A Hierarchical Imprecise Probability Approach to Reliability Assessment of Large Language Models

Robab Aghazadeh-Chakherlou[a], Qing Guo[b,c], Siddartha Khastgir[a], Peter Popov[d], Xiaoge Zhang[e], Xingyu Zhao[a,*]

[a] *WMG, University of Warwick, Coventry, United Kingdom*
[b] *Center for Frontier AI Research, A*STAR, Singapore, Singapore*
[c] *School of Computing, National University of Singapore, Singapore, Singapore*
[d] *Center for Software Reliability, City St George's, University of London, London, United Kingdom*
[e] *Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Kowloon, Hong Kong*

Abstract

Large Language Models (LLMs) are increasingly deployed across diverse domains, raising the need for rigorous reliability assessment methods. Existing benchmark-based evaluations primarily offer descriptive statistics of model accuracy over datasets, providing limited insight into the probabilistic behavior of LLMs under real operational conditions. This paper introduces HIP-LLM, a **H**ierarchical **I**mprecise **P**robability framework for modeling and inferring **LLM** reliability. Building upon the foundations of software reliability engineering, HIP-LLM defines LLM reliability as the probability of failure-free operation over a specified number of future tasks under a given Operational Profile (OP). HIP-LLM represents dependencies across (sub-)domains hierarchically, enabling multi-level inference from subdomain to system-level reliability. HIP-LLM embeds imprecise priors to capture epistemic uncertainty and incorporates OPs to reflect usage contexts. It derives posterior reliability envelopes that quantify uncertainty across priors and data. Experiments on multiple benchmark datasets demonstrate that HIP-LLM offers a more nuanced and standardized reliability characterization than existing benchmark and state-of-the-art approaches. A publicly accessible repository of HIP-LLM is provided.

Keywords:
Large Language Model; Software Reliability; Hierarchical Bayesian Inference; Operational Profile; Epistemic Uncertainty; Imprecise Probability

1. Introduction

Large language models (LLMs) are increasingly applied across a wide range of tasks, from general-purpose reasoning to highly specialized applications. For instance, recent studies [1, 2, 3, 4, 5, 6] employ LLMs as tools to support reliability analysis and safety assurance for safety-critical systems. This expanding scope of application underscores the urgent need for comprehensive and rigorous evaluation methods to assess the *reliability* of LLMs themselves,

*Corresponding author: Xingyu Zhao, xingyu.zhao@warwick.ac.uk

given the high-stakes nature of those applications. While numerous studies have proposed domain-specific performance evaluations in areas such as coding [7, 8, 9], medicine [10], and system safety [11, 12, 13], others have focused on broader properties of LLMs, including safety (avoiding harmful content) [14, 15, 16], robustness (resisting adversarial attacks such as jailbreaks) [17, 18], fairness [19], and privacy [20, 21]. However, there remains a notable lack of research dedicated to reliability, which motivates this work.

While software *reliability* is highly interrelated with the aforementioned properties (a formal definition and its distinction to other properties will be discussed in later sections), it is a unique property that emphasizes the failure probabilities, specified operational time and conditions, and statistically valid uncertainty quantification. As a "user-centric" property [22, 23, 24], the delivered reliability depends on the behavior of end-users and how the software is expected to be used in practice. This notion is made measurable through the *Operational Profile* (OP) [25, 26], which specifies a probability distribution over the types of demands that users actually place on the software. The recent report from OpenAI [27] provide the most comprehensive data on how user-groups use ChatGPT, which aligns with the idea of OPs.

Despite software reliability assessment has been studied for decades [28, 29], state-of-the-art LLM evaluation largely relies on benchmarks [30]. This creates several research gaps:

Gap-1 (static benchmarks vs dynamic operational use): While we acknowledge the usefulness of benchmarks, they are primarily designed for purposes such as comparing and ranking LLMs on given tasks under fixed datasets, neglecting the variability and dynamic nature of users' behaviors in real-world applications. Thus, benchmarks cannot serve the purpose of assessing reliability, especially when the OP of the users is different from the data distribution (implicitly) represented by the benchmark dataset.

Gap-2 (independent benchmarks vs hierarchical dependencies): While LLMs are generally designed for broad use (covering various task types such as legal reasoning and coding), these tasks naturally exist at different levels of abstraction. It is often useful to organize them hierarchically, dividing the input space into domains and subdomains [31]. High-level domains[1] (e.g., law vs. coding) can be treated as largely independent, whereas subdomains within a domain (e.g., coding in Python vs. coding in C++) are more likely to exhibit dependencies. In such cases, performance in one subdomain may influence users' confidence in another, whereas cross-domain effects are minimal. Reliability assessment should therefore reflect these dependency relationships and support evaluation at multiple abstraction levels, depending on the assessor's role. For instance, a LLM vendor may be interested in system-wide reliability across all domains, a programmer may focus on reliability in coding tasks, and a Python developer may be specifically concerned with the reliability of Python coding. Existing benchmarks, however, typically evaluate LLM capabilities either in isolation or by simply averaging performance across tasks, that obscure these dependency relationships [31].

[1]While the precise definition of what constitutes a domain or subdomain is an application-oriented question that may vary case by case reflecting the assessor's domain knowledge, in this reliability modeling work we do not prescribe specific taxonomies. Instead, we introduce a general hierarchical modeling structure that can accommodate different dependency assumptions (cf. Section 5 for more discussion).

Gap-3 (descriptive statistics vs statistical inference): While descriptive statistics summarize the observed dataset by reporting point-estimate benchmark scores, they do not formally model *how the data were generated*. Even when variance is reported as a way of "quantifying uncertainty", it quantifies variability *within the sample itself*. In contrast, statistical inference treats the observed results as samples from *an underlying stochastic process or population* and uses probabilistic modeling to estimate the parameters of that process [32]. This inferential perspective enables generalization beyond the fixed benchmark *dataset* and supports principled uncertainty quantification about reliability which concerns a underlying *population* of future inputs [31].

Indeed, acknowledging these research gaps, recent studies [31, 32] attempt to bridge them by applying both "frequentist" and Bayesian statistical inference methods. However, despite representing important first steps, they do not fully resolve the aforementioned gaps (e.g., not explicitly incorporate the OP) and introduce new gaps:

Gap-4 (failure probability vs probability of future failure-free runs) While both [31, 32] examined the metric of failure probability[2], they did not extend it to the more practical and widely adopted definition of reliability that predicts the *probability of failure-free runs* over a specified future operational time [33, 34, 35].

Gap-5 (non-informative vs informative priors): The work [32] is a frequentist approach that cannot embed prior knowledge explicitly like Bayesian methods. The work [31] employs non-informative priors in its Bayesian models, without facilitating the embedding of any informative prior knowledge that end-users may have.

To bridge the five research gaps collectively, we propose HIP-LLM, a hierarchical imprecise probability approach to assess LLM reliability. HIP-LLM devises a more rigorous and versatile assessment with the following key features. First, similar to [31], it structures the model into independent high-level domains, with each domain further divided into dependent subdomains. In this way, reliability can be evaluated at multiple levels of detail while preserving statistical dependencies, so that information from one subdomain contributes to the inference about other subdomains within the same domain. We then incorporate OPs as operational weights at each level of the hierarchy (subdomain $\rightarrow$ domain $\rightarrow$ general-purpose LLM) to reflect the dynamic operational use of subdomains and domains in practice. We assume tasks constitute independent and identically distributed (i.i.d.) trials drawn from the specified OP. This assumption is appropriate for reset or single-task scenarios, in which each task is executed in a fresh session without shared context or memory—settings commonly encountered in offline evaluation like benchmarking[3] Accordingly, HIP-LLM is intended for LLM usage scenarios in which each task is executed in a new chat session without retaining memory from previous sessions, thereby aligning with the assumption of a "reset" LLM.

In contrast to [31], which relies on non-informative priors, HIP-LLM enables the incorporation of informative prior knowledge. Since selecting a single prior in Bayesian inference is

[2]We later show that this is a special case of our reliability model.

[3]In contrast, *long-context* or *agentic workflows*, where task outcomes are sequentially dependent through memory or tool use, violate the i.i.d. assumption and require fundamentally different modeling approaches to our HIP-LLM. We provide more discussions later in Remark 1 and Section 5.

often controversial due to its encoding of epistemic uncertainty, we adopt *Imprecise Probability* [36, 37] that represents uncertainty about the prior itself without committing to a single distribution. Consequently, reliability metrics (defined in terms of both failure probability and the probability of failure-free runs in future operations) are expressed through *posterior distribution envelopes* at different levels of the hierarchy, reflecting both uncertainties from the data and the prior knowledge.

In summary, the contributions of this work are as follows:

- We formally define the reliability assessment problem for LLMs in accordance with established software reliability standards, while delineating its distinction from related properties.
- We propose HIP-LLM, a hierarchical imprecise probability model that explicitly addresses key research gaps in the current state-of-the-art, by modeling OPs, hierarchical dependencies, statistically principled uncertainty quantification on reliability (both failure probability and the probability future failure-free runs over a specified operational time), and imprecise prior knowledge.
- We release a public repository containing all experimental data, source code, and models at `https://github.com/aghazadehchakherlou-web/llm-imprecise-bayes`.

The remainder of the paper is organized as follows. Section 2 introduces preliminaries and reviews related works on LLM assessment. Section 3 provides the modeling details of HIP-LLM. Section 4 evaluates and illustrates HIP-LLM via experiments on datasets. Section 5 discusses the limitations and assumptions, and finally Section 6 concludes the paper.

2. Preliminaries and Related Works

2.1. Software Reliability

According to American National Standards Institute (ANSI), software reliability is defined as [38]:

Definition 1 (Software reliability)**.** *The probability of failure-free software operation for a specified period of time in a specified environment.*

This probabilistic definition is also largely adopted by software reliability engineering literature [28], which makes reliability amenable to statistical modeling and permits risk-aware aggregation across heterogeneous operational uses. The specific probabilistic metric used to model reliability is domain-dependent and determined by the operational nature of the software. For example, continuous-time systems are continuously operated in the active control of a process, whereas on-demand systems are only invoked upon receipt of discrete demands. In the latter case, such as nuclear power protection systems, the probability of failure on demand (*pfd*) has been adopted in standards [23, 39, 40] and extensively studied [41, 33, 42].

Although related, reliability is distinct from safety, security, and accuracy. Reliability focuses on the *probability* of failures (relative to a specification), a stochastic and usage-weighted concept by OPs [42]. Safety concerns *critical failures* with *catastrophic consequences*, i.e.,

whether a failure can lead to unacceptable *harm*. For instance, a system can be reliable but unsafe (if frequent but non-harmful failures are tolerated while rare failures produce catastrophic harm), or safe but unreliable (if it fails frequently but never produce catastrophic harm by fail-safe design)[4]. Security is concerned with resisting *malicious threats*: it normally assumes an explicit threat model and seeks to prevent or mitigate intentional compromises). While security fundamentally requires the analysis of a malicious threat actor, reliability and safety can be studied within benign operational environments where failure originates from sources such as design flaws and implementation bugs. Within the AI/ML community, accuracy is typically a narrower performance metric, defined as a descriptive statistic (proportion of correct predictions) over a fixed and given dataset. However, such a dataset does not necessarily represent the unknown ground-truth population of inputs that the model will face upon deployment, and accuracy does not quantify the uncertainties of failure-free operations over future time like reliability.

A central insight from software reliability engineering is that reliability is *user-context dependent*: the same software can exhibit very different delivered reliability under different patterns of use. The OP formalizes this dependency, which is defined as [23, 25, 28]:

Definition 2 (Operational Profile (OP))**.** *An OP is a probability distribution of inputs to a software system, representing the relative frequencies with which different input sequences expected occur during actual operation.*

There are some caveats regarding the term "inputs" in the definition of OP, particularly when considering factors such as software memory and dependencies[5]. We refer readers to [23] for a detailed discussion. For LLMs, OPs serve an analogous role [27]: they specify the distribution of task types (e.g., factual question answering, coding, summarization) and their relative importance in deployment. Incorporating OPs into LLM reliability analysis ensures that evaluation metrics capture not just aggregate benchmark scores, but also how well a model performs across the realistic mix of tasks it is expected to face.

2.2. Benchmark-Based Evaluation of LLMs via Descriptive Statistics

Most evaluation methods for LLMs report only point estimates such as accuracy scores. Question-answering benchmarks such as MMLU [43] and RACE [44] measure how often a model picks the right answer in multiple-choice questions. Code benchmarks like DS-1000 [8], InterCode [7], and HumanEval Pro/MBPP Pro [9] test whether generated programs run correctly. Assistant and autonomy benchmarks such as GAIA [45] and H-CAST [46] look at overall task completion—whether the system can finish a complex job successfully. Model cards and reports for Claude 3 [47], GPT-4o [48], and code models [49] publish these benchmark results as single numbers per model. Broader frameworks like HELM [50] add

[4]We note that for systems in which all failures are catastrophic (e.g., safety-critical systems), safety and reliability assessments do not require different statistical reasoning, such as when measuring the probability that a system will operate safely (i.e., without critical failures) over a given mission.

[5]In practice, because software systems may possess memory, their success/failure often depends on entire sequences of inputs rather than isolated ones. Accordingly, "inputs" may denote a complete stimulus, which may encompass the full sequence of interactions within a task, or even the cumulative sequence of inputs since the system was last reinitialized.

other measures such as calibration, robustness, fairness, and efficiency, but still mainly give point values. In real-world use, Clio [21] collects statistics on what kinds of tasks people actually do with AI with a focus on privacy. Method studies like multilingual chain-of-thought [51] and ReAct [52] suggest ways to improve reasoning and interaction, judged by benchmark accuracy.

Recent studies widely acknowledge that benchmark-based evaluations for LLMs are valuable but inherently limited due to the new characteristics and challenges introduced by LLMs. Benchmarks facilitate progress tracking and model comparison, yet they capture only a partial view of LLM capabilities. Chang et al. [30] note that traditional benchmarks (reusing fixed task sets for ranking models) have driven early advances but are now approaching their limits due to data repetition, task saturation[6], and limited assessment of reasoning or interactivity. Liu et al. [53] emphasize that standard benchmarks often overlook moral, safety, and social dimensions of model behavior. In domain-specific contexts, Croxford et al. [10] show that medical benchmarks miss crucial aspects such as factual accuracy and clinical reasoning, highlighting the need for domain prior knowledge. Saleh et al. [54] observe that efficiency and performance benchmarks remain fragmented and inconsistent. From a safety perspective, Liu et al. [55] warn that toxicity and robustness tests are often too narrow and easily gamed, recommending flexible, real-world evaluations. Finally, Ye et al. [56] introduce psychometric principles, arguing that current benchmarks frequently lack checks for reliability and fairness.

While we acknowledge the usefulness of LLM benchmarks for progress tracking of LLM versions, comparison and ranking of LLMs, they cannot be used to make claims on LLM reliability. Our HIP-LLM approach goes beyond benchmarks, by framing the reliability assessment of LLMs as statistical inference problems.

2.3. Statistical Inference for Evaluating LLMs

Benchmark-based descriptive statistics simply summarize the collected data, whereas statistical inference draws probabilistic conclusions about the *underlying population or process* that generated those data, thereby enabling generalization, prediction and uncertainty quantification. In the AI/ML community, many studies have proposed statistically principled approaches to quantify uncertainty for *individual inputs*, such as text prompts for LLMs, using methods like conformal prediction [57, 58]. However, these approaches focus on instance-level uncertainty and do not capture uncertainty at the operational reliability level. To the best of our knowledge, only two existing works extend uncertainty quantification to this broader reliability perspective for LLMs.

In the Anthropic report [32], Miller frames LLM evaluations as statistical experiments and argues for reporting uncertainty alongside benchmark scores. Using classical inference tools, the study constructs confidence intervals via the Central Limit Theorem, applies clustered standard errors for correlated items, recommends paired inference for model comparisons, and develops power analysis and variance reduction techniques. This work emphasizes that benchmark outcomes should be treated as data drawn from an underlying population, moving beyond descriptive statistics.

HiBayES [31] applies hierarchical Bayesian generalized linear models that: (a) use a Bayesian statistical inference (multilevel Binomial/Poisson Generalized Linear Models) and

[6]Over time, models are trained on similar data, inflating scores and diminishing benchmark usefulness.

captures the nested structure of LLM evaluations; (b) employ partial pooling to account for dependencies across levels; (c) provide a fully probabilistic framework, yielding full posterior distributions to quantify uncertainties, rather than relying on point-estimates.

Despite these advances, important limitations remain. Neither Miller's frequentist framework nor HiBayES explicitly defines/models the OP (even though both implicitly assume that the data represent it). Moreover, neither approach can effectively incorporate (potentially imprecise) prior knowledge: Miller's work is restricted to classical frequentist inference, while HiBayES relies on non-informative priors. Both also focus on the narrower notion of failure probability as reliability, overlooking the more general and standardized reliability definition in terms of the probability of failure-free runs over future operations. The latter requires predictive inference that accounts for the propagation of uncertainties into the future: a more demanding but practical reliability claim [33, 34]. Miller accounts for clustered dependence through variance corrections, but does not introduce hierarchical latent variables or multi-level generative models as in HiBayES or HIP-LLM. Table 1 summarizes the key features and provides a comparative overview of the three methods (HIP-LLM, HiBayES [31], and Miller [32]).

2.4. Robust Bayesian Analysis

Robust Bayesian analysis represents a general framework for investigating the sensitivity of posterior measures to uncertainties in the inputs of Bayesian inference [59, 60]. Regarding uncertainties in priors, while several dedicated methods have been proposed [61, 33, 62, 63], Imprecise Probability has emerged as one of the most widely adopted approaches [36, 37, 64, 65, 66]. It addresses the problem of prior uncertainty by avoiding reliance on *a single* prior distribution, rather representing *a credal set* of plausible priors and deriving posterior bounds that reflect this epistemic uncertainty.

To illustrate the main idea of the Imprecise Probability framework, consider a simple coin-flipping problem where we wish to estimate the probability of heads θ with data D ($n = 10$, $k = 3$ heads). As shown in Table 2, instead of a single point estimate or a single Beta posterior distribution, the *posterior envelope* (i.e., a set of posterior distributions) of Imprecise Probability[7] reflects both the observed data and the epistemic uncertainty arising from imprecise prior knowledge (represented by the set of Beta priors).

Table 2: Classical Bayesian vs Imprecise Probability for a coin-flipping example.

Feature	**Classical Bayesian**	**Imprecise Probability**
Prior	$\text{Beta}(\alpha = 2, \beta = 2)$	$\text{Beta}(\alpha, \beta),\ \alpha \in [1, 3], \beta \in [1, 3]$
Posterior	$\theta \mid D \sim \text{Beta}(5, 9)$	$\theta \mid D \sim \text{Beta}(3 + \alpha, 7 + \beta),\ \alpha \in [1, 3], \beta \in [1, 3]$
Poster. mean	$\mathbb{E}[\theta \mid D] = 0.36$	$\mathbb{E}[\theta \mid D] \in [0.31, 0.38]$

Our hierarchical solution, HIP-LLM, is a robust Bayesian approach that addresses the

[7]For simplicity and illustrative purpose, the example used here is based on the model of *imprecise Linearly Updated Conjugate prior Knowledge (iLUCK)* [65]) which leverages the conjugacy for analytical posterior results. Our HIP-LLM is not using this iLUCK model due to the non-linearity of the hierarchical probabilistic model proposed in Section 3.

Table 1: Comparison of LLM evaluation frameworks highlighting differences in reliability definition, operational profile modeling, and uncertainty treatment

Aspect	**Miller [32]**	**HiBayES** [31]	**HIP-LLM**
Statistical framework	Frequentist	Bayesian	Bayesian
Primary goal	Uncertainty-aware benchmark reporting	Uncertainty-aware, hierarchical accuracy estimation	Uncertainty-aware reliability assessment under OP with embedded priors
Metric under estimation	Failure probability (per task)	Failure probability (per task)	Reliability (probability of failure-free future tasks)
OP	Implicit (dataset assumed representative)	Implicit (dataset assumed representative)	Explicit, multi-level OPs
Hierarchical structure	None	LLM, domains, subdomains	LLM, domains, subdomains
Dependence handling	Clustered standard errors	Partial pooling	Partial pooling
Prior specification	None	Uniform precise priors	Informative imprecise priors (credal sets)
Output	Confidence intervals	Single posterior distribution	Posterior envelopes

aforementioned gaps. Similar to HiBayES, it adopts a hierarchical structure, but instead of yielding a single posterior, it reports posterior envelopes over imprecise priors and uncertain OPs (as variables).

3. The Model: HIP-LLM

The proposed method HIP-LLM stands for Hierarchical Imprecise Probability for Large Language Models reliability assessment. Before introducing the framework, we first formally define the reliability of LLMs.

LLMs are software; therefore, to ensure compatibility with the more general and standardized definition of software reliability (Def. 1), we define LLM reliability as follows:

Definition 3 (LLM reliability)**.** *The probability that an LLM produces failure-free responses over a specified number of future tasks (sequences of closely related queries), under specified (sub-)domains and operational environment.*

The definition of LLM reliability closely parallels the standardized definition of software reliability, as both emphasize *probabilistic* reasoning and the requirement of *failure-free performance* in future operations. The key distinctions arise from the nature of LLMs as "on-demand" software. Whereas classical software reliability is typically framed in terms of "operational time" that covers continuously operated systems like controllers over clock-time, LLM reliability is defined more explicitly in terms of a specified number of future discrete tasks (each consisting of sequences of related queries). Furthermore, while both definitions account for operational conditions, the LLM context requires explicit reference to domains and subdomains, reflecting the general-purpose and multi-domain design of LLMs. Finally, in traditional software, failure is typically defined as a deviation from the specification, whereas there is no specification for LLMs and thus "failures of LLMs"[8] are often informally/implicitly characterized by factual errors (hallucinations) [67, 68] or divergence from human-expert answers [69, 70].

3.1. Problem Formulation

While we provide a table of notations in Appendix C, according to Def. 3, LLM reliability can be formalised as follows:

Definition 4 (Formalised LLM reliability)**.** *Let $\mathcal{X}$ denote the input-space of all possible tasks for a given LLM, and let π be the OP, i.e., a probability distribution over $\mathcal{X}$ that reflects the likelihood of encountering each task $x \in \mathcal{X}$ in practice. Consider a sequence of $n \geq 1$ tasks i.i.d. according to the OP π, let $\mathbb{I}(x_\tau) \in \{0, 1\}$ indicate success (1) or failure (0) on the τ-th task x_τ, then the LLM reliability is:*

$$R(n, \pi) = \Pr_{x_\tau \sim \pi} \left(\bigcap_{\tau=1}^{n} \{\mathbb{I}(x_\tau) = 1\} \right) \tag{1}$$

Intuitively, $R(n, \pi)$ represents the probability that the LLM will operate failure-free across the next n i.i.d. tasks according to the OP π. Importantly, we make the following remarks:

Remark 1 (The i.i.d. assumption vs. contextual memory)**.** *A key assumption of the above definition is the task failures/successes are i.i.d. Bernoulli trials. Such modeling is not uncommon in reliability modeling, especially for critical on-demand systems. A typical justification is when demands are rare, and the states/memory of the software and its operational environment are effectively "reset" in-between [23, 71]. In the context of LLMs, we carefully define our reliability metric in terms of i.i.d "tasks", rather than individual prompts which are often contextually dependent. A task may consist of a sequence of related prompts aimed at achieving a single task goal. We acknowledge that modern LLMs (e.g., ChatGPT) typically retain chat history as contextual memory, which can violate the i.i.d. assumption between tasks. However, most LLMs provide the option to start a new chat session (for a new single task) without any memory and history from previous chat sessions, aligning with the assumption of "resetting" the LLM. We note such "reset" settings are commonly encountered in LLM offline evaluation like benchmarking [30]. In contrast, long-context or agentic*

[8] A complete and formal characterization of what constitutes a "failure" for LLMs remains an open research question, and out of the scope of this paper. Cf. Section 5 for discussions.

workflows, in which tasks are sequentially dependent through memory retention or tool use, violate this assumption. We note, however, that such dependence is not unique to LLMs and has long been recognized in software reliability modeling, particularly in industrial control software, where sensor readings form continuous and dependent trajectories through the input space. In such settings, two common approaches are typically adopted: either modeling the system at a higher level of abstraction by treating an entire trajectory as a single trial[9]*, or employing alternative stateful reliability models (which we leave as an future work).*

Remark 2 (Failure probability vs. future reliability)**.** *While the reliability metric $R(n, \pi)$ represents the probability of correctly processing the next n future tasks under the OP π, the special case $1 - R(1, \pi)$ (where $n = 1$) represents the failure probability [33, 34, 35] that studied by, e.g., [31, 32]. Accordingly, throughout this paper, we use the term* failure probability *to denote $1-R(1, \pi)$ and* the probability of *failure-free runs (of n future tasks) to denote $R(n, \pi)$, in order to distinguish these two notions. The more general term* reliability *is used to refer to either quantity when the intended meaning is clear from the surrounding context.*

Remark 3 (General purpose reliability vs. domain-specific reliability)**.** *Since the input-space $\mathcal{X}$ represents all possible LLM tasks, so $R(n, \pi)$ is the general-purpose reliability of the LLM under study. For (sub-)domain specific reliability, we need to partition $\mathcal{X}$ and derive "local" OPs; then the lower level (sub-)domain specific reliability can be similarly derived like Eq.* (1)*.*

Remark 4 (Binary failure vs. non-binary scoring)**.** *In traditional software reliability engineering, failures are naturally defined as binary events, success or failure, according to an explicit system specification. Similarly, most LLM benchmarks introduce task-specific specifications, implemented via automated evaluators or human annotations, which yield binary outcomes for scoring each prompted task. Consistent with this established practice, we model LLM reliability based on binary failure in Def. 4. However, unlike traditional software, what constitutes a "failure" for an LLM can be domain dependent, and either objective with ground truth answers (for, e.g., coding, math, and factual Q&A) or inherently subjective (e.g., for creative writing). The external scoring mechanisms used to determine LLM failures may themselves be noisy or inconsistent, a challenge that is not unique to LLMs but well known in software reliability modeling as the problem of imperfect test oracles. Accordingly, in HIP-LLM Def. 4, failures are conceptually assumed to be well defined, and the framework is designed to operate correctly under a perfect test-oracle. The implications of imperfect or noisy scoring mechanisms (i.e., the imperfect test-oracle problem) are orthogonal to this formulation and are acknowledged as an important direction for future work.*

To do statistical inference for the reliability metric defined in Eq. (1) with assumptions in aforementioned remarks, a simplified "textbook" Bayesian model would be the Beta-Binominal one (which also used as one of the baselines in our experiments). This

[9] A widely adopted abstraction in industrial software reliability (e.g., protection systems in nuclear power plants) is to treat a continuous trajectory of dependent sensor readings as a single unit of processing [23, 28], which may result in either success or failure. Under this abstraction, Bernoulli trials remain appropriate despite strong dependence among inputs along the trajectory. The same rationale applies to LLMs, where an interaction session consisting of multiple dependent prompts/tasks can be treated as a single fallible trial.

Beta-Binomial estimator applies to a single domain with precise prior knowledge. Let $\theta := \Pr_{x\sim\pi}(I(x) = 1)$ denote the (unknown) probability of success on a random task drawn from the OP π. Given N i.i.d. evaluated tasks with C successes and $N - C$ failures, assume a Binomial likelihood

$$C, N \mid \theta \sim \text{Binomial}(N, \theta) = \theta^C (1-\theta)^{N-C},$$

and a prior $\Pr(\theta)$. By Bayes' rule, the posterior distribution of θ is

$$\Pr(\theta \mid C, N) = \frac{\theta^C (1-\theta)^{N-C} \Pr(\theta)}{\int_0^1 \theta^C (1-\theta)^{N-C} \Pr(\theta)\, d\theta}. \tag{2}$$

Similarly for the future reliability of passing n^F tasks:

$$\Pr(R(n^F, \pi) \mid C, N) = \frac{\theta^{n^F} \theta^C (1-\theta)^{N-C} \Pr(\theta)}{\int_0^1 \theta^C (1-\theta)^{N-C} \Pr(\theta)} \tag{3}$$

If $\Pr(\theta) = \text{Beta}(\alpha, \beta)$, conjugacy yields the closed-form posterior

$$\theta \mid C, N \sim \text{Beta}(\alpha + C,\ \beta + N - C).$$

For future reliability, thanks to the conjugacy again, the posterior mean reliability for n^F future tasks is therefore:

$$\mathbb{E}[R(n^F, \pi) \mid \mathcal{D}] = \mathbb{E}[\theta^{n^F} \mid \mathcal{D}] = \frac{B(\alpha + C + n^F,\ \beta + N - C)}{B(\alpha + C,\ \beta + N - C)},$$

where $B(\cdot, \cdot)$ denotes the Beta function. Similarly, the posterior PDF and CDF can also be derived and we omit them for brevity.

To more rigorously assess the formally defined LLM reliability, coping with the aforementioned Remarks and Gaps, the next subsection introduces our proposed solution HIP-LLM. It models the LLM as a hierarchical structure consisting of independent domains, each containing statistically dependent subdomains (cf. Fig. 1).

Remark 5 (The need of hierarchical modeling on (sub-)domain (in-)dependencies)**.** *The modeled dependencies and independencies represent the epistemic structure of our hierarchical Bayesian model. That is, observing failures in one (sub-)domain may or may not update our beliefs about the reliability of other (sub-)domains. One possible example of justification*[10] *is: We model coding and law as independent because they rely on distinct competencies of the LLM—coding on formal, symbolic reasoning and syntax manipulation, and law on narrative understanding and normative interpretation. Since these skills draw from largely separate representations and training data, failures in one domain provide little information about failures in the other. In Bayesian terms, their failure probabilities can be treated as a* priori independent parameters*, reflecting separate latent skill dimensions of the model. On the other hand, sub-domains failure probabilities are modeled as dependent parameters given their shared LLM competencies.*

[10]Cf. Section 5 for discussions on the validity of this hierarchal dependency assumption.

Our goal is to infer the posterior distributions of future reliability (and its special case, the failure probability), at the subdomain, domain, and general-purpose LLM levels, based on observed correct responses from tasks within each subdomain.

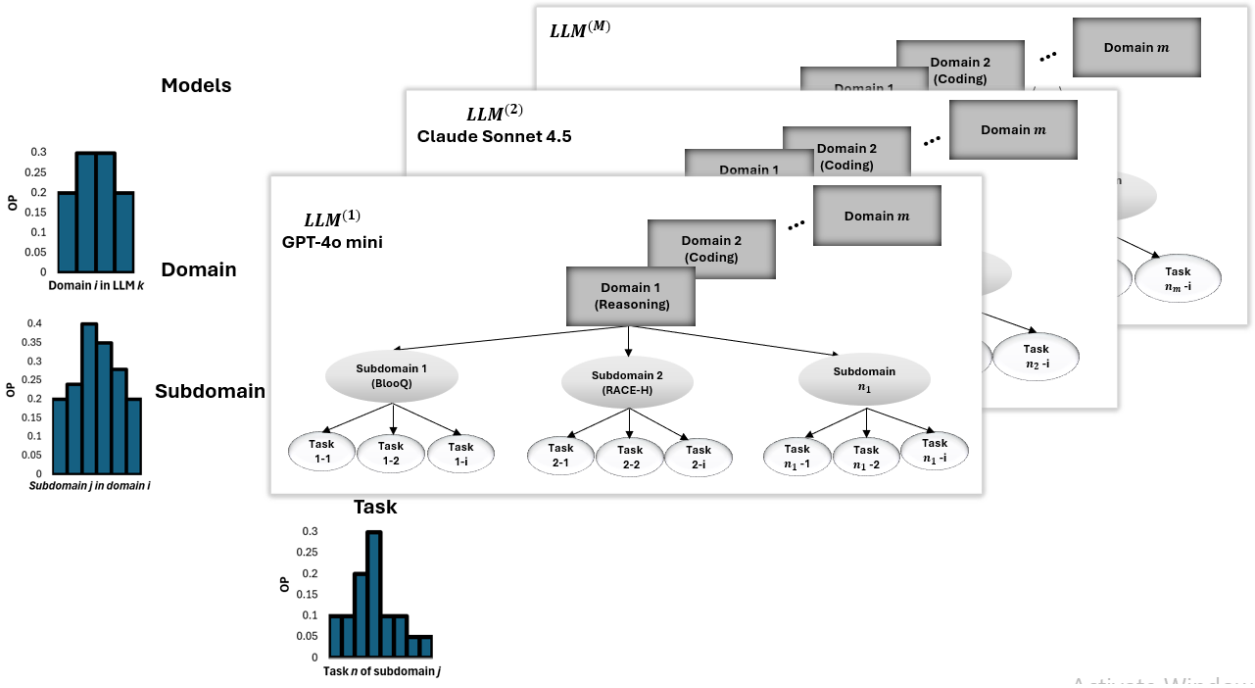


Figure 1: Schematic representation of the hierarchical LLM, domain, and subdomain structure for reliability estimation for M LLM models. Rectangles indicate independent components, while ovals indicate dependent components within the hierarchy.

3.2. Proposed Solution

Consider a hierarchical structure of an LLM comprising independent domains $D_1, D_2, \ldots, D_m$, where each domain D_i contains statistically dependent subdomains $S_{i1}, \ldots, S_{in_i}$ (Fig. 2). We wish to infer the posterior distributions over subdomain, domain and LLM level reliabilities by observing C_{ij} correct responses out of N_{ij} trials (tasks) in each subdomain.

Fig. 2 presents a detailed view of the hierarchical structure (subdomains $\rightarrow$ domains $\rightarrow$ LLM), with assumed priors and parameters. We assume that subdomain reliabilities within a domain are *dependent through a shared prior*, and that domain reliabilities are *aggregated* from their subdomains according to task-specific OPs. Our goal is to construct a principled hierarchical Bayesian model that supports information sharing across dependent subdomains through partial pooling (Sec. 3.2.1) and uncertainty quantification via Imprecise Probability (Sec. 3.2.2).

Figures 1 and 2 illustrate a general hierarchical structure comprising multiple LLM instances ($LLM^{(1)}, LLM^{(2)}, \ldots, LLM^{(M)}$). For clarity, however, Theorems 1–6 focus on the reliability assessment of a *single* LLM system. Accordingly, we omit the superscript (k) and use unindexed symbols (D_i, S_{ij}, p_L). Extending the framework to multiple LLMs is straightforward—apply it to each system independently and compare their posterior distributions.

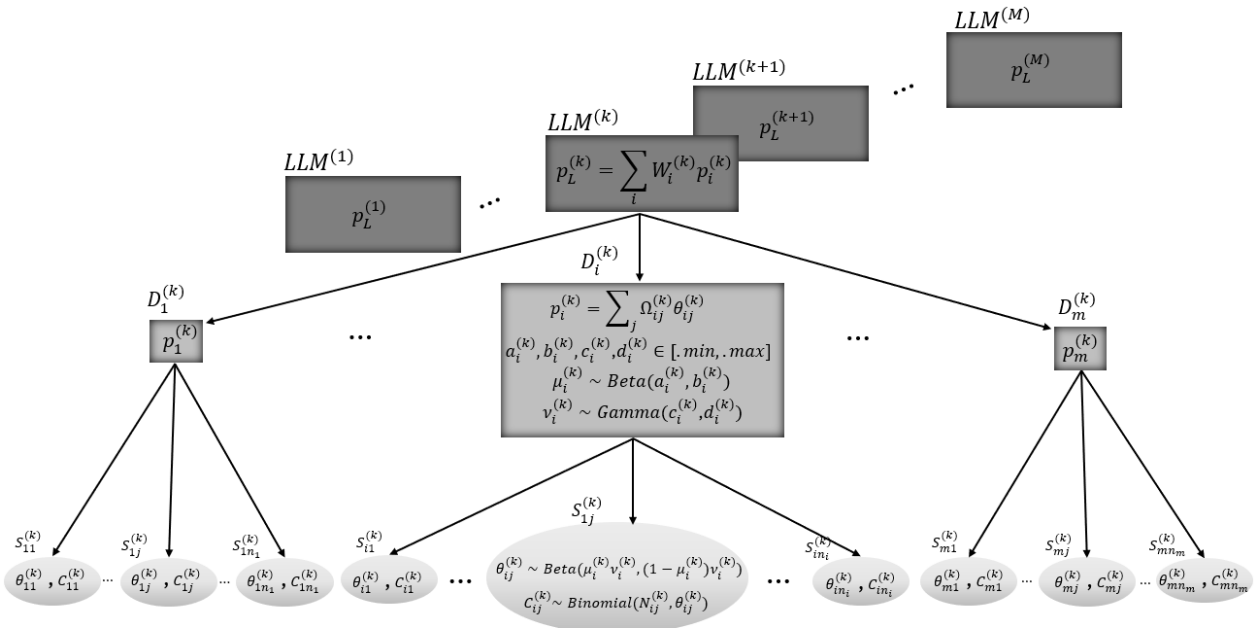


Figure 2: Hierarchical structure with independent domains and dependent subdomains. For readability, parameters and priors are shown only for one subdomain S_{ij} and its parent domain D_i under a representative LLM_k; the remaining subdomains, domains ($i = 1, \ldots, m$), and LLMs ($k = 1, \ldots, M$) are identical, differing only in their indices. Rectangles indicate independent components, while ovals indicate dependent components within the hierarchy.

3.2.1. Hierarchical Bayesian Framework for LLM Reliability Modeling

As seen in Fig. 2, inference proceeds hierarchically from the bottom up. At the subdomain level, the observed data consist of the number of correct responses C_{ij} out of total trials N_{ij}. These update the subdomain reliabilities θ_{ij}, which are modeled with a Binomial likelihood[11] and a Beta prior[12]. The Beta prior is parameterized by domain-level hyperparameters (μ_i, ν_i)[13], representing the expected reliability and the prior strength within domain D_i. These hyperparameters are in turn governed by domain-specific hyperpriors:

[11]The assumption of independent Bernoulli trials with constant success probability θ_{ij} may not capture all real dependencies, but it serves as an effective approximation for modeling subdomain outcomes [32, 31, 62]. Aggregating across N_{ij} trials then yields the Binomial likelihood, whose support $\{0, \ldots, N_{ij}\}$ matches the possible counts of correct responses observed in subdomain S_{ij}.

$$C_{ij} \mid \theta_{ij}, N_{ij} \;\sim\; \text{Binomial}(N_{ij}, \theta_{ij}).$$

[12]We place a Beta prior on subdomain reliability:

$$\theta_{ij} \sim \text{Beta}(\alpha_i, \beta_i).$$

The Beta distribution is the most common choice for probabilities bounded in $[0, 1]$, and it is conjugate to the Binomial likelihood, ensuring closed-form updates and computational stability.

[13]To make priors more intuitive and interpretable, we use a reparameterization. Instead of specifying the Beta prior directly in terms of (α_i, β_i), we express it as

$$\theta_{ij} \mid \mu_i, \nu_i \;\sim\; \text{Beta}(\mu_i \nu_i, (1-\mu_i)\nu_i),$$

$\mu_i \sim \text{Beta}(a_i, b_i)$[14], and $\nu_i \sim \text{Gamma}$[15]$(c_i, \text{rate}$[16]$ = d_i)$.

Because the hyperparameters (μ_i, ν_i) are shared across all subdomains in domain D_i, the information is pooled: subdomains with few observations are stabilized by drawing on evidence from other subdomains in the same domain, while subdomains with many observations are influenced mainly by their own data. This hierarchical setup achieves the desired dependence within domains, while domains remain independent (cf. Section 5 for discussions on this setup).

Once posterior subdomain reliabilities θ_{ij} are inferred, they are aggregated into domain-level reliabilities $p_i = \sum_j \Omega_{ij}\theta_{ij}$ using OP weights Ω_{ij}, which reflect the practical importance of subdomains. These domain-level posteriors are then further combined into the overall LLM reliability, $p_L = \sum_i W_i p_i$, using domain operational weights W_i. Note, OPs at different levels are represented by the variables Ω_{ij} and W_i, which, for a given LLM use case, can be instantiated either as fixed constants (when usage is certain) or as probability distributions that encode uncertainty about how the LLM will be used in practice [72, 73, 74, 75].

3.2.2. Uncertainty Handling via Imprecise Probability

To address the epistemic uncertainty in prior specification, we adopt an Imprecise Probability approach: we specify the hyper-hyper-parameters a_i, b_i, c_i, d_i as intervals (Eq. 4) rather than point values. This produces posterior envelopes (lower and upper bounds) at the subdomain, domain, and overall LLM levels.

$$\begin{aligned} a_i &\in [a_i^{\min}, a_i^{\max}], \quad b_i \in [b_i^{\min}, b_i^{\max}], \\ c_i &\in [c_i^{\min}, c_i^{\max}], \quad d_i \in [d_i^{\min}, d_i^{\max}] \end{aligned} \tag{4}$$

At the subdomain level, we compute posterior bounds for each task type by considering all admissible hyperparameter configurations:

$$\underline{Pr}(\theta_{ij} \mid C_i) \leq Pr(\theta_{ij} \mid C_i) \leq \overline{Pr}(\theta_{ij} \mid C_i) \tag{5}$$

Note, C_i denotes the data is domain i. Since we consider dependencies among subdomains, the posterior θ_{ij} is a function of all data C_i in domain i.

---

where μ_i denotes the expected reliability (prior mean, $\mu_i = \mathbb{E}[\theta_{ij} \mid \mu_i, \nu_i]$) and ν_i denotes the prior strength, reflecting the confidence in μ_i (equivalent sample size or pseudo-counts, $\nu_i = \alpha_i + \beta_i$). This reparameterization makes prior beliefs easier to specify and justify.

[14]This choice reflects the idea that μ_i itself is a probability lying in $(0, 1)$, and the Beta distribution provides a flexible family of shapes that can express different prior beliefs about domain reliability, ranging from diffuse to highly concentrated around particular values (e.g., favoring higher values, lower values, or balanced around 0.5).

[15]This treats ν_i as a positive random variable ($\nu_i > 0$) reflecting how tightly subdomains within a domain are assumed to cluster around μ_i. When ν_i is small, the prior is diffuse and allows substantial variation across subdomains (weak pooling). When ν_i is large, the prior concentrates mass near μ_i and subdomains are tightly clustered (strong pooling).

[16]The Gamma distribution is commonly written in two equivalent forms: shape–rate and shape–scale. We use the *shape–rate* form, $\nu_i \sim \text{Gamma}(c_i, \text{rate} = d_i)$, with $\mathbb{E}[\nu_i] = c_i/d_i$ and $\text{Var}[\nu_i] = c_i/d_i^2$. If a library expects the *shape–scale* form, set $\theta_i = 1/d_i$ and write $\nu_i \sim \text{Gamma}(c_i, \text{scale} = \theta_i)$; the two parameterizations are mathematically identical under $\theta_i = 1/d_i$. The only practical concern is clarity and reproducibility—accidentally treating a rate as a scale (or vice versa) would change the prior's mean and variance.

Later (Theorem. 2 and Theorem. 3) we will discuss that the closed-form densities (like subdomain level) do not exist for domain and LLM level reliability. We represent the cumulative distribution function (CDF) envelopes for the domain and overall LLM levels.

At the domain level, uncertainty propagates upward through OP weights $p_i = \sum_j \Omega_{ij}\theta_{ij}$, producing domain reliability bounds:

$$\underline{F}_{p_i}(t \mid C_i) = \inf_{h_i \in \mathcal{A}_i} F_{p_i}(t \mid C_i, h_i), \quad \overline{F}_{p_i}(t \mid C_i) = \sup_{h_i \in \mathcal{A}_i} F_{p_i}(t \mid C_i, h_i), \quad t \in [0,1]$$

Here and in the following, $t \in [0,1]$ denotes a generic probability threshold at which the CDF of the corresponding (non-failure probability or reliability) random variable is evaluated.

The domain level posterior p_i only considers C_i, given our assumption on cross-domain independence.

At the LLM level, uncertainty aggregates across all k domains through domain weights $p_L = \sum_{i=1}^{k} W_i p_i$, while respecting cross-domain independence, resulting in system-level reliability bounds:

$$\underline{F}_{p_L}(t \mid \text{data}) = \inf_{\mathcal{H} \in \mathcal{A}_{\text{LLM}}} F_{p_L}(t \mid \text{data}, \mathcal{H}),$$
$$\overline{F}_{p_L}(t \mid \text{data}) = \sup_{\mathcal{H} \in \mathcal{A}_{\text{LLM}}} F_{p_L}(t \mid \text{data}, \mathcal{H})$$

where, $t \in [0,1]$.

While in the next subsection, we develop Theorems 1–3 of deriving posterior sets for those non-failure probability variables θ_{ij}s, p_is and p_L, the posterior distribution sets for future reliability of passing next n^F tasks at each level, e.g., for a domain i:

$$\underline{Pr}((\sum_j \Omega_{ij}\theta_{ij})^{n^F} \mid C_i) \leq Pr((\sum_j \Omega_{ij}\theta_{ij})^{n^F} \mid C_i) \leq \overline{Pr}((\sum_j \Omega_{ij}\theta_{ij})^{n^F} \mid C_i) \tag{6}$$

can also be derived, as shown in our Theorems 4–6.

3.2.3. Theorems

The following theorems are the main mathematical results of HIP-LLM. Intuitively, given the imprecise prior knowledge encoded by the hyperparameters, the theorems derive the posterior distributions of non-failure probabilities at different hierarchical levels, as well as the future reliability, based on the probabilistic reasoning model illustrated in Fig. 2, conditioned on the observed task failure data.

Theorem 1 (Sub-domain level non-failure probability)**.** *For subdomain S_{ij} in domain D_i, let $C_i = \{(C_{ik}, N_{ik})\}_{k=1}^{n_i}$ be the observed data. Let the admissible set of hyperparameters be*

$$\mathcal{A}_i \;=\; [a_i^{\min}, a_i^{\max}] \times [b_i^{\min}, b_i^{\max}] \times [c_i^{\min}, c_i^{\max}] \times [d_i^{\min}, d_i^{\max}],$$

and write $h_i = (a_i, b_i, c_i, d_i)$. Then, for any *$h_i \in \mathcal{A}_i$, the marginal posterior density of θ_{ij} is*

$$Pr(\theta_{ij} \mid C_i, h_i) = \frac{f_{\text{marg}}(\theta_{ij}, C_i; h_i)}{Z_{\text{marg}}(h_i)},$$

where f_{marg} (unnormalized posterior) and Z_{marg} (normalizing constant) are

$$f_{\text{marg}}(\theta_{ij}, C_i; h_i) = \int_0^1 \int_0^\infty \left[\prod_{k \neq j} \int_0^1 d\theta_{ik} \right] L(\boldsymbol{\theta}_i)\, Pr(\boldsymbol{\theta}_i \mid \mu_i, \nu_i)\, Pr(\mu_i, \nu_i \mid h_i)\, d\mu_i\, d\nu_i$$

$$Z_{\text{marg}}(h_i) = \int_0^1 \int_0^\infty Pr(C_i \mid \mu_i, \nu_i)\, Pr(\mu_i, \nu_i \mid h_i)\, d\mu_i\, d\nu_i,$$

with $L(\boldsymbol{\theta}_i) = Pr(C_i \mid \boldsymbol{\theta}_i)$.

The imprecise marginal posterior is characterized by the lower/upper envelopes

$$\underline{Pr}(\theta_{ij} \mid C_i) = \inf_{h_i \in \mathcal{A}_i} Pr(\theta_{ij} \mid C_i, h_i), \qquad \overline{Pr}(\theta_{ij} \mid C_i) = \sup_{h_i \in \mathcal{A}_i} Pr(\theta_{ij} \mid C_i, h_i).$$

The proof of Theorem 1 is presented at Appendix Appendix A.2. In this theorem, the subdomain posterior $Pr(\theta_{ij} \mid C_i, h_i)$ has a closed-form density because of conjugacy: the Beta prior combined with the Binomial likelihood yields a mixture of Beta distributions after marginalizing over the hyperparameters (μ_i, ν_i), which can be expressed and evaluated as a proper probability density function.

However, for later Theorems closed-form densities do not exist because $p_i = \sum_j \Omega_{ij}\theta_{ij}$ and $p_L = \sum_i W_i p_i$ are weighted sums of dependent random variables—the distribution of a sum of Beta random variables has no analytical form except in trivial cases[17]. Computing such densities would require intractable multi-dimensional integrals. The CDF formulation sidesteps this problem: $F_{p_i}(t \mid C_i, h_i) = \int_0^1 \int_0^\infty F_{p_i}(t \mid \mu_i, \nu_i, C_i)\, Pr(\mu_i, \nu_i \mid C_i, h_i)\, d\mu_i\, d\nu_i$ only requires a two-dimensional integral over (μ_i, ν_i), where the conditional CDF can be computed via Monte Carlo sampling of independent Betas. Since CDFs provide all necessary information for practical reliability assessment (probabilities, quantiles, expectations), they are the natural representation when densities are unavailable.

Theorem 2 (Domain level posterior non-failure probability)**.** *For domain D_i with local OP weights Ω_{ij} (where $\sum_{j=1}^{n_i} \Omega_{ij} = 1$), let $p_i = \sum_{j=1}^{n_i} \Omega_{ij}\theta_{ij}$ be the domain-level non-failure probability. Define the admissible set of hyper-hyper-parameters*

$$\mathcal{A}_i = [a_i^{\min}, a_i^{\max}] \times [b_i^{\min}, b_i^{\max}] \times [c_i^{\min}, c_i^{\max}] \times [d_i^{\min}, d_i^{\max}],$$

and write $h_i = (a_i, b_i, c_i, d_i)$.

Then, for any $h_i \in \mathcal{A}_i$, the posterior distribution of p_i is characterized by its CDF:

$$F_{p_i}(t \mid C_i, h_i) = \Pr(p_i \leq t \mid C_i, h_i) = \int_0^1 \int_0^\infty F_{p_i}(t \mid \mu_i, \nu_i, C_i)\, Pr(\mu_i, \nu_i \mid C_i, h_i)\, d\mu_i\, d\nu_i$$

where

[17] Assuming the weights are constants. When the weights are modeled as random variables with their own probability distributions, the same problem persists (if not harder).

- $F_{p_i}(t \mid \mu_i, \nu_i, C_i)$ *is the conditional CDF of* $p_i = \sum_{j=1}^{n_i} \Omega_{ij}\theta_{ij}$ *given that* $\theta_{ij} \mid \mu_i, \nu_i, C_i \overset{ind.}{\sim} Beta(C_{ij} + \mu_i\nu_i, N_{ij} - C_{ij} + (1-\mu_i)\nu_i)$ *for* $j = 1, \ldots, n_i$,

- $Pr(\mu_i, \nu_i \mid C_i, h_i)$ *is the hyper-posterior obtained via Bayes' rule:*

$$Pr(\mu_i, \nu_i \mid C_i, h_i) = \frac{Pr(C_i \mid \mu_i, \nu_i)\, Beta(\mu_i \mid a_i, b_i)\, Gamma(\nu_i \mid c_i, rate = d_i)}{\int_0^1 \int_0^\infty Pr(C_i \mid \mu, \nu)\, Beta(\mu \mid a_i, b_i)\, Gamma(\nu \mid c_i, rate = d_i)\, d\mu\, d\nu}$$

The imprecise domain posterior is characterized by CDF envelopes:

$$\underline{F}_{p_i}(t \mid C_i) = \inf_{h_i \in \mathcal{A}_i} F_{p_i}(t \mid C_i, h_i), \qquad \overline{F}_{p_i}(t \mid C_i) = \sup_{h_i \in \mathcal{A}_i} F_{p_i}(t \mid C_i, h_i).$$

For the proof details, we refer reads to Appendix A.3.

Theorem 3 (LLM-level posterior non-failure probability)**.** *For the LLM system with domain weights* W_i *(where* $\sum_{i=1}^m W_i = 1$*), let* $p_L = \sum_{i=1}^m W_i p_i$ *be the LLM-level failure probability and* $data = \{C_1, \ldots, C_m\}$ *the observed data across all domains. Assume cross-domain independence.*

Define the domain-level admissible sets

$$\mathcal{A}_i = [a_i^{\min}, a_i^{\max}] \times [b_i^{\min}, b_i^{\max}] \times [c_i^{\min}, c_i^{\max}] \times [d_i^{\min}, d_i^{\max}]$$

and write $h_i = (a_i, b_i, c_i, d_i)$ *for* $i = 1, \ldots, m$*. Define the LLM-level admissible set as the Cartesian product*

$$\mathcal{A}_{LLM} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_m,$$

and collect the domain hyperparameters as $\mathcal{H} = (h_1, \ldots, h_m) \in \mathcal{A}_{LLM}$*.*

Then, for any $\mathcal{H} \in \mathcal{A}_{LLM}$*, the posterior distribution of* p_L *is characterized by its CDF:*

$$F_{p_L}(t \mid data, \mathcal{H}) = Pr(p_L \le t \mid data, \mathcal{H}) = \int \cdots \int G\big(t \mid \{\mu_i, \nu_i\}_{i=1}^m, data\big) \prod_{i=1}^m Pr(\mu_i, \nu_i \mid C_i, h_i) \prod_{i=1}^m d\mu_i\, d\nu_i$$

where

- $G(t \mid \{\mu_i, \nu_i\}_{i=1}^m, data)$ *is the conditional CDF of* $p_L = \sum_{i=1}^m W_i p_i$ *given all hyperparameters, defined as*

$$G\big(t \mid \{\mu_i, \nu_i\}_{i=1}^m, data\big) = \int_{\mathcal{R}_L(t)} \prod_{i=1}^m f_{p_i}(p_i \mid \mu_i, \nu_i, C_i)\, dp_1 \cdots dp_m,$$

where $\mathcal{R}_L(t) := \{(p_1, \ldots, p_m) \in (0,1)^m : \sum_{i=1}^m W_i p_i \le t\}$*, and* $f_{p_i}(\cdot \mid \mu_i, \nu_i, C_i)$ *is the conditional density of* $p_i = \sum_{j=1}^{n_i} \Omega_{ij}\theta_{ij}$ *under* $\theta_{ij} \mid \mu_i, \nu_i, C_i \overset{ind.}{\sim} Beta(C_{ij} + \mu_i\nu_i, N_{ij} - C_{ij} + (1-\mu_i)\nu_i)$*,*

- *$Pr(\mu_i, \nu_i \mid C_i, h_i)$ is the domain-level hyper-posterior for domain i:*

$$Pr(\mu_i, \nu_i \mid C_i, h_i) = \frac{Pr(C_i \mid \mu_i, \nu_i)\, Beta(\mu_i \mid a_i, b_i)\, Gamma(\nu_i \mid c_i, rate = d_i)}{\int_0^1 \int_0^\infty Pr(C_i \mid \mu, \nu)\, Beta(\mu \mid a_i, b_i)\, Gamma(\nu \mid c_i, rate = d_i)\, d\mu\, d\nu}$$

- *Cross-domain independence ensures $Pr(\{\mu_i, \nu_i\}_{i=1}^m \mid data, \mathcal{H}) = \prod_{i=1}^m Pr(\mu_i, \nu_i \mid C_i, h_i)$.*

The imprecise LLM posterior is characterized by CDF envelopes:

$$\underline{F}_{p_L}(t \mid data) = \inf_{\mathcal{H} \in \mathcal{A}_{LLM}} F_{p_L}(t \mid data, \mathcal{H})$$
$$\overline{F}_{p_L}(t \mid data) = \sup_{\mathcal{H} \in \mathcal{A}_{LLM}} F_{p_L}(t \mid data, \mathcal{H})$$

For more details, see Section Appendix A.4.

Theorems 1, 2, and 3 characterize the posterior distribution of non-failure probability θ_{ij} or aggregated p_i, p_L. However, in practice, we often care about the reliability over *a specified number of consecutive future operations*, e.g., "what is the probability that an LLM succeeds on the next 10 tasks in a row?" or "what is the probability that the LLM succeeds on the next 20 coding tasks?".

The following set of theorems extends the aforementioned theorems to characterize the full posterior distribution of reliability for n^F consecutive future operations, i.e., the probability that the LLM operates n^F consecutive failure-free tasks. Conditioning is on the observed evaluation data across subdomains S_{ij}: C_{ij} correct generations out of N_{ij} prompts.

Theorem 4 (Subdomain posterior reliability for n^F future operations)**.** *For subdomain S_{ij} in domain D_i, let $C_i = \{(C_{ik}, N_{ik})\}_{k=1}^{n_i}$ denote all observed data in the domain. Let the admissible set of hyperparameters be*

$$\mathcal{A}_i = [a_i^{\min}, a_i^{\max}] \times [b_i^{\min}, b_i^{\max}] \times [c_i^{\min}, c_i^{\max}] \times [d_i^{\min}, d_i^{\max}],$$

and write $h_i = (a_i, b_i, c_i, d_i)$. Define the reliability random variable[18] *$R_{ij}(n^F) = \theta_{ij}^{n^F}$, representing the probability of n^F consecutive failure-free operations in subdomain S_{ij}.*

For any $h_i \in \mathcal{A}_i$, the posterior CDF of $R_{ij}(n^F)$ is:

$$F_{R_{ij}(n^F)}(t \mid C_i, h_i) = \Pr(\theta_{ij}^{n^F} \le t \mid C_i, h_i) = \int_0^{t^{1/n^F}} Pr(\theta_{ij} \mid C_i, h_i)\, d\theta_{ij},$$

where $Pr(\theta_{ij} \mid C_i, h_i)$ is the marginal posterior density from Theorem 1.

The imprecise posterior distribution is characterized by the CDF envelopes:

$$\underline{F}_{R_{ij}(n^F)}(t \mid C_i) = \inf_{h_i \in \mathcal{A}_i} F_{R_{ij}(n^F)}(t \mid C_i, h_i),$$
$$\overline{F}_{R_{ij}(n^F)}(t \mid C_i) = \sup_{h_i \in \mathcal{A}_i} F_{R_{ij}(n^F)}(t \mid C_i, h_i)$$

[18] Comparing to the reliability definition in Def. 4, the OPs are omitted as we assume they are fixed constants in these theorems.

The transformation $\theta_{ij} \mapsto \theta_{ij}^{n^F}$ generally does not yield a closed-form density. We therefore characterize $R_{ij}(n^F)$ through its CDF. See Appendix A.5 computational methods.

Theorem 5 (Domain posterior reliability for n^F future operations)**.** *For domain D_i with local OP weights Ω_{ij} (where $\sum_{j=1}^{n_i} \Omega_{ij} = 1$), let $C_i = \{(C_{ik}, N_{ik})\}_{k=1}^{n_i}$ denote all observed data in the domain. Let the admissible set of hyperparameters be*

$$\mathcal{A}_i = [a_i^{\min}, a_i^{\max}] \times [b_i^{\min}, b_i^{\max}] \times [c_i^{\min}, c_i^{\max}] \times [d_i^{\min}, d_i^{\max}],$$

and write $h_i = (a_i, b_i, c_i, d_i)$. Define the domain-level reliability

$$p_i = \sum_{j=1}^{n_i} \Omega_{ij}\theta_{ij}, \quad R_i(n^F) = p_i^{n^F},$$

representing the probability of n^F consecutive failure-free operations at the domain level.

For any $h_i \in \mathcal{A}_i$, the posterior distribution of $R_i(n^F)$ is characterized by its CDF:

$$F_{R_i(n^F)}(t \mid C_i, h_i) = \Pr\left(\left[\sum_{j=1}^{n_i} \Omega_{ij}\theta_{ij}\right]^{n^F} \leq t \,\middle|\, C_i, h_i\right),$$

computed by integrating over the joint posterior $Pr(\boldsymbol{\theta}_i \mid C_i, h_i)$.

The imprecise posterior distribution is characterized by the CDF envelopes:

$$\underline{F}_{R_i(n^F)}(t \mid C_i) = \inf_{h_i \in \mathcal{A}_i} F_{R_i(n^F)}(t \mid C_i, h_i),$$

$$\overline{F}_{R_i(n^F)}(t \mid C_i) = \sup_{h_i \in \mathcal{A}_i} F_{R_i(n^F)}(t \mid C_i, h_i)$$

The CDF can be computed via numerical integration over the joint posterior $Pr(\boldsymbol{\theta}_i \mid C_i, h_i)$ and Monte Carlo sampling, cf. Appendix A.5.

Theorem 6 (LLM posterior reliability for n^F future operations)**.** *For the LLM system with an OP of domain weights W_i (where $\sum_{i=1}^{m} W_i = 1$), let data $= \{C_1, \ldots, C_m\}$ denote all observed data across domains. Assume cross-domain independence.*

Define the domain-level admissible sets

$$\mathcal{A}_i = [a_i^{\min}, a_i^{\max}] \times [b_i^{\min}, b_i^{\max}] \times [c_i^{\min}, c_i^{\max}] \times [d_i^{\min}, d_i^{\max}]$$

and write $h_i = (a_i, b_i, c_i, d_i)$ for $i = 1, \ldots, m$. Define the LLM-level admissible set as the Cartesian product

$$\mathcal{A}_{LLM} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_m,$$

and collect the domain hyperparameters as $\mathcal{H} = (h_1, \ldots, h_m) \in \mathcal{A}_{LLM}$.

Define the LLM-level reliability

$$p_L = \sum_{i=1}^{m} W_i p_i, \quad R_L(n^F) = p_L^{n^F},$$

representing the probability of n^F consecutive failure-free operations at the LLM level.

For any $\mathcal{H} \in \mathcal{A}_{LLM}$, the posterior distribution of $R_L(n^F)$ is characterized by its CDF:

$$F_{R_L(n^F)}(t \mid data, \mathcal{H}) = \Pr\left(\left[\sum_{i=1}^{m} W_i p_i \right]^{n^F} \leq t \,\middle|\, data, \mathcal{H} \right),$$

computed by integrating over $\prod_{i=1}^{m} Pr(p_i \mid C_i, h_i)$.

The imprecise posterior distribution is characterized by the CDF envelopes:

$$\underline{F}_{R_L(n^F)}(t \mid data) = \inf_{\mathcal{H} \in \mathcal{A}_{LLM}} F_{R_L(n^F)}(t \mid data, \mathcal{H}),$$

$$\overline{F}_{R_L(n^F)}(t \mid data) = \sup_{\mathcal{H} \in \mathcal{A}_{LLM}} F_{R_L(n^F)}(t \mid data, \mathcal{H})$$

Again, while the closed-form density is not available, the CDF can be computed via numerical integration and Monte Carlo sampling. See Appendix A.5 for details.

Remark 6 (Translate vague prior knowledge to hyperparameters)**.** *In HIP-LLM, hyperparameters are not tuning constants but variables used to represent uncertainty about prior knowledge. At the lowest level, θ_{ij} indicates "how reliable" (i.e., the probability that the model succeeds on a randomly drawn task from the subdomain) of S_{ij}. Subdomains within the same domain D_i are assumed to be related, and this dependence is captured by assuming that their non-failure probabilities are drawn from a shared Beta distribution governed by two domain-level parameters: μ_i and ν_i. Here, μ_i represents the expected reliability of the LLM in domain D_i, while ν_i represents how confident we are in this expectation, or equivalently, how strongly subdomain reliabilities are expected to cluster around μ_i. Since assessors typically do not know the exact values of μ_i and ν_i, HIP-LLM places distributions on them: $\mu_i \sim \text{Beta}(a_i, b_i)$ and $\nu_i \sim \text{Gamma}(c_i, d_i)$. The hyperparameters (a_i, b_i, c_i, d_i) therefore encode high-level and (likely) vague/imperfect expert beliefs rather than performance data.*

To ensure that these hyperparameters correspond consistently to the intended prior beliefs, it is convenient to express (a_i, b_i) and (c_i, d_i) through an auxiliary mean–strength parameterization*. Let: **a)** $r_i \in (0, 1)$ denote the assessor's belief about the expected reliability in domain D_i, and $s_i > 0$ denote the associated confidence level (interpretable as an equivalent number of prior observations); **b)** $e_i > 0$ denote the assessor's expected clustering strength (i.e., how strongly subdomain reliabilities are expected to concentrate around the domain mean), and $f_i > 0$ represent the associated confidence in this expectation.*

The Beta parameters can then be written as follows:

$$a_i = r_i s_i, \qquad b_i = (1 - r_i) s_i \tag{7}$$

Under this parameterization the prior mean is

$$\mathbb{E}[\mu_i] = \frac{a_i}{a_i + b_i} = r_i,$$

while the parameter $s_i = a_i + b_i$ controls the strength (concentration) of the prior. Similarly, the Gamma parameters can be written as

$$c_i = f_i, \qquad d_i = \frac{f_i}{e_i}. \tag{8}$$

Under this reparameterization, the prior mean of the clustering strength becomes

$$\mathbb{E}[\nu_i] = \frac{c_i}{d_i} = e_i,$$

while the parameter f_i controls the concentration of the Gamma prior around e_i. Larger values of f_i correspond to stronger confidence in the expected clustering strength.

For example, suppose an assessor believes that the reliability of the LLM in the reasoning domain is likely between 0.8 and 0.9, but with only moderate confidence. This belief can be expressed by specifying $r_i \in [0.8, 0.9]$ and $s_i \in [30, 100]$. The corresponding Beta parameters therefore lie in the ranges $a_i \in [24, 90]$ and $b_i \in [3, 20]$. At the same time, the assessor may believe that the reliabilities of different reasoning subdomains (e.g., logical reasoning, multi-step reasoning etc.) should not differ drastically, but may still vary to some extent. This belief can be expressed by specifying $e_i \in [30, 100]$ and $f_i \in [3, 10]$ which determine the corresponding Gamma hyperparameters through $c_i = f_i$ and $d_i = f_i/e_i$ (i.e., $c_i \in [3, 10]$ and $d_i \in [0.03, 0.33]$). Together, the hyperparameters (a_i, b_i, c_i, d_i) therefore encode the assessor's belief that the overall reliability of the domain is high (around 0.8–0.9), while allowing moderate variation between individual subdomains.

It is important to note that the parameters s_i and f_i represent different levels of epistemic confidence in the hierarchical model. The parameter s_i reflects the assessor's confidence in the expected domain reliability μ_i, whereas ν_i controls how similar the reliabilities of individual subdomains are expected to be. Consequently, the model separates two distinct aspects of prior knowledge: beliefs about the overall reliability of the domain and beliefs about the degree of similarity among its subdomains. In this representation, r_i and e_i denote the assessor's expected values for the domain reliability and the clustering strength, respectively, while s_i and f_i encode the associated confidence levels. This hierarchical structure therefore allows assessors to express both their expectations and their uncertainty about these quantities in a transparent manner.

Allowing these quantities to vary within intervals rather than fixing them to single values enables HIP-LLM to represent epistemic uncertainty about prior beliefs following the imprecise probability framework [36, 37]. This uncertainty is then propagated through the hierarchical model, resulting in posterior reliability bounds rather than overconfident point estimates.

4. Evaluation

To demonstrate and evaluate our HIP-LLM, we empirically investigate five research questions (RQs) in this section.

4.1. Research Questions

RQ1 (Effectiveness): How effectively can HIP-LLM assess and compare posterior reliability distributions across different levels of the hierarchy, considering uncertainties propagated from subdomains to domains, and finally to general-purpose LLMs? In this RQ, we aim to demonstrate the use case of our HIP-LLM as a reliability assessment tool.

RQ2 (Sensitivity to hyperparameters): How sensitive are the posterior estimates of HIP-LLM to the hyperparameters (a_i, b_i, c_i, d_i)? These hyperparameters represent the

assessors' (imprecise) prior knowledge, thus understanding their sensitivity to the posteriors may provide insights on how prior knowledge can be elicited.

RQ3 (Sensitivity to OPs): How sensitive are the posterior reliability estimates of HIP-LLM to variations in the OPs that characterize operational usage of LLMs at various levels? The delivered and perceived reliability of an LLM depends on how it will be used, i.e. the OP. We hypothesize that general-purpose LLMs may exhibit lower sensitivity to variations in OPs, whereas LLMs trained for specific (sub-)domains are likely to be more sensitive. We investigate and demonstrate how HIP-LLM can characterize and quantify such OP-dependent reliability variations.

RQ4 (Predictability): How can HIP-LLM predicts future reliability of passing next n^F tasks? While RQ1–RQ3 focus on failure probabilities, which is a special case of reliability, we additionally aim to demonstrate HIP-LLM's capability to predict the probability of successfully completing n^F future tasks and to quantify how this reliability varies as the reliability requirement n^F changes.

RQ5 (Comparison to baselines): How does HIP-LLM compare to established and state-of-the-art Bayesian reliability estimators? To compare the accuracy of different models against a "ground truth" reliability, we conduct synthetic simulation experiments where the "ground truth" OP and failure probabilities of sub-domains are assumed known.

RQ6 (Failure definitions): How can HIP-LLM cope with alternative definitions of task success, such as different pass@k criteria? As per Remark 4, the definition of failures of a LLM may vary and subject to uncertainties. While we do not formally study LLM failure definitions, we demonstrate how different failure definitions can be incorporated in HIP-LLM.

RQ7 (Robustness to memory effects): How robust are HIP-LLM's posterior estimates to violations of the i.i.d. assumption caused by memory-induced dependence during LLM evaluation? While HIP-LLM is designed for reset, single-task scenarios, real-world evaluation data may not be strictly generated under this assumption. It is therefore critical to assess whether its posterior estimates remain meaningful when independence is only approximately satisfied or partially violated by memory-induced dependencies.

RQ8 (Scalability): How does the computational time and hardware RAM cost of HIP-LLM scale with the number of domains, subdomains, hyperparameter configurations, and Monte Carlo samples? Understanding the scalability of HIP-LLM with these key parameters is essential to determine if it can be applied to large-scale evaluations in practice.

4.2. Experimental Setup

OPs. To emphasize the role of OPs, *we "simulate" the OPs by assigning probability distributions over these datasets for sampling and by specifying operational weights across (sub-)domains.* Without loss of generality, and consistent with our three-level hierarchy, we define the task-level OP as a uniform distribution and assign operational weights at the subdomain and domain levels proportionally to their dataset sizes. That said, assessors may use alternative distributions as the 3-level OPs when additional information is available, such as those approximated from historical usage data [24, 76] or user behavior reports [27]. As demonstrated in RQ3, where we vary the operational weights.

Remark 7 (Simulated OP from benchmarks vs. real-world OP)**.** *Similar to many software reliability modeling studies [28], HIP-LLM assumes that the OP is specified and reflects real*

usage conditions. In the absence of real operational data, and following the same experimental practice as [31, 32], we leverage existing benchmark datasets to "simulate" OPs for fair comparison with baselines. Specifically, benchmark datasets are treated as sampling frames, and proportional dataset sizes are adopted as a proxy OP. This choice neither implies that OP acquisition is solved nor suggests that such proxies are suitable for real deployments; rather, it serves to demonstrate how HIP-LLM integrates OPs when they are provided and to enable comparison with existing benchmark-based methods.

Estimating realistic and evolving OPs from usage data is a well-studied problem in software reliability and is usually treated separately from reliability modeling [77, 25]. In practice, an OP for an LLM system can be constructed using a simple workflow based on real usage data:

1. ***Log collection:*** *Collect interaction logs or telemetry from deployed systems containing user prompts or tasks.*
2. ***Task mapping:*** *Assign each task to a domain or subdomain (e.g., coding, reasoning, summarization) using predefined rules or automated classifiers.*
3. ***Frequency estimation:*** *Count how often each task category occurs to estimate empirical frequencies.*
4. ***OP construction and update:*** *Convert these frequencies into probabilities that define the OP, and periodically update the OP as new logs become available.*

While we believe existing OP estimation techniques may be applicable to LLMs, deriving OPs from real-world LLM usage data introduces new challenges and warrants dedicated future investigation.

Data. Specifically, we evaluate our hierarchical framework by simulating LLM operational data from four widely used benchmarks, structured into two domains with two subdomains each (same as HiBayEs [31]), instantiated by the following datasets:

- **Domain 1 (Coding)**
  - **Subdomain**$_{11}$: basic Python programming tasks with unit-test based evaluation (MBPP [9]).
  - **Subdomain**$_{12}$: data-science oriented Python problems involving libraries such as `pandas` and `numpy` (DS-1000 [8]).
- **Domain 2 (Reasoning)**:
  - **Subdomain**$_{21}$: reading comprehension problems where the LLM must answer yes/no questions given short passages (BoolQ [78]).
  - **Subdomain**$_{22}$: high school level reading comprehension problems with multiple-choice answers (RACE-H [44]).

All experiments were conducted using publicly available APIs of GPT-4o, GPT-4o-mini (from OpenAI), Claude Sonnet 4.5, and Claude Haiku 3.5 (from Anthropic)[19]. Table 3 reports

[19]For academic research purposes only which is permitted under both vendors' terms of service for research publications.

sample accuracies per subdomain, i.e., the proportions of correct responses under `Pass@1`[20]. To operationalize the i.i.d. Bernoulli trail assumption on tasks, all LLM evaluations were conducted with cleared context between tasks. Each task in MBPP, DS-1000, BoolQ, and RACE-H was processed as an independent API call with no conversational history, ensuring that outcomes are not influenced by previous interactions.

Table 3: Evaluation results using Pass@1. Entries are per-subdomain accuracies (C_{ij}/N_{ij}). Rightmost column shows the row mean across models.

Dom	**Subdom (Data.)**	4o-mini	4o	sonnet-4.5	haiku-3.5	Mean
Dom_1	Subdom_{11} (MBPP)	0.440	0.471	0.450	0.447	0.452
	Subdom_{12} (DS-1000)	0.490	0.420	0.493	0.483	0.472
	Mean	0.465	0.446	0.472	0.465	–
Dom_2	Subdom_{21} (BoolQ)	0.890	0.909	0.900	0.883	0.896
	Subdom_{22} (RACE-H)	0.820	0.552	0.840	0.859	0.768
	Mean	0.855	0.731	0.87	0.871	–
LLM Mean		0.661	0.585	0.671	0.668	–

From the benchmark accuracy scores, all LLMs appear to perform better in the Reasoning domain than in the Coding domain. Their performances across most subdomains are similar, except for RACE-H, where model OpenAI-4o shows a clear weakness. The small variance between accuracy values for corresponding subdomains across models suggests that the performance differences are more domain-driven than model-driven.

Hyperparameters. All the hyperparameters we are using for generating the set of figures in the next subsection is shown in Appendix B.

Interpretation of Posterior CDF Envelopes. CDF envelopes are used to jointly represent data uncertainty and epistemic uncertainty arising from imprecise prior knowledge. Since closed-form densities are generally unavailable for weighted sums of dependent reliability parameters, posterior uncertainty is characterized numerically through CDF envelopes, which support probabilistic queries and risk-aware interpretation across all levels of the hierarchy. Unless stated otherwise, all figures in this section follow this convention.

Throughout the experimental section, where appropriate, we visualize posterior uncertainty using CDF envelopes. These CDFs are constructed from observed benchmark evaluation data, consisting of C_{ij} correct responses out of N_{ij} tasks for each subdomain S_{ij}. Depending on the level of aggregation shown, a CDF represents the posterior distribution of a non-failure probability at the subdomain level (θ_{ij}), the domain level (p_i), or the overall LLM level (p_L). Domain- and LLM-level CDFs are obtained by aggregating lower-level reliability parameters using the specified OP weights.

Monte Carlo sampling is used because the hierarchical aggregation of dependent subdomain reliabilities and imprecise priors leads to posterior distributions that are analytically intractable but can be evaluated efficiently and accurately via numerical sampling.

[20]Produce 1 solution per task. Score as correct if that solution passes verification.

Baseline Experimental Configuration. Unless otherwise stated, all empirical experiments are conducted using a fixed baseline configuration with $m = 2$ domains, $n = 2$ subdomains per domain, $K = 160$ hyperparameter configurations per domain, $S = 3000$ Monte Carlo samples per configuration, a (μ, ν) integration grid of size $G = 2000$, a fixed CDF evaluation grid of size $T = 201$, and a capped number of LLM-level configuration pairings $K_{\text{total}} \leq 512$. Domain-level parameters μ_i and ν_i are treated as latent variables and numerically integrated over a grid, while the hyper-hyperparameters (a_i, b_i, c_i, d_i) are sampled from fixed intervals, with $a_i, b_i \in [1, 12]$ and $c_i, d_i \in [1, 25]$, to represent imprecise prior knowledge.

All experiments were implemented in Python 3.12.12 using NumPy 2.0.2, pandas 2.2.2, and matplotlib 3.10.0. The experiments were executed in the Google Colab environment on a CPU runtime with an Intel(R) Xeon(R) CPU @ 2.20GHz and approximately 13 GB RAM. Experiments were performed in a single-process setting without explicit parallelization or GPU acceleration.

4.3. Results and Analysis

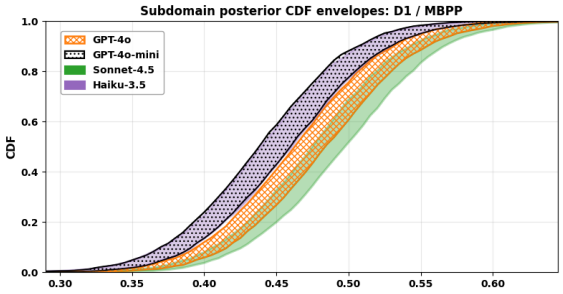
This section reports and analyses the empirical results to the RQs.

4.3.1. ***RQ1****(Effectiveness)*

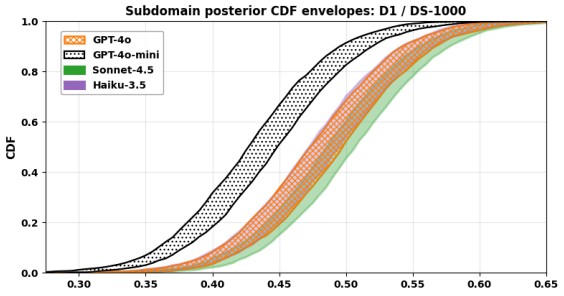
Fig. 3 shows posterior CDF envelopes for the four subdomains (MBPP, DS–1000, BoolQ, RACE–H) across the four LLMs where a right–shifted CDF indicates higher reliability and a tighter band indicates greater certainty.

Coding tasks. On MBPP (top left), the envelopes for GPT–4o–mini and Haiku 3.5 are nearly indistinguishable and lie to the left of GPT–4o, while Sonnet 4.5 is right–most but still overlapping with–4o. On DS–1000 (top right), GPT–4o–mini lies clearly to the left, while Haiku 3.5 overlaps almost entirely with GPT–4o—their envelopes coincide so closely that Haiku 3.5 is barely distinguishable. Sonnet 4.5 remains right–most, with only partial overlap with GPT–4o and Haiku 3.5.

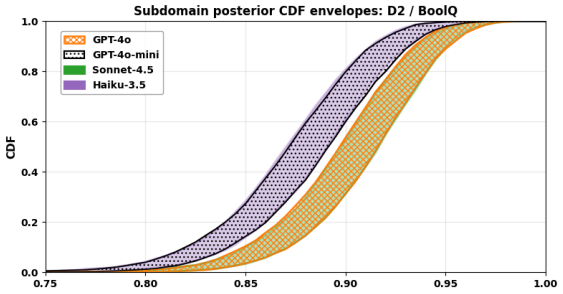
Reasoning tasks. On BoolQ (bottom left), GPT–4o–mini is almost identical to Haiku 3.5 on the left, while Sonnet 4.5 and GPT–4o nearly coincide on the right. Hence the envelopes separate into two close pairs: mini $\approx$ Haiku 3.5 $<$ Sonnet 4.5 $\approx$ 4o. On RACE–H (bottom right), Haiku 3.5 lies clearly left–most, GPT–4o–mini overlaps partly with Haiku but extends rightward, followed by GPT–4o, and finally Sonnet 4.5 on the far right.



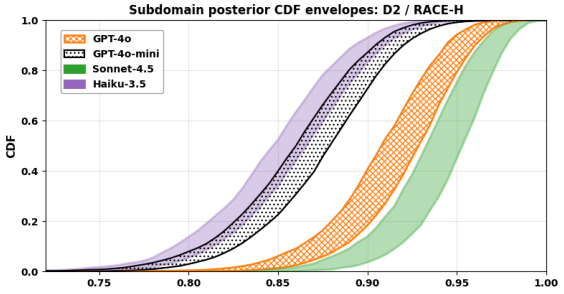
(a) Posterior CDF envelopes of the non-failure probability for Subdom_{11} (MBPP dataset in Dom_1) across four models.



(b) Posterior CDF envelopes of the non-failure probability for Subdom_{12} (DS-1000 dataset in Dom_1) across four models.



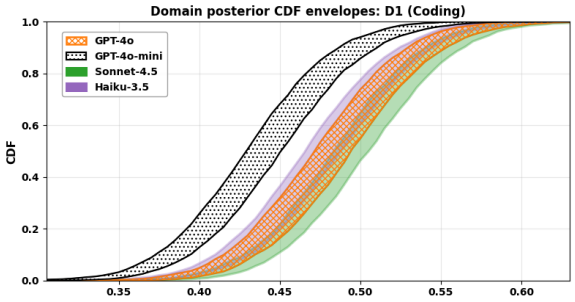
(c) Posterior CDF envelopes of the non-failure probability for Subdom_{21} (BlooQ dataset in Dom_2) across four models.



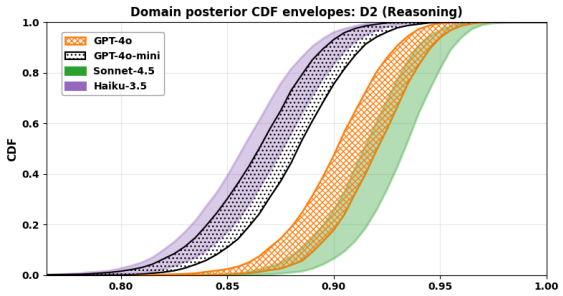
(d) Posterior CDF envelopes of the non-failure probability for Subdom_{22} (RACE-H dataset in Dom_2) across four models.

Figure 3: Posterior CDF envelopes of non-failure probability at the subdomain level. Rows correspond to domains (D_1 (coding): MBPP, DS-1000; D_2 (reasoning): BoolQ, RACE-H). Subdomains within the same domain are statistically dependent through shared domain-level hyperparameters (μ_i, ν_i), where μ_i represents the average domain reliability and ν_i controls the strength of coupling (partial pooling) among subdomains. As a result, observations from one subdomain inform the inferred reliability of the others. Domains are assumed statistically independent. A question-oriented interpretation of figure: After observing the evaluation data, how do the four models differ in terms of their subdomain-level non-failure probability?

Fig. 4 aggregates subdomains within each domain via the operational weights (Ω_{ij}) and reports the posterior CDFs of the domain reliabilities (p_i). Note, for simplicity, we just assign the operational weights (Ω_{ij}) proportionally according to the dataset sizes of the two sub-domains in each domain i. Similarly to sub-domain results, we may observe and compare domain-level non-failure probabilities p_i. Again, comparing to accuracy scores (point estimates) and HiBayEs (single posterior distributions), our HIP-LLM yields distribution envelopes that considers more types of uncertainties.



(a) Posterior CDF envelopes of non-failure probability $p_1 = \sum_j \Omega_{1j}\theta_{1j}$ for Dom_1 (Coding) across four LLMs, with subdomain-level operational weights $\Omega_{1\cdot} = (0.204, 0.796)$.



(b) Posterior CDF envelopes of non-failure probability $p_2 = \sum_j \Omega_{2j}\theta_{2j}$ for Dom_2 (reasoning) across four models, with subdomain-level operational weights (0.483, 0.517).

Figure 4: Posterior CDF envelopes of non-failure probability of domain-level ($p_i = \sum_j \Omega_{ij}\theta_{ij}$). A question-oriented interpretation of figure: After observing the evaluation data, how do the four models differ in terms of their domain level non-failure probability?

Fig. 5 aggregates both domains into the overall LLM reliability p_L using the cross–domain operational weights $W = [0.149, 0.851]$ for [Coding, Reasoning]. As before, the weights are simply assigned proportionally according to the dataset sizes of domains. The figure shows, 4o-mini and Haiku bands overlap almost completely, indicating near–equivalent reliability, while GPT–4o and Sonnet 4.5 partially overlap, reflecting moderate but consistent uncertainty between them. Overall, Sonnet 4.5 remains most reliable (which is consistent with Table 3).

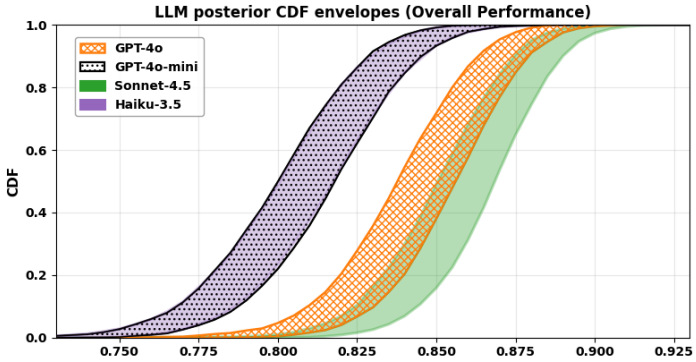
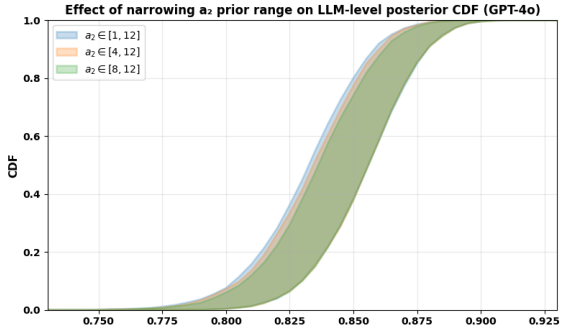
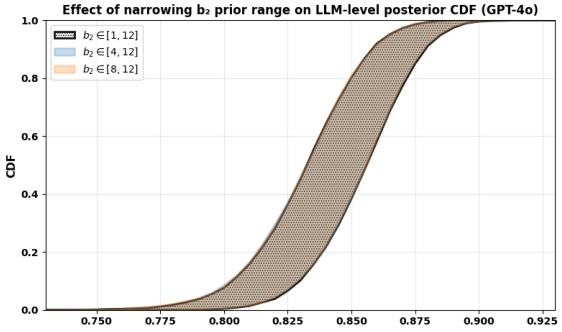


Figure 5: Posterior CDF envelope of non-failure probability of overall LLM level ($p_L = \sum_i W_i p_i$) with domain weights $W = (0.149, 0.851)$ across four models. A question-oriented interpretation of figure: *Engineering question:* After observing the evaluation data, how do the four LLMs differ in terms of their LLM-level non-failure probability?

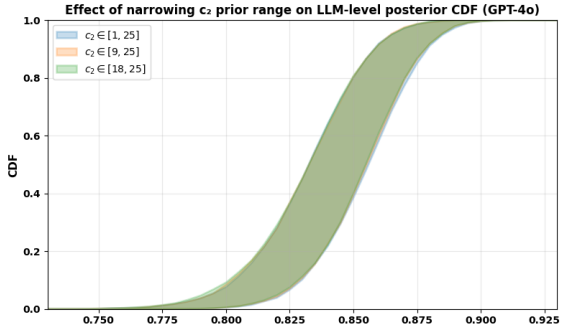
4.3.2. ***RQ2*** *(Sensitivity to hyperparameters)*



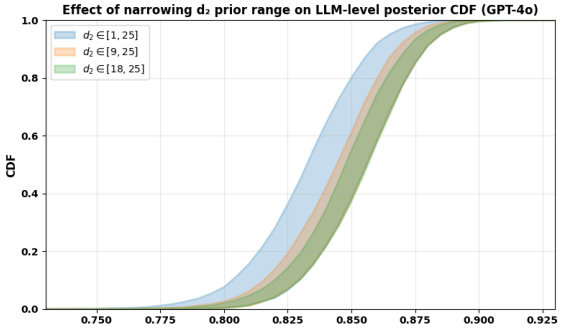
(a) Sensitivity of posterior CDF envelops of non-failure probability ($p_L = \sum_{i=1}^{m} W_i p_i$) to a.



(b) Sensitivity of posterior CDF envelops of non-failure probability of LLM ($p_L = \sum_{i=1}^{m} W_i p_i$) to b.



(c) Sensitivity of posterior CDF envelops of non-failure probability of LLM ($p_L = \sum_{i=1}^{m} W_i p_i$) to c.



(d) Sensitivity of posterior CDF envelops of non-failure probability of LLM ($p_L = \sum_{i=1}^{m} W_i p_i$) to d.

Figure 6: Effect of variations in the hyperparameter values (a, b, c, and d) on the posterior CDF envelops of non-failure probability of LLM ($p_L = \sum_{i=1}^{m} W_i p_i$). A question-oriented interpretation of the figures: "What are the LLM-level posterior CDF envelopes of the non-failure probability as domain-level hyperparameters vary within their admissible ranges, after observing the testing data?"

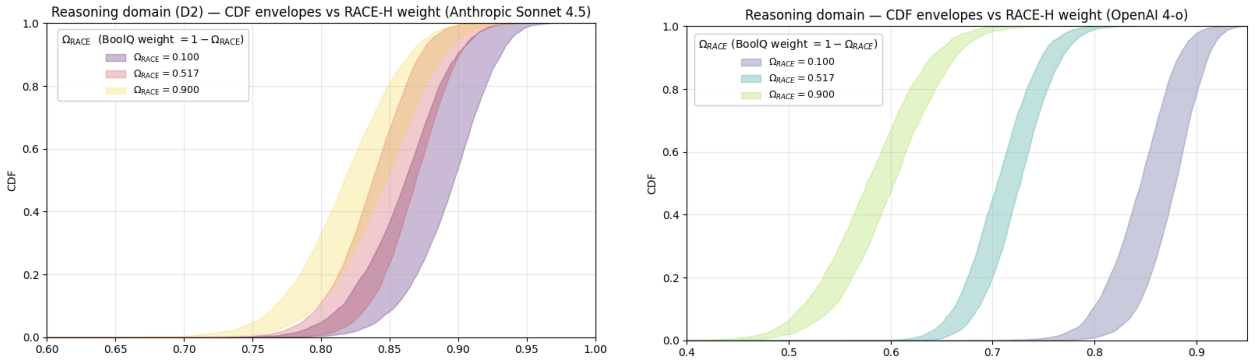
Corresponding to Remark 6, the sensitivity analysis in Fig. 6 highlights that different hyperparameters encode qualitatively different aspects of expert elicitation and therefore require different levels of care when specified. Variations in a_i and b_i reflect differences in expert expectations about the average domain-level reliability and mainly affect the location of the posterior reliability envelopes. As such, specifying these parameters requires careful consideration of how optimistic or pessimistic prior beliefs are justified by available evidence. In contrast, the hyperparameters c_i and d_i encode expert beliefs about confidence and pooling strength across subdomains, and directly control the width of the posterior envelopes.

Fig. 6a illustrates the sensitivity of the LLM-level posterior reliability to the hyperparameter a_i in the prior $\mu_i \sim \text{Beta}(a_i, b_i)$. For example, keeping $b_i = 4$, an expert belief corresponding to $a_i = 8$ yields a prior mean $\mu_i = 0.67$, while a more optimistic belief expressed by $a_i = 16$ increases the mean to $\mu_i = 0.80$. When propagated through the hierarchical model and updated with the same evaluation data, these beliefs lead to a rightward shift of the posterior CDF envelope, indicating a higher central estimate of p_L, while the envelope width remains similar due to the constraining effect of the data. Fig. 6b shows the complementary effect of varying b_i. With $a_2 = 12$, allowing $b_2 \in [1, 12]$ corresponds to prior means $\mu_2 \in [0.50, 0.92]$, whereas narrowing the range to $b_2 \in [4, 12]$ and $[8, 12]$ restricts μ_2 to

$[0.50, 0.75]$ and $[0.50, 0.60]$, respectively.

Fig. 6c and Fig. 6d illustrate the sensitivity of the LLM-level posterior reliability distributions to the hyperparameters c_i and d_i. Allowing wide ranges such as $c_2, d_2 \in [1, 25]$ corresponds to admitting both weak and strong confidence scenarios, while progressively narrowing these ranges (e.g., $c_2 \in [9, 25]$ or $[18, 25]$, and similarly for d_2) excludes low- or high-confidence assumptions. When propagated through the hierarchical model and updated with the same evaluation data, these changes leave the central location of the posterior CDF largely unchanged but systematically tighten or widen the envelope, indicating that c_i and d_i primarily control epistemic uncertainty rather than the expected reliability level. The stronger effect of d_i occurs because it directly reduces the pooling strength ν_i, while c_i mainly affects how concentrated this belief is. As a result, changing d_i more strongly weakens pooling across subdomains and leads to larger changes in the width of the posterior reliability envelopes.

4.3.3. ***RQ3*** *(Sensitivity to OPs)*



(a) Posterior CDF envelopes of non-failure probability for the Reasoning domain (D_2) under alternative operational weights on RACE–H, for Anthropic Sonnet 4.5.

(b) Posterior CDF envelopes of non-failure probability for the Reasoning domain (D_2) under alternative operational weights on RACE–H, for OpenAI 4-o.

Figure 7: Posterior CDF envelopes of non-failure probability for the Reasoning domain (D_2) under alternative operational weights on RACE–H, $\Omega_{\text{RACE}} \in \{0.10, 0.517, 0.90\}$, with $\Omega_{\text{BoolQ}} = 1 - \Omega_{\text{RACE}}$. **(a)** Sonnet 4.5; **(b)** OpenAI 4-o. A question-oriented interpretation of figure: How does the inferred domain-level non-failure probability change as the operational profile shifts emphasis between RACE–H and BoolQ?

Fig. 7 shows posterior CDF envelopes for the Reasoning-domain reliability p_2 under three OPs, $\Omega_{\text{RACE}} \in \{0.10, 0.517, 0.90\}$ with $\Omega_{\text{BoolQ}} = 1 - \Omega_{\text{RACE}}$. Fig. 7a reports Anthropic Sonnet 4.5; Fig. 7b reports OpenAI 4-o.

(a) Sonnet 4.5. As Ω_{RACE} increases the CDF shifts slightly left (lower delivered p_2), with strong overlap among bands. This weak sensitivity to the OP is consistent with the subdomain accuracies being both high and similar ($\theta_{\text{BoolQ}} \approx 0.90$ vs. $\theta_{\text{RACE-H}} \approx 0.84$): the mixture $p_2 = \Omega_{\text{BoolQ}}\theta_{\text{BoolQ}} + \Omega_{\text{RACE}}\theta_{\text{RACE-H}}$ changes only modestly as weight moves from BoolQ to RACE–H. The mild widening of the envelope at higher Ω_{RACE} reflects slightly larger posterior uncertainty on RACE–H relative to BoolQ.

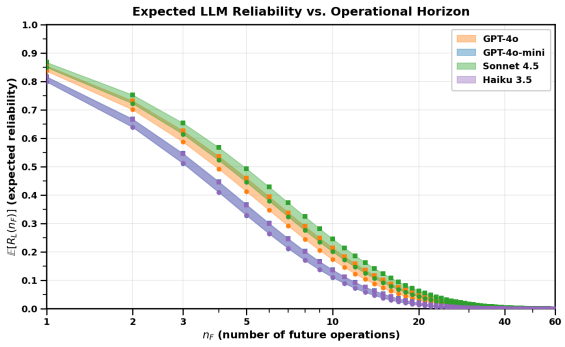
(b) OpenAI 4-o. In contrast, the envelopes are well separated and move substantially left as Ω_{RACE} increases. This strong dependence on the operational weighting arises because 4-o's subdomain accuracies differ widely ($\theta_{\text{BoolQ}} \approx 0.91$ vs. $\theta_{\text{RACE-H}} \approx 0.55$), so the mixture p_2 is dominated by whichever subdomain receives more weight.

Overall, the posterior CDF envelopes for OpenAI 4-o exhibit a strong dependence on the BoolQ/RACE–H weighting, whereas Sonnet 4.5 is comparatively invariant to Ω_{RACE}. This arises because GPT-4-o's subdomain accuracies differ widely (0.91 vs 0.55), producing a hierarchical posterior with pronounced weight sensitivity. These results suggest that GPT-4-o's reasoning performance is more uneven across benchmark types, while Sonnet-4.5 demonstrates domain-level robustness.

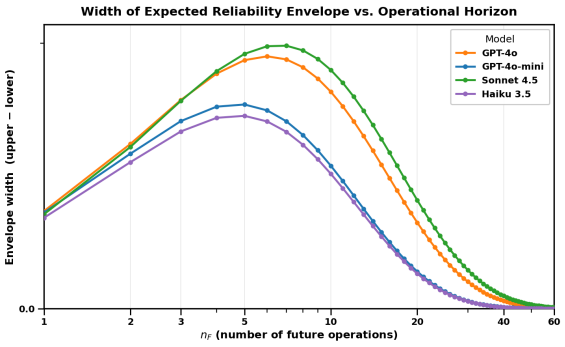
From a practical point of view, these results show that changing the operational weights can meaningfully change the reliability experienced by users, especially for OpenAI 4-o. The operational weights represent how often different types of tasks occur in real use, so changing them corresponds to a change in user behavior or application context. For 4-o, different task mixes lead to clearly different reliability levels, meaning that the same model can appear reliable in one application but much less so in another. In contrast, Sonnet 4.5 shows relatively stable reliability across different task mixes, indicating more uniform performance. This highlights that understanding the expected task distribution is important when deploying LLMs, particularly for models whose performance varies strongly across subdomains.

4.3.4. ***RQ4*** *(Predictability)*

Fig. 8a presents the expected LLM-level reliability $\mathbb{E}[R_L(n^F)]$ as a function of the operational horizon n^F, where n^F denotes the number of consecutive operations in the future. The horizontal axis displays n^F on a logarithmic scale, while the vertical axis shows the expected reliability—the posterior mean probability that the LLM successfully completes all next n^F consecutive tasks without failure.



(a) Imprecise posterior envelopes of the expected LLM reliability ($\mathbb{E}[R_L(n^F)]$) versus the required number of failure-free future tasks n^F for reliability claims. Imprecise posterior envelopes show decay in reliability over consecutive tasks across four models.



(b) Envelope widths (upper–lower) of expected reliability $\mathbb{E}[R_L(n^F)]$. Each curve represents the spread (upper–lower bound) of expected reliability as the number of future operations increases. The bell-shaped profile indicates that uncertainty grows with operational horizon before decaying as overall reliability converges toward zero.

Figure 8: Expected reliability envelopes and their width as functions of number of failure-free future tasks required by assessors (i.e., $\mathbb{E}[R_L(n^F)]$ and [upper bound of envelop − lower bound of envelop]). Question-oriented interpretation of the figure: "what is the posterior expected reliability of passing next n^F future tasks, after seeing testing data?"

At $n^F = 1$ all models exhibit high expected reliability, $\mathbb{E}[R_L(1)] \approx 0.80$–$0.87$, i.e., a high probability of producing a correct answer on the very next task (the non-failure probability). As the horizon length n^F increases, expected reliability drops sharply for every model. Across all horizons the ordering is stable: Sonnet-4.5 achieves the highest reliability, GPT-4o follows

closely, while Haiku-3.5 and GPT-4o-mini form the lower pair (with GPT-4o-mini typically the lowest). However, as shown in As Fig. 8b, the uncertainty (measured by the interval width) of Sonnet-4.5's reliability prediction is also the highest.

As Fig. 8b shows, at $n^F = 1$ the envelopes are quite tight, indicating low uncertainty. The bands widen modestly for small–to–medium horizons (i.e., when n^F increases from 2 to ~ 6), as a result of compounded prediction uncertainties. As n^F increases, all curves converge toward zero, indicating reduced uncertainty in the reliability prediction. Intuitively, this suggests greater confidence that the LLMs are unlikely to pass all n^F future tasks.

From a user perspective, Fig. 8 illustrates how reliability claims vary with the required number of future failure-free tasks n^F: as n^F increases, the claim becomes more stringent, leading to a corresponding decrease in confidence. This provides practical guidance for selecting models based on intended usage. For example, in healthcare decision support, an assessor may require a very high reliability of failure-free performance for a small number of critical diagnostic queries, favoring models with stronger short-horizon reliability guarantees (i.e., a higher but more steeply declining curve). In legal document analysis, a compliance team may comprise reliability a bit over a longer sequence of tasks (i.e., a lower but flatter curve), such as cross-checking clauses or summarizing precedents. In this way, Fig. 8 illustrates how model reliability scales with the required number of future failure-free tasks and supports informed deployment decisions across different application scenarios.

4.3.5. ***RQ5*** *(Comparison to baselines)*

We evaluate the proposed HIP-LLM method against three Bayesian baselines using controlled synthetic experiments in which the *ground-truth system reliability* (GT) is known. The system consists of two domains, each comprising two subdomains. Each subdomain is characterized by a reliability θ_{ij} and an OP weight OP_{ij}. The overall system reliability is defined as the OP-weighted average of subdomain reliabilities[21],

$$p_L = \sum_{i,j} \mathrm{OP}_{ij}\, \theta_{ij}. \tag{9}$$

In Bayesian inference, the posterior distribution reflects a combination of prior beliefs and observed data, with their relative influence governed by the amount of available evidence. As the sample size increases, the posterior becomes increasingly dominated by the data rather than the prior[22]. Thus, to illustrate HIP-LLM's ability to incorporate informative prior and to examine the influence of priors on posterior estimates, we consider two distinct sample-size regimes:

[21] In the paper, the OP is defined hierarchically, with W_i representing the importance of each domain and Ω_{ij} representing the relative importance of subdomain j within domain i, so that the system reliability is given by $p_L = \sum_i W_i \sum_j \Omega_{ij}\, \theta_{ij}$. In the experiments, this structure is written in an equivalent but simpler form by specifying directly the combined subdomain weights $\mathrm{OP}_{ij} = W_i\, \Omega_{ij}$. The system reliability is then computed as $p_L = \sum_{i,j} \mathrm{OP}_{ij}\, \theta_{ij}$, which is algebraically identical to the hierarchical expression. Under this representation, domain weights are not fixed or ignored: they are implicitly given by $W_i = \sum_j \mathrm{OP}_{ij}$, and the within-domain weights are recovered as $\Omega_{ij} = \mathrm{OP}_{ij}/W_i$. This choice simplifies the experimental setup by defining the OP at the subdomain level, while remaining fully consistent with the hierarchical formulation used in the paper.

[22] In the limiting cases, when no data are observed the posterior coincides with the prior, whereas with an infinite amount of data the posterior converges to "frequentist" estimates driven solely by the data.

- **Small-N:** [100, 500, 1000, 300] samples per subdomain, where priors may have more influence on the posterior.

- **Large-N:** [1000, 5000, 10000, 3000] samples per subdomain, where data are expected to dominate more than the priors.

For each regime, synthetic binomial observations are generated from the same ground-truth subdomain reliabilities. Inference is performed under three OP scenarios:

- **Dataset-based OP ($\mathbf{OP^{data}}$):** OP weights are set proportional to dataset sizes, introducing a structural mismatch with respect to the ground truth OP. This setting represents a naive scenario in which the OP is ignored and assessors rely solely on (benchmark) datasets for reliability assessment.

- **Approximated OP ($\mathbf{OP^{approx}}$):** The ground truth OP is perturbed by $\pm 20\%$ noise and renormalized, representing a scenario in which assessors recognize the importance of OPs and apply existing OP estimation methods to approximate them, rather than relying solely on evaluation datasets (i.e., $\mathrm{OP}^{\mathrm{data}}$), while still incurring estimation errors.

- **Ground-truth OP ($\mathbf{OP^{GT}}$)**: The ground truth OP is used directly, representing an idealized oracle scenario in which the assessors know the OP for certain.

We compare the following inference methods:

- **BB-UnInf**: Independent Beta–Binomial model with non-informative priors (Beta(1,1)).

- **BB-Inf**: Independent Beta–Binomial model with informative priors centered on the GT reliabilities.

- **HiBayES ([31]):** A hierarchical Bayesian model with partial pooling across subdomains.

- **HIP-LLM:** The proposed imprecise-probability approach, producing an envelope over posterior distributions induced by a family of priors.

The criteria for comparison are as follows:

- **Median ($\hat{p}_L^{\mathrm{med}}$):** For methods producing a single posterior distribution (BB-UnInf, BB-Inf, HiBayES), the median is defined as the empirical median of the posterior samples of p_L. For HIP-LLM, which induces a *set of posterior distributions* indexed by hyperparameter configurations $h \in \mathcal{H}$, we first compute the median of each posterior,

$$m_h = Q_{0.50}\big(p_L^{(h)}\big),$$

and then report the envelope of posterior medians,

$$\hat{p}_L^{\mathrm{med}} \in \left[\min_{h\in\mathcal{H}} m_h,\ \max_{h\in\mathcal{H}} m_h\right].$$

- **Error:** For single-posterior methods, the error is defined as the absolute deviation between the posterior median and the ground-truth reliability,

$$\text{Error} = \left| \hat{p}_L^{\text{med}} - p_L^{\text{GT}} \right|.$$

  For HIP-LLM, the error is computed *per posterior envelope* as

$$e_h = \left| m_h - p_L^{\text{GT}} \right|,$$

  and the reported error corresponds to the envelope of admissible errors,

$$\text{Error} \in \left[\min_{h \in \mathcal{H}} e_h,\ \max_{h \in \mathcal{H}} e_h \right].$$

- **Credible Interval (CI):** For single-posterior methods, a Bayesian 90% credible interval is defined as

$$\text{CI}_{0.90} = \left[Q_{0.05}(p_L),\ Q_{0.95}(p_L) \right],$$

  where $Q_q(\cdot)$ denotes the empirical quantile.

  For HIP-LLM, which produces a family of posteriors $\{p_L^{(h)}\}_{h \in \mathcal{H}}$, we report the widest credible interval compatible with the assumed prior uncertainty,

$$\left[\min_{h \in \mathcal{H}} Q_{0.05}\left(p_L^{(h)}\right),\ \max_{h \in \mathcal{H}} Q_{0.95}\left(p_L^{(h)}\right) \right].$$

Tables 4 and 5 illustrate how system reliability estimates are influenced by the OP and prior assumptions. In the Small-N regime, where data are limited, all methods slightly underestimate the ground-truth reliability ($p_L^{\text{GT}} = 0.5860$), with posterior medians typically between 0.57 and 0.58. In this setting, both OP mismatch and prior choice matter: using an approximate or dataset-based OP increases error compared to the ground-truth OP, and informative priors can reduce point error when they are correctly aligned with the GT subdomain reliabilities. This reduction, however, is due to favorable prior specification rather than inherent robustness and would not persist if the prior were misspecified.

Table 4: Posterior estimates under the Small-N ([100, 500, 1000, 300]) regime ($p_L^{\mathrm{GT}} = 0.5860$).

OP	Method	Median value/ bound	Error value/bound	90% Interval
$\mathrm{OP}^{\mathrm{data}}$	BB-UnInf	0.5733	0.0127	[0.5549, 0.5917]
	BB-Inf	0.5749	0.0111	[0.5569, 0.5924]
	HiBayES	0.5732	0.0128	[0.5552, 0.5915]
	HIP-LLM	[0.5713, 0.5751]	[0.0109, 0.0147]	[0.5528, 0.5937]
$\mathrm{OP}^{\mathrm{approx}}$	BB-UnInf	0.5740	0.0120	[0.5460, 0.6009]
	BB-Inf	0.5756	0.0104	[0.5509, 0.5995]
	HiBayES	0.5712	0.0148	[0.5438, 0.5966]
	HIP-LLM	[0.5698, 0.5774]	[0.0086, 0.0162]	[0.5421, 0.6048]
$\mathrm{OP}^{\mathrm{GT}}$	BB-UnInf	0.5782	0.0078	[0.5486, 0.6062]
	BB-Inf	0.5796	0.0064	[0.5538, 0.6042]
	HiBayES	0.5750	0.0110	[0.5459, 0.6011]
	HIP-LLM	[0.5738, 0.5816]	[0.0044, 0.0122]	[0.5446, 0.6100]

HIP-LLM behaves differently in the Small-N regime by reporting intervals rather than single reliability values. For example, it produces ranges of plausible medians such as [0.5713, 0.5751] under the dataset-based OP and [0.5738, 0.5816] under the ground-truth OP. These wider bounds reflect epistemic uncertainty about prior assumptions and make explicit that multiple reliability estimates are consistent with the data. Hierarchical modeling through HiBayES also smooths estimates across subdomains, but can introduce additional bias when data are scarce.

In the Large-N regime, the effect of priors and modeling choices largely disappears. All methods produce nearly identical posterior medians across OP scenarios, and uncertainty intervals become very narrow. For HIP-LLM, the median interval collapses to a tight range (e.g., [0.5860, 0.5866] under the dataset-based OP), showing that prior uncertainty no longer plays a significant role.

When comparing methods, HIP-LLM intervals should be interpreted as robustness bounds rather than standard credible intervals; for conservative decision making, the lower bound of the HIP-LLM median interval provides a worst-case reliability estimate consistent with the available data.

4.3.6. ***RQ6*** *(Sensitivity to the definition of success/failure)*

A key modeling assumption in HIP-LLM is the use of binary success/failure labels at the subdomain level, summarized by the observed success counts C_{ij} and total trials N_{ij}. In practice, however, the notion of "success" is not unique and may depend on the chosen evaluation criterion.

To assess the robustness of HIP-LLM to alternative success definitions, we conduct a controlled experiment on a single dataset (MBPP) and a single model (Claude Sonnet 4.5). We compare two common criteria: *Pass@1*, where only the first generated solution is evaluated, and *Pass@3*, where a task is considered successful if at least one out of three independent generations passes all tests. All other experimental conditions are held fixed.

Table 5: Posterior estimates under the Large-N ([1000, 5000, 10000, 3000]) regime ($p_L^{\mathrm{GT}} = 0.5860$).

OP	Method	Median value/ bound	Error value/bound	90% Interval
$\mathrm{OP}^{\mathrm{data}}$	BB-UnInf	0.5907	0.0047	[0.5849, 0.5965]
	BB-Inf	0.5908	0.0048	[0.5849, 0.5965]
	HiBayES	0.5909	0.0049	[0.5846, 0.5969]
	HIP-LLM	[0.5860, 0.5866]	[0.0000, 0.0006]	[0.5846, 0.5968]
$\mathrm{OP}^{\mathrm{approx}}$	BB-UnInf	0.5887	0.0027	[0.5801, 0.5973]
	BB-Inf	0.5885	0.0025	[0.5799, 0.5970]
	HiBayES	0.5887	0.0027	[0.5800, 0.5980]
	HIP-LLM	[0.5752, 0.5764]	[0.0096, 0.0108]	[0.5794, 0.5978]
$\mathrm{OP}^{\mathrm{GT}}$	BB-UnInf	0.5928	0.0068	[0.5836, 0.6017]
	BB-Inf	0.5926	0.0066	[0.5837, 0.6015]
	HiBayES	0.5929	0.0069	[0.5838, 0.6025]
	HIP-LLM	[0.5784, 0.5797]	[0.0063, 0.0076]	[0.5832, 0.6024]

Figure 9 shows the subdomain-level posterior CDF envelopes for the true success probability $\frac{C}{N}$ under Pass@1 and Pass@3. As expected, the more permissive Pass@3 criterion yields a higher observed success count and a right-shifted posterior envelope. However, the two envelopes exhibit substantial overlap, indicating that the inferred reliability is only moderately sensitive to the choice of success definition.

4.3.7. ***RQ7*** *(Robustness to Memory Effects)*

Our formal reliability definition models task outcomes as i.i.d. Bernoulli trials under a fixed OP. In practice, however, LLM usage may violate independence because the model can condition on previously observed conversational context (i.e., *memory*). To assess the practical impact of such dependence on HIP-LLM, we conducted two complementary experiments: (i) an empirical evaluation to quantify memory growth when memory is not reset after each task and its relation to dependence strength; and (ii) a targeted sensitivity analysis that injects memory-induced dependence into one of subdomains (Subdom_{21}-BoolQ) and examines how the resulting dependence shifts posterior envelopes across hierarchy levels.

We evaluated one subdomain (Subdom_{21}-BoolQ)[23] in a single continuous session in which the entire conversation history was retained in every API call. For each query, we recorded the API-reported memory. In this setting, memory grows monotonically with the number of questions and provides an operational proxy for the strength of cross-item dependence introduced by persistent context (Fig. 10a). After 299 questions, the retained context reached $\approx 2.06 \times 10^5$ bytes, i.e., ≈ 201 KB, illustrating that long-context evaluation can induce non-negligible memory accumulation even within a single benchmark run.

Using the above proxy, we observed that the dependence strength induced by memory

[23]We conducted this experiment on 300 tasks from the BoolQ dataset. To preserve the OP proportions across subdomains, we kept the same relative weights for the other datasets as in the main evaluation.

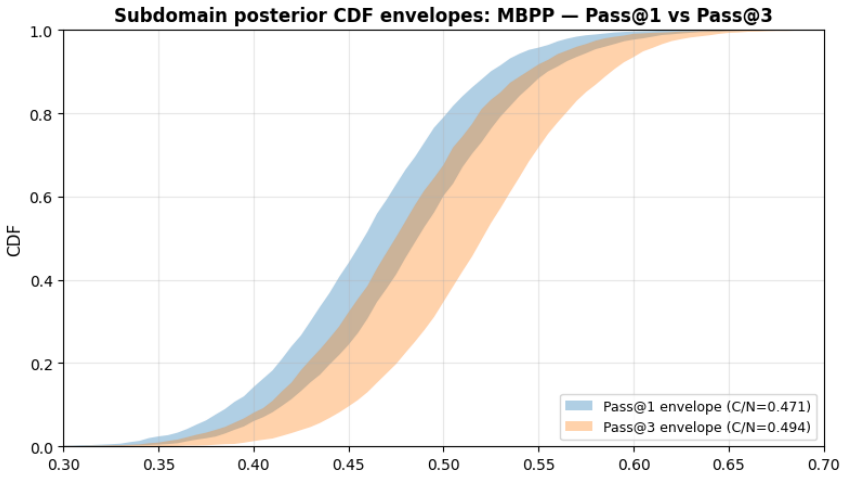


Figure 9: Subdomain-level posterior CDF envelopes of non-failure probability for MBPP under alternative success definitions. Results are shown for Claude Sonnet 4.5 on MBPP ($N = 257$), using identical tasks, prompts, and tests. Pass@1 yields $\frac{C}{N} = 0.471$, while Pass@3 yields $\frac{C}{N} = 0.494$. Envelopes reflect imprecise hierarchical inference with identical hyper-hyperparameter intervals; only the success definition differs. A question-oriented interpretation of figure: How does the inferred subdomain-level non-failure probability change when alternative success (failure) definitions, such as Pass@1 versus Pass@3, are used on the same evaluation data?

increases approximately linearly with the retained context size. This motivates a parsimonious sensitivity model in which the dependence parameter is a linear function of memory. The linear mapping is used only as a stress-test mechanism to quantify how departures from independence could propagate into the inferred reliability distributions; it does not change the definition of reliability itself, but probes the robustness of posterior conclusions when the i.i.d. approximation is imperfect.

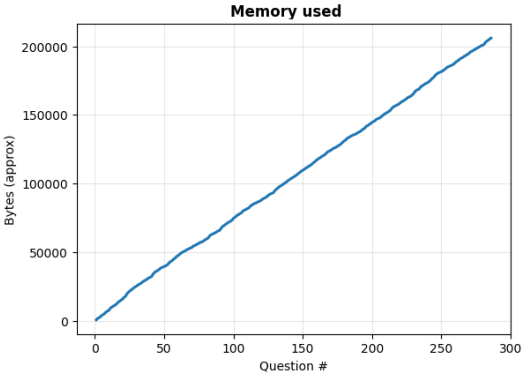
We then injected memory-induced dependence into BoolQ according to the linear dependence model and compared posterior CDF envelopes under three BoolQ θ settings,

$$\theta_{\text{BoolQ}} \in \{0.915,\ 0.940,\ 0.945\},$$

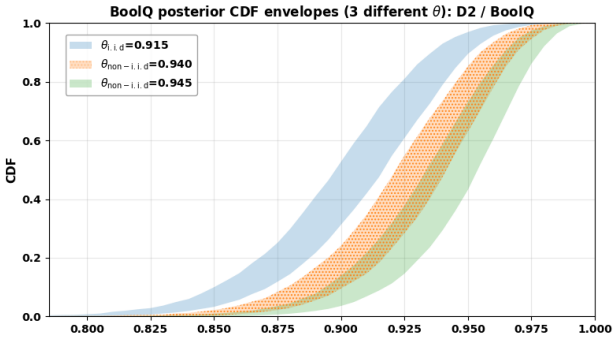
where $\theta_{\text{BoolQ}} = 0.915$ corresponds to the *i.i.d. assumption* (no memory-induced dependence), while $\theta_{\text{BoolQ}} = 0.940$ and 0.945 represent non-i.i.d. regimes induced by increasing retained context. The remaining subdomain accuracies (MBPP, DS-1000, RACE-H) were kept fixed, and the same hierarchical structure and operational-profile aggregation were preserved (subdomain $\rightarrow$ domain via Ω_{ij}, and domain $\rightarrow$ LLM via W_i). For each θ_{BoolQ}, we propagated uncertainty through the hierarchy under the same imprecise hyperparameter intervals and computed CDF envelopes at three levels: (i) BoolQ subdomain, (ii) Reasoning domain ($D2$), and (iii) overall LLM level.

Across all three levels, the envelopes shift smoothly with θ_{BoolQ}: increasing θ_{BoolQ} yields a consistent rightward shift (stochastically larger reliability), while decreasing θ_{BoolQ} shifts the envelopes leftward.

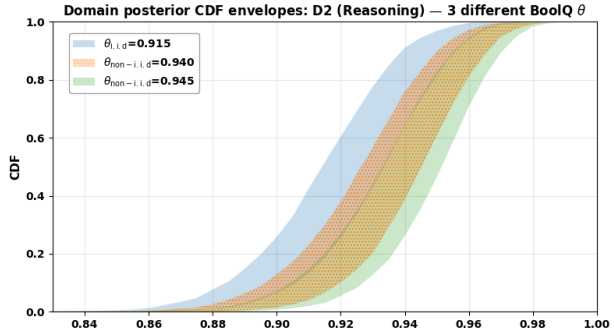
These experiments support two practical conclusions: (1) persistent context can induce measurable dependence during benchmark-style evaluation, and the effect scales with memory; and (2) within a realistic range around the observed reliability levels, our posterior



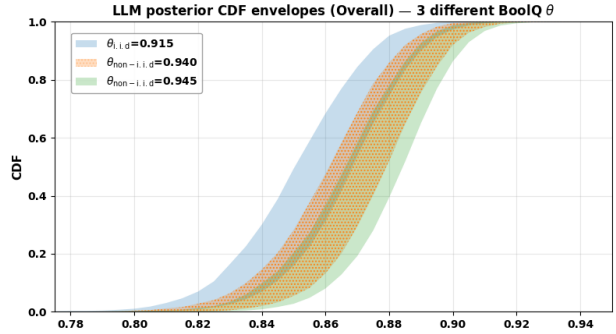
(a) Growth of retained conversational context as a function of the number of BoolQ tasks evaluated in a single continuous session. Memory size (in bytes) increases monotonically and serves as an operational proxy for the strength of memory-induced dependence.



(b) Posterior CDF envelopes of non-failure probability for the BoolQ subdomain under three accuracy settings: $\theta_{\text{BoolQ}} = 0.915$ (i.i.d.), and $\theta_{\text{BoolQ}} = 0.940, 0.945$ (non-i.i.d.). Increasing memory-induced dependence produces smooth rightward shifts of the envelope.



(c) Posterior CDF envelopes of non-failure probability at the Reasoning domain level (Dom2) resulting from propagation of BoolQ dependence through subdomain aggregation with weights Ω_{ij}.



(d) Posterior CDF envelopes of non-failure probability at the LLM level after domain aggregation with operational-profile weights W_i, illustrating the attenuated but consistent impact of subdomain-level dependence.

Figure 10: Sensitivity of posterior CDF envelopes of non-failure probability to memory-induced non-i.i.d. dependence. The top-left panel (10a) quantifies memory accumulation during same-session evaluation, motivating a linear dependence proxy. The remaining panels show how increasing dependence in the BoolQ subdomain propagates through the hierarchical reliability model, yielding smooth and controlled shifts of envelopes at subdomain, domain, and LLM levels.

envelope conclusions remain stable, with dependence primarily producing controlled shifts rather than qualitative changes in domain- and LLM-level reliability characterization.

4.3.8. ***RQ8*** *(Scalability)*

This section characterizes the computational cost of the proposed HIP-LLM inference pipeline. The analysis is performed at the level of the algorithmic procedure (rather than for any single experimental figure) and is structured to (i) derive theoretical time and memory scaling laws from the inference steps, and (ii) empirically validate these scaling laws by controlled parameter sweeps.

Let m denote the number of domains and $\bar{n}$ the average number of subdomains per domain[24]. For each domain i, HIP-LLM represents hyperparameters on a discretized grid over (μ_i, ν_i) of size $G = n_\mu n_\nu$. Imprecision at the hyperparameter level is handled by evaluating a finite set of K hyperparameter configurations (or candidates) over the grid. To propagate uncertainty from subdomain to domain and from domain to system level, the method uses Monte Carlo sampling with S samples per configuration. Posterior envelopes are reported on a fixed evaluation grid (e.g., a CDF grid) of size T.

The inference is decomposed into three hierarchical stages:

1. **Subdomain level:** For each domain and each hyperparameter configuration, HIP-LLM evaluates subdomain posteriors conditional on (μ_i, ν_i) on the grid and constructs the corresponding imprecise posterior set.
2. **Domain level:** For each configuration, HIP-LLM propagates subdomain uncertainty to the domain level via Monte Carlo sampling by drawing subdomain reliability variables and aggregating them using OP weights Ω_{ij}.
3. **LLM level:** HIP-LLM aggregates domain-level reliability to the system level using domain weights W_i, while preserving imprecision by computing lower/upper envelopes across the explored hyperparameter configurations.

Theoretical time complexity is obtained by counting the dominant operations required by the above stages as functions of $(m, \bar{n}, G, K, S, T)$. Subdomain-level cost is driven by grid-based hyperposterior evaluation and any intra-domain coupling required to compute subdomain posteriors under shared hyperparameters. Domain-level cost is driven by Monte Carlo propagation, which scales with the number of subdomains drawn per sample and the number of samples. LLM-level cost is driven by repeating aggregation across hyperparameter configurations and computing envelope statistics on the evaluation grid. Memory complexity is characterized by the storage required for (i) grid-based posterior arrays per domain/configuration, (ii) Monte Carlo samples (if stored) or streaming statistics (if not stored), and (iii) envelope representations on the evaluation grid.

To validate the theoretical scaling, we use controlled experiments in which all numerical parameters are fixed to baseline values and then varied one at a time. Specifically, we run repeated executions while sweeping m, $\bar{n}$, K, S, and G independently, and record wall-clock

[24]Although different domains may have different numbers of subdomains, the computational cost is mainly determined by the typical workload of a domain. For this reason, we describe the scaling in terms of the average number of subdomains $\bar{n}$. This keeps the formulas simple, reflects the observed runtime in practice, and avoids overly pessimistic worst-case estimates.

time and peak memory usage for each run. This isolates the marginal effect of each parameter and enables direct comparison with the predicted scaling laws. To quantify scaling exponents, we fit a power-law model of the form time $= c\,x^{\alpha}$ (and analogously for memory), where x is the swept parameter[25] and α[26] is the inferred scaling exponent. Here c is a constant which sets the scale.

Many steps of HIP-LLM can be computed independently across domains and hyperparameter configurations, making the method naturally parallelizable. Parallel execution reduces wall-clock runtime in practice, while the underlying asymptotic computational complexity remains unchanged.

The following experiment empirically validates the theoretical computational complexity analysis of the HIP-LLM inference pipeline. We report controlled scalability experiments that measure wall-clock runtime and peak memory usage while varying key algorithmic parameters one at a time. The results are used to confirm the predicted asymptotic scaling laws[27] and to identify the dominant computational stages in practice.

Scalability plots. (Fig. 11a- Fig. 11e) showing runtime as a function of each swept parameter (m, $\bar{n}$, K, S, and G), together with fitted scaling laws. Across all sweeps, the empirical results closely match the theoretical scaling predictions and confirm that subdomain-level inference dominates the computational cost of HIP-LLM in practice.

Baseline timing breakdown. (Fig. 11f) reporting the relative contribution of each inference stage under the reference configuration. The results show that subdomain-level posterior computation dominates runtime, accounting for over 99% of total execution time, while domain- and LLM-level aggregation contributes a negligible fraction. This explains the weak empirical dependence on S and highlights the subdomain inference stage as the primary computational bottleneck.

Memory Complexity. According to Tab. 6 empirical results confirm the expected memory behavior of HIP-LLM. Peak memory usage scales approximately linearly with the number of domains m, the average number of subdomains $\bar{n}$, the number of hyperparameter configurations K, and the number of Monte Carlo samples S, due to the storage of posterior quantities and sampled reliabilities. In contrast, memory usage is effectively independent of the integration grid size G, indicating that the (μ, ν) grid is reused across configurations rather than stored per sample.

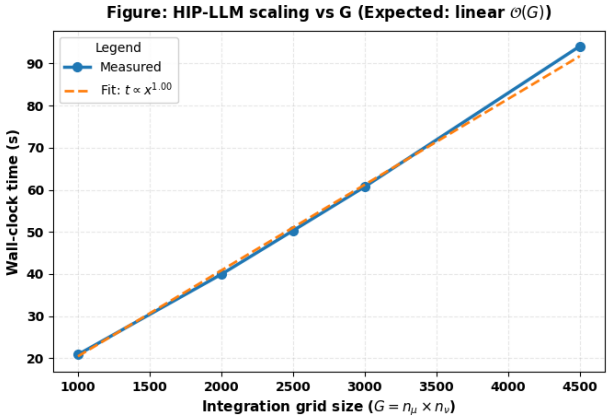
5. Discussion

In this section, we discuss the assumptions and limitations of the proposed HIP-LLM.

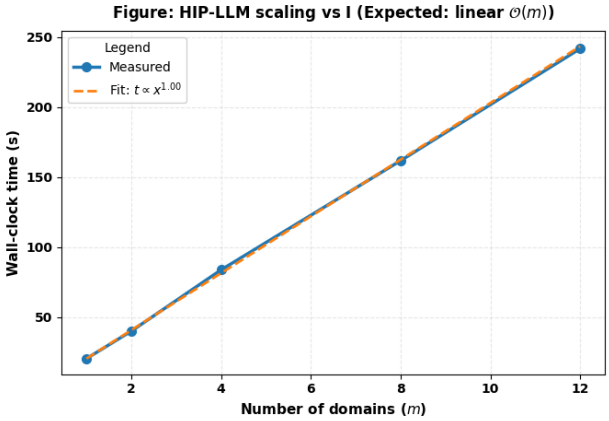
[25] The parameter we change (e.g. number of subdomains, samples, domains, etc.)

[26] The parameter that shows how fast time grows when x increases. for instance, $\alpha = 1$ corresponds to linear growth, while $\alpha = 2$ corresponds to quadratic growth, and so on.

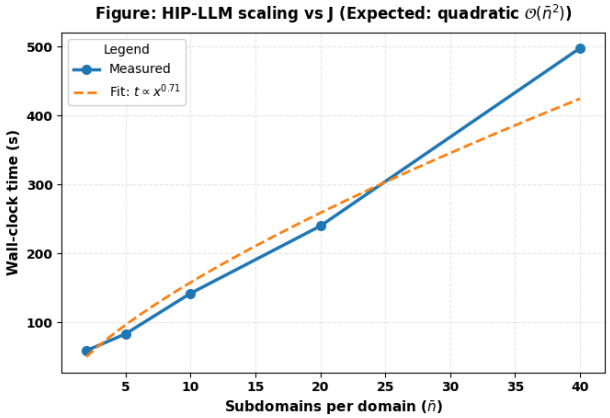
[27] Do the measured runtimes grow the way the theory says they should?



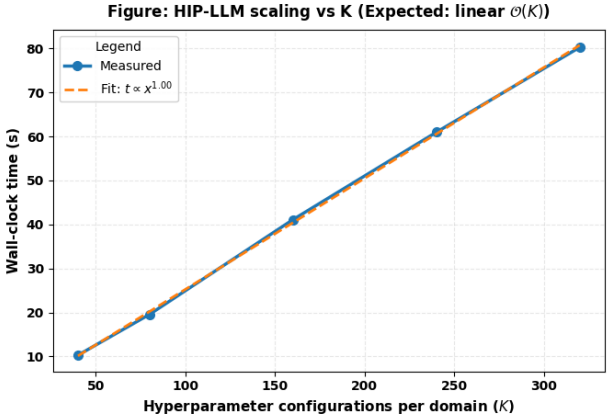
(a) Scaling with the (μ, ν) integration grid size $G = n_\mu n_\nu$, showing linear growth in wall-clock time, consistent with the theoretical $\mathcal{O}(G)$ complexity.



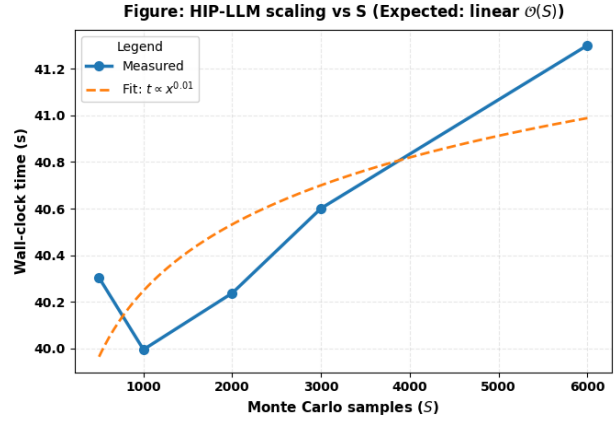
(b) Scaling with the number of domains m, demonstrating linear runtime growth due to independent per-domain inference, in agreement with the expected $\mathcal{O}(m)$ behavior.



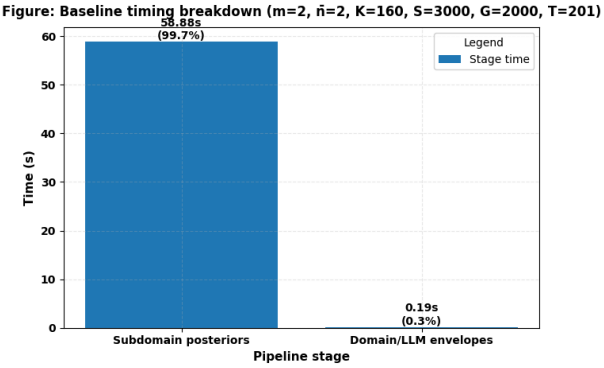
(c) Scaling with the average number of subdomains per domain $\bar{n}$, exhibiting sub-quadratic growth over the tested range, reflecting practical efficiencies relative to the worst-case $\mathcal{O}(\bar{n}^2)$ bound.



(d) Scaling with the number of hyperparameter configurations per domain K, confirming linear complexity due to independent evaluation across configurations.



(e) Scaling with the number of Monte Carlo samples S, showing weak dependence on S and indicating that Monte Carlo propagation is not the dominant computational cost in the tested regime.



(f) Baseline timing breakdown for the full HIP-LLM pipeline, showing that subdomain posterior computation dominates the total runtime, while domain- and LLM-level envelope aggregation contributes a negligible fraction.

Figure 11: Empirical scalability of the HIP-LLM inference pipeline under controlled parameter sweeps. Each subfigure varies a single parameter while keeping all others fixed at the baseline configuration ($m = 2$, $\bar{n} = 2$, $K = 160$, $S = 3000$, $G = 2000$, $T = 201$, $K_{\text{total}} \leq 512$). Measured runtimes are compared against power-law fits to validate theoretical complexity predictions and to identify dominant computational stages.

Table 6: Peak memory usage under controlled scalability experiments. Each row reports the measured peak memory while sweeping one parameter and keeping all others fixed at the baseline configuration.

Swept parameter	Range	Peak memory (MB)
Number of domains m	$1 \to 12$	$24.6 \to 146.7$
Subdomains per domain $\bar{n}$	$2 \to 40$	$35.8 \to 314.2$
Monte Carlo samples S	$500 \to 6000$	$7.6 \to 69.4$
Hyperparameter configs K	$40 \to 320$	$19.1 \to 57.8$
Grid size G	$1000 \to 4500$	≈ 35.7 (constant)

Assumptions on hierarchical dependencies of failure probabilities. Inspired by HiBayES [31], we introduce a hierarchical dependency structure where dependencies are defined at the level of failure probabilities across related task groups. Specifically, high-level domains (e.g., coding vs. law) are modeled as independent because their underlying competencies and training data sources are substantially distinct. In contrast, subdomains within a domain (e.g., coding in C++ vs. coding in Python) are modeled as having dependent failure probabilities, reflecting shared latent skills, representations, or reasoning patterns that influence performance across related tasks. This abstraction allows tractable modeling of correlated reliability while acknowledging that the true dependencies among knowledge areas are more complex and fluid. While we do not prescribe specific taxonomies of domain/sub-domains for LLMs in this paper, our formulation is intended to be without loss of generality: users of HIP-LLM can redefine their own "domains" to reflect their own assumptions about dependence. For example, if users believe that two domains (say, A and B) are not fully independent, they may treat them as two subdomains under a newly introduced higher-level synthetic domain. Such flexibility allows HIP-LLM to adapt to different operational interpretations or empirical evidence of dependency, reflecting the assessor's domain knowledge.

Note, such (in)dependencies at the parameter level (i.e., correlations or independence between the failure probabilities of domains or subdomains) are different from task-level outcome dependencies, which we discuss next.

Assumptions of i.i.d. Bernoulli Trails for Task Outcomes. It is important to distinguish the assumption of independent task outcomes from the (in)dependencies among failure probabilities discussed earlier. The latter concern parameter-level relationships, i.e., how the long-run failure rates across domains or subdomains may co-vary due to shared competencies, whereas the former concerns instance-level independence of individual task results, given those parameters. Even if the failure probabilities of two sub-domains are correlated random variables, we can still assume that, conditional on those probabilities, each task outcome within a sub-domain is an independent Bernoulli trial.

As discussed earlier in Def. 4 and Remark 1, the i.i.d. assumption is pragmatically justified in the context of LLM evaluations when tasks are executed in isolation, such as *initiating new chat sessions without contextual memory*. Nevertheless, caution is warranted: in practice, the way tasks are designed, sequenced, or batched can introduce subtle dependencies that violate this idealization, e.g., in long-context, agentic workflows. Assessors should therefore ensure testing procedures (e.g., resetting each task session) align with the independence assumption, a common setup in offline benchmarking.

Indeed, the i.i.d. Bernoulli trial assumption remains a simplification, motivated by mathematical convenience and interpretability. However, this assumption is not introduced uniquely by our HIP-LLM, rather that models with the i.i.d. assumption have a long history of being used in software reliability assessment. Thayer *et al.*'s [79] model was one of the earliest, used in early works on random testing [80]. Regulatory bodies also have recommended using the i.i.d. assumption in reliability assessments when appropriate [40]. Having said that, weakening the i.i.d. assumption has been a long-standing research topic in the software reliability community. Numerous models have been proposed to capture dependence among successive tests, including a binary Markov chain model [81], a Markov renewal process [82, 83], the Markov-based model with benign-failure states [84], and Bayesian models examining how sensitive reliability claims are to potential dependencies of test outcomes [71, 85].

A road-map for extending HIP-LLM beyond i.i.d. task outcomes can build directly on aforementioned, established software reliability models. More concretely, a first step is treating a multi-turn interaction as the unit of analysis and allowing within-session dependence while modeling across-session dynamics via latent states. One can introduce Markov/hidden-state structures, e.g., hierarchical hidden Markov model or Markov renewal formulations where latent "interaction states" evolve over time and govern task-level failure probabilities.

What constitutes a failure for LLMs. Reliability, by definition, concerns "failures". In traditional software systems, failures are relatively straightforward to define, as they can be explicitly linked to violations of formal specifications or requirements. However, for AI systems, and particularly for LLMs, no such explicit specification exists. As a result, a complete and formal characterization of what constitutes a failure for LLMs remains an open research challenge. The notion of "failure" can vary across applications and stakeholders. For instance, factual errors, inconsistency[28], harmful or biased content, or divergence from human judgment may all be considered failures depending on the context [30].

The choice of scoring criteria for defining LLM failures is inherently application dependent and should align with domain-specific specifications (if any) and evaluation norms. For example, in code generation, failures are typically defined in terms of functional correctness, such as compilation success or passing predefined test cases. In mathematical problem solving, correctness is often determined by agreement with a unique ground-truth answer or a formally verifiable solution. In contrast, in legal analysis, failure definitions may rely on expert judgment regarding factual accuracy, consistency with legal principles, or omission of critical considerations. For more open-ended tasks such as email drafting or creative writing, failures are often defined relative to stylistic, pragmatic, or preference-based criteria, frequently assessed through human evaluation. These examples illustrate that failure definitions for LLMs are not universal but must be specified in accordance with the intended use and domain requirements, reinforcing the need to interpret reliability estimates as conditional on the chosen scoring criteria.

That said, in this work, our focus is not on prescribing a specific definition of LLM

[28]Most existing LLM evaluation studies focus on one-shot accuracy metrics (e.g., Pass@1), whereas real-world usage often involves iterative prompting. Inconsistencies across multiple outputs can undermine user trust and thus regarded as a failure.

failure but on *providing a reliability assessment modeling framework that is agnostic to failure definitions and failure detections (i.e., defining the test oracle).* For example, the reliability definition in Def. 4 can be readily extended from binary failures to non-binary scores by replacing the indicator function with a continuous scoring function (e.g., taking values in $[0, 1]$ through probabilistic or multi-rater failure models), thereby capturing scoring noise or subjectivity (commonly referred to as the imperfect test oracle problem in traditional software reliability). HIP-LLM can accommodate any well-defined notion of failure, enabling users to instantiate the model with the criteria most relevant to their domain or evaluation objectives.

How can OPs be specified in practice. OPs are fundamental to software reliability assessment, and the same principle applies to LLM reliability. HIP-LLM provides explicit hierarchical "interfaces" for integrating OPs; however, in the absence of access to real operational data, this study relies on simulated OPs derived from existing benchmark datasets. Specifically, benchmark datasets are treated as sampling frames that approximate the unknown operational data distribution (cf. Remark 7), with task-level distributions assumed within each subdomain and operational weights assigned across subdomains and domains. This experimental choice is made solely to demonstrate how HIP-LLM incorporates OPs and to enable fair comparison with existing benchmark-based methods, and *should not be interpreted as a solution to OP acquisition.* Accurately estimating OPs from real usage data is beyond the scope of this reliability modeling study and is a well-recognized, separate research problem in software reliability engineering, for which numerous methodologies have been proposed (e.g., [25, 26, 28]). The concept of OP has also been discussed and measured in AI software contexts [24, 76, 86, 87].

Importantly, the challenge of modeling LLM OPs does not undermine the proposed framework; rather, it highlights why OPs are essential to reliability assessment. In established software reliability practice, OPs are progressively refined as new information becomes available (e.g., execution logs, usage telemetry, or expert-informed adjustments) without requiring changes to the underlying reliability model [25, 26, 28, 77]. As more representative datasets, user interaction logs, or task-distribution estimates for LLMs become available [27], OPs can be updated accordingly, allowing reliability estimates to be recalibrated toward actual field usage. Such separation and "decoupling design" of HIP-LLM between evidence (testing results) and usage assumptions (OPs) enables HIP-LLM to evolve naturally with deployment conditions, improving fidelity without *redesigning* the model.

While we acknowledge that our present treatment of OP uncertainty remains simplified (for demonstration/comparison), more complex modeling on OP uncertainties, e.g., dynamic OPs, will form our important future works. Notably, a range of reliability models in traditional software engineering explicitly account for OP uncertainty and dynamics, including Bayesian updating, adaptive OP adjustment, and black-box or time-varying usage models [72, 26, 73, 74, 75]. These established techniques provide a foundation that can be retrofitted into HIP-LLM to support uncertain and dynamical OPs in LLM deployments.

How to embed prior knowledge. HIP-LLM models uncertainty at three levels: the subdomain success rate (θ_{ij}), how much domains share information (μ_i, ν_i), and the top-level parameters (a_i, b_i, c_i, d_i) that control the importance of those priors.

As a Bayesian framework, HIP-LLM requires priors to initiate inference, and these priors may substantially influence the resulting posteriors, particularly when data are limited [88, 89]. The challenge of eliciting meaningful priors is *not unique* to HIP-LLM but is common to all Bayesian models. Unlike HiBayES [31], which employs noninformative priors, HIP-LLM follows the principles of robust Bayesian analysis [59, 60, 90] by adopting the Imprecise Probability framework. This approach enables the representation of uncertainty about prior beliefs through sets of priors rather than committing to a single distribution. Prior research has shown that Imprecise Probability can facilitate expert elicitation by supporting structured "thought experiments" [65, 91, 92] allowing experts to express vague, partial, or interval-valued prior beliefs. Similarly, HIP-LLM's prior layer can be informed by such thought experiments and expert judgment. While we acknowledge that eliciting priors in practice remains challenging, HIP-LLM at least offers a practical "what-if" analysis tool by allowing assessors to explore how different prior assumptions might influence posterior reliability claims, for example: if my prior belief is A, then after observing evidence B, the posterior reliability would be C.

On the practicality of HIP-LLM. HIP-LLM is designed to replace current LLM assessment practices that rely primarily on benchmark scores. As highlighted by the 5 gaps identified in the introduction, benchmark-based evaluations often ignore OPs, overlook dependencies between related domains, and cannot incorporate prior knowledge, limiting their relevance to real-world use. In contrast, HIP-LLM produces decision-oriented reliability assessments by integrating OPs, hierarchical domain dependencies, and epistemic uncertainty and priors. Rather than reporting isolated scores (which, strictly speaking, cannot be regarded as "reliability" under standardized definitions such as ANSI [38]), HIP-LLM enables rigorous reliability claims such as "*the confidence that a model will complete a specified number of future tasks without failure under a given usage profile*". This allows stakeholders to compare LLMs within specific domains, aggregate reliability across domains, or assess how reliability changes across different user groups or usage patterns. Such capabilities support informed offline decision-making, including model selection, risk assessment, and regulatory certification prior to deployment. While this work focuses on offline assessment, HIP-LLM could potentially be extended to online reliability monitoring with appropriate engineering optimizations for efficiency (e.g., caching and sliding data windows), which we leave for future work.

6. Conclusion

This paper presents HIP-LLM, a hierarchical Bayesian framework with imprecise probability for evaluating the reliability of LLMs. While HIP-LLM acknowledges the general-purpose nature of contemporary LLMs, it also adheres to the standardized definition of software reliability which inherently depends on specific applications and operational contexts. We reconcile these seemingly conflicting aspects by introducing a multi-level hierarchical structure of evaluation tasks that explicitly models dependencies among subdomains, integrates information through partial pooling, and incorporates OPs at multiple levels of abstraction. Moreover, as a Bayesian inference method, HIP-LLM respects the difficulty of selecting priors. Instead of simply using non-informative priors, HIP-LLM provides the mechanism to

embed imprecise prior knowledge and reports posterior envelopes (rather than single distributions). Our experiments on common LLMs demonstrate that these features enable a more nuanced and standardized reliability estimation compared to existing benchmarks and state-of-the-art methods.

While we believe HIP-LLM represents an important first step toward a principled reliability assessment framework for LLMs (and more broadly, for multimodal generative AI) several key limitations remain (cf. Section 5). These include the challenge of formally defining what constitutes a failure for an LLM, accurately estimating OPs, and systematically eliciting imprecise prior knowledge. Moreover, the applicability of HIP-LLM is limited to reset or single-task usage scenarios conforming to the i.i.d. assumption (cf. Remark 1); and should not be overgeneralized to long-context or agentic LLM workflows (unless such workflows are modeled at a higher level of abstraction, treating the entire interaction trajectory as a single trial), which require dependency-aware reliability models—an important future work. More future extensions should aim to develop methods for handling uncertain and dynamically evolving OPs, and integrating more refined mechanisms for prior elicitation.

Acknowledgment

SK and XZ have received funding from the European Union's EU Framework Program for Research and Innovation Europe Horizon (grant agreement No 101202457). XZ's contribution is also supported by the UK EPSRC New Investigator Award [EP/Z536568/1]. SK's contribution is supported by the UKRI Future Leaders Fellowship Grant [MR/S035176/1].

Views and opinions expressed are those of the authors only and do not necessarily reflect those of the European Union or European Research Executive Agency (REA). Neither the European Union nor the granting authority can be held responsible for them.

References

[1] Z. Pang, Y. Luan, J. Chen, T. Li, Parinfogpt: An LLM-based two-stage framework for reliability assessment of rotating machine under partial information, Reliability Engineering & System Safety 250 (2024) 110312.

[2] X. Xiao, P. Chen, B. Qi, H. Zhao, J. Liang, J. Tong, H. Wang, Krail: A knowledge-driven framework for human reliability analysis integrating idheas-data and large language models, Reliability Engineering & System Safety (2025) 111585.

[3] S. Zheng, K. Pan, J. Liu, Y. Chen, Empirical study on fine-tuning pre-trained large language models for fault diagnosis of complex systems, Reliability Engineering & System Safety 252 (2024) 110382.

[4] U. Gohar, M. C. Hunter, R. R. Lutz, M. B. Cohen, Codefeater: Using llms to find defeaters in assurance cases, in: proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering, 2024, pp. 2262–2267.

[5] A. Murugesan, I. Wong, J. Arias, R. Stroud, S. Varadarajan, E. Salazar, G. Gupta, R. Bloomfield, J. Rushby, Automating semantic analysis of system assurance cases using goal-directed asp, Theory and Practice of Logic Programming 24 (4) (2024) 805–824.

[6] B. Sultan, L. Apvrille, AI-driven consistency of SysML diagrams, in: proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems, 2024, pp. 149–159.

[7] J. Yang, A. Prabhakar, K. Narasimhan, S. Yao, Intercode: Standardizing and benchmarking interactive coding with execution feedback, Advances in Neural Information Processing Systems 36 (2023) 23826–23854.

[8] Y. Lai, C. Li, Y. Wang, T. Zhang, R. Zhong, L. Zettlemoyer, W.-t. Yih, D. Fried, S. Wang, T. Yu, DS-1000: A natural and reliable benchmark for data science code generation, in: International Conference on Machine Learning, PMLR, 2023, pp. 18319–18345.

[9] Z. Yu, Y. Zhao, A. Cohan, X.-P. Zhang, HumanEval pro and MBPP pro: Evaluating Large Language Models on self-invoking code generation, arXiv preprint arXiv:2412.21199 (2024).

[10] E. Croxford, Y. Gao, N. Pellegrino, K. Wong, G. Wills, E. First, F. Liao, C. Goswami, B. Patterson, M. Afshar, Current and future state of evaluation of large language models for medical summarization tasks, Npj health systems 2 (1) (2025) 6.

[11] S. Charalampidou, A. Zeleskidis, I. M. Dokas, Hazard analysis in the era of AI: Assessing the usefulness of ChatGPT4 in stpa hazard analysis, Safety Science 178 (2024) 106608.

[12] G. K. Kaya, D. Bovell, M. Sujan, G. Braithwaite, Large language models powered system safety assessment: applying STPA and FRAM, Safety Science 191 (2025) 106960.

[13] Y. Qi, X. Zhao, S. Khastgir, X. Huang, Safety analysis in the era of large language models: a case study of STPA using ChatGPT, Machine Learning with Applications 19 (2025) 100622.

[14] Z. Zhang, L. Lei, L. Wu, R. Sun, Y. Huang, C. Long, X. Liu, X. Lei, J. Tang, M. Huang, SafetyBench: Evaluating the safety of large language models, in: proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics, ACL, 2024, pp. 15537–15553.

[15] X. Liu, Y. Zhu, J. Gu, Y. Lan, C. Yang, Y. Qiao, Mm-safetybench: A benchmark for safety evaluation of multimodal large language models, in: European Conference on Computer Vision, Springer, 2024, pp. 386–403.

[16] Y. Mou, S. Zhang, W. Ye, Sg-bench: Evaluating llm safety generalization across diverse tasks and prompt types, Advances in Neural Information Processing Systems 37 (2024) 123032–123054.

[17] Center for AI Standards and Innovation (CAISI), Evaluation of deepseek AI models, Report, National Institute of Standards and Technology (2025).

[18] T. Yu, Y. Jing, X. Zhang, W. Jiang, W. Wu, Y. Wang, W. Hu, B. Du, D. Tao, Benchmarking reasoning robustness in large language models, arXiv preprint arXiv:2503.04550 (2025).

[19] S. Wang, P. Wang, T. Zhou, Y. Dong, Z. Tan, J. Li, Ceb: Compositional evaluation benchmark for fairness in Large Language Models, in: International Conference on Representation Learning, Vol. 2025, 2025, pp. 22627–22668.

[20] Q. Li, J. Hong, C. Xie, J. Tan, R. Xin, J. Hou, X. Yin, Z. Wang, D. Hendrycks, Z. Wang, B. Li, B. He, D. Song, LLM-PBE: Assessing data privacy in Large Language Models, Proceedings of the VLDB Endow. 17 (11) (2024) 3201–3214.

[21] A. Tamkin, M. McCain, K. Handa, E. Durmus, L. Lovitt, A. Rathi, S. Huang, A. Mountfield, J. Hong, S. Ritchie, et al., Clio: Privacy-preserving insights into real-world AI use, arXiv preprint arXiv:2412.13678 (2024).

[22] B. Littlewood, L. Strigini, Software reliability and dependability: a roadmap, in: Proceedings of the Conference on the Future of Software Engineering, 2000, pp. 175–188.

[23] L. Strigini, B. Littlewood, Guidelines for statistical testing, Project Report PASCON/WO6-CCN2/TN12, City University London (1997).

[24] Y. Dong, W. Huang, V. Bharti, V. Cox, A. Banks, S. Wang, X. Zhao, S. Schewe, X. Huang, Reliability assessment and safety arguments for machine learning components in system assurance, ACM Transactions on Embedded Computing Systems 22 (3) (2023) 1–48.

[25] J. Musa, Operational profiles in software-reliability engineering, IEEE Software 10 (2) (1993) 14–32.

[26] J. D. Musa, Adjusting measured field failure intensity for operational profile variation, in: Proceedings of IEEE Int. Symposium on Software Reliability Engineering, 1994, pp. 330–333.

[27] A. Chatterji, T. Cunningham, D. J. Deming, Z. Hitzig, C. Ong, C. Y. Shan, K. Wadman, How people use chatgpt, Tech. rep., National Bureau of Economic Research (2025).

[28] M. R. Lyu, et al., Handbook of software reliability engineering, Vol. 222, IEEE computer society press Los Alamitos, 1996.

[29] B. Littlewood, L. Strigini, Validation of ultra-high dependability for software-based systems, Comm. of the ACM 36 (1993) 69–80.

[30] Y. Chang, X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, H. Chen, X. Yi, C. Wang, Y. Wang, et al., A survey on evaluation of large language models, ACM transactions on intelligent systems and technology 15 (3) (2024) 1–45.

[31] L. Luettgau, H. Coppock, M. Dubois, C. Summerfield, C. Ududec, Hibayes: A Hierarchical Bayesian Modeling Framework for AI Evaluation Statistics, arXiv preprint arXiv:2505.05602 (2025).

[32] E. Miller, Adding error bars to evals: A statistical approach to language model evaluations, arXiv preprint arXiv:2411.00640 (2024).

[33] L. Strigini, A. Povyakalo, Software fault-freeness and reliability predictions, in: Computer Safety, Reliability, and Security, Vol. 8153 of LNCS, Springer Berlin Heidelberg, 2013, pp. 106–117.

[34] P. Bishop, A. Povyakalo, L. Strigini, Bootstrapping confidence in future safety from past safe operation, in: IEEE 33rd Int.Symp. on Software Reliability Engineering, IEEE, 2022, pp. 97–108.

[35] X. Zhao, A. Banks, J. Sharp, V. Robu, D. Flynn, M. Fisher, X. Huang, A safety framework for critical systems utilising deep neural networks, in: International Conference on Computer Safety, Reliability, and Security, Springer, 2020, pp. 244–259.

[36] T. Augustin, F. P. Coolen, G. De Cooman, M. C. Troffaes, Introduction to imprecise probabilities, John Wiley & Sons, 2014.

[37] M. C. Troffaes, Decision making under uncertainty using imprecise probabilities, International journal of approximate reasoning 45 (1) (2007) 17–29.

[38] ANSI/IEEE, Standard glossary of software engineering terminology, Tech. rep., STD-729-1991 (1991).

[39] IEC, IEC61508, Functional Safety of Electrical/ Electronic/Programmable Electronic Safety Related Systems, 2010.

[40] C. Atwood, J. LaChance, H. Martz, D. Anderson, M. Englehardt, D. Whitehead, T. Wheeler, Handbook of parameter estimation for probabilistic risk assessment, Report NUREG/CR-6823, U.S. Nuclear Regulatory Commission, Washington, DC (2003).

[41] B. Littlewood, J. Rushby, Reasoning about the reliability of diverse two-channel systems in which one channel is 'possibly perfect', IEEE Tran. on Software Engineering 38 (5) (2012) 1178–1194.

[42] X. Zhao, B. Littlewood, A. Povyakalo, L. Strigini, D. Wright, Modeling the probability of failure on demand (pfd) of a 1-out-of-2 system in which one channel is "quasi-perfect", Reliability Engineering & System Safety 158 (2017) 230–245.

[43] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, J. Steinhardt, Measuring massive multitask language understanding, arXiv preprint arXiv:2009.03300 (2020).

[44] G. Lai, Q. Xie, H. Liu, Y. Yang, E. Hovy, Race: Large-scale reading comprehension dataset from examinations, arXiv preprint arXiv:1704.04683 (2017).

[45] G. Mialon, C. Fourrier, T. Wolf, Y. LeCun, T. Scialom, Gaia: a benchmark for general ai assistants, in: The Twelfth International Conference on Learning Representations, 2023.

[46] D. Rein, J. Becker, A. Deng, S. Nix, C. Canal, D. O'Connel, P. Arnott, R. Bloom, T. Broadley, K. Garcia, et al., HCAST: Human-calibrated autonomy software tasks, arXiv preprint arXiv:2503.17354 (2025).

[47] A. Anthropic, The claude 3 model family: Opus, sonnet, haiku, Claude-3 Model Card 1 (1) (2024) 4.

[48] A. Hurst, A. Lerer, A. P. Goucher, A. Perelman, A. Ramesh, A. Clark, A. Ostrow, A. Welihinda, A. Hayes, A. Radford, et al., Gpt-4o system card, arXiv preprint arXiv:2410.21276 (2024).

[49] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. D. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, et al., Evaluating large language models trained on code, arXiv preprint arXiv:2107.03374 (2021).

[50] P. Liang, R. Bommasani, T. Lee, D. Tsipras, D. Soylu, M. Yasunaga, Y. Zhang, D. Narayanan, Y. Wu, A. Kumar, et al., Holistic evaluation of language models, arXiv preprint arXiv:2211.09110 (2022).

[51] F. Shi, M. Suzgun, M. Freitag, X. Wang, S. Srivats, S. Vosoughi, H. W. Chung, Y. Tay, S. Ruder, D. Zhou, et al., Language models are multilingual chain-of-thought reasoners, arXiv preprint arXiv:2210.03057 (2022).

[52] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, Y. Cao, React: Synergizing reasoning and acting in language models, in: International Conference on Learning Representations (ICLR), 2023.

[53] Y. Liu, Y. Yao, J.-F. Ton, X. Zhang, R. Guo, H. Cheng, Y. Klochkov, M. F. Taufiq, H. Li, Trustworthy llms: a survey and guideline for evaluating large language models' alignment, arXiv preprint arXiv:2308.05374 (2023).

[54] Y. Saleh, M. Abu Talib, Q. Nasir, F. Dakalbab, Evaluating large language models: a systematic review of efficiency, applications, and future directions, Frontiers in Computer Science 7 (2025) 1523699.

[55] S. Liu, C. Li, J. Qiu, X. Zhang, F. Huang, L. Zhang, Y. Hei, P. S. Yu, The Scales of Justitia: A Comprehensive Survey on Safety Evaluation of LLMs, arXiv preprint arXiv:2506.11094 (2025).

[56] H. Ye, J. Jin, Y. Xie, X. Zhang, G. Song, Large language model psychometrics: A systematic review of evaluation, validation, and enhancement, arXiv preprint arXiv:2505.08245 (2025).

[57] V. Quach, A. Fisch, T. Schuster, A. Yala, J. H. Sohn, T. S. Jaakkola, R. Barzilay, Conformal Language Modeling, in: The 12th International Conference on Learning Representations (ICLR'24).

[58] Z. Wang, J. Duan, L. Cheng, Y. Zhang, Q. Wang, X. Shi, K. Xu, H. T. Shen, X. Zhu, ConU: Conformal Uncertainty in Large Language Models with Correctness Coverage Guarantees, in: Findings of the Association for Computational Linguistics: EMNLP 2024, Association for Computational Linguistics, Miami, Florida, USA, 2024, pp. 6886–6898.

[59] J. Berger, E. Moreno, L. Pericchi, M. Bayarri, J. Bernardo, J. Cano, J. Horra, J. Martín, D. Rios, B. Betrò, A. Dasgupta, P. Gustafson, L. Wasserman, J. Kadane, C. Srinivasan, M. Lavine, A. O'Hagan, W. Polasek, C. Robert, S. Sivaganesan, An overview of robust Bayesian analysis, Test 3 (1994) 5–124.

[60] D. Insua, F. Ruggeri, Robust Bayesian Analysis, Lecture Notes in Statistics, Springer New York, 2012.

[61] P. Bishop, R. Bloomfield, B. Littlewood, A. Povyakalo, D. Wright, Toward a formalism for conservative claims about the dependability of software-based systems, IEEE Transactions on Software Engineering 37 (5) (2010) 708–717.

[62] X. Zhao, K. Salako, L. Strigini, V. Robu, D. Flynn, Assessing safety-critical systems from operational testing: A study on autonomous vehicles, Information and Software Techn. 128 (2020) 106393.

[63] R. A. Chakherlou, L. Strigini, Impact of Prior Beliefs on Dependability Prediction for a Changed System Using Pre-change Operational Evidence, in: 24th International Conference on Software Quality, Reliability and Security, IEEE, 2024, pp. 572–583.

[64] L. V. Utkin, F. P. A. Coolen, Imprecise probabilistic inference for software run reliability growth models., Journal of Uncertain Systems. 12 (4) (2018) 292–308.

[65] G. Walter, T. Augustin, Imprecision and prior-data conflict in generalized bayesian inference, Journal of Statistical Theory and Practice 3 (1) (2009) 255–271.

[66] X. Zhao, R. Calinescu, S. Gerasimou, V. Robu, D. Flynn, Interval change-point detection for runtime probabilistic model checking, in: Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering, 2020, pp. 163–174.

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

[68] M. Dahl, V. Magesh, M. Suzgun, D. E. Ho, Hallucinating law: Legal mistakes with large language models are pervasive, Law, regulation, and policy (2024).

[69] X. Huang, W. Ruan, et al., A survey of safety and trustworthiness of large language models through the lens of verification and validation, Artificial Intelligence Review 57 (7) (2024) 175.

[70] Y. Zhang, Y. Tang, W. Ruan, X. Huang, S. Khastgir, P. Jennings, X. Zhao, Protip: Probabilistic robustness verification on text-to-image diffusion models against stochastic perturbation, in: European Conference on Computer Vision, Springer, 2024, pp. 455–472.

[71] K. Salako, X. Zhao, The unnecessity of assuming statistically independent tests in bayesian software reliability assessments, IEEE Transactions on Software Engineering 49 (4) (2023) 2829–2838.

[72] P. Bishop, A. Povyakalo, Deriving a frequentist conservative confidence bound for probability of failure per demand for systems with different operational and test profiles, Reliability Engineering & System Safety 158 (2017) 246–253.

[73] R. Pietrantuono, P. Popov, S. Russo, Reliability assessment of service-based software under operational profile uncertainty, Reliability Engineering & System Safety 204 (2020) 107193.

[74] P. Popov, Why black-box bayesian safety assessment of autonomous vehicles is problematic and what can be done about it?, IEEE Transactions on Intelligent VehiclesIn press (2025).

[75] P. Popov, Dynamic safety assessment of Autonomous Vehicle based on Multivariate Bayesian Inference (DyAVSA), Journal of Reliable Intelligent Environments 11 (3) (2025) 1–23.

[76] W. Huang, X. Zhao, A. Banks, V. Cox, X. Huang, Hierarchical distribution-aware testing of deep learning, ACM Transactions on Software Engineering and Methodology 33 (2) (2023) 1–35.

[77] C. Smidts, C. Mutha, M. Rodríguez, M. J. Gerber, Software testing with an operational profile: OP definition, ACM Computing Surveys (CSUR) 46 (3) (2014) 1–39.

[78] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, K. Toutanova, BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions, in: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Vol. 1, 2019, pp. 2924–2936.

[79] R. A. Thayer, M. Lipow, E. C. Nelson, Software Reliability, North-Holland, 1978.

[80] J. W. Duran, S. C. Ntafos, An Evaluation of Random Testing, IEEE Transactions on Software Engineering SE-10 (4) (1984) 438–444.

[81] S. Chen, S. Mills, A binary Markov process model for random testing, IEEE Transactions on Software Engineering 22 (3) (1996) 218–223.

[82] K. Goseva-Popstojanova, K. S. Trivedi, Failure correlation in software reliability models, IEEE Transactions on Reliability 49 (1) (2000) 37–48.

[83] Y. Dai, M. Xie, K. Poh, Modeling and analysis of correlated software failures of multiple types, IEEE Transactions on Reliability 54 (1) (2005) 100–106.

[84] A. Bondavalli, S. Chiaradonna, F. Di Giandomenico, L. Strigini, Dependability models for iterative software considering correlation between successive inputs, in: Proceedings of IEEE International. Computer Performance and Dependability Symposium, IEEE, Erlangen, Germany, 1995, pp. 13–21.

[85] K. Salako, X. Zhao, Demonstrating software reliability using possibly correlated tests: Insights from a conservative bayesian approach, Quality and Reliability Engineering International 40 (3) (2024) 1197–1220.

[86] X. Zhao, W. Huang, A. Banks, V. Cox, D. Flynn, S. Schewe, X. Huang, Assessing the reliability of deep learning classifiers through robustness evaluation and operational profiles, in: AISafety-IJCAI'21, Vol. 2916, CEUR-WS, 2021, p. 16.

[87] X. Zhao, R. Aghazadeh-Chakherlou, C.-H. Cheng, P. Popov, L. Strigini, On the need for a statistical foundation in scenario-based testing of autonomous vehicles, in: Proceedings of the IEEE International Conference on Intelligent Transportation Systems (ITSC), Gold Coast, Australia, 2025.

[88] P. Bishop, R. Bloomfield, B. Littlewood, A. Povyakalo, D. Wright, Toward a formalism for conservative claims about the dependability of software-based systems, IEEE Transactions on Software Engineering 37 (5) (2011) 708–717.

[89] X. Zhao, V. Robu, D. Flynn, K. Salako, L. Strigini, Assessing the safety and reliability of autonomous vehicles from road testing, in: International Symposium on Software Reliability Engineering (ISSRE), 2019, pp. 13–23.

[90] G. Walter, L. J. M. Aslett, F. P. A. Coolen, Bayesian nonparametric system reliability using sets of priors, International Journal of Approximate Reasoning 80 (2017) 67–88.

[91] X. Zhao, V. Robu, D. Flynn, F. Dinmohammadi, M. Fisher, M. Webster, Probabilistic model checking of robots deployed in extreme environments, in: Proceedings of the 33rd AAAI Conference on Artificial Intelligence, Vol. 33, Honolulu, Hawaii, USA, 2019, pp. 8076–8084.

[92] X. Zhao, S. Gerasimou, R. Calinescu, C. Imrie, V. Robu, D. Flynn, Bayesian learning for the robust verification of autonomous robots, Communications Engineering 3 (1) (2024) 18.

Appendix A. Mathematical Derivations

This appendix contains the proofs and supplementary information for the theorems presented in the main text.

Appendix A.1. Common Framework and Notation

In a LLM, for a domain D_i with subdomains $S_{i1}, \ldots, S_{in_i}$. For each subdomain $j = 1, \ldots, n_i$:

- **Parameter:** $\boldsymbol{\theta}_i = (\theta_{i1}, \ldots, \theta_{in_i}) \in (0,1)^{n_i}$ (subdomain reliability)
- **Parameter:** n^F = number of trials in the future
- **Data:** $C_i = \{(C_{ik}, N_{ik})\}_{k=1}^{n_i}$ (observed correct responses and trial counts)
- **Hierarchical prior.**

$$\begin{aligned} C_{ij} \mid \theta_{ij}, N_{ij} &\sim \text{Binomial}(N_{ij}, \theta_{ij}), \\ \theta_{ij} \mid \mu_i, \nu_i &\overset{\text{iid}}{\sim} \text{Beta}(\mu_i\nu_i,\, (1-\mu_i)\nu_i), \\ \mu_i &\sim \text{Beta}(a_i, b_i), \qquad \nu_i \sim \text{Gamma}(c_i, \text{rate} = d_i), \end{aligned}$$

  with $\mu_i \in (0,1)$, $\nu_i > 0$ (domain-level parameters), and hyper- hyperparameters $h_i = (a_i, b_i, c_i, d_i)$.

- **Admissible hyperparameters (imprecise prior).**

$$\mathcal{A}_i = [a_i^{\min}, a_i^{\max}] \times [b_i^{\min}, b_i^{\max}] \times [c_i^{\min}, c_i^{\max}] \times [d_i^{\min}, d_i^{\max}].$$

- **Common Likelihood:**

$$L(\boldsymbol{\theta}_i) = Pr(C_i \mid \boldsymbol{\theta}_i) = \prod_{j=1}^{n_i} \binom{N_{ij}}{C_{ij}} \theta_{ij}^{C_{ij}} (1-\theta_{ij})^{N_{ij}-C_{ij}}$$

Appendix A.2. Proof of Theorem 1

Theorem 1 For subdomain S_{ij} in domain D_i, let $C_i = \{(C_{ik}, N_{ik})\}_{k=1}^{n_i}$ be the observed data. Let the admissible set of hyperparameters be

$$\mathcal{A}_i \;=\; [a_i^{\min}, a_i^{\max}] \times [b_i^{\min}, b_i^{\max}] \times [c_i^{\min}, c_i^{\max}] \times [d_i^{\min}, d_i^{\max}],$$

and write $h_i = (a_i, b_i, c_i, d_i)$. Then, *for any* $h_i \in \mathcal{A}_i$, the marginal posterior density of θ_{ij} is

$$Pr(\theta_{ij} \mid C_i, h_i) = \frac{f_{\text{marg}}(\theta_{ij}, C_i; h_i)}{Z_{\text{marg}}(h_i)},$$

where f_{marg} (unnormalized posterior) and Z_{marg} (normalizing constant) are

$$f_{\text{marg}}(\theta_{ij}, C_i; h_i) = \int_0^1 \int_0^\infty \left[\prod_{k \neq j} \int_0^1 d\theta_{ik}\right] L(\boldsymbol{\theta}_i)\, Pr(\boldsymbol{\theta}_i \mid \mu_i, \nu_i)\, Pr(\mu_i, \nu_i \mid h_i)\, d\mu_i\, d\nu_i$$

$$Z_{\text{marg}}(h_i) = \int_0^1 \int_0^\infty Pr(C_i \mid \mu_i, \nu_i)\, Pr(\mu_i, \nu_i \mid h_i)\, d\mu_i\, d\nu_i,$$

with $L(\boldsymbol{\theta}_i) = Pr(C_i \mid \boldsymbol{\theta}_i)$.

The imprecise marginal posterior is characterized by the lower/upper envelopes

$$\underline{Pr}(\theta_{ij} \mid C_i) = \inf_{h_i \in \mathcal{A}_i} Pr(\theta_{ij} \mid C_i, h_i), \qquad \overline{Pr}(\theta_{ij} \mid C_i) = \sup_{h_i \in \mathcal{A}_i} Pr(\theta_{ij} \mid C_i, h_i).$$

Proof of Theorm 1. We proceed step by step.

Step 1 — Likelihood and hierarchical prior densities. The joint data likelihood factorizes over subdomains:

$$L(\boldsymbol{\theta_i}) = Pr(C_i \mid \theta_i) = \prod_{k=1}^{n_i} \binom{N_{ik}}{C_{ik}} \theta_{ik}^{C_{ik}} (1-\theta_{ik})^{N_{ik}-C_{ik}}. \tag{A.1}$$

Given (μ_i, ν_i), the prior on θ_i factorizes as

$$Pr(\boldsymbol{\theta_i} \mid \mu_i, \nu_i) = \prod_{k=1}^{n_i} \frac{\Gamma(\nu_i)}{\Gamma(\mu_i\nu_i)\Gamma((1-\mu_i)\nu_i)}\, \theta_{ik}^{\mu_i\nu_i - 1} (1-\theta_{ik})^{(1-\mu_i)\nu_i - 1}.$$

Given the domain-level parameters (μ_i, ν_i), the subdomain reliabilities are conditionally independent and identically distributed:

$$\theta_{ij} \mid \mu_i, \nu_i \overset{\text{iid}}{\sim} \text{Beta}\big(\mu_i\nu_i,\, (1-\mu_i)\nu_i\big), \qquad j = 1, \ldots, n_i.$$

Here $\mu_i \in (0,1)$ is the prior mean and $\nu_i > 0$ is the prior strength (concentration).

For each k, the Beta pdf (Single-coordinate density) is:

$$Pr(\theta_{ik} \mid \mu_i, \nu_i) = \frac{\Gamma(\nu_i)}{\Gamma(\mu_i\nu_i)\,\Gamma((1-\mu_i)\nu_i)}\, \theta_{ik}^{\mu_i\nu_i - 1}\, (1-\theta_{ik})^{\,(1-\mu_i)\nu_i - 1}, \qquad \theta_{ik} \in (0,1).$$

Since $\{\theta_{ik}\}_{k=1}^{n_i}$ are conditionally independent given (μ_i, ν_i),

$$\begin{aligned} Pr(\boldsymbol{\theta}_i \mid \mu_i, \nu_i) &= \prod_{k=1}^{n_i} Pr(\theta_{ik} \mid \mu_i, \nu_i) \\ &= \prod_{k=1}^{n_i} \text{Beta}(\theta_{ik} \mid \mu_i\nu_i, (1-\mu_i)\nu_i) \\ &= \left(\frac{\Gamma(\nu_i)}{\Gamma(\mu_i\nu_i)\,\Gamma((1-\mu_i)\nu_i)}\right)^{n_i} \prod_{k=1}^{n_i} \theta_{ik}^{\mu_i\nu_i - 1}\, (1-\theta_{ik})^{\,(1-\mu_i)\nu_i - 1} \end{aligned} \tag{A.2}$$

Given hyperparameters $h_i = (a_i, b_i, c_i, d_i)$, we take μ_i and ν_i to be independent:

$$Pr(\mu_i, \nu_i \mid h_i) \;=\; Pr(\mu_i \mid h_i)\, Pr(\nu_i \mid h_i).$$

We assume:

$$\mu_i \sim \text{Beta}(a_i, b_i), \quad \mu_i \in (0,1), \qquad \nu_i \sim \text{Gamma}(c_i, \text{rate} = d_i), \quad \nu_i > 0.$$

Explicitly,

$$Pr(\mu_i \mid h_i) = \frac{\Gamma(a_i + b_i)}{\Gamma(a_i)\Gamma(b_i)}\, \mu_i^{a_i - 1}(1 - \mu_i)^{\, b_i - 1}, \qquad Pr(\nu_i \mid h_i) = \frac{d_i^{c_i}}{\Gamma(c_i)}\, \nu_i^{c_i - 1}\, e^{-d_i \nu_i}$$

Thus,

$$\begin{aligned} Pr(\mu_i, \nu_i \mid h_i) &= \mathrm{Beta}(\mu_i \mid a_i, b_i)\ \mathrm{Gamma}(\nu_i \mid c_i, \mathrm{rate} = d_i) \\ &= \frac{\Gamma(a_i + b_i)}{\Gamma(a_i)\,\Gamma(b_i)}\ \frac{d_i^{c_i}}{\Gamma(c_i)}\ \mu_i^{a_i - 1}(1 - \mu_i)^{\, b_i - 1}\ \nu_i^{c_i - 1} e^{-d_i \nu_i} \\ &= \frac{1}{\mathrm{B}(a_i, b_i)}\ \frac{d_i^{c_i}}{\Gamma(c_i)}\ \mu_i^{a_i - 1}(1 - \mu_i)^{\, b_i - 1}\ \nu_i^{c_i - 1} e^{-d_i \nu_i} \end{aligned} \tag{A.3}$$

where $\mathrm{B}(a_i, b_i) = \dfrac{\Gamma(a_i)\Gamma(b_i)}{\Gamma(a_i + b_i)}$.

The hyperprior factorizes:

$$Pr(\mu_i, \nu_i \mid h_i) = \frac{\Gamma(a_i + b_i)}{\Gamma(a_i)\Gamma(b_i)} \mu_i^{a_i - 1}(1 - \mu_i)^{b_i - 1} \cdot \frac{d_i^{c_i}}{\Gamma(c_i)} \nu_i^{c_i - 1} e^{-d_i \nu_i}.$$

Step 2 — Normalizing constant. By definition,

$$Z_{\mathrm{marg}}(C_i, h_i) = \int_0^1 Pr(\theta_{ij} \mid C_i, h_i)\, d\theta_{ij} = \int_0^1 \int_0^\infty Pr(C_i \mid \mu_i, \nu_i)\, Pr(\mu_i, \nu_i \mid h_i)\, d\mu_i\, d\nu_i,$$

where

$$Pr(C_i \mid \mu_i, \nu_i) = \int_{(0,1)^{n_i}} L(\theta_i)\, Pr(\theta_i \mid \mu_i, \nu_i)\, d\theta_i.$$

Start from the definition:

$$Pr(\theta_{ij} \mid C_i, h_i) = \frac{f_{\mathrm{marg}}(\theta_{ij}, C_i; h_i)}{Z_{\mathrm{marg}}(C_i, h_i)}$$

where

$$f_{\mathrm{marg}}(\theta_{ij}, C_i; h_i) = \int_0^1 \int_0^\infty \left[\prod_{k \neq j} \int_0^1 d\theta_{ik} \right] L(\boldsymbol{\theta}_i)\, Pr(\boldsymbol{\theta}_i \mid \mu_i, \nu_i)\, Pr(\mu_i, \nu_i \mid h_i)\, d\mu_i\, d\nu_i$$

The posterior integrates to 1:

$$\int_0^1 Pr(\theta_{ij} \mid C_i, h_i)\, d\theta_{ij} = 1 \implies \frac{1}{Z_{\mathrm{marg}}(C_i, h_i)} \int_0^1 f_{\mathrm{marg}}(\theta_{ij}, C_i; h_i)\, d\theta_{ij} = 1$$

Hence

$$Z_{\mathrm{marg}}(C_i, h_i) = \int_0^1 f_{\mathrm{marg}}(\theta_{ij}, C_i; h_i)\, d\theta_{ij}$$

By expanding the integral:

$$Z_{\text{marg}}(C_i, h_i) = \int_0^1 \int_0^1 \int_0^\infty \left[\prod_{k \neq j} \int_0^1 d\theta_{ik} \right] L(\boldsymbol{\theta}_i)\, Pr(\boldsymbol{\theta}_i \mid \mu_i, \nu_i)\, Pr(\mu_i, \nu_i \mid h_i)\, d\mu_i\, d\nu_i\, d\theta_{ij}$$
$$= \int_0^1 \int_0^\infty \left[\int_{(0,1)^{n_i}} L(\boldsymbol{\theta}_i)\, Pr(\boldsymbol{\theta}_i \mid \mu_i, \nu_i)\, d\boldsymbol{\theta}_i \right] Pr(\mu_i, \nu_i \mid h_i)\, d\mu_i\, d\nu_i. \tag{A.4}$$

By substituting Eqs. A.1, A.2, A.3 into A.4 we have:

$$Z_{\text{marg}}(C_i, h_i) = \left[\prod_{j=1}^{n_i} \binom{N_{ij}}{C_{ij}} \right] \frac{d_i^{c_i}}{\mathrm{B}(a_i, b_i)\, \Gamma(c_i)} \int_0^1 \int_0^\infty \mu_i^{a_i - 1} (1 - \mu_i)^{b_i - 1}\, \nu_i^{c_i - 1} e^{-d_i \nu_i}$$
$$\times \left(\frac{\Gamma(\nu_i)}{\Gamma(\alpha_i)\, \Gamma(\beta_i)} \right)^{n_i} \prod_{j=1}^{n_i} \frac{\Gamma\big(C_{ij} + \alpha_i\big)\, \Gamma\big(N_{ij} - C_{ij} + \beta_i\big)}{\Gamma\big(N_{ij} + \nu_i\big)}\, d\nu_i\, d\mu_i \tag{A.5}$$

Step 3 — Imprecise posterior. For any fixed $h_i = (a_i, b_i, c_i, d_i) \in \mathcal{A}_i$, Steps 1–2 give the precise (pointwise) posterior density

$$Pr(\theta_{ij} \mid C_i, h_i) = \frac{f_{\text{marg}}(\theta_{ij}, C_i; h_i)}{Z_{\text{marg}}(C_i, h_i)},$$

The hyperparameter vector h_i is not known exactly but only to lie within the admissible hyperrectangle

$$\mathcal{A}_i = [a_i^{\min}, a_i^{\max}] \times [b_i^{\min}, b_i^{\max}] \times [c_i^{\min}, c_i^{\max}] \times [d_i^{\min}, d_i^{\max}].$$

Hence the family $\mathcal{F}_i = \{Pr(\theta_{ij} \mid C_i, h_i) : h_i \in \mathcal{A}_i\}$ represents all posterior densities consistent with our prior ignorance about h_i.

Following the theory of imprecise probabilities, we define the *lower* and *upper* posterior densities (envelopes) as the pointwise infimum and supremum over this family:

$$\underline{Pr}(\theta_{ij} \mid C_i) = \inf_{h_i \in \mathcal{A}_i} Pr(\theta_{ij} \mid C_i, h_i), \qquad \overline{Pr}(\theta_{ij} \mid C_i) = \sup_{h_i \in \mathcal{A}_i} Pr(\theta_{ij} \mid C_i, h_i).$$

Operationally, these envelopes are obtained by evaluating the closed-form posterior $Pr(\theta_{ij} \mid C_i, h_i)$ at the extremal corners of $\mathcal{A}_i$ or through numerical optimization if the extrema occur in the interior. The resulting pair $(\underline{Pr}, \overline{Pr})$ bounds all admissible precise posteriors and fully characterizes the imprecise posterior belief about θ_{ij} given the uncertainty in (a_i, b_i, c_i, d_i).

□

Appendix A.3. Proof of Theorem 2

Theorem 2 For domain D_i with local OP weights Ω_{ij} (where $\sum_{j=1}^{n_i} \Omega_{ij} = 1$), let $p_i = \sum_{j=1}^{n_i} \Omega_{ij} \theta_{ij}$ be the domain-level non-failure probability. Define the admissible set of hyper-hyper-parameters

$$\mathcal{A}_i = [a_i^{\min}, a_i^{\max}] \times [b_i^{\min}, b_i^{\max}] \times [c_i^{\min}, c_i^{\max}] \times [d_i^{\min}, d_i^{\max}],$$

and write $h_i = (a_i, b_i, c_i, d_i)$.

Then, for any $h_i \in \mathcal{A}_i$, the posterior distribution of p_i is characterized by its CDF:

$$F_{p_i}(t \mid C_i, h_i) = \Pr(p_i \le t \mid C_i, h_i) \\ = \int_0^1 \int_0^\infty F_{p_i}(t \mid \mu_i, \nu_i, C_i)\, Pr(\mu_i, \nu_i \mid C_i, h_i)\, d\mu_i\, d\nu_i$$

where

- $F_{p_i}(t \mid \mu_i, \nu_i, C_i)$ is the conditional CDF of $p_i = \sum_{j=1}^{n_i} \Omega_{ij}\theta_{ij}$ given that $\theta_{ij} \mid \mu_i, \nu_i, C_i \overset{\text{ind.}}{\sim} \text{Beta}(C_{ij} + \mu_i\nu_i, N_{ij} - C_{ij} + (1-\mu_i)\nu_i)$ for $j = 1, \ldots, n_i$,
- $Pr(\mu_i, \nu_i \mid C_i, h_i)$ is the hyper-posterior obtained via Bayes' rule:

$$Pr(\mu_i, \nu_i \mid C_i, h_i) = \\ \frac{Pr(C_i \mid \mu_i, \nu_i)\, \text{Beta}(\mu_i \mid a_i, b_i)\, \text{Gamma}(\nu_i \mid c_i, \text{rate} = d_i)}{\int_0^1 \int_0^\infty Pr(C_i \mid \mu, \nu)\, \text{Beta}(\mu \mid a_i, b_i)\, \text{Gamma}(\nu \mid c_i, \text{rate} = d_i)\, d\mu\, d\nu}$$

The imprecise domain posterior is characterized by CDF envelopes:

$$\underline{F}_{p_i}(t \mid C_i) = \inf_{h_i \in \mathcal{A}_i} F_{p_i}(t \mid C_i, h_i), \qquad \overline{F}_{p_i}(t \mid C_i) = \sup_{h_i \in \mathcal{A}_i} F_{p_i}(t \mid C_i, h_i).$$

The domain reliability $p_i = \sum_{j=1}^{n_i} \Omega_{ij}\theta_{ij}$ is a weighted sum of subdomain reliabilities, whose joint posterior distribution $Pr(\boldsymbol{\theta}_i \mid C_i, h_i)$ has no tractable closed form. However, the hierarchical structure of our model provides a natural decomposition: we can express this joint posterior as a mixture over the shared hyperparameters (μ_i, ν_i), where conditionally on these hyperparameters, the subdomains become independent with Beta posteriors. This decomposition, formalized in Lemma 7, is the key to deriving the domain-level CDF. The proof proceeds by (1) expressing the domain CDF as an integral of the joint posterior over a weighted-sum constraint region, (2) applying the hierarchical decomposition from the lemma, (3) exchanging the order of integration via Fubini's theorem, and (4) recognizing the inner integral as a conditional CDF, yielding a tractable mixture representation.

Lemma 7 (Hierarchical decomposition of subdomain joint posterior)**.** *Under the hierarchical model with hyperparameters $h_i = (a_i, b_i, c_i, d_i)$, the joint posterior of subdomain reliabilities $\boldsymbol{\theta}_i = (\theta_{i1}, \ldots, \theta_{in_i})$ admits the decomposition*

$$Pr(\boldsymbol{\theta}_i \mid C_i, h_i) = \int_0^1 \int_0^\infty Pr(\boldsymbol{\theta}_i \mid \mu_i, \nu_i, C_i)\, Pr(\mu_i, \nu_i \mid C_i, h_i)\, d\mu_i\, d\nu_i, \tag{A.6}$$

where:

1. *The conditional distribution factorizes into independent Betas:*

$$Pr(\boldsymbol{\theta}_i \mid \mu_i, \nu_i, C_i) = \prod_{j=1}^{n_i} Beta(\theta_{ij} \mid C_{ij} + \mu_i\nu_i, N_{ij} - C_{ij} + (1-\mu_i)\nu_i).$$

2. *The hyper-posterior is obtained via Bayes' rule:*

$$Pr(\mu_i, \nu_i \mid C_i, h_i) = \frac{Pr(C_i \mid \mu_i, \nu_i)\, Pr(\mu_i, \nu_i \mid h_i)}{Pr(C_i \mid h_i)}, \tag{A.7}$$
$$Pr(\mu_i, \nu_i \mid h_i) = Beta(\mu_i \mid a_i, b_i) \times Gamma(\nu_i \mid c_i, rate = d_i),$$

where the marginal likelihood is

$$Pr(C_i \mid \mu_i, \nu_i) = \prod_{j=1}^{n_i} \binom{N_{ij}}{C_{ij}} \frac{B(C_{ij} + \mu_i\nu_i, N_{ij} - C_{ij} + (1-\mu_i)\nu_i)}{B(\mu_i\nu_i, (1-\mu_i)\nu_i)}, \tag{A.8}$$

and the evidence is

$$Pr(C_i \mid h_i) = \int_0^1 \int_0^\infty Pr(C_i \mid \mu, \nu)\, Beta(\mu \mid a_i, b_i)\, Gamma(\nu \mid c_i, rate = d_i)\, d\mu\, d\nu. \tag{A.9}$$

Proof of Lemma 7. The decomposition follows directly from the law of total probability and the hierarchical prior structure. By the tower property of conditional expectation applied to densities:

$$\begin{aligned} Pr(\boldsymbol{\theta}_i \mid C_i, h_i) = & \\ & \int_0^1 \int_0^\infty Pr(\boldsymbol{\theta}_i, \mu_i, \nu_i \mid C_i, h_i)\, d\mu_i\, d\nu_i = \\ & \int_0^1 \int_0^\infty Pr(\boldsymbol{\theta}_i \mid \mu_i, \nu_i, C_i)\, Pr(\mu_i, \nu_i \mid C_i, h_i)\, d\mu_i\, d\nu_i \end{aligned}$$

The conditional independence of $\{\theta_{ij}\}$ given (μ_i, ν_i, C_i) follows from the hierarchical prior:

$$\theta_{ij} \mid \mu_i, \nu_i \overset{\text{iid}}{\sim} \text{Beta}(\mu_i\nu_i, (1-\mu_i)\nu_i), \quad j = 1, \ldots, n_i,$$

combined with independent Binomial likelihoods $C_{ij} \mid \theta_{ij}, N_{ij} \sim \text{Binomial}(N_{ij}, \theta_{ij})$. By conjugacy, the Beta-Binomial update yields:

$$\theta_{ij} \mid \mu_i, \nu_i, C_i \sim \text{Beta}(C_{ij} + \mu_i\nu_i, N_{ij} - C_{ij} + (1-\mu_i)\nu_i).$$

The hyper-posterior (A.7) follows from Bayes' rule. The marginal likelihood (A.8) is the Beta-Binomial distribution for each subdomain, obtained by integrating the Binomial likelihood against the Beta prior. The evidence (A.9) is the normalization constant ensuring the hyper-posterior integrates to one. □

Proof of Theorem 2. We derive the posterior CDF of the domain reliability $p_i = \sum_{j=1}^{n_i} \Omega_{ij}\theta_{ij}$ using the hierarchical decomposition from Lemma 7.

Step 1: CDF definition. For any $t \in [0, 1]$, the posterior CDF of p_i is

$$F_{p_i}(t \mid C_i, h_i) = \Pr(p_i \le t \mid C_i, h_i) = \Pr\left(\sum_{j=1}^{n_i} \Omega_{ij}\theta_{ij} \le t \,\middle|\, C_i, h_i \right).$$

Define the constraint region

$$\mathcal{R}_i(t) := \left\{ \boldsymbol{\theta}_i \in (0,1)^{n_i} : \sum_{j=1}^{n_i} \Omega_{ij}\theta_{ij} \leq t \right\}.$$

Then

$$F_{p_i}(t \mid C_i, h_i) = \int_{\mathcal{R}_i(t)} Pr(\boldsymbol{\theta}_i \mid C_i, h_i)\, d\boldsymbol{\theta}_i.$$

Step 2: Apply hierarchical decomposition. Substituting the decomposition from Lemma 7 (equation (A.6)):

$$F_{p_i}(t \mid C_i, h_i) = \int_{\mathcal{R}_i(t)} \left[\int_0^1 \int_0^\infty Pr(\boldsymbol{\theta}_i \mid \mu_i, \nu_i, C_i)\, Pr(\mu_i, \nu_i \mid C_i, h_i)\, d\mu_i\, d\nu_i \right] d\boldsymbol{\theta}_i.$$

Step 3: Change of integration order. By Fubini's theorem (applicable since all integrands are non-negative and integrate to finite values):

$$F_{p_i}(t \mid C_i, h_i) = \int_0^1 \int_0^\infty \left[\int_{\mathcal{R}_i(t)} Pr(\boldsymbol{\theta}_i \mid \mu_i, \nu_i, C_i)\, d\boldsymbol{\theta}_i \right] Pr(\mu_i, \nu_i \mid C_i, h_i)\, d\mu_i\, d\nu_i.$$

Step 4: Conditional CDF. The inner integral is the conditional CDF of p_i given (μ_i, ν_i) and C_i:

$$F_{p_i}(t \mid \mu_i, \nu_i, C_i) = \int_{\mathcal{R}_i(t)} Pr(\boldsymbol{\theta}_i \mid \mu_i, \nu_i, C_i)\, d\boldsymbol{\theta}_i = \Pr\left(\sum_{j=1}^{n_i} \Omega_{ij}\theta_{ij} \leq t \,\middle|\, \mu_i, \nu_i, C_i \right),$$

where $\theta_{ij} \mid \mu_i, \nu_i, C_i$ are independent Beta random variables as specified in Lemma 7.

Step 5: Mixture representation. Combining Steps 3 and 4:

$$F_{p_i}(t \mid C_i, h_i) = \int_0^1 \int_0^\infty F_{p_i}(t \mid \mu_i, \nu_i, C_i)\, Pr(\mu_i, \nu_i \mid C_i, h_i)\, d\mu_i\, d\nu_i.$$

where

- $F_{p_i}(t \mid \mu_i, \nu_i, C_i)$ is the conditional CDF of $p_i = \sum_{j=1}^{n_i} \Omega_{ij}\theta_{ij}$ given that $\theta_{ij} \mid \mu_i, \nu_i, C_i \overset{\text{ind.}}{\sim} \text{Beta}(C_{ij} + \mu_i\nu_i, N_{ij} - C_{ij} + (1-\mu_i)\nu_i)$ for $j = 1, \ldots, n_i$,
- $Pr(\mu_i, \nu_i \mid C_i, h_i)$ is the hyper-posterior obtained via Bayes' rule:

$$Pr(\mu_i, \nu_i \mid C_i, h_i) = \frac{Pr(C_i \mid \mu_i, \nu_i)\, \text{Beta}(\mu_i \mid a_i, b_i)\, \text{Gamma}(\nu_i \mid c_i, \text{rate} = d_i)}{\int_0^1 \int_0^\infty Pr(C_i \mid \mu, \nu)\, \text{Beta}(\mu \mid a_i, b_i)\, \text{Gamma}(\nu \mid c_i, \text{rate} = d_i)\, d\mu\, d\nu}$$

This expresses the domain posterior CDF as a weighted average of conditional CDFs, where the weights are given by the hyper-posterior $Pr(\mu_i, \nu_i \mid C_i, h_i)$.

Step 6: Imprecise probability bounds. For each $h_i \in \mathcal{A}_i$, the mixture formula defines a valid CDF $F_{p_i}(\cdot \mid C_i, h_i)$. The imprecise posterior is characterized by pointwise envelopes over the admissible set:

$$\underline{F}_{p_i}(t \mid C_i) = \inf_{h_i \in \mathcal{A}_i} F_{p_i}(t \mid C_i, h_i), \quad \overline{F}_{p_i}(t \mid C_i) = \sup_{h_i \in \mathcal{A}_i} F_{p_i}(t \mid C_i, h_i), \quad t \in [0,1]$$

□

Appendix A.4. Proof of Theorem 3

Theorem 3 For the LLM system with domain weights W_i (where $\sum_{i=1}^m W_i = 1$), let $p_L = \sum_{i=1}^m W_i p_i$ be the LLM-level failure probability and data $= \{C_1, \ldots, C_m\}$ the observed data across all domains. Assume cross-domain independence.

Define the domain-level admissible sets

$$\mathcal{A}_i = [a_i^{\min}, a_i^{\max}] \times [b_i^{\min}, b_i^{\max}] \times [c_i^{\min}, c_i^{\max}] \times [d_i^{\min}, d_i^{\max}]$$

and write $h_i = (a_i, b_i, c_i, d_i)$ for $i = 1, \ldots, m$. Define the LLM-level admissible set as the Cartesian product

$$\mathcal{A}_{\text{LLM}} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_m,$$

and collect the domain hyperparameters as $\mathcal{H} = (h_1, \ldots, h_m) \in \mathcal{A}_{\text{LLM}}$.

Then, for any $\mathcal{H} \in \mathcal{A}_{\text{LLM}}$, the posterior distribution of p_L is characterized by its CDF:

$$F_{p_L}(t \mid \text{data}, \mathcal{H}) = Pr(p_L \le t \mid \text{data}, \mathcal{H}) = \int \cdots \int G\big(t \mid \{\mu_i, \nu_i\}_{i=1}^m, \text{data}\big) \prod_{i=1}^m Pr(\mu_i, \nu_i \mid C_i, h_i) \prod_{i=1}^m d\mu_i \, d\nu_i$$

where

- $G(t \mid \{\mu_i, \nu_i\}_{i=1}^m, \text{data})$ is the conditional CDF of $p_L = \sum_{i=1}^m W_i p_i$ given all hyperparameters, defined as

$$G\big(t \mid \{\mu_i, \nu_i\}_{i=1}^m, \text{data}\big) = \int_{\mathcal{R}_L(t)} \prod_{i=1}^m f_{p_i}(p_i \mid \mu_i, \nu_i, C_i) \, dp_1 \cdots dp_m,$$

  where $\mathcal{R}_L(t) := \{(p_1, \ldots, p_m) \in (0,1)^m : \sum_{i=1}^m W_i p_i \le t\}$, and $f_{p_i}(\cdot \mid \mu_i, \nu_i, C_i)$ is the conditional density of $p_i = \sum_{j=1}^{n_i} \Omega_{ij} \theta_{ij}$ under $\theta_{ij} \mid \mu_i, \nu_i, C_i \overset{\text{ind.}}{\sim} \text{Beta}(C_{ij} + \mu_i \nu_i, N_{ij} - C_{ij} + (1 - \mu_i)\nu_i)$,

- $Pr(\mu_i, \nu_i \mid C_i, h_i)$ is the domain-level hyper-posterior for domain i:

$$Pr(\mu_i, \nu_i \mid C_i, h_i) = \frac{Pr(C_i \mid \mu_i, \nu_i) \, \text{Beta}(\mu_i \mid a_i, b_i) \, \text{Gamma}(\nu_i \mid c_i, \text{rate} = d_i)}{\int_0^1 \int_0^\infty Pr(C_i \mid \mu, \nu) \, \text{Beta}(\mu \mid a_i, b_i) \, \text{Gamma}(\nu \mid c_i, \text{rate} = d_i) \, d\mu \, d\nu}$$

- Cross-domain independence ensures $Pr(\{\mu_i, \nu_i\}_{i=1}^m \mid \text{data}, \mathcal{H}) = \prod_{i=1}^m Pr(\mu_i, \nu_i \mid C_i, h_i)$.

The imprecise LLM posterior is characterized by CDF envelopes:

$$\underline{F}_{p_L}(t \mid \text{data}) = \inf_{\mathcal{H} \in \mathcal{A}_{\text{LLM}}} F_{p_L}(t \mid \text{data}, \mathcal{H})$$
$$\overline{F}_{p_L}(t \mid \text{data}) = \sup_{\mathcal{H} \in \mathcal{A}_{\text{LLM}}} F_{p_L}(t \mid \text{data}, \mathcal{H})$$

Proof of Theorem 3. We derive the posterior distribution of the LLM reliability $p_L = \sum_{i=1}^m W_i p_i$ by characterizing its CDF, building on the domain-level results from Theorem 2.

Step 1: CDF definition. For any $t \in [0, 1]$, the posterior CDF of p_L is

$$F_{p_L}(t \mid \text{data}, \mathcal{H}) = \Pr(p_L \leq t \mid \text{data}, \mathcal{H}) = \Pr\left(\sum_{i=1}^m W_i p_i \leq t \,\middle|\, \text{data}, \mathcal{H} \right).$$

Define the region

$$\mathcal{R}_L(t) := \left\{ (p_1, \ldots, p_m) \in (0,1)^m : \sum_{i=1}^m W_i p_i \leq t \right\}.$$

Then

$$F_{p_L}(t \mid \text{data}, \mathcal{H}) = \int_{\mathcal{R}_L(t)} \prod_{i=1}^m Pr(p_i \mid C_i, h_i)\, dp_1 \cdots dp_m,$$

where cross-domain independence ensures the joint density factorizes as

$$Pr(p_1, \ldots, p_m \mid \text{data}, \mathcal{H}) = \prod_{i=1}^m Pr(p_i \mid C_i, h_i).$$

Step 2: Domain-level mixture representation. From Theorem 2, each domain posterior can be written as

$$Pr(p_i \mid C_i, h_i) = \int_0^1 \int_0^\infty f_{p_i}(p_i \mid \mu_i, \nu_i, C_i)\, Pr(\mu_i, \nu_i \mid C_i, h_i)\, d\mu_i\, d\nu_i,$$

where $f_{p_i}(\cdot \mid \mu_i, \nu_i, C_i)$ is the conditional density of $p_i = \sum_{j=1}^{n_i} \Omega_{ij} \theta_{ij}$ given that

$$\theta_{ij} \mid \mu_i, \nu_i, C_i \overset{\text{ind.}}{\sim} \text{Beta}(C_{ij} + \mu_i \nu_i, N_{ij} - C_{ij} + (1 - \mu_i)\nu_i), \quad j = 1, \ldots, n_i.$$

Step 3: Hierarchical decomposition across domains. Substituting the domain-level mixture into the LLM-level CDF:

$$F_{p_L}(t \mid \text{data}, \mathcal{H}) = \int_{\mathcal{R}_L(t)} \prod_{i=1}^m \left[\int_0^1 \int_0^\infty f_{p_i}(p_i \mid \mu_i, \nu_i, C_i)\, Pr(\mu_i, \nu_i \mid C_i, h_i)\, d\mu_i\, d\nu_i \right] dp_1 \cdots dp_m$$

Step 4: Change of integration order. By Fubini's theorem (applicable because all integrands are non-negative and integrate to finite values), we can exchange the order of integration:

$$F_{p_L}(t \mid \text{data}, \mathcal{H}) = \int_0^1 \cdots \int_0^1 \int_0^\infty \cdots \int_0^\infty \left[\int_{\mathcal{R}_L(t)} \prod_{i=1}^m f_{p_i}(p_i \mid \mu_i, \nu_i, C_i)\, dp_1 \cdots dp_m \right] \times \prod_{i=1}^m Pr(\mu_i, \nu_i \mid C_i, h_i) \prod_{i=1}^m d\mu_i\, d\nu_i.$$

Step 5: Conditional CDF. The inner integral is the conditional CDF of p_L given all hyperparameters:

$$\begin{aligned} G\big(t \mid \{\mu_i, \nu_i\}_{i=1}^m, \text{data}\big) &:= \int_{\mathcal{R}_L(t)} \prod_{i=1}^m f_{p_i}(p_i \mid \mu_i, \nu_i, C_i)\, dp_1 \cdots dp_m \\ &= \Pr\left(\sum_{i=1}^m W_i p_i \le t \,\middle|\, \{\mu_i, \nu_i\}_{i=1}^m, \text{data} \right), \end{aligned}$$

where $p_i \mid \mu_i, \nu_i, C_i$ are independent across domains, each distributed according to the weighted sum of independent Betas as specified in Step 2.

Step 6: Mixture representation. Combining Steps 4 and 5:

$$F_{p_L}(t \mid \text{data}, \mathcal{H}) = \int \cdots \int G\big(t \mid \{\mu_i, \nu_i\}_{i=1}^m, \text{data}\big) \prod_{i=1}^m Pr(\mu_i, \nu_i \mid C_i, h_i) \prod_{i=1}^m d\mu_i\, d\nu_i \tag{A.10}$$

where the integrals range over $\mu_i \in (0,1)$ and $\nu_i \in (0,\infty)$ for all $i = 1, \ldots, m$.

This expresses the LLM posterior CDF as a weighted average of conditional CDFs, where the weights are given by the product of independent domain-level hyper-posteriors.

Step 7: Imprecise probability bounds. For each $\mathcal{H} \in \mathcal{A}_{\text{LLM}}$, the formula above defines a valid CDF $F_{p_L}(\cdot \mid \text{data}, \mathcal{H})$. The imprecise posterior is characterized by pointwise envelopes:

$$\begin{aligned} \underline{F}_{p_L}(t \mid \text{data}) &= \inf_{\mathcal{H} \in \mathcal{A}_{\text{LLM}}} F_{p_L}(t \mid \text{data}, \mathcal{H}), \\ \overline{F}_{p_L}(t \mid \text{data}) &= \sup_{\mathcal{H} \in \mathcal{A}_{\text{LLM}}} F_{p_L}(t \mid \text{data}, \mathcal{H}) \end{aligned}$$

where, $t \in [0,1]$ □

Appendix A.5. Reliability for Multiple Consecutive Operations

Theorems 1, 2, and 3 establish posterior distributions for the reliability parameters θ_{ij}, p_i, and p_L at the subdomain, domain, and LLM levels, respectively. These parameters represent the probability of success on a *single* future operation. However, in practical reliability assessment, we often need to evaluate performance over *multiple consecutive operations*.

This appendix clarifies the mathematical framework for extending the case $n^F = 1$ to $n^F > 1$ (reliability with the required number of future failure-free tasks), which is formalized in Theorems 4, 5, and 6. We explain:

1. Why reliability over n^F operations takes the form $\theta_{ij}^{n^F}$ (or $p_i^{n^F}$, $p_L^{n^F}$)
2. Why we characterize these distributions through CDFs rather than closed-form densities
3. How to derive the CDF integral formulas
4. How this framework extends across all hierarchical levels

Appendix A.5.1. Definition: Reliability with the required n^F failure-free future tasks

At the subdomain level, we define reliability for n^F consecutive future operations as:

$$R_{ij}(n^F) = \theta_{ij}^{n^F}$$

Assume that:

- Each task in subdomain S_{ij} succeeds independently with probability θ_{ij}
- Operations are identically distributed (i.i.d. assumption)
- We observe n^F consecutive operations

Under these assumptions, the probability that *all* n^F operations succeed is:

$$\Pr(\text{all } n^F \text{ operations succeed}) = \underbrace{\theta_{ij} \times \theta_{ij} \times \cdots \times \theta_{ij}}_{n^F \text{ times}} = \theta_{ij}^{n^F}$$

Because θ_{ij} is uncertain—it is a random variable with posterior distribution $Pr(\theta_{ij} \mid C_i, h_i)$ from Theorem 1—the n^F-operation reliability $R_{ij}(n^F) = \theta_{ij}^{n^F}$ is also a random variable. We must therefore characterize its full posterior distribution.

Appendix A.5.2. Why the CDF Approach

Unlike the posteriors in Theorems 1–3, which characterize reliability for a *single* future operation, the distribution of $R_{ij}(n^F) = \theta_{ij}^{n^F}$ generally does not have a closed-form probability density function, even when θ_{ij} follows a Beta distribution. This is because:

- The transformation $\theta_{ij} \mapsto \theta_{ij}^{n^F}$ is nonlinear for $n^F \neq 1$
- At higher hierarchical levels (domain, LLM), we have weighted sums of dependent random variables raised to powers, making closed forms even less tractable

Therefore, we characterize the distribution of $R_{ij}(n^F)$ through its *CDF*, which has a tractable integral representation and is sufficient for all practical reliability calculations (e.g., computing probabilities, quantiles, expected values).

Appendix A.5.3. Deriving the CDF Formula

What does CDF answers: "What is the probability that n^F-operation reliability is at most r?"

$$F_{R_{ij}(n^F)}(t \mid C_i, h_i) = \Pr(R_{ij}(n^F) \leq t \mid C_i, h_i) = \Pr(\theta_{ij}^{n^F} \leq t \mid C_i, h_i)$$

To compute this probability, we use the following key observation. If $\theta_{ij}^{n^F} \leq t$, then taking the n^F-th root of both sides:

$$\theta_{ij} \leq t^{1/n^F}$$

Since $\theta_{ij} \in (0, 1)$ and $t \in [0, 1]$, both sides are positive, and the function $x \mapsto x^{1/n^F}$ is monotonically increasing on $[0, 1]$ for $n^F > 0$. Therefore, the inequality is preserved under this transformation.

It follows that:

$$\Pr(\theta_{ij}^{n^F} \leq t \mid C_i, h_i) = \Pr(\theta_{ij} \leq t^{1/n^F} \mid C_i, h_i)$$

The right-hand side is simply the CDF of the posterior distribution $Pr(\theta_{ij} \mid C_i, h_i)$ from Theorem 1, evaluated at t^{1/n^F}:

$$\Pr(\theta_{ij} \leq t^{1/n^F} \mid C_i, h_i) = \int_0^{t^{1/n^F}} Pr(\theta_{ij} \mid C_i, h_i)\, d\theta_{ij}$$

This gives us the formula in Theorem 4.

Appendix A.5.4. Extension to Domain and LLM Levels

The same logic extends to domain and LLM levels:

- **Domain level:** $R_i(n^F) = p_i^{n^F}$ where $p_i = \sum_{j=1}^{n_i} \Omega_{ij}\theta_{ij}$
- **LLM level:** $R_L(n^F) = p_L^{n^F}$ where $p_L = \sum_{i=1}^{m} W_i p_i$

At these levels, closed forms are even less available because we must:

1. Compute the distribution of weighted sums of dependent Beta random variables (from Theorems 2 and 3)
2. Apply the power transformation to obtain $R_i(n^F)$ or $R_L(n^F)$

Both steps lack analytical solutions, so we compute CDFs via numerical integration or Monte Carlo sampling.

Appendix B. Global numerical settings used in all figures

All figures in this paper are produced under the same imprecise hierarchical Bayes setup and numerical settings:

- **Domains/subdomains.** D1 (Coding) = {MBPP, DS-1000}, D2 (Reasoning) = {BoolQ, RACE-H}.

- **Aggregation weights.** Within-domain weights $\Omega_1 = [0.204, 0.796]$ (MBPP, DS-1000) and $\Omega_2 = [0.483, 0.517]$ (BoolQ, RACE-H); LLM-level weights $W = [0.149, 0.851]$ (D1, D2).

- **Hyperpriors and imprecision ranges (per domain i).**
  $\mu_i \sim \text{Beta}(a_i, b_i),\ \ \nu_i \sim \text{Gamma}(c_i, \text{rate} = d_i),\ \ \text{with } a_i, b_i \in [1, 12] \text{ and } c_i, d_i \in [1, 25].$

- **Simulation** For each domain and each sampled hyperparameter configuration: *(1)* sample (μ_i, ν_i) from the discretized posterior; *(2)* draw subdomain accuracies θ_{ij} from their Beta posteriors; *(3)* aggregate to p_i and p_L; *(4)* compute either empirical CDFs of $Z \in \{\theta_{ij}, p_i, p_L\}$ on a fixed grid, or Monte Carlo expectations such as $\widehat{\mathbb{E}}[R_L(n^F)] = \frac{1}{S}\sum_{s=1}^{S}\left(p_L^{(s)}\right)^{n^F}$. Repeating over K hyperparameter configurations yields a family of CDFs/expectations; the plotted bands are the *pointwise min–max* across configurations (epistemic uncertainty).

- **CDF envelopes.** For any quantity $Z \in \{\theta_{ij}, p_i, p_L\}$, we compute an empirical CDF for each configuration, and plot the pointwise min–max envelope $[F_{\min}(t), F_{\max}(t)]$ across configurations.

Appendix C. List of Mathematical Notations

Symbol	Description
$\mathcal{X}$	Input-space of all possible tasks for an LLM
π	Operational Profile (OP), probability distribution over tasks
n	Sequence of tasks
$I(x_\tau)$	Indicator function: 1 for success, 0 for failure on task τ
x_τ	The τ-th task in a sequence of n tasks
$R(n, \pi)$	LLM reliability: probability of failure-free operation over n tasks
M	Number of LLM models being evaluated
m	Number of independent domains
D_i	Domain i $(i = 1, \ldots, m)$
n_i	Number of subdomains in domain i
S_{ij}	Subdomain j in domain i $(j = 1, \ldots, n_i)$
C_{ij}	Number of correct responses in subdomain S_{ij}
N_{ij}	Number of trials (tasks) in subdomain S_{ij}
C_i	Set of observed data in domain i: $\{(C_{ik}, N_{ik})\}_{k=1}^{n_i}$
data	Observed data across all domains: $\{C_1, \ldots, C_m\}$
θ_{ij}	Subdomain reliability (success probability) for S_{ij}
$\boldsymbol{\theta}_i$	Vector of subdomain reliabilities: $(\theta_{i1}, \ldots, \theta_{in_i})$
p_i	Domain-level non-failure probability (posterior)
p_L	LLM-level non-failure probability (posterior)
$R_{ij}(n^F)$	Subdomain reliability for n^F consecutive tasks
$R_i(n^F)$	Domain reliability for n^F consecutive tasks
$R_L(n^F)$	LLM reliability for n^F consecutive tasks
Ω_{ij}	Operational weight for subdomain j in domain i
W_i	Operational weight for domain i
μ_i	Expected reliability (prior mean) for domain i
ν_i	Prior strength (concentration parameter) for domain i
a_i, b_i	Hyperparameters of the Beta prior for μ_i
c_i, d_i	Hyperparameters of Gamma prior for ν_i
h_i	Domain-level hyperparameter tuple (a_i, b_i, c_i, d_i)
$\mathcal{A}_i$	Admissible set of domain-level hyperparameters for domain i
$\mathcal{H}$	Tuple of all domain hyperparameters: $(h_1, \ldots, h_m)$
$\mathcal{A}_{\text{LLM}}$	LLM-level admissible hyperparameter set: $\mathcal{A}_1 \times \cdots \times \mathcal{A}_m$
$F_{p_i}(t \mid C_i, h_i)$	Posterior CDF of domain-level non-failure probability p_i
$[\underline{F}_{p_i}(t \mid C_i), \overline{F}_{p_i}(t \mid C_i)]$	Interval-valued CDF of domain-level non-failure probability p_i
$F_{p_L}(t \mid \text{data}, \mathcal{H})$	Posterior CDF of LLM-level non-failure probability p_L
$[\underline{F}_{p_L}(t \mid \text{data}), \overline{F}_{p_L}(t \mid \text{data})]$	Interval-valued CDF of domain-level non-failure probability p_L
$F_{R_{ij}(n^F)}(t \mid C_i, h_i)$	Posterior CDF of subdomain reliability for n^F consecutive tasks
$F_{R_i(n^F)}(t \mid C_i, h_i)$	Posterior CDF of domain-level reliability for n^F consecutive tasks
$F_{R_L(n^F)}(t \mid \text{data}, \mathcal{H})$	Posterior CDF of LLM-level reliability for n^F consecutive tasks

Symbol	Description
$f_{\text{marg}}(\theta_{ij}, C_i; h_i)$	Unnormalized marginal posterior for θ_{ij}
$Z_{\text{marg}}(h_i)$	Normalizing constant for marginal posterior
$f_{p_i}(\cdot \mid \mu_i, \nu_i, C_i)$	Conditional density of domain reliability
$G(t \mid \{\mu_i, \nu_i\}_{i=1}^m, \text{data})$	onditional CDF of p_L given all domain hyperparameters
$\mathcal{R}_i(t)$	Constraint region $\{(p_1, \ldots, p_m) : \sum_j \Omega_{ij} p_{ij} \leq t\}$ for domain i
$\mathcal{R}_L(t)$	Constraint region induced by threshold t for LLM-level CDF evaluation
$f_{\text{marg}}(\theta_{ij}, C_i; h_i)$	Unnormalized marginal posterior for θ_{ij}
$Z_{\text{marg}}(h_i)$	Normalizing constant for marginal posterior
$f_{p_i}(\cdot \mid \mu_i, \nu_i, C_i)$	Conditional density of domain-level non-failure probability p_i
$G(t \mid \{\mu_i, \nu_i\}_{i=1}^m, \text{data})$	Conditional CDF of LLM-level non-failure probability p_L given all domain hyperparameters
$\mathcal{R}_i(t)$	Constraint region for domain-level CDF evaluation of p_i
$\mathcal{R}_L(t)$	Constraint region for LLM-level CDF evaluation of p_L
t	Probability threshold at which the corresponding CDF is evaluated