supervisory signals related to task knowledge. We employ Supervised Fine-Tuning (SFT) and Direct Preference Optimization (DPO), with the data construction process illustrated in Figure 2 (2).

**SFT Data Construction** We assume that task-specific knowledge boundary of the model can be approximated by repeatedly sampling the model's outputs on the training data. Based on this, our SFT data construction approach is intuitive. We conduct multiple rounds of sampling for each training sample. Predictions that are consistently accurate are labeled as "sure," while those exhibiting errors or omissions receive an "unsafe" label. Importantly, we refrain from correcting erroneous outputs to avoid introducing supervisory signals that could compromise the model's task performance.

The number of sampling rounds is a hyperparameter denoted as  $i$ . A larger  $i$  results in a more conservative model, as the proportion of "unsafe" labels in the dataset increases with greater  $i$ . In our study, we set  $i = 1$  by default to accurately reflect the model's realistic capability boundary.

**DPO Preference Data Construction** DPO enables the model to learn directly from preference pairs, making the design of an effective preference strategy crucial. Given our alignment objective, it is intuitive to use Weighted-F1 as the preference score function. The  $\alpha$  values in Weighted-F1 significantly influence the model’s behavior and performance. Through experimentation (see Section 5.6), we determined that setting  $\alpha_1$  to 0.25 and  $\alpha_2$  to 0.75 best balances the model’s immediate helpfulness and its potential performance after refinement. Therefore, this setting will be used by default unless specified. To enhance the stability of DPO training, we ensure that all data is generated through multi-sampling, aligning with the model’s output distribution.

### 4.3 Refinement Model

We train automated refinement models for each dataset to further optimize by reasoning on unsure predictions, utilizing supervised fine-tuning (SFT) based on the LLAMA3 8B model. Overall, our refinement process operates at a fine-grained level, with each unsure prediction being individually refined. We employ a multi-step reasoning paradigm similar to the Chain of Thought (Wei et al., 2022) methodology and use GPT-4o (Hurst et al., 2024) to generate the reasoning process. This process is illustrated in Figure 2 (3), and detailed information can be found in Appendix A.

## 5 Experiments

We conduct experiments focusing on three core research questions:

**RQ1** Does our proposed reliability training pipeline outperform current methods in enhancing self-awareness abilities?

**RQ2** Does our EKBM framework improve the overall performance of LLMs on complex tasks?

**RQ3** Does our approach demonstrate strong scalability, performing well across foundation models with various performance?

### 5.1 Experiments Setup

#### 5.1.1 Model and Baselines

For our experiments, we utilize the LLAMA3 8B model and the Qwen2.5 7B model as the backbone.

For reliability alignment, we conducted **Reliability-SFT** and integrated DPO to create two additional variants: **Reliability-SFT+DPO Joint Training**, which undergoes joint training

with SFT and DPO, and **Reliability-SFT+DPO Post Training**, where DPO training follows initial SFT training, optimizing the model beyond the Reliability-SFT foundation. The key distinction lies in the different source distributions of preference data sampled for DPO.

To evaluate the reliability of our proposed methods, we compare their performance against several baseline approaches, including three prompt-based and three uncertainty-based methods.

**Prompt-based** methods include three approaches: Direct, Verbose, and Self-Reflection (SR). The Direct method predicts outputs without incorporating any confidence labels, while Verbose generates confidence labels alongside predictions based on predefined principles. SR evaluates prediction reliability through individual reflections.

**Uncertainty-based** methods consist of three approaches: Token Probability (Prob), Self-Consistency (SC), and P(True) (Kadavath et al., 2022). These methods classify outputs as "sure" or "unsure" using different techniques. Prob calculates average token-level probabilities, SC assesses the frequency of predictions across multiple samples, and P(True) determines the probability of the "True" token during model self-evaluation.

The details of the baselines can be found in E.1.

#### 5.1.2 Dataset

We evaluate our methods on the Dialogue State Tracking (DST) task in task-oriented dialogue systems, which involves extracting slot-value pairs from multi-turn dialogues based on a predefined ontology (Xu et al., 2024a). Common evaluation metrics for DST include Joint Goal Accuracy (JGA) and Slot-F1. For JGA, a sample is deemed correct only if all predictions are accurate without omissions, while Slot-F1 offers a slot-level assessment.

To ensure a robust evaluation, we select three datasets: MultiWOZ-2.4 (Ye et al., 2021), BiTOD (Lin et al.), and SGD (Rastogi et al., 2020). For BiTOD we only utilize the English version. For SGD, we randomly select 10,000 test samples.

#### 5.1.3 Training Details

We modify the DeepSpeed-Chat (Yao et al., 2023) framework to accommodate our experimental setup. We adopt most of the default training parameters provided by DeepSpeed-Chat except that the learning rate for DPO training is set to 1e-7 and the maximum sequence length is set to 8096 on the SGD

