BRIDGE: Predicting Human Task Completion Time From Model Performance

Fengyuan Liu [*1 2] **Jay Gala** [*1 2] **Nilaksh** [1 3]
Dzmitry Bahdanau [1 2 4 5] **Siva Reddy** [1 2 5 6] **Hugo Larochelle** [1]

Abstract

Evaluating the real-world capabilities of AI systems requires grounding benchmark performance in human-interpretable measures of task difficulty. Existing approaches that rely on direct human task completion time annotations are costly, noisy, and difficult to scale across benchmarks. In this work, we propose BRIDGE, a unified psychometric framework that learns a latent difficulty scale from model responses and anchors it to human task completion time. Using a two-parameter logistic Item Response Theory model, we jointly estimate latent task difficulty and model capability from model performance data across multiple benchmarks. We demonstrate that latent task difficulty varies linearly with the logarithm of human completion time, allowing human task completion time to be inferred for new benchmarks from model performance alone. Leveraging this alignment, we forecast frontier model capabilities in terms of human task length and independently reproduce METR's exponential scaling results, with the 50% solvable task horizon doubling approximately every 6 months. Code is available at McGill-NLP/BRIDGE.

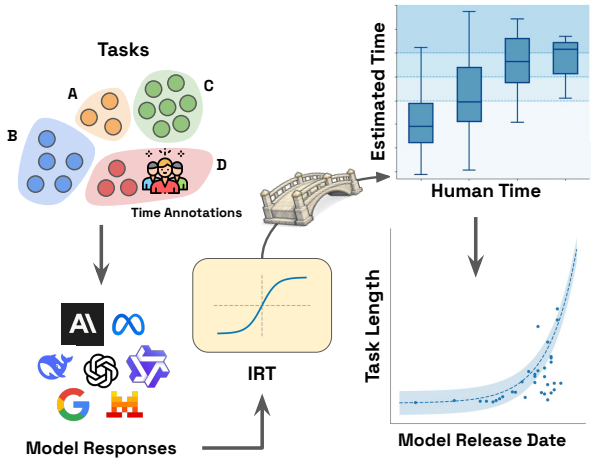

Figure 1. Overview of BRIDGE. Model responses across different benchmarks (clustered by colors) are used to fit a two-parameter logistic Item Response Theory (2PL IRT) model, estimating latent task difficulty and model capability. Calibrating latent difficulty against tasks with known human task completion times yields accurate predictions of human task duration for new benchmarks. We leverage this alignment to forecast frontier model capabilities in terms of human task length even in the absence of human task duration annotations.

1. Introduction

As Artificial Intelligence (AI) systems are increasingly deployed in open-ended, real-world settings, their capabilities are typically reported via benchmark scores and aggregate metrics. However, such scores are difficult to interpret as measures of *task difficulty* or *real-world effort*: improvements may reflect gains on short, routine instances while leaving longer, multi-step tasks largely unchanged, and comparable score changes can correspond to very different shifts in practical capability. Consequently, benchmark performance alone provides limited guidance about what AI systems can reliably do or how quickly practical capability is improving. A more actionable evaluation should express performance on human-aligned scales, most directly, the time a human would require to complete a task.

A prominent attempt to express AI capability in human terms is by Kwa et al. (2026) at METR, which measures performance in terms of the length of tasks AI agents can complete and reports exponential growth, with task-length horizons doubling roughly every 7 months. While compelling, this paradigm depends on human task completion time annotations from people with relevant expertise. These annotations are expensive collect and are difficult to extend consistently across diverse benchmarks. As model capabilities continue to scale, relying on new human studies to anchor each benchmark becomes increasingly impractical, creating a growing gap between benchmark-centric

[*]Equal contribution [1]Mila - Quebec AI Institute [2]McGill University [3]Polytechnique Montréal [4]Periodic Labs [5]Canada CIFAR AI Chair [6]ServiceNow Research. Correspondence to: {fengyuan.liu,jay.gala}@mila.quebec.

Proceedings of the 43rd International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

evaluation and human-centric notions of difficulty.

In this work, we introduce BRIDGE,[1] a unified psychometric framework (illustrated in Figure 1) that addresses this gap by aligning task difficulty for humans, measured by task completion time, with task difficulty for models, measured by benchmark performance.

Item Response Theory (IRT) (Baker, 2001), and in particular the two-parameter logistic (2PL) model, has been adopted to analyze large language model (LLM) performance and benchmarks (Lalor et al., 2019; Rodriguez et al., 2021; Hofmann et al., 2025). We adopt a 2PL IRT model to jointly estimate the latent difficulty of individual tasks and the capability of individual models using a binarized indicator (success vs. failure) as performance data across multiple benchmarks.

Our central empirical finding is that IRT-inferred latent task difficulty closely tracks human completion time: across benchmarks with annotations, latent difficulty varies approximately linearly with the logarithm of human task duration. This relationship anchors the latent scale to a human-interpretable time axis and enables prediction of human task completion time from model performance alone. After calibrating the scale using existing human annotations (e.g., METR), we can map IRT difficulty values for new benchmarks onto this time scale without conducting new human studies. We show that for newly introduced tasks (e.g., SWE-bench (Jimenez et al., 2024) and Cybench (Zhang et al., 2025)), our predicted completion times align well with both available human annotations and qualitative expectations, indicating that the IRT-based latent scale effectively transfers difficulty estimates between humans and models.

Building on this alignment, we use BRIDGE to forecast frontier model capabilities in terms of human task-length horizons using only model performance logs, without requiring human task-time annotations. We estimate that solvable task length continues to grow exponentially, with a doubling time of approximately 6 months. These trends are consistent with the findings of Kwa et al. (2026), providing independent validation that model-centric data can recover human-centric task length forecasting.

2. Background and Motivation

2.1. Measuring capability through human task completion time

Estimating task completion time provides a meaningful bridge between benchmarks and real-world applications, enabling AI capability to be expressed in human-interpretable units and to forecast progress over different time horizons (Ngo, 2023). METR (Kwa et al., 2026) operationalized this idea through the "50%-task-completion time horizon," defined as the duration of tasks that an AI system can complete with 50% probability. Using this metric, they showed that the frontier task-length horizon has grown exponentially in recent years. Their analysis relied on human time annotations for 170 tasks, collected from annotators with relevant domain knowledge but no task-specific context.

Although the METR framework provides a valuable human-centered scale, its reliance on direct time annotations tightly couples each benchmark to a bespoke human study. This coupling makes it difficult to propagate a calibrated notion of difficulty across benchmarks with different formats, domains, or task distributions. METR's subsequent work partially addresses cross-domain transfer by estimating human completion time under alternative modeling assumptions; however, this approach still depends on benchmark-specific heuristics and strong priors about task structure (METR, