Belief-Guided Inference Control for Large Language Model Services via Verifiable Observations

Wenhao Yuan[1], Chenchen Lin[2], Jian Chen[1], Jinfeng Xu[1], Shuo Yang[1], Edith Cheuk Han Ngai[1,*]
[1]The University of Hong Kong, [2]Sun Yat-sen University
wenhao.yuan@connect.hku.hk, chngai@eee.hku.hk

Abstract

In black-box large language model (LLM) services, response reliability is often only partially observable at decision time, while stronger inference pathways incur substantial computational cost, inducing a budgeted sequential decision problem: for each request, the system should decide whether the default low-cost response is sufficiently reliable or whether additional computation should be allocated to improve response quality. In this paper, we propose **Ver**ifiable **O**bservations for Risk-aware **I**nference **C**ontrol (VEROIC), a framework for adaptive inference control in black-box LLM settings, which formulates request-time control as a *partially observable Markov decision process* to capture partial observability and sequential budget coupling. It constructs a lightweight verifiable observation channel from the input-output pair by aggregating heterogeneous quality signals into a belief state over latent response reliability, which is then used by a budget-aware policy to decide whether to return the default output or trigger a higher-cost inference pathway. Experiments on diverse tasks show that VEROIC achieves improved quality-cost trade-offs, stronger risk estimation and calibration, and more robust long-horizon inference control than competitive baselines.

1 Introduction

Large Language Models (LLMs) have become an important substrate for intelligent systems, supporting applications such as programming assistance (Chen et al., 2025; Dinh et al., 2023), content creation (Choi et al., 2025; Cheng et al., 2025; Padmakumar and He, 2024), and scientific reasoning (Zhang et al., 2025; Lyu et al., 2025; Si et al., 2025). In practical deployments, however, LLMs are accessed as black-box services rather than as fully controlled models. Users and downstream systems observe only the input-output interface, while the reliability of each response is not directly observable at decision time and must instead be inferred from imperfect observable evidence (Liang et al., 2023; Ouyang et al., 2022). Meanwhile, stronger inference pathways, such as larger models or longer reasoning, can improve response quality but incur additional latency and cost.

Such black-box LLM services pose a practical online control problem. At request time, the system cannot directly observe whether the default low-cost response is reliable and must instead rely on imperfect response-side signals to decide whether additional computation should be allocated (Kadavath et al., 2022; Kuhn et al., 2023). Stronger inference paths, such as escalation to a larger model, or iterative refinement, can improve response reliability, but they also incur nontrivial latency and monetary cost (Dohan et al., 2022; Ong et al., 2025). Uniformly applying the strongest configuration is inefficient, while always accepting the default response risks failures. Moreover, the observable evidence available after the default response is often noisy and only partially informative, so myopic one-step routing may be insufficient. The objective is therefore not unconditional quality maximization, but adaptive control of the long-term quality–cost trade-off under partial observability.

Existing work has shown that LLM behavior can be variable and difficult to characterize from surface outputs alone (Richter et al., 2025; Sun et al., 2025; Kemmerzell et al., 2025; Balayn et al., 2025; Yuan et al., 2026). However, many existing approaches still evaluate or optimize LLM systems under the assumption that response quality can be inferred reliably either from observable outputs or from static offline benchmarks, which is violated in black-box online use: the response that appears superficially plausible may still be unreliable, while a conservative response may already be adequate for the task. This creates a gap between offline evaluation and the online control problem faced by real

*Corresponding author.

LLM services, giving rise to two core challenges. ***I. Partial and noisy reliability observations:*** the true reliability of a response is usually not directly observable when the system must act. Lightweight signals extracted from the input-output pair, such as verifiable checks or proxy indicators, are informative but imperfect, and their reliability depends on context. ***II. Budgeted long-horizon inference control:*** when stronger inference paths are costly, the system must repeatedly decide whether to allocate additional computation using limited evidence, balancing quality gains against inference cost under a budget constraint, leading to the question: *How can a black-box LLM service maintain calibrated estimates of latent response reliability and allocate inference resources adaptively over time to optimize the long-term quality-cost trade-off?*

To address this problem, we propose **Ver**ifiable **O**bservations for Risk-aware **I**nference **C**ontrol (VEROIC) in black-box LLM settings. Specifically, we formulate request-time inference management as a partially observable Markov decision process (POMDP) in § 3, where the latent state captures the unobserved reliability condition of the default response at decision time. To address Challenge **I**, we construct calibrated lightweight reliability signals from the input-output pair and integrate them through the structured observation model and Bayesian belief updates in § 4 to infer a belief state over latent response reliability. To address Challenge **II**, we use the belief state to parameterize a budget-aware control policy to decide whether to return the default response or escalate to a higher-cost inference path, enabling adaptive inference allocation under partial observability and sequential budget constraints. Unlike conventional routing methods, VEROIC explicitly treats response reliability as a latent variable and performs inference control through belief-based decision making. Our main contributions are summarized as follows:

- We formulate the black-box LLM inference management as a partially observable sequential decision problem.
- We propose VEROIC, a novel belief-based framework that infers latent response reliability from verifiable checks and proxy signals for budget-aware escalation.
- Experiments show that VEROIC improves quality-cost trade-offs, risk estimation, calibration, and control robustness over baselines.

2 Related Works

Uncertainty Estimation and Calibration in LLMs. A growing body of work studies uncertainty estimation and calibration in large language models (Xia et al., 2025; Liu et al., 2025b; Tao et al., 2025; Xiong et al., 2024). Prior work examines whether LLMs can express calibrated confidence and how confidence signals align with actual correctness (Xiong et al., 2024; Liu et al., 2024, 2025a). Related studies also investigate observable indicators of unreliability, such as hallucination, inconsistency, and unsupported claims, often with auxiliary evaluators or self-consistency signals (Sriramanan et al., 2024; Du et al., 2024; Simhi et al., 2025). However, these methods mostly operate at the level of individual outputs and do not address sequential, budget-aware inference allocation.

Adaptive Inference Routing and Cost-Aware LLM Systems. Recent work explores routing and cost-aware control strategies for improving the quality-cost trade-off of LLM systems. FRUGALGPT routes requests across models with different cost-quality operating points (Chen et al., 2024), while more recent work studies adaptive routing based on certainty or confidence signals (Lu et al., 2025). Related approaches use observable output features, uncertainty scores, or lightweight verifiers to decide when stronger inference is needed (Lu et al., 2025; Chen et al., 2024). Most such methods map current observable features directly to one-step decisions, without explicitly modeling latent response reliability or maintaining a belief state.

Decision Making under Partial Observability. A related line of work highlights that observable outputs alone are often insufficient for reliably characterizing LLM behavior, especially under uncertainty, hidden service variation, or imperfect monitoring (Liang et al., 2023; Sun et al., 2025). Other studies show that evaluation or monitoring signals can be incomplete or misleading, making one-step decisions based only on surface observations unreliable (Denison et al., 2024; Ji et al., 2025; Wang et al., 2025). In contrast to static evaluation or episodic monitoring protocols, our setting focuses on sequential decision making under partial observability, where inference actions are coupled through a shared computation budget. VEROIC builds on this perspective by maintaining a posterior belief over latent response reliability and using it to guide budget-aware escalation over time.

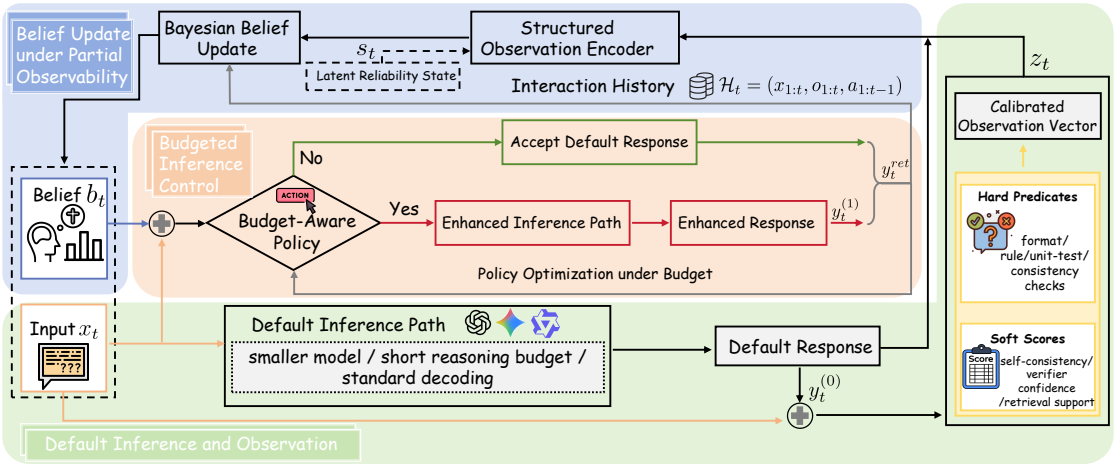

Figure 1: The framework overview of our proposed VEROIC.

3 Problem Formulation

We study adaptive inference control in black-box LLM settings. At each step t, the system receives a request $x_t \in \mathcal{X}$ and first produces a default low-cost response $y_t^{(0)} \in \mathcal{Y}$, such as one generated by a smaller model or a shorter reasoning budget. While the default response is inexpensive, its true reliability is generally not directly observable at decision time. A stronger inference path can improve response reliability, but incurs additional cost. We consider an online setting in which requests arrive sequentially and the enhanced inference budget is shared across the interaction horizon, so earlier decisions affect later computation allocation. The core problem is therefore to decide, for each request, whether to accept the default response or allocate additional inference computation.

We formulate the problem as a *partially observable Markov decision process*, denoted by $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \Omega, \mathcal{T}, \mathcal{O}, \mathcal{R}, \gamma)$. The latent state $s_t \in \mathcal{S}$ captures the unobserved reliability condition of the default response for the current request and is influenced by factors that are not directly visible to the controller, including request difficulty, inference uncertainty, and other hidden factors affecting response reliability. We instantiate the latent state space as a binary set $\mathcal{S} = \{0, 1\}$, where $s_t = 1$ indicates that the default response is reliable and $s_t = 0$ otherwise. Since the latent state is not directly observable, the controller maintains a belief state $b_t(s) = \mathbb{P}(s_t = s \mid \mathcal{H}_t)$, where $\mathcal{H}_t$ denotes the interaction history up to step t. The belief state summarizes the currently available evidence about latent response reliability and therefore serves as the decision state under partial observability. The action space $\mathcal{A} = \{0, 1\}$, where $a_t = 0$ means that the system accepts the default response and returns $y_t^{(0)}$ as the final output, while $a_t = 1$ means that the system triggers an enhanced inference path and obtains an alternative response $y_t^{(1)}$, which may be produced by a stronger model or a larger reasoning budget. The observation space is $\Omega := \mathcal{Y} \times (0, 1)^k$, and the observation at step t is written as $o_t = (y_t^{(0)}, z_t)$, where $y_t^{(0)}$ is the default response and $z_t \in (0, 1)^k$ is a lightweight quality observation vector constructed from $(x_t, y_t^{(0)})$. After generating the default response, the controller extracts the lightweight observation vector and updates its belief over the latent state. The control policy is then defined by $\pi(a_t \mid b_t, x_t)$, where the belief state summarizes the current observation together with the interaction history, and x_t is retained to preserve request-specific contextual information not fully captured by the low-dimensional belief state. The transition kernel $\mathcal{T}(s_{t+1} \mid s_t, x_t, a_t)$ models the sequential evolution of latent reliability conditions across requests and can capture service-side non-stationarity when such effects are present. The observation kernel $\mathcal{O}(o_t \mid s_t, x_t)$ connects the latent state to the observable default response and quality signals. We will specify the observation in § 4.

Within the POMDP formulation, the controller uses the current belief state to decide whether to accept the default response or invoke enhanced inference. The controller seeks a policy that optimizes long-term utility under a computation budget. The reward trades off the quality of the response against the additional cost of enhanced inference, while optionally penalizing decisions made under high estimated risk. The resulting objective is to allocate stronger inference selectively, rather than

uniformly, so as to optimize the long-term quality-cost trade-off under partial observability.

4 Belief-Based Inference Control

4.1 Lightweight Observation Construction

To achieve belief-based inference control without incurring the cost of stronger inference on every request, we construct a lightweight quality observation channel from the input-output pair $(x_t, y_t^{(0)})$. The goal is to approximate the latent quality condition of the default response using low-cost but informative signals that are available immediately after the default inference pass. Since no single signal is sufficiently reliable across tasks and requests, we combine heterogeneous evidence sources into a unified observation vector for downstream belief updating. (i) *Hard verifiable checks* h_j are Boolean predicates that capture discrete properties that can be programmatically evaluated, such as format validity, rule satisfaction, or unit-test results. Their instantaneous realizations are defined as

$$\tilde{h}_t^{(j)} = \mathbb{1}\{h_j(x_t, y_t^{(0)})\}, j \in [k_h]. \quad (1)$$

(ii) *Soft proxy scores* g_i are bounded score functions that provide continuous quality-related evidence, such as self-consistency, verifier confidence, or retrieval support, which produce the raw scores

$$\tilde{g}_t^{(i)} = g_i(x_t, y_t^{(0)}), i \in [k_g]. \quad (2)$$

Due to the heterogeneous and noisy signals, we further smooth and calibrate them before belief updating. For hard signals, we maintain a sliding-window summary over the most recent W realizations to reduce sensitivity to one-step noise and to capture short-term temporal consistency. For soft scores, we apply a monotone calibration map so that different proxy dimensions become comparable as probabilistic indicators of latent response reliability. Let $\Xi_t^{(j)} = \sum_{\ell=\max(1,t-W+1)}^{t} \tilde{h}_\ell^{(j)}$ denote the windowed count for the j-th hard signal. We then define the components as

$$z_{h,t}^{(j)} = \frac{\alpha_{0j} + \Xi_t^{(j)}}{\alpha_{0j} + \beta_{0j} + W}, z_{g,t}^{(i)} = \Gamma_i(\tilde{g}_t^{(i)}; \vartheta_{i,t}), \quad (3)$$

where $(\alpha_{0j}, \beta_{0j})$ are Beta-Bernoulli pseudocounts, and Γ_i is the calibration function with parameters $\vartheta_{i,t}$. By collecting all the calibrated components, it yields the observation vector

$$z_t = (z_{h,t}^{(1:k_h)}, z_{g,t}^{(1:k_g)}) \in (0,1)^k, k = k_h + k_g, \quad (4)$$

which serves as a lightweight observation of the latent quality state associated with the default response. Importantly, z_t is not treated as a direct quality label; rather, it provides noisy but continuous evidence for posterior belief updating under partial observability, enabling the controller to estimate response reliability and determine whether enhanced inference is warranted.

4.2 Structured Observation Model

Given x_t, the system first produces $y_t^{(0)}$ and then constructs the full observation tuple $o_t = (y_t^{(0)}, z_t) \in \Omega$. As the true response-generation mechanism of a black-box LLM service is inaccessible, we introduce a structured observation encoder that maps the default response and proxy signals into state-dependent evidence for posterior belief updating. Let $\psi_t := (\{\Xi_t^{(j)}\}_{j=1}^{k_h}, \{\vartheta_{i,t}\}_{i=1}^{k_g})$ denote the observation-side calibration state associated with the hard and soft signals. We define a state-conditioned compatibility score

$$\ell_\phi(s_t; x_t, y_t^{(0)}, z_t, \psi_t) = \ell_y(s_t; x_t, y_t^{(0)}) + \ell_z(s_t; x_t, y_t^{(0)}, z_t, \psi_t), \quad (5)$$

where ℓ_y captures semantic evidence extracted from the default response, and ℓ_z captures auxiliary reliability evidence provided by the calibrated observation vector. This decomposition separates information contained in the surface response from information provided by external quality proxies, which is important because a plausible-looking response may still be unreliable, while lightweight proxy signals may reveal latent reliability risk not directly visible from the text alone. Within ℓ_z, the hard components $z_{h,t}$ are treated as deterministic calibrated summaries induced by the sliding-window calibration state ψ_t, whereas the soft components $z_{g,t}$ are processed through state-dependent compatibility functions. We use the factorized form

$$\ell_z(s_t; x_t, y_t^{(0)}, z_t, \psi_t) = \ell_h(x_t, y_t^{(0)}, z_{h,t}; \psi_t) + \sum_{i=1}^{k_g} \ell_g^{(i)}(s_t; x_t, y_t^{(0)}, z_{g,t}^{(i)}; \psi_t), \quad (6)$$

where ℓ_h summarizes the contribution of the calibrated hard evidence and $\ell_g^{(i)}$ measures how compatible the i-th soft proxy dimension is with the latent reliability state. ℓ_y, ℓ_h, and $\ell_g^{(i)}$ can be instantiated by lightweight parametric scoring modules over response features and calibrated proxy inputs. Next, we define the observation likelihood up to normalization as $\mathcal{O}(o_t \mid s_t, x_t; \psi_t) \propto$

$\exp(\ell_\phi(s_t; x_t, y_t^{(0)}, z_t, \psi_t))$, which provides a learnable interface that converts response content and calibrated proxy signals into posterior evidence over the latent state. In practice, the encoder parameters ϕ are fitted using sparse verification feedback, delayed task-level supervision, or off-policy trajectories collected during policy optimization. As a result, the observation model remains directly implementable in black-box online settings while preserving the state-based structure needed for belief updating and downstream inference control.

4.3 Bayesian Belief Dynamics

Since the true quality of the response is unobservable, the controller maintains a Bayesian belief over latent reliability states throughout the online interaction process. Given the history up to step t, $\mathcal{H}_t = (x_{1:t}, o_{1:t}, a_{1:t-1})$, the belief state is defined as $b_t(s) = \mathbb{P}(s_t = s \mid \mathcal{H}_t)$, $s \in \mathcal{S}$, which summarizes the currently available evidence about latent response reliability and serves as the sufficient decision state under partial observability. At each step t, the system receives the request x_t, executes the default low-cost inference path, and constructs the observation tuple o_t. The controller then updates its belief through an approximate Bayesian prediction-filtering procedure. First, the previous posterior is propagated through the latent transition model to obtain the predictive belief $\hat{b}_t(s_t) = \sum_{s_{t-1}\in\mathcal{S}} \mathcal{T}(s_t \mid s_{t-1}, x_{t-1}, a_{t-1}) b_{t-1}(s_{t-1})$, capturing temporal persistence in latent reliability conditions induced by the online deployment process. Once observing the response and quality observation vector, the controller performs Bayesian filtering with the structured observation encoder:

$$b_t(s_t) = \frac{\mathcal{O}(o_t \mid s_t, x_t; \psi_t)\, \hat{b}_t(s_t)}{\sum_{s'\in\mathcal{S}} \mathcal{O}(o_t \mid s', x_t; \psi_t)\, \hat{b}_t(s')}, \quad (7)$$

which combines two complementary sources of information and shows how the observation encoder transforms the default response and proxy evidence into posterior mass over the latent reliability state. The default response $y_t^{(0)}$ provides semantic evidence about the quality of the current output, while z_t contributes lightweight but informative proxy signals about latent reliability risk. Their combination refines the predictive belief into a posterior estimate that is more stable than any single noisy signal alone, thereby turning imperfect per-request observations into a decision state suitable for long-horizon, budget-aware inference control.

4.4 Budgeted Risk-Aware Policy Optimization

Based on the current belief state, the controller decides whether to accept the default response or invoke enhanced inference. We optimize this policy under a long-term computation budget, with the goal of balancing realized task utility, inference cost, and residual reliability risk under partial observability. Let y_t^{ret} denote the final returned response, which equals the default output $y_t^{(0)}$ when $a_t = 0$ and the enhanced output $y_t^{(1)}$ when $a_t = 1$. We define the per-step reward as

$$r_t = V(y_t^{\text{ret}}) - C(a_t) - \kappa R(b_t), \quad (8)$$

where $V(\cdot)$ denotes an observable utility surrogate instantiated from task-specific feedback, evaluation signals, or delayed supervision available during training or policy optimization, $C(a_t)$ denotes the additional computation cost induced by enhanced inference, and $\kappa \geq 0$ controls the penalty placed on residual reliability risk at decision time. The belief-dependent risk term is defined as $R(b_t) := \sum_{s\in\mathcal{S}} b_t(s)\varrho(s)$, where $\varrho(s) \geq 0$ is a state-dependent risk score reflecting the unreliability associated with latent state s. To prevent excessive escalation to high-cost inference, we impose a discounted computation budget $\mathbb{E}_\pi[\sum_{t\geq 0} \gamma^t C(a_t)] \leq B$, where discount factor $\gamma \in (0, 1)$ and B is the effective long-run budget. Equivalently, the constraint can be written as $\mathbb{E}_\pi[\sum_{t\geq 0} \gamma^t (B(1-\gamma) - C(a_t))] \geq 0$, which enables a Lagrangian formulation of the constrained control objective. To encourage sufficient exploration during policy learning, we further incorporate a Shannon entropy regularizer $\mathcal{H}(\pi)$, which mitigates premature convergence to degenerate escalation policies under uncertain beliefs. The resulting optimization objective is

$$\max_\pi \mathbb{E}_\pi[\sum\nolimits_{t\geq 0} \gamma^t (r_t + \mu_d(B(1-\gamma) - C(a_t)) + \rho\mathcal{H}(\pi(\cdot \mid b_t, x_t)))], \quad (9)$$

where $\mu_d \geq 0$ is the dual variable associated with the computation-budget constraint and $\rho \geq 0$ is the entropy weight. The entropy term is given by $\mathcal{H}(\pi(\cdot \mid b_t, x_t)) = -\sum_{a\in\mathcal{A}} \pi(a \mid b_t, x_t) \log \pi(a \mid b_t, x_t)$. In practice, we optimize the belief-conditioned policy with actor-critic style policy gradient updates under the Lagrangian objective. The objective encourages the policy to allocate stronger inference only when the expected quality gain justifies the additional computation

cost. When the posterior belief indicates that the default response is likely reliable, the controller is encouraged to preserve computation by accepting the low-cost output; when the belief places substantial mass on high-risk states, the controller is encouraged to trigger enhanced inference despite its additional cost. In this way, the learned policy achieves adaptive long-horizon control of the quality-cost trade-off under partial observability.

5 Experiments

5.1 Experimental Setups

Datasets. We evaluate our method on benchmark datasets covering diverse reasoning paradigms. **Math Reasoning** involves solving problems that require multi-step symbolic manipulation and logical deduction. We adopt GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) for this category. **Code Generation** evaluates the ability to synthesize executable programs from natural language specifications, and we consider HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021). **Question Answering** (QA) requires combining factual knowledge with reasoning over one or multiple pieces of evidence, testing both inference accuracy and factual consistency. We evaluate on HotpotQA (Yang et al., 2018), 2WikiMHQA (Ho et al., 2020), and PopQA (Mallen et al., 2023).

Baselines. We compare our method against several representative baselines. FIXED, RANDOM, and PERIODIC are budget-matched heuristic strategies. OUTPUT-TRIGGER (Lu et al., 2025; Chen et al., 2024) is a rule-based router that escalates on predefined output-side signals, such as failed checks. CONFIDENCE-THRESHOLD (Xiong et al., 2024; Tao et al., 2025) escalates when a single confidence or risk score crosses a budget-calibrated threshold. RISK PREDICTOR (Liu et al., 2024, 2025b) trains a supervised model on the observation features to predict default-response failure risk, and escalates when the predicted risk exceeds a budget-calibrated threshold.

Evaluation Metrics. We evaluate the method from three complementary perspectives. **Task Utility**: For reasoning and QA tasks, we report EM and F1 scores; for code generation tasks, we report Pass@1. **Risk Estimation and Calibration**: We report AUROC and AUPRC to measure the ability of the signals to identify low-quality or high-risk outputs, and evaluate probabilistic calibration using the Brier score, negative log-likelihood (NLL), and expected calibration error (ECE). **Long-Horizon Control Performance**: To assess robustness, we report the low-quality occurrence rate (Occ), the conditional value-at-risk (CVaR) to capture tail risk, and the recovery delay (RecD), which measures recovery latency after severe low-quality events.

Inference Settings. We consider two inference settings to evaluate adaptive inference control. (i) *Cross-Model Inference Switching* (CMIS). The system uses a lower-cost model as the default inference path and a stronger model as the enhanced inference path (e.g., LLaMA-3.1-8B $\rightarrow$ 70B). (ii) *Intra-Model Inference Scaling* (IMIS). The underlying base model remains unchanged, but the system varies the effective inference budget within the same model, such as reasoning depth, decoding budget, or refinement strength, yielding different quality-cost operating points.

Implementation Details. We evaluate the controller under a sequential interaction protocol with a fixed episode length $H = 500$. For each dataset, we uniformly sample 500 instances without replacement to form one episode and reveal them one step at a time. At each step, the controller has access only to the current request, the default response, and the interaction history up to that point, but not to future requests. Unless otherwise specified, results are reported under the CMIS setting with two backbone pairs: LLaMA-3.1-8B $\rightarrow$ LLaMA-3.1-70B and Qwen-2.5-7B $\rightarrow$ Qwen-2.5-32B, where the smaller model serves as the default low-cost inference path, and the larger model serves as the enhanced inference path. We set the enhanced inference budget to $\alpha = 0.05$ and the discount factor to $\gamma = 0.99$. In budget-sensitivity experiments, we sweep $\alpha \in \{0.01, 0.02, 0.05, 0.10, 0.20\}$. We set the sliding-window length to $W = 20$. The enhanced inference action is implemented using a higher-quality inference pipeline, instantiated here by a higher-capacity model or a larger reasoning budget. We instantiate $k_h = 4$ (format and constraint checks, consistency or unit-test style checks, answerability or groundedness heuristics, and safety or compliance checks) with $k_g = 2$ (self-consistency and an external lightweight verifier). The inference control policy is trained using an on-policy reinforcement learning algorithm and is frozen during evaluation. We set the enhanced inference cost to $C = 0.20$, the risk penalty weight to $\kappa = 1.0$, the entropy weight to $\rho = 0.01$, and

Table 1: Performance of VEROIC and other baselines on various tasks (Math Reasoning, Code Generation, and Question Answering) under the CMIS setting. The best result in each column is marked in **bold**.

Inference Paths	Methods	GSM8K	MATH	HumanEval	MBPP	HotpotQA		2WikiMHQA		PopQA	
		EM ↑	EM ↑	Pass@1 ↑	Pass@1 ↑	EM ↑	F1 ↑	EM ↑	F1 ↑	EM ↑	F1 ↑
LLaMA-3.1-8B ↓ LLaMA-3.1-70B	FIXED	86.8	45.8	65.5	67.2	51.6	68.7	41.2	57.6	31.1	47.4
	RANDOM	88.9	49.4	69.7	70.3	53.8	70.6	43.4	59.7	33.5	49.2
	PERIODIC	89.4	49.8	69.3	70.6	54.7	70.8	43.9	60.3	33.2	49.6
	OUTPUT-TRIGGER	90.1	51.2	71.3	71.8	55.3	71.6	44.7	61.4	34.2	50.2
	CONFIDENCE-THRESHOLD	90.4	51.5	71.6	72.2	55.6	71.9	44.8	61.6	34.2	50.7
	RISK PREDICTOR	91.2	52.1	72.3	72.6	55.9	72.3	45.3	62.0	34.9	51.0
	VEROIC	**92.4**	**52.7**	**72.9**	**73.2**	**56.2**	**72.8**	**45.8**	**62.6**	**35.3**	**51.5**
Qwen-2.5-7B ↓ Qwen-2.5-32B	FIXED	85.6	44.8	62.8	66.2	51.2	68.3	40.5	56.9	30.2	46.8
	RANDOM	87.9	48.5	66.4	69.6	53.1	70.2	42.8	59.2	32.5	48.7
	PERIODIC	88.1	48.9	66.8	70.1	53.3	70.3	43.1	59.5	32.6	48.9
	OUTPUT-TRIGGER	89.4	50.6	68.7	71.8	54.6	71.3	44.5	60.8	33.7	49.8
	CONFIDENCE-THRESHOLD	89.7	50.9	68.9	72.0	54.9	71.6	44.7	61.0	33.9	50.1
	RISK PREDICTOR	90.6	51.8	70.1	72.3	55.3	71.9	45.1	61.8	34.3	50.5
	VEROIC	**91.4**	**52.3**	**71.2**	**72.5**	**55.8**	**72.1**	**45.8**	**62.3**	**34.8**	**50.8**

Table 2: Risk estimation and probability calibration performance for identifying high-risk or low-quality outputs that may benefit from enhanced inference under the CMIS setting.

Methods	GSM8K					HumanEval					HotpotQA				
	AUROC ↑	AUPRC ↑	Brier ↓	NLL ↓	ECE ↓	AUROC ↑	AUPRC ↑	Brier ↓	NLL ↓	ECE ↓	AUROC ↑	AUPRC ↑	Brier ↓	NLL ↓	ECE ↓
FIXED	0.71	0.32	0.185	0.523	0.085	0.66	0.26	0.205	0.584	0.096	0.64	0.23	0.218	0.615	0.114
RANDOM	0.73	0.41	0.173	0.475	0.076	0.72	0.33	0.197	0.529	0.084	0.69	0.28	0.192	0.563	0.093
PERIODIC	0.77	0.43	0.162	0.467	0.065	0.74	0.38	0.182	0.522	0.076	0.72	0.31	0.185	0.548	0.085
OUTPUT-TRIGGER	0.83	0.48	0.149	0.447	0.062	0.77	0.41	0.171	0.495	0.072	0.77	0.38	0.180	0.528	0.079
CONFIDENCE-THRESHOLD	0.84	0.49	0.152	0.452	0.066	0.78	0.42	0.175	0.505	0.074	0.78	0.39	0.183	0.535	0.082
RISK PREDICTOR	0.85	0.51	0.144	0.432	0.058	0.80	0.46	0.166	0.482	0.065	0.80	0.41	0.172	0.520	0.072
VEROIC	0.87	0.52	0.139	0.417	0.048	0.81	0.48	0.160	0.473	0.057	0.81	0.43	0.166	0.513	0.064

Table 3: Long-horizon control performance with the LLaMA inference-path pair under the CMIS setting.

Methods	GSM8K			HumanEval			HotpotQA		
	Occ ↓	CVaR ↓	RecD ↓	Occ ↓	CVaR ↓	RecD ↓	Occ ↓	CVaR ↓	RecD ↓
FIXED	0.42	0.38	210	0.48	0.41	243	0.52	0.48	276
RANDOM	0.33	0.26	145	0.39	0.34	187	0.45	0.37	216
PERIODIC	0.25	0.22	134	0.34	0.28	166	0.39	0.30	193
OUTPUT-TRIGGER	0.20	0.17	109	0.26	0.22	145	0.31	0.27	168
CONFIDENCE-THRESHOLD	0.19	0.16	112	0.25	0.21	150	0.31	0.26	174
RISK PREDICTOR	0.17	0.14	98	0.23	0.19	133	0.27	0.23	158
VEROIC	0.13	0.11	77	0.19	0.17	114	0.22	0.19	128

the learning rate to $\eta = 0.05$.

5.2 Main Results

We report the performance on Math Reasoning, Code Generation, and Question Answering tasks under the CMIS setting in Table 1. Across both inference-path pairs, VEROIC consistently achieves the strongest overall performance. While the stronger routing baselines CONFIDENCE-THRESHOLD and RISK PREDICTOR already outperform heuristic strategies, VEROIC still provides additional gains. On Math Reasoning benchmarks, VEROIC achieves the highest EM across both backbone pairs, suggesting more effective allocation of additional computation to difficult reasoning instances. For Code Generation tasks, it attains the best Pass@1, indicating that belief-based control more accurately identifies cases that benefit from enhanced inference. On Question Answering benchmarks, VEROIC again obtains the best EM and F1 scores, with particularly clear gains on multi-hop QA, where selective escalation is especially beneficial under partial observability.

Table 2 reports risk estimation and probability calibration performance under the CMIS setting with the LLaMA inference-path pair. Across all three benchmarks, VEROIC consistently achieves the best results on both discrimination and calibration metrics. While the stronger routing baselines CONFIDENCE-THRESHOLD and RISK PREDICTOR already improve substantially over heuristic strategies, VEROIC still provides further gains in both AUROC/AUPRC and calibration quality, indicating a stronger ability to identify high-risk outputs that are likely to benefit from enhanced inference. In particular, VEROIC achieves the highest AUROC and AUPRC on GSM8K, HumanEval, and HotpotQA, suggesting that belief-based inference more effectively separates outputs that require escalation from those that can be safely accepted under the default low-cost path. It also achieves the lowest Brier score, NLL, and ECE across all three tasks, showing that its predicted risk scores are not only more discriminative but also better calibrated to actual output reliability.

In Table 3, we demonstrate long-horizon con-

Table 4: Ablation study of our proposed VEROIC under the CMIS setting.

Methods	GSM8K				HumanEval				HotpotQA				
	EM ↑	Occ ↓	CVaR ↓	RecD ↓	Pass@1 ↑	Occ ↓	CVaR ↓	RecD ↓	EM ↑	F1 ↑	Occ ↓	CVaR ↓	RecD ↓
w/o Risk Signals	91.2	0.23	0.18	106	71.2	0.31	0.25	169	54.3	71.8	0.29	0.25	158
w/o Enhanced Inference	88.7	0.48	0.42	239	66.9	0.55	0.48	275	52.5	68.5	0.52	0.47	263
w/o Adaptive Policy	90.4	0.30	0.24	142	69.4	0.29	0.25	187	54.1	70.6	0.35	0.31	185
VEROIC	92.4	0.13	0.11	77	72.9	0.19	0.17	114	56.2	72.8	0.22	0.19	128

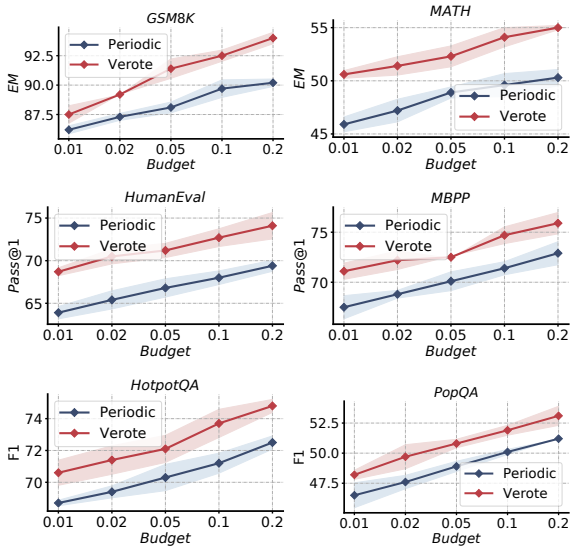

Figure 2: Effect of the enhanced-inference budget on task performance across six benchmark datasets under the CMIS setting with the Qwen inference-path pair.

trol performance under the CMIS setting with the LLaMA inference-path pair. Across all three benchmarks, VEROIC achieves the lowest Occ, CVaR, and RecD, indicating that belief-based control improves long-horizon robustness beyond single-step uncertainty thresholding and supervised risk prediction. Consistent improvements on different tasks suggest that the learned policy yields more stable sequential inference behavior across domains.

5.3 Ablation Studies

We conduct the ablation study in Table 4. Removing any major component degrades both task utility and long-horizon control performance, showing that the gains of VEROIC arise from the joint effect of risk estimation, enhanced inference, and adaptive control. In particular, removing the risk signals leads to consistent drops in task performance together with higher Occ, CVaR, and RecD across all tasks, indicating that calibrated observation signals are essential for identifying cases that warrant enhanced inference. Removing enhanced inference causes the largest performance degradation and the worst long-horizon risk metrics, confirming that the default low-cost path alone is insufficient for maintaining reliable performance on difficult instances. Replacing the adaptive policy with a non-adaptive alternative also degrades both utility and robustness, suggesting that risk estimation must be coupled with belief-based, budget-aware decision making in order to fully realize its benefits.

5.4 Parameter Sensitivity Analysis

Figure 2 analyzes the effect of the enhanced-inference budget α. As the budget increases, both PERIODIC and VEROIC exhibit monotonic performance improvements across all tasks, reflecting the expected benefit of making enhanced inference available to a larger fraction of requests. However, VEROIC consistently outperforms PERIODIC at every budget level, indicating that its advantage comes not merely from increased compute usage, but from better identifying which high-risk outputs warrant escalation, which is particularly meaningful in the low-budget regime, where selective allocation matters most. Overall, results show that VEROIC achieves a more favorable performance trade-off and validates the effectiveness of belief-based adaptive inference control.

6 Conclusion

In this paper, we propose VEROIC, a belief-based framework for adaptive inference control in black-box LLM services. By treating response reliability as a latent variable, VEROIC enables the system to infer whether the default low-cost output is trustworthy from observable signals and to decide when enhanced inference is warranted under a computation budget. We introduce a lightweight observation channel with a structured observation model and Bayesian belief updates for estimating latent response reliability under partial observability. We further formulate budget-aware inference allocation as a long-horizon control problem, in which the policy selectively invokes stronger inference paths. Extensive experiments across diverse benchmarks, multiple backbones, and different inference settings show that VEROIC consistently achieves better performance over competitive baselines.

Limitations

This work exhibits several limitations worth noting. First, our experiments use benchmark-based sequential evaluation rather than real deployed black-box LLM services, and therefore may not fully capture service-side non-stationarity or user-facing interaction dynamics. Second, the performance of VEROIC depends on the quality of the constructed observation signals; weak or miscalibrated hard checks and proxy scores may lead to inaccurate belief estimates and suboptimal escalation decisions. Third, our current instantiation uses a binary latent reliability state and lightweight scoring modules, which may not capture finer-grained reliability structure in more complex settings. Finally, the framework introduces additional modeling and training overhead, so its practical value depends on the deployment context and the underlying quality-cost trade-off.

Acknowledgments

This work was supported by the UGC General Research Fund no. 17209822 and the Innovation and Technology Commission Fund no. ITS/383/23FP from Hong Kong.

GenAI Usage Disclosure

This work is entirely original and was conducted by the authors. Generative AI tools were not used to produce any content of the work; they were used solely to assist with language refinement and improve clarity and quality of the text.

References

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Agathe Balayn, Mireia Yurrita, Fanny Rancourt, Fabio Casati, and Ujwal Gadiraju. 2025. Unpacking trust dynamics in the llm supply chain: An empirical exploration to foster trustworthy llm production & use. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI '25, New York, NY, USA. Association for Computing Machinery.

Lingjiao Chen, Matei Zaharia, and James Zou. 2024. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*. Featured Certification.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Valerie Chen, Alan Zhu, Sebastian Zhao, Hussein Mozannar, David Sontag, and Ameet Talwalkar. 2025. Need help? designing proactive ai assistants for programming. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI '25, New York, NY, USA. Association for Computing Machinery.

Zihao Cheng, Li Zhou, Feng Jiang, Benyou Wang, and Haizhou Li. 2025. Beyond binary: Towards fine-grained llm-generated text detection via role recognition and involvement measurement. In *Proceedings of the ACM on Web Conference 2025*, WWW '25, page 2677–2688, New York, NY, USA. Association for Computing Machinery.

Yoonseo Choi, Eun Jeong Kang, Seulgi Choi, Min Kyung Lee, and Juho Kim. 2025. Proxona: Supporting creators' sensemaking and ideation with llm-powered audience personas. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI '25, New York, NY, USA. Association for Computing Machinery.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Carson Denison, Monte MacDiarmid, Fazl Barez, David Duvenaud, Shauna Kravec, Samuel Marks, Nicholas Schiefer, Ryan Soklaski, Alex Tamkin, Jared Kaplan, and 1 others. 2024. Sycophancy to subterfuge: Investigating reward-tampering in large language models. *arXiv preprint arXiv:2406.10162*.

Tuan Dinh, Jinman Zhao, Samson Tan, Renato Negrinho, Leonard Lausen, Sheng Zha, and George Karypis. 2023. Large language models of code fail at completing code with potential bugs. *Advances in Neural Information Processing Systems*, 36:41386–41412.

David Dohan, Winnie Xu, Aitor Lewkowycz, Jacob Austin, David Bieber, Raphael Gontijo Lopes, Yuhuai Wu, Henryk Michalewski, Rif A. Saurous, Jascha Sohl-dickstein, Kevin Murphy, and Charles Sutton. 2022. Language model cascades.

Xuefeng Du, Chaowei Xiao, and Sharon Li. 2024. Haloscope: Harnessing unlabeled llm generations for hallucination detection. *Advances in Neural Information Processing Systems*, 37:102948–102972.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jiaming Ji, Wenqi Chen, Kaile Wang, Donghai Hong, Sitong Fang, Boyuan Chen, Jiayi Zhou, Juntao Dai, Sirui Han, Yike Guo, and 1 others. 2025. Mitigating deceptive alignment via self-monitoring. *arXiv preprint arXiv:2505.18807*.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, and 17 others. 2022. Language models (mostly) know what they know. *Preprint*, arXiv:2207.05221.

Nils Kemmerzell, Annika Schreiner, Haroon Khalid, Michael Schalk, and Letizia Bordoli. 2025. Towards a better understanding of evaluating trustworthiness in ai systems. *ACM Computing Surveys*, 57(9):1–38.

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*.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D Manning, Christopher Re, Diana Acosta-Navas, Drew A. Hudson, and 31 others. 2023. Holistic evaluation of language models. *Transactions on Machine Learning Research*.

Gabrielle Kaili-May Liu, Gal Yona, Avi Caciularu, Idan Szpektor, Tim G. J. Rudner, and Arman Cohan. 2025a. MetaFaith: Faithful natural language uncertainty expression in LLMs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 29612–29656, Suzhou, China. Association for Computational Linguistics.

Shudong Liu, Zhaocong Li, Xuebo Liu, Runzhe Zhan, Derek F Wong, Lidia S Chao, and Min Zhang. 2024. Can llms learn uncertainty on their own? expressing uncertainty effectively in a self-training manner. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21635–21645, Miami, Florida, USA. Association for Computational Linguistics.

Xiaoou Liu, Tiejin Chen, Longchao Da, Chacha Chen, Zhen Lin, and Hua Wei. 2025b. Uncertainty quantification and confidence calibration in large language models: A survey. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 6107–6117, New York, NY, USA. Association for Computing Machinery.

Jinghui Lu, Haiyang Yu, Siliang Xu, Shiwei Ran, Guozhi Tang, Siqi Wang, Bin Shan, Teng Fu, Hao Feng, Jingqun Tang, and 1 others. 2025. Prolonged reasoning is not all you need: Certainty-based adaptive routing for efficient llm/mllm reasoning. *arXiv preprint arXiv:2505.15154*.

Bohan Lyu, Yadi Cao, Duncan Watson-Parris, Leon Bergen, Taylor Berg-Kirkpatrick, and Rose Yu. 2025. Adapting while learning: Grounding LLMs for scientific problems with tool usage adaptation. In *Forty-second International Conference on Machine Learning*.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822, Toronto, Canada. Association for Computational Linguistics.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. 2025. Routellm: Learning to route llms with preference data. *Preprint*, arXiv:2406.18665.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Vishakh Padmakumar and He He. 2024. Does writing with language models reduce content diversity? In *The Twelfth International Conference on Learning Representations*.

Leo Richter, Xuanli He, Pasquale Minervini, and Matt Kusner. 2025. An auditing test to detect behavioral shift in language models. In *The Thirteenth International Conference on Learning Representations*.

Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. 2025. Can LLMs generate novel research ideas? a large-scale human study with 100+ NLP researchers. In *The Thirteenth International Conference on Learning Representations*.

Adi Simhi, Itay Itzhak, Fazl Barez, Gabriel Stanovsky, and Yonatan Belinkov. 2025. Trust me, i'm wrong: Llms hallucinate with certainty despite knowing the

answer. *Findings of the Association for Computational Linguistics: EMNLP*, 2025:14665–14688.

Gaurang Sriramanan, Siddhant Bharti, Vinu Sankar Sadasivan, Shoumik Saha, Priyatham Kattakinda, and Soheil Feizi. 2024. Llm-check: Investigating detection of hallucinations in large language models. *Advances in Neural Information Processing Systems*, 37:34188–34216.

Guoheng Sun, Ziyao Wang, Xuandong Zhao, Bowei Tian, Zheyu Shen, Yexiao He, Jinming Xing, and Ang Li. 2025. Invisible tokens, visible bills: The urgent need to audit hidden operations in opaque llm services. *arXiv preprint arXiv:2505.18471*.

Linwei Tao, Yi-Fan Yeh, Minjing Dong, Tao Huang, Jialin Yu, Philip Torr, and Chang Xu. 2025. Revisiting uncertainty estimation and calibration of large language models. In *39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Scaling Environments for Agents (SEA)*.

Ren-Jian Wang, Ke Xue, Zeyu Qin, Ziniu Li, Sheng Tang, Hao-Tian Li, Shengcai Liu, and Chao Qian. 2025. Quality-diversity red-teaming: Automated generation of high-quality and diverse attackers for large language models. *arXiv preprint arXiv:2506.07121*.

Zhiqiu Xia, Jinxuan Xu, Yuqian Zhang, and Hang Liu. 2025. A survey of uncertainty estimation methods on large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 21381–21396, Vienna, Austria. Association for Computational Linguistics.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, YIFEI LI, Jie Fu, Junxian He, and Bryan Hooi. 2024. Can LLMs express their uncertainty? an empirical evaluation of confidence elicitation in LLMs. In *The Twelfth International Conference on Learning Representations*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380.

Wenhao Yuan, Chenchen Lin, Jian Chen, Jinfeng Xu, Xuehe Wang, and Edith Cheuk Han Ngai. 2026. Verify before you commit: Towards faithful reasoning in llm agents via self-auditing. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, San Diego, California, United States. Association for Computational Linguistics.

Qiang Zhang, Keyan Ding, Tianwen Lv, Xinda Wang, Qingyu Yin, Yiwen Zhang, Jing Yu, Yuhao Wang, Xiaotong Li, Zhuoyi Xiang, Xiang Zhuang, Zeyuan Wang, Ming Qin, Mengyao Zhang, Jinlu Zhang, Jiyu Cui, Renjun Xu, Hongyang Chen, Xiaohui Fan, and 2 others. 2025. Scientific large language models: A survey on biological & chemical domains. *ACM Comput. Surv.*, 57(6).

A Additional Experimental Results

A.1 Dataset Details

GSM8K (Grade School Math 8K) (Cobbe et al., 2021) is a benchmark designed to evaluate a model's ability to perform multi-step numerical reasoning expressed in natural language. It consists of approximately 8,500 grade-school–level math word problems, each requiring a sequence of arithmetic operations to reach the final answer. The problems emphasize logical decomposition, intermediate reasoning steps, and precise numerical computation rather than advanced mathematical knowledge. GSM8K has become a standard benchmark for assessing chain-of-thought reasoning and mathematical problem-solving capabilities of large language models.

The **MATH** dataset (Hendrycks et al., 2021) is a large-scale benchmark targeting advanced mathematical reasoning across a wide range of difficulty levels. It contains problems drawn from high school mathematics competitions, covering topics such as algebra, geometry, number theory, probability, and calculus. Each problem is paired with a detailed solution, enabling evaluation not only of final answer corRecDness but also of structured reasoning processes. Compared to GSM8K, MATH presents significantly higher complexity and abstraction, making it a challenging testbed for assessing symbolic reasoning and deep mathematical understanding.

HumanEval (Chen et al., 2021) is a benchmark specifically designed to evaluate code generation and program synthesis capabilities of language models. It consists of hand-written Python function signatures accompanied by natural language docstrings that describe the intended functionality. Models are required to generate executable code that satisfies hidden unit tests. HumanEval focuses on functional corRecDness rather than surface-level similarity, and is widely used to assess a model's ability to reason about algorithms, control flow, and edge cases in programming tasks.

The **MBPP** (Mostly Basic Python Problems) dataset (Austin et al., 2021) evaluates fundamental programming skills through short, self-contained Python coding tasks. Each problem includes a natural language description, a function signature, and a small set of test cases. Compared to HumanEval, MBPP targets more elementary programming concepts such as loops, conditionals, and basic data structures, while still requiring precise implementation. MBPP is commonly used to measure a model's robustness in basic code synthesis and general-purpose programming reasoning.

HotpotQA (Yang et al., 2018) is a multi-hop question answering benchmark that requires reasoning across multiple documents. Unlike single-hop QA datasets, questions in HotpotQA cannot be answered by a single paragraph alone; instead, models must retrieve and integrate information from two or more supporting facts. The dataset is constructed from Wikipedia and includes both answer supervision and supporting fact annotations, enabling evaluation of not only answer accuracy but also evidence selection and reasoning chains.

2WikiMHQA (Ho et al., 2020) (Two-Wikipedia Multi-Hop Question Answering) is a large-scale benchmark for explicit multi-hop reasoning over Wikipedia articles. Each question is designed to require reasoning over exactly two linked documents, forming a clear two-hop inference path. Compared to HotpotQA, 2WikiMHQA offers more controlled reasoning structures, making it particularly suitable for analyzing a model's ability to perform structured document linking and intermediate inference in multi-hop QA settings.

PopQA (Mallen et al., 2023) is an open-domain question answering dataset focused on popular factual knowledge, such as widely known entities, events, and concepts. Questions are designed to be answerable using commonly available world knowledge rather than specialized or obscure facts. PopQA emphasizes knowledge recall and factual accuracy, and is often used to evaluate the breadth and reliability of a model's internal knowledge, especially under zero-shot or retrieval-free settings.

A.2 Baseline Details

We compare VEROIC against several representative baselines for adaptive inference allocation.

FIXED triggers enhanced inference at a constant rate throughout the interaction horizon, independent of the current request, output signals, or interaction history. This baseline represents static budget allocation policies that spend a fixed fraction of the enhanced-inference budget without adaptation. While simple and predictable, FIXED cannot concentrate additional computation on high-risk requests and therefore yields limited efficiency under partial observability.

RANDOM triggers enhanced inference by independently sampling a Bernoulli random variable at each interaction step with a fixed probability. Com-

pared with FIXED, random allocation introduces stochasticity and avoids a fully deterministic escalation schedule. However, it remains history-agnostic and does not make use of observable quality signals or inferred risk, leading to inefficient budget use over long horizons.

PERIODIC deterministically invokes enhanced inference every $1/\alpha$ steps, ensuring uniform budget usage over time. This baseline reflects scheduled resource allocation policies that provide regular coverage but do not respond to request difficulty or response quality. As a result, PERIODIC may waste additional computation on easy cases while missing difficult requests that arise between escalation intervals.

OUTPUT-TRIGGER (Lu et al., 2025; Chen et al., 2024) is a rule-based routing baseline that triggers enhanced inference based on predefined output-side signals, such as failed checks, low confidence indicators, or other observable warning patterns. This class of methods represents uncertainty-aware or post-hoc routing strategies that condition escalation on surface-level response statistics. While more adaptive than rate-based baselines, OUTPUT-TRIGGER relies on hand-designed rules and does not maintain a belief over latent response reliability.

CONFIDENCE-THRESHOLD (Xiong et al., 2024; Tao et al., 2025) is a single-score uncertainty routing baseline that triggers enhanced inference when the current confidence or risk score crosses a budget-calibrated threshold. Compared with OUTPUT-TRIGGER, it replaces heuristic rule combinations with a single continuous uncertainty signal, but still makes decisions from the current step only and does not incorporate belief propagation or history-dependent inference.

RISK PREDICTOR (Liu et al., 2024, 2025b) is a supervised routing baseline that trains a predictor on the same observation features used by VEROIC to estimate the failure risk of the default response. Enhanced inference is triggered when the predicted risk exceeds a budget-calibrated threshold. This baseline provides a stronger learned alternative to heuristic routing, but it remains a one-step decision rule and does not explicitly model latent reliability dynamics through belief updates.

A.3 Model Instantiation.

We instantiate the structured observation encoder with lightweight task-agnostic scoring modules. Specifically, $\ell_y(s_t; x_t, y_t^{(0)})$ is implemented as a linear score over response-side features extracted from the default output, including response length, format validity indicators, self-consistency statistics, and verifier-related features. The hard-evidence term $\ell_h(x_t, y_t^{(0)}, z_{h,t}; \psi_t)$ is implemented as a linear score over the calibrated hard-predicate vector $z_{h,t}$. Each soft-evidence term $\ell_g^{(i)}(s_t; x_t, y_t^{(0)}, z_{g,t}^{(i)}; \psi_t)$ is implemented as a state-dependent affine score over the calibrated soft signal $z_{g,t}^{(i)}$. All encoder parameters are trained jointly with the control policy. For soft-signal calibration, we use a monotone affine-sigmoid map $\Gamma_i(\tilde{g}_t^{(i)}; \vartheta_i) = \sigma(a_i \tilde{g}_t^{(i)} + b_i)$, where $\vartheta_i = (a_i, b_i)$ denotes the calibration parameters for the i-th soft proxy. In practice, the calibration parameters are fitted on a held-out validation split using task-level correctness labels and then kept fixed during policy learning and evaluation. Hard signals are calibrated with Beta-Bernoulli smoothing using the sliding-window counts.

A.4 Policy Optimization and Utility.

We optimize the belief-conditioned policy using an on-policy actor-critic algorithm with entropy regularization under the Lagrangian objective. The actor takes (b_t, x_t) as input and outputs the probability of escalation, while the critic estimates the state value for variance reduction. Both actor and critic are implemented as lightweight two-layer MLPs over the concatenated belief-state and request-feature representation. We use Adam for optimization, and select model checkpoints based on validation performance under the target budget. Unless otherwise specified, the trained policy is frozen at evaluation time. The utility surrogate $V(y_t^{ret})$ is instantiated from task-specific correctness signals. For GSM8K and MATH, we use exact-match correctness. For HumanEval and MBPP, we use unit-test pass/fail outcomes. For HotpotQA, 2WikiMHQA, and PopQA, we use task-level QA scores derived from exact match and token-level F1, with the scalar reward computed from the corresponding evaluation score. These observable task signals are used only for training and evaluation, while test-time control decisions rely solely on the default response and the constructed observation vector.

A.5 Metric Details

We evaluate VEROIC from three complementary perspectives: task utility, risk estimation and cali-

bration, and long-horizon control performance.

Task Utility. For math reasoning and question answering tasks, we report Exact Match (EM) and F1 scores. For code generation tasks, we report Pass@1. These metrics are computed on the final returned response y_t^{ret} and measure task performance under different inference-control policies.

Risk Estimation and Probability Calibration. We evaluate how well the framework identifies low-quality or high-risk default responses that may benefit from enhanced inference. At each interaction step t, the predicted risk score is given by the belief-induced unreliability probability, $\hat{p}_t := b_t(s = 0)$, where b_t is the posterior belief over latent reliability states and $s = 0$ denotes that the default response is unreliable. For offline evaluation, we define a binary error label $e_t := \mathbb{1}\{y_t^{(0)} \text{ is low-quality}\}$, where the exact criterion depends on the task-specific evaluation signal of the default response. We report AUROC and AUPRC to measure the discriminative ability of $\hat{p}_t$ in identifying high-risk cases. To assess probabilistic calibration, we additionally report the Brier score,

$$\text{Brier} := \frac{1}{H}\sum\nolimits_{t=1}^{H}(\hat{p}_t - e_t)^2, \tag{10}$$

the negative log-likelihood (NLL),

$$\text{NLL} := -\frac{1}{H}\sum_{t=1}^{H}\left[e_t \log \hat{p}_t + (1-e_t)\log(1-\hat{p}_t)\right], \tag{11}$$

and the expected calibration error (ECE), computed by partitioning $\hat{p}_t$ into confidence bins and measuring the discrepancy between empirical frequencies and predicted probabilities.

Long-Horizon Control Performance. To evaluate robustness under sequential budgeted interaction, we report three long-horizon control metrics. The low-quality occurrence rate (Occ) is the fraction of interaction steps whose final returned response is low-quality:

$$\text{Occ} := \frac{1}{H}\sum_{t=1}^{H}\mathbb{1}\{y_t^{\text{ret}} \text{ is low-quality}\}. \tag{12}$$

To capture tail risk, we compute the conditional value-at-risk (CVaR) of episode-level low-quality outcomes at confidence level β across multiple sampled interaction episodes. Finally, the recovery delay (RecD) measures the average number of subsequent steps required for the system to recover after a severe low-quality event, thereby quantifying recovery latency under prolonged interaction.

A.6 Used Prompts

Verification Prompt for Enhanced Inference Control

You are an evaluator.

Your task is to examine the given task input and the model output, and decide whether the output should be considered **unreliable or unacceptable** for this task according to the provided evaluation criteria. This judgment is used only as a verification signal for inference control. Do **not** provide a corrected answer, alternative solution, or extra reasoning beyond what is needed to justify your judgment.

Task Input:
{input_task}

Model Output:
{model_output_y}

Optional Evidence (may be empty):
{evidence_bundle}

Evaluation Criteria (apply all that are relevant):

- **Correctness:** The output answers the question or solves the task correctly.
- **Constraint Satisfaction:** The output follows required format, constraints, and instructions in the input.
- **Consistency:** The output is internally consistent and does not contain contradictory statements.
- **Evidence Consistency (if evidence is provided):** Claims that depend on the evidence do not contradict the provided evidence.
- **Code Validity (if code task):** The solution is executable or compilable and passes the provided checks, if any are given in the evidence.

Binary Decision Rule:

- Output `Verdict: 1` if the output is **unreliable/unacceptable** (fails any key criterion above in a way that would meaningfully harm task success).

- Output `Verdict: 0` if the output is **acceptable** (meets the key criteria; minor stylistic issues are acceptable).

Output Format (strict JSON):

```
{"Verdict": 0 or 1,
 "Reason": "<one short checkable
 reason>",
 "KeyEvidence": "<what you checked:
 e.g., 'final answer incorrect',
 'fails constraint X', 'contradicts
 passage #2', 'unit test failed'>"}
```

Rules:

- Keep the `Reason` and `KeyEvidence` short and checkable.
- Do not output a corrected answer or additional solution steps.
- If the task cannot be evaluated from the given information, default to `Verdict: 0` and explain what is missing in `KeyEvidence`.

Soft Scoring Prompt for Lightweight Verification

You are an evaluator.

Your task is to assess the quality and reliability of the given model output for the task and return a **continuous score in** $[0, 1]$ that reflects how well the output satisfies the evaluation criteria. A higher score indicates a more reliable output. Do **not** provide a correct answer or additional solution.

Task Input:
{input_task}

Model Output:
{model_output_y}

Optional Evidence (may be empty):
{evidence_bundle}

Evaluation Criteria (consider all that are relevant):

- **Correctness:** The output answers the task correctly or follows a valid solution strategy.
- **Constraint Satisfaction:** The output follows the required formats, constraints, and instructions.
- **Consistency:** The output is internally coherent and free of contradictions.
- **Evidence Consistency (if evidence is provided):** Claims are consistent with the given evidence.
- **Code Validity (if code task):** The solution is syntactically correct and aligns with expected behavior.

Scoring Instruction:

- Return a single scalar `Score` in $[0, 1]$.
- `Score` should reflect overall output reliability:
  - 0.9–1.0: clearly correct and reliable.
  - 0.6–0.8: mostly correct with minor issues.
  - 0.3–0.5: partially correct or uncertain.
  - 0.0–0.2: largely incorrect or unreliable.
- Use intermediate values when appropriate.

Output Format (strict JSON):

```
{"Score": <float in [0,1]>,
 "Rationale": "<one short checkable
 reason>"}
```

Rules:

- Keep the `Rationale` brief and based on observable properties.
- Do not include a correct solution or extra reasoning.
- If the task cannot be reliably evaluated with the given information, return a conservative score and explain why.

IMIS High-Effort Service Prompt

System Instruction

You are a careful assistant operating under a high-effort inference setting.

When responding to user requests:

- Read the task carefully and follow all instructions.
- Deliberate carefully before producing the final answer.
- Ensure internal consistency between the reasoning process and the final output.
- Verify critical steps, calculations, and logical transitions when needed.
- Prioritize correctness and completeness over brevity.

Produce your best possible answer to the user's request.

IMIS Low-Effort Service Prompt

System Instruction

You are an efficient assistant operating under a low-effort inference setting.

When responding to user requests:

- Provide a direct and fluent response to the query.
- Aim for a reasonable and plausible answer.
- Prioritize efficiency and avoid unnecessary elaboration.
- Do not perform extensive verification unless it is clearly required.
- Keep the response concise and readable.

Respond directly to the user's request.