Method Type	Method	MultiWOZ-2.4				BiTOD				SGD			
		Prec <sub>sure</sub> ↑	Rec <sub>total</sub> ↑	Opti. F1 ↑	Qual. F1 ↑	Prec <sub>sure</sub> ↑	Rec <sub>total</sub> ↑	Opti. F1 ↑	Qual. F1 ↑	Prec <sub>sure</sub> ↑	Rec <sub>total</sub> ↑	Opti. F1 ↑	Qual. F1 ↑
<i>LLAMA3-8B (Dubey et al., 2024)</i>													
Prompt	Direct	72.48	77.98	73.38	73.38	73.37	64.83	65.39	65.39	26.46	28.71	25.12	25.12
	Verbose	85.95	85.85	80.88	53.34	91.53	76.52	78.09	53.56	69.59	27.93	29.25	14.98
	SR	86.43	81.34	81.34	61.43	92.39	76.32	78.84	48.60	65.32	33.61	33.37	21.30
Uncertainty	Prob	80.05	80.66	77.74	69.03	83.50	72.44	73.40	60.30	36.93	32.28	27.74	22.86
	SC	80.11	89.51	84.15	64.48	83.16	75.48	77.33	62.93	84.66	41.06	42.76	16.55
	P(True)	72.82	79.00	73.50	70.81	72.44	66.26	65.97	63.06	24.78	29.93	26.26	26.24
Reliability (Ours)	SFT	94.04	91.75	91.65	82.44	95.66	88.25	88.56	83.08	86.74	85.51	82.66	61.64
	+ DPO Joint	<b>94.52</b>	92.16	<b>92.83</b>	83.50	<b>95.84</b>	<b>88.85</b>	<b>88.89</b>	83.11	<b>89.07</b>	<b>86.31</b>	<b>85.89</b>	61.42
	+ DPO Post	93.59	<b>93.10</b>	91.75	<b>83.55</b>	95.17	88.00	88.35	<b>85.78</b>	82.26	83.47	79.70	<b>64.14</b>
<i>Qwen2.5-7B (Yang et al., 2024)</i>													
Prompt	Direct	69.42	71.37	67.22	67.22	71.19	65.31	65.98	65.98	33.43	36.11	33.40	33.40
	Verbose	83.02	71.14	69.05	58.88	78.62	77.22	75.23	64.78	65.78	31.80	32.13	21.74
	SR	73.54	72.08	69.69	67.87	73.73	66.48	67.90	66.34	35.87	36.82	35.13	34.05
Uncertainty	Prob	77.12	74.47	72.02	65.19	84.57	74.43	75.49	61.11	46.66	38.91	34.83	29.36
	SC	83.33	90.20	84.67	57.00	87.65	74.83	77.42	51.43	71.47	61.08	53.38	25.22
	P(True)	70.03	71.44	67.49	67.24	72.11	65.66	66.13	65.65	33.60	36.40	33.67	33.56
Reliability (Ours)	SFT	94.16	91.93	91.98	82.03	96.08	89.13	89.33	83.01	90.69	83.86	83.09	64.63
	+ DPO Joint	<b>95.29</b>	<b>92.88</b>	<b>93.03</b>	80.86	<b>95.65</b>	<b>89.86</b>	<b>89.51</b>	82.00	<b>91.23</b>	<b>85.78</b>	<b>83.95</b>	64.48
	+ DPO Post	94.02	92.23	92.12	<b>82.89</b>	<b>96.15</b>	89.29	<b>89.60</b>	<b>84.12</b>	86.70	84.61	81.00	<b>67.51</b>

Table 1: Reliability performance on three DST datasets. *Method denotation:* DPO Joint: SFT and DPO Joint Tranining. DPO Post: DPO Post Training. *Metric denotation:* Prec<sub>sure</sub>: precision of "sure" predictions. Rec<sub>total</sub>: recall of the "sure" and "unsure" predictions. Opti. F1: Optimal-F1. Qual. F1: Quality-F1.

dataset and 2048 on the others. By default, we utilize 1,000samples for Reliability-SFT Training and 2,000samples for DPO Training. All models are trained for 1 epoch using NVIDIA A800 GPUs.

## 5.2 Reliability Evaluation

The results of our experiments on model reliability are summarized in Table 1. We compare our reliability-training pipeline with baseline algorithms, emphasizing the model’s self-awareness ability. For **RQ1**, our method significantly outperforms traditional prompt-based and uncertainty-based methods across various datasets, markedly enhancing self-awareness.

Our approach consistently achieves higher Quality-F1 scores, indicating improved effectiveness in categorizing predictions as "sure" or "unsure" based on confidence. Notably, a high Precisionsure indicates reliable "sure" predictions, fostering user trust with minimal verification. Simultaneously, a high Recalltotal ensures that the "unsure" category encompasses a broad range of potentially correct outputs, forming a solid basis for further refinement. Collectively, these metrics contribute to a higher Optimal-F1 score, representing the theoretical maximum post-refinement performance. In other words, our system offers improved performance by effectively balancing precision and recall, ensuring both immediate reliability and the potential for improved post-refinement accuracy.

In our approach, methods incorporating DPO generally outperform SFT-only methods. Specifically, models trained with DPO Joint Training achieve higher Optimal-F1 scores, while those undergoing Post Training yield superior Quality-F1

scores. This suggests that Joint Training models are more conservative, relying on subsequent refinement, whereas Post Training models exhibit greater confidence, yielding predictions of generally higher quality. This discrepancy arises from the differing sampling distributions of DPO data; Joint Training samples from a pre-SFT distribution, where performance is relatively poor, leading to a dataset with a higher proportion of "unsure" predictions, resulting in more conservative model behavior. We will further explore the differences between these two approaches in subsequent sections.

Notably, some baselines, such as Self-Consistency, often achieve higher Precision<sub>sure</sub>, Recall<sub>total</sub>, and Optimal-F1 scores compared to others, suggesting initial superiority. However, this indicates an undesirable scenario: an overly conservative model that frequently labels predictions as "unsure." This approach leads to a suboptimal balance between prediction types, evidenced by its significantly lower Quality-F1 score. In extreme cases, a model could assign "unsure" to all predictions, thereby maximizing Optimal-F1. However, this would merely shift the burden to the refinement stage, reduce immediate usability and fails to balance reliability and helpfulness.

## 5.3 Refinement for Unsure Prediction

We trained an automatic refinement model for each dataset as described in Section 4.3. We refine the predictions labeled as "unsure" from the initial stage and merge the refined outputs with the "sure" predictions to generate the final results. The overall evaluation results are presented in Table

2. For **RQ2**, It can be observed that our method, after refinement, achieves significantly better performance with substantial improvements in Slot-F1 and JGA.

Method Type	Method	MultiWOZ-2.4		BiTOD		SGD	
		Slot-F1 ↑	JGA ↑	Slot-F1 ↑	JGA ↑	Slot-F1 ↑	JGA ↑
<i>LLAMA3-8B</i>							
Prompt	Direct	78.86	36.01	82.40	36.46	32.41	13.58
	Verbose	76.80	36.69	76.93	27.99	23.11	9.00
	SR	78.68	29.72	76.50	30.74	29.90	11.25
Uncertainty	Prob	75.89	24.19	73.88	24.86	28.48	9.63
	SC	70.98	20.70	69.98	24.86	25.60	4.39
	P(True)	73.88	19.18	67.07	19.03	26.29	7.34
Reliability (Ours)	SFT	85.95	53.76	85.21	60.09	74.04	27.62
	+ DPO Joint	86.47	54.31	84.53	61.37	75.47	<b>29.18</b>
	+ DPO Post	<b>86.55</b>	<b>56.33</b>	<b>85.87</b>	<b>64.95</b>	73.96	28.13
<i>Owen2.5-7B</i>							
Prompt	Direct	78.66	36.35	83.23	37.39	42.02	16.93
	Verbose	67.59	26.45	74.37	27.38	27.79	9.19
	SR	68.88	19.09	67.18	19.15	34.77	8.78
Uncertainty	Prob	74.18	23.56	73.97	25.64	35.91	10.81
	SC	59.34	23.45	65.60	28.31	34.34	3.06
	P(True)	67.52	17.05	66.25	17.99	33.66	8.07
Reliability (Ours)	SFT	84.50	51.90	85.29	58.39	74.86	28.02
	+ DPO Joint	87.06	52.41	85.44	62.31	74.85	27.06
	+ DPO Post	<b>87.57</b>	<b>53.19</b>	<b>85.86</b>	<b>63.61</b>	<b>74.95</b>	<b>28.51</b>

Table 2: Overall performance after refinement. Note: for Direct baseline we conduct a fully refinement since there's no confidence label.

It is important to note that, referring to Section 5.2, despite a higher theoretical performance limit (Optimal-F1), DPO Joint Training models exhibit suboptimal Slot-F1 and JGA after refinement to Post Training models. This discrepancy stems from the limitations of the refinement model. Since the refinement model is not ideal, excessive "unseen" predictions can lead to residual errors after refinement, incurring significant costs without proportional gains. This highlights the importance of a balanced "sure" and "unseen" classification. An ideal model should minimize unnecessary "unseen" predictions and refine only when truly warranted, maximizing both efficiency and overall performance.

#### 5.4 Scalability Analysis

To evaluate the scalability of our methods, we conducted experiments on foundation models with varying task capabilities. We regard the original LLAMA3 8B model as a lower-performance model (denoted as Low) and applied supervised fine-tuning (SFT) on a subset of the dataset to obtain models with improved performance. We trained two foundation models using 1,000 samples (Medium) and 10,000 samples (High) and compared their performance against baseline algorithms, with results shown in Figure 3. For

**RQ3**, our methods consistently outperform baselines across different foundation models after refinement. Detailed results are available in Appendix C, leading to consistent conclusions.

Additionally, as illustrated in Appendix Table 4, the advantage of our DPO Post Training method over Joint Training increases with the improvement of foundation model performance. This is due to the offline nature of our DPO algorithm; as foundational model performance improves, the limitations of the Joint Training data distribution relative to Post Training become more pronounced, impacting overall performance. Notably, in the SGD dataset, DPO Joint Training becomes less beneficial as the foundation model's performance rises.

#### 5.5 Cost Analysis

In this work, "Cost" refers to the additional overhead from the refinement process. We define the cost of refining an "unseen" prediction as a constant value of 1, equating Cost to the number of "unseen" predictions. For better comparison, we express Cost as the proportion of "unseen" predictions.

We compare the performance and Cost of various baselines, as shown in Figure 4. Results are presented only for the MultiWOZ-2.4 dataset, with complete findings available in Appendix D. Our methods demonstrate significant performance improvements while maintaining a relatively low Cost. Furthermore, as illustrated in Appendix Table 3, Cost decreases significantly with improved foundation model performance. This aligns with intuitive expectations: as the model strengthens, the proportion of uncertain predictions diminishes.

#### 5.6 DPO Preference Strategies

We construct our DPO preference data using Weighted-F1 as the ranking metric, as detailed in Section 3.2.1. Weighted-F1 includes two adjustable  $\alpha$  parameters:  $\alpha_1$  influences Weighted Precision, while  $\alpha_2$  affects Weighted Recall. Using the LLAMA3 8B Reliability-SFT model, we generated DPO preference data with various  $\alpha$  values and conducted Post Training. Test results on the MultiWOZ-2.4 dataset are shown in Figures 5a and 5b. The model achieves optimal performance with  $\alpha_1$  set to 0.25 and  $\alpha_2$  to 0.75, reaching near-best results at a relatively low cost.

Analysis indicates that increasing  $\alpha_1$  makes Weighted Precision more sensitive to errors in "unseen" predictions, reducing tolerance and generating fewer "unseen" outputs. In contrast, increasing

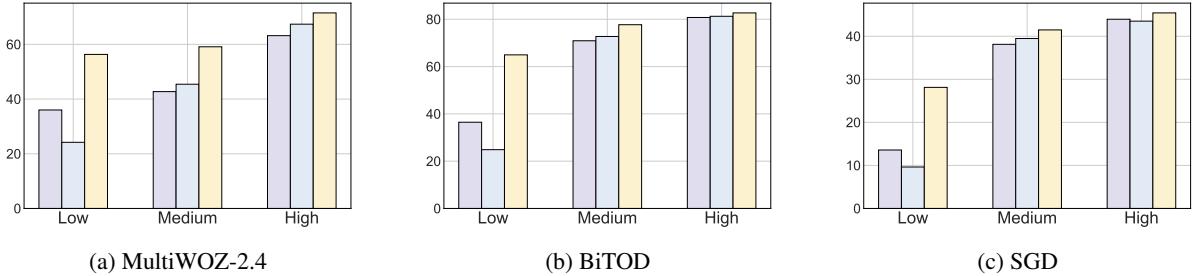


Figure 3: Comparison of different methods under multiple foundation models of various task ability. Figure (a), (b) and (c) illustrates the **JGA after refinement (%)** on three datasets. The terms Low, Medium, and High represent the task ability of the foundation models, referring to the original LLAMA3-8b model, a model trained on 1,000 samples, and a model trained on 10,000 samples, respectively. Legend: ■ Prompt-based Method (Direct), ■ Uncertainty-based Method (Prob), ■ Reliability Method (ours, SFT+DPO Post Training).

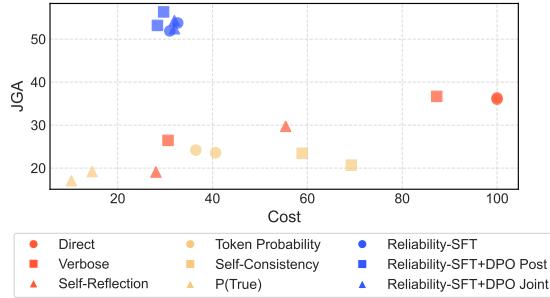


Figure 4: Comparison of different methods on performance with corresponding cost on MultiWOZ-2.4.

$\alpha_2$  raises the reward for successful recall from "unseen" predictions, encouraging the model to produce more "unseen" predictions and adopt a more conservative stance. Practically, the optimal alpha settings depend on the scenario and refinement model performance. When the refinement model is underperforming, it may be unwise to incur significant costs for marginal performance gains. The flexible adjustment of both  $\alpha$  parameters allows for adaptation to diverse real-world situations.

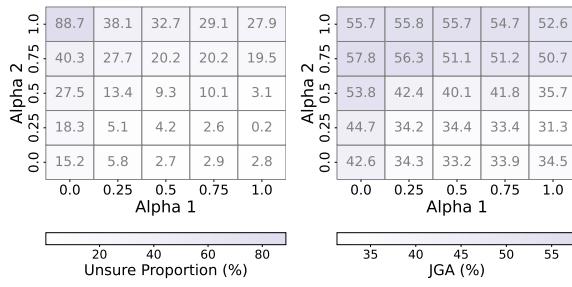


Figure 5: Comparison of different DPO preference strategies on model behavior and performance.

## 5.7 Refinement Model comparison

We employ various refinement models to enhance the predictions of the LLAMA3 Reliability-

SFT+DPO Post model, comparing their performance as shown in Table 6. In addition to our trained refinement models, we utilize GPT-4o and the recently popular DeepSeek series ([DeepSeek-AI et al., 2025](#)), including the distilled 8B and 70B versions, all exhibiting strong reasoning capabilities. The results indicate significant performance variations among the different refinement models, with the DeepSeek 8B model underperforming, while the 70B model closely matches GPT-4o. Furthermore, our task-specific fine-tuning with CoT substantially improves refinement accuracy.

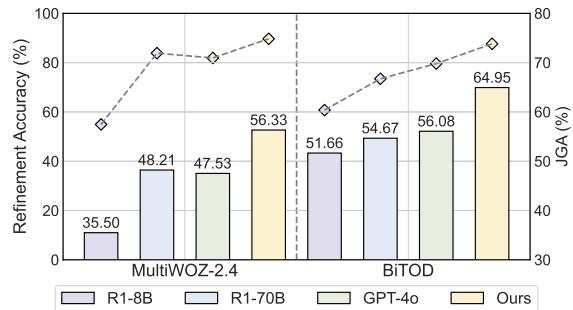


Figure 6: Comparison of different refinement models. The line chart represents "Refinement Accuracy", while the histogram represents "JGA".

## 6 Conclusion

In this work, we propose a two-stage framework that enhances the reliability of large language models by integrating fast and slow thinking paradigms. Through precise confidence assessment and high-accuracy refinement, we achieve a balance between reliability and usability. Extensive evaluations demonstrate that our method significantly improves model self-awareness and performance across complex tasks, establishing a new paradigm for enhancing the reliability of language models in error-sensitive applications.

## 7 Limitations

While our EKBM framework demonstrates promising results, several limitations warrant discussion. First, the refinement stage incurs computational overhead proportional to the volume of "unsure" predictions, which may hinder scalability in latency-sensitive applications. Second, the efficacy of refinement critically depends on the performance of the secondary model; suboptimal refinement can propagate errors, offsetting gains from uncertainty labeling. Third, the manual tuning of Weighted-F1 parameters ( $\alpha_1, \alpha_2$ ) introduces scenario-specific biases, limiting generalizability. Finally, our evaluation focuses on dialogue state tracking tasks—broader validation across diverse domains is necessary to assess universal applicability. Addressing these challenges could further enhance the framework's practicality and robustness.

## References

- Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv:2311.05232*.
- Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrom, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.
- Ziwei Ji, Tiezheng Yu, Yan Xu, Nayeon Lee, Etsuko Ishii, and Pascale Fung. 2023. Towards mitigating llm hallucination via self reflection. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1827–1843.
- 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*.
- Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.
- 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.
- Moxin Li, Yong Zhao, Yang Deng, Wenxuan Zhang, Shuaiyi Li, Wenya Xie, See-Kiong Ng, and Tat-Seng Chua. 2024b. Knowledge boundary of large language models: A survey. *arXiv preprint arXiv:2412.12472*.
- Z Lin, A Madotto, GI Winata, P Xu, F Jiang, Y Hu, C Shi, and P Fung. Bitod: A bilingual multi-domain dataset for task-oriented dialogue modeling. arxiv 2021. *arXiv preprint arXiv:2106.02787*.
- Alisa Liu, Zhaofeng Wu, Julian Michael, Alane Suhr, Peter West, Alexander Koller, Swabha Swayamdipta, Noah A Smith, and Yejin Choi. 2023. We're afraid language models aren't modeling ambiguity. *arXiv preprint arXiv:2304.14399*.
- Potsawee Manakul, Adian Liusie, and Mark JF Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*.
- Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8689–8696.
- Daya Guo DeepSeek-AI, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.
- Hsiu-Yuan Huang, Yutong Yang, Zhaoxi Zhang, Sanwoo Lee, and Yunfang Wu. 2024. A survey of uncertainty estimation in llms: Theory meets practice. *arXiv preprint arXiv:2410.15326*.

- Shuchang Tao, Liuyi Yao, Hanxing Ding, Yuexiang Xie, Qi Cao, Fei Sun, Jinyang Gao, Huawei Shen, and Bolin Ding. 2024. When to trust llms: Aligning confidence with response quality. *arXiv preprint arXiv:2404.17287*.
- Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D Manning. 2023. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. *arXiv preprint arXiv:2305.14975*.
- SM Tonmoy, SM Zaman, Vinija Jain, Anku Rani, Vipula Rawte, Aman Chadha, and Amitava Das. 2024. A comprehensive survey of hallucination mitigation techniques in large language models. *arXiv preprint arXiv:2401.01313*.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.
- Hongshen Xu, Ruisheng Cao, Su Zhu, Sheng Jiang, Hanchong Zhang, Lu Chen, and Kai Yu. 2024a. **A birgat model for multi-intent spoken language understanding with hierarchical semantic frames.** *Preprint*, arXiv:2402.18258.
- Hongshen Xu, Zichen Zhu, Lei Pan, Zihan Wang, Su Zhu, Da Ma, Ruisheng Cao, Lu Chen, and Kai Yu. 2025. **Reducing tool hallucination via reliability alignment.** *Preprint*, arXiv:2412.04141.
- Hongshen Xu, Zichen Zhu, Situo Zhang, Da Ma, Shuai Fan, Lu Chen, and Kai Yu. 2024b. Rejection improves reliability: Training llms to refuse unknown questions using rl from knowledge feedback. *arXiv preprint arXiv:2403.18349*.
- An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.
- Zhewei Yao, Reza Yazdani Aminabadi, Olatunji Ruwase, Samyam Rajbhandari, Xiaoxia Wu, Ammar Ahmad Awan, Jeff Rasley, Minjia Zhang, Conglong Li, Connor Holmes, et al. 2023. Deepspeed-chat: Easy, fast and affordable rlhf training of chatgpt-like models at all scales. *arXiv preprint arXiv:2308.01320*.
- Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. 2021. Multiwoz 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. *arXiv preprint arXiv:2104.00773*.
- Xunjian Yin, Xu Zhang, Jie Ruan, and Xiaojun Wan. 2024. Benchmarking knowledge boundary for large language model: A different perspective on model evaluation. *arXiv preprint arXiv:2402.11493*.
- Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. 2023. Siren’s song in the ai ocean: a survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*.
- Bin Zhou, Xinyu Li, Tianyuan Liu, Kaizhou Xu, Wei Liu, and Jinsong Bao. 2024a. Causalkgpt: industrial structure causal knowledge-enhanced large language model for cause analysis of quality problems in aerospace product manufacturing. *Advanced Engineering Informatics*, 59:102333.
- Kaitlyn Zhou, Jena D Hwang, Xiang Ren, and Maarten Sap. 2024b. Relying on the unreliable: The impact of language models’ reluctance to express uncertainty. *arXiv preprint arXiv:2401.06730*.

## A Refinement Model

We trained an automated refinement model for each dataset. The refinement model conducts in-depth reasoning on unsure predictions based on task-specific criteria, operating within a defined action space. Our refinement models are built upon SFT of LLAMA3 8B utilizing data constructed with OpenAI’s GPT-4o.

**Action Space** The action space of the refinement model includes three actions: remove, reserve, and correct. A prediction should be reserved if entirely correct, should be corrected if partially correct and should be removed if entirely unnecessary. For any unsure predictions, a proper action from these three options is sufficient to achieve the required refinement.

**Data Construction** We adopt a multi-step reasoning Chain-of-Thought (CoT) paradigm. The specific reasoning instructions and processes are detailed in Table 5. For data construction, we used OpenAI’s GPT-4o model. During the data generation process, we provided the model with the prediction to be refined and the corresponding golden label, then instructed it to generate a reasoning process based on predefined principles, as shown in Table 6. We combined the instruction, input, the prediction to be refined, the thinking steps, and the refinement result into a final training sample. For each dataset, we sampled 5,000 examples from the model’s unsure predictions as well as from randomly selected data in the training set, ensuring a balanced distribution of data across three actions.

## B Baselines Details

Here we detail the specifics of the baselines.

### Prompt-based

- Direct: Utilizes prompts similar to those in Table 8 to guide the model in generating predictions directly, without incorporating confidence labels.
- Verbose: Employs prompts akin to those in Figure 8 to direct the model, which outputs confidence labels (sure or unsure) alongside predictions based on its self-assessment.
- Self-Reflection (SR): Builds on predictions from the Direct baseline, allowing the model to reflect on each prediction to assess its confidence level.

## Uncertainty-based

- Token Probability (Prob): Computes average token-level probabilities, classifying outputs as "sure" or "unsure" based on a threshold  $t$ , with probabilities normalized and  $t$  set to 0.8.
- Self-Consistency (SC): Samples  $N$  times, classifying predictions that appear consistently at proportion  $p$  as "sure," while others are marked as "unsure." We conduct ten samples ( $N = 10$ ) and set  $p$  to 50%.
- P(True): Based on predictions from the Direct baseline, this method evaluates the correctness of each prediction individually, marking those with a higher probability of responding "True" than "False" as "sure," otherwise as "unsure."

Method Type	Method	MultiWOZ-2.4		BiTOD		SGD	
		LLAMA	Qwen	LLAMA	Qwen	LLAMA	Qwen
Prompt	Direct	100.00	100.00	100.00	100.00	100.00	100.00
	Verbose	87.28	30.62	80.69	38.05	90.30	74.07
	SR	55.43	28.07	95.50	8.55	84.77	15.07
Uncertainty	Prob	36.47	40.65	50.46	52.91	56.68	53.01
	SC	69.26	58.91	58.05	62.59	96.80	92.71
	P(True)	14.63	10.22	13.35	1.82	1.52	1.71
Reliability	SFT	32.67	30.98	20.03	21.78	51.24	55.96
	+ DPO Joint	31.96	31.98	21.19	23.63	57.72	45.29
	(Ours)	27.69	28.38	10.54	18.14	40.68	53.10

Table 3: Cost (proportion of unsure predictions, %) comparison of different methods.

## C Scalability Analysis

The detailed results are presented in Table 4, demonstrating that our method maintains superiority across various capability baselines, achieving more reliable outcomes (higher Precision<sub>sure</sub>, Recall<sub>total</sub>, and Quality-F1) and enhanced overall performance post-refinement (higher JGA).

## D Refinement Cost

The detailed overall results of cost are presented in Table 3. The relationships between cost and performance for the BiTOD and SGD datasets are illustrated in Figure 7 and Figure 8, respectively.

## E LLM Instructions

The instructions used in this work are presented in this chapter, taking the MultiWOZ-2.4 dataset as an example. The instructions for the BiTOD dataset are similar.

Method Type	Method	MultiWOZ-2.4						BiTOD				SGD		
		Prec <sub>sure</sub> ↑	Rec <sub>total</sub> ↑	Qual. F1 ↑	JGA ↑	Prec <sub>sure</sub> ↑	Rec <sub>total</sub> ↑	Qual. F1 ↑	JGA ↑	Prec <sub>sure</sub> ↑	Rec <sub>total</sub> ↑	Qual. F1 ↑	JGA ↑	
<i>1000Samples-Trained LLAMA3 Based</i>														
Prompt	Direct	91.07	86.28	87.84	42.73	93.01	92.78	92.82	70.91	82.28	85.10	82.77	38.13	
Uncertainty	Prob SC	91.84 92.68	86.52 <b>92.37</b>	86.84 84.31	45.43 43.18	94.93 95.70	93.55 95.12	93.66 92.05	72.73 70.37	84.45 86.61	85.24 92.30	82.80 78.71	39.48 28.62	
Reliability (Ours)	SFT + DPO Joint + DPO Post	95.42 95.11 <b>96.33</b>	92.11 90.68 92.15	89.86 89.05 <b>89.91</b>	55.62 58.94 <b>59.12</b>	94.97 96.06 <b>96.22</b>	94.74 95.60 95.43	93.80 95.11 <b>95.43</b>	73.64 75.36 <b>77.70</b>	85.16 81.66 <b>86.67</b>	85.14 82.82 <b>85.44</b>	81.06 79.69 <b>83.27</b>	39.23 37.62 <b>41.48</b>	
<i>10000Samples-Trained LLAMA3 Based</i>														
Prompt	Direct	94.51	93.30	93.63	63.19	95.35	95.14	94.34	80.76	83.24	83.75	83.11	43.96	
Uncertainty	Prob SC	95.77 96.24	94.06 <b>96.40</b>	92.95 92.13	67.39 67.61	95.95 <b>97.83</b>	95.53 <b>97.81</b>	95.23 93.42	81.26 78.04	84.37 <b>88.13</b>	84.61 <b>92.42</b>	83.08 79.26	43.52 38.20	
Reliability (Ours)	SFT + DPO Joint + DPO Post	96.98 96.98 <b>97.71</b>	95.86 95.61 96.32	94.40 94.71 <b>94.80</b>	70.85 70.84 <b>71.51</b>	96.44 97.15 96.57	96.20 96.65 96.35	96.03 95.96 <b>96.59</b>	80.54 82.35 <b>82.69</b>	85.30 82.31 86.29	83.92 82.36 85.53	83.57 80.75 <b>84.21</b>	45.11 40.48 <b>45.43</b>	

Table 4: Reliability performance of methods on models with higher task capability. Note: the JGA column indicates results after refinement.

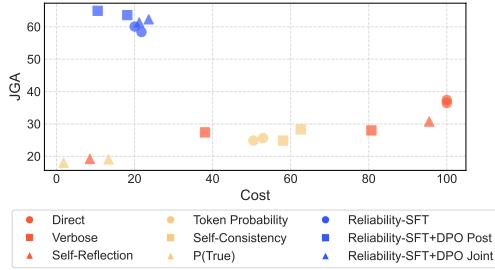


Figure 7: Comparison of different methods on performance with corresponding cost on BiTOD.

## E.1 Inference Instruction

The prompts used with the original untrained LLAMA3 model are shown in Tables 7 and 8. Among them, the Direct Inference Instruction requires the model to directly output the prediction results, while the Reliability Inference Instruction additionally requires the model to output a confidence label alongside the prediction.

## E.2 Training Instruction

The prompts used with the trained LLAMA3 model are shown in Tables 9 and 10. The Direct Training Instruction is designed to train foundation models capable of directly outputting prediction results. In contrast, the Reliability Training Instruction is designed to train models capable of explicitly outputting confidence labels alongside predictions.

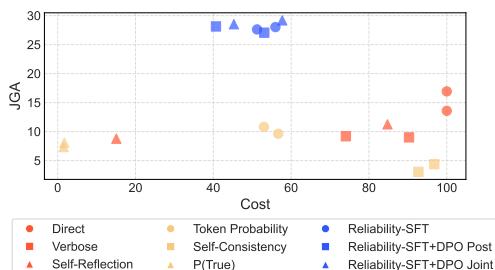


Figure 8: Comparison of different methods on performance with corresponding cost on SGD.

---

## Refinement Examples

---

Give you a dialog history between USER and SYSTEM, and a slot-value pair extracted from the dialog, I want you to analyze on the dialog history and review/refine the given dialog state slot-value pair to make them more accurate and reliable.

### **## Domain Slot Spaces**

```
{  
    "hotel": {  
        "name": {  
            "data_type": str,  
            "example": "hamilton lodge"  
        },  
        ...  
    },  
    ...  
}
```

### **## Question**

#### **Dialog history:**

USER: Hello. I need train to London liverpool Street. SYSTEM: Where are you departing from and do you have a time preference? USER: Yes, I'd like to leave Cambridge sometime after 7:00. SYSTEM: I have one that leaves at 7:59 and 4 more that depart every two hours after. USER: The 7:59 will be fine.

#### **Predicted slot-value pair:**

train: leaveat = 7:59

#### **Refinement:**

Thinking steps:

step1: structure aspect: "leaveat" is one of the slots in the train domain and the value is incorrectly specified as "7:59" (formation error). Precise time should strictly follow the "HH:MM" format.

step2: semantic aspect: "leaveat" is not a "name"-related slot, so the value must be extracted from the user's direct intent. Although the system provides a train that departs at 7:59, the user only mentions that he'd like to leave Cambridge after 7:00. Considering the formation, the correct value should be 07:00.

step3: domain train correct; slot leaveat correct; value 7:59 incorrect.

step4: the predicted slot-value pair is incorrect in both structure and semantic, and the value should be corrected.

**Refine action:** correct

**Refined slot-value pair:** train: leaveat = 07:00

---

Table 5: An example of the Refinement process.

---

## GPT4o Prompt For Refinement Thinking Steps Generation

---

Give you a dialog history between USER and SYSTEM, a predicted slot-value pair and a gold refinement, I want you to analyze on the dialog history, refer to the examples I've provided and the golden refinement, and then continue to provide the thinking steps following the predefined principles.

Firstly, we use json dict to describe the slots and their corresponding value space. Then, we will specify the requirements you need to comply and provide some examples. Last, we will present you with the dialog history, the predicted dialog state slot-value pair and the gold refinement for you to perform continuation.

### ## Domain Slot Spaces

```
{  
    "hotel": {  
        "name": {  
            "data_type": str,  
            "example": "hamilton lodge"  
        },  
        ...  
    },  
    ...  
}
```

### ## Meta Requirements

1. Analyze the dialog history and the predicted slot-value pair carefully, correct and refine the pair according to the following Detailed Refinement Principles.
2. For the given slot-value pair, you should only take one of the actions of "reserve", "remove" or "correct" according to the following principles.
  - a. reserve: if the slot is relevant to the dialog history and the value is correctly specified, you should keep the pair.
  - b. remove: if the slot is irrelevant to the given dialog history or there's not corresponding value could be extracted from the history, you should print "none" indicating that the pair has been removed.
  - c. correct: if the slot is relevant to the dialog history and the value is incorrectly specified, you should extract the correct value and replace it.
3. You need to think and reason in the same way as the Examples. Please refer to the Examples for formatting requirement. Make sure the output is all lowercase. Only focus on the given slot-value pair and do not consider any other slot. only provide the reasonable thinking steps, do not provide any extra prefixes or suffixes.

### ## Detailed Refinement Principles

...

### ## Examples

...

### ## Question

**Dialog history:**

USER: ... SYSTEM: ...

**Predicted slot-value pair:**

train: leaveat = 7:59

**Refinement:**

**Refine action:** correct

**Refined slot-value pair:** train: leaveat = 07:00

**Thinking steps:**

---

Table 6: The prompt used to instruct GPT-4o to generate thinking steps for refinement

---

## Direct Inference Instruction

---

Give you a dialog history between USER and SYSTEM, I want you to analyze on it and generate the dialog state.

Firstly, we use json dict to describe the slots and their corresponding value space in each domain. Then, we will specify the requirements you need to comply. Last, we demonstrate some use cases.

## Domain and Slot Space

The available ontology of the dialog state is as follows:

```
{  
    "hotel": {  
        "name": {  
            "data_type": str,  
            "example": "hamilton lodge"  
        },  
        ...  
    },  
    ...  
}
```

## Requirements

1. Analyze the dialog history carefully and fill the relevant domains and slots.
2. For slots with a specified value range, responses must fall within the provided range. For slots without specified value range, the answer must be extracted from the history. Set value as "dontcare" if user doesn't have a preference.
3. You only need to consider the domains and slots that are relevant to the conversation history. Do not include those irrelevant in your response and avoid presenting empty domains or slots.
4. Your answer should also be in one-line jsonl format and make sure the output is all lowercase. Do not provide any extra prefixes or suffixes or any explanations.

## Output Dialog State Example

```
{"hotel": {"area": "centre", "name": "alexander bed and breakfast", "parking": "yes", "type": "guest-house"}, "attraction": {"name": "kambar"}}
```

## Examples

shots...

---

Table 7: Instruction used in Direct, Token probability and Self-consistency baselines.

---

## **Reliability Inference Instruction**

---

Give you a dialog history between USER and SYSTEM, I want you to analyze on it and generate the dialog state.

Firstly, we use json dict to describe the slots and their corresponding value space in each domain. Then, we will specify the requirements you need to comply. Last, we demonstrate some use cases.

### **## Domain and Slot Space**

The available ontology of the dialog state is as follows:

```
{  
    "hotel": {  
        "name": {  
            "data_type": str,  
            "example": "hamilton lodge"  
        },  
        ...  
    },  
    ...  
}
```

### **## Requirements**

1. Analyze the dialog history carefully and fill the relevant domains and slots.
2. For slots with a specified value range, responses must fall within the provided range. For slots without specified value range, the answer must be extracted from the history. Set value as "dontcare" if user doesn't have a preference.
3. You should try to cover as many domains and slot-value pairs relevant to the conversation history as possible. Mark each slot with either "sure" or "unsure" according to your confidence in it. For slot-value pairs that you are pretty sure that they are truly relevant and should be included, and meanwhile you have great confidence in the correctness of the value, you should tag it with "sure". Otherwise, you should tag the slot-value pair with "unsure".
4. Principles:
  - a. Goal one: Achieving as close to 100% accuracy as possible in those "sure" slot-value pairs.
  - b. Goal two: The "sure" part along with the "unsure" part should cover all possible slots involved in the dialog history as much as possible.
  - c. Heavy penalization: Providing incorrect slot-value pairs in the "sure" part.
  - d. Heavy penalization: Missing slot-value pairs that should be extracted.
  - e. Light penalization: Providing incorrect or redundant slot-value pairs in the "unsure" part.
5. Your answer should also be in one-line jsonl format and make sure the output is all lowercase. Do not provide any extra prefixes or suffixes or any explanations. Please refer to the Output Dialog State Example for formatting requirement.

### **## Output Dialog State Example**

```
{"hotel": {"area": {"value": "centre", "confidence": "sure"}, "name": {"value": "alexander bed and breakfast", "confidence": "sure"}, "parking": {"value": "yes", "confidence": "unsure"}, "type": {"value": "guesthouse", "confidence": "sure"}}, "attraction": {"name": {"value": "kambar", "confidence": "unsure"}}}
```

### **## Examples**

shots...

---

Table 8: Instruction used with original LLAMA model in Verbose baselines.

---

## **Direct Training Instruction**

---

Generate the dialogue state based on the given dialogue context.

---

Table 9: Instruction used in Direct SFT Training.

---

**Reliability Training Instruction**

---

Generate the dialogue state based on the given dialogue context. Ensure the results are as reliable as possible, the 'sure' parts are as accurate as possible, and the overall coverage includes all relevant slots.

---

Table 10: Instruction used in Reliable Training, guiding the model to explicitly extinguish between sure and unsure responses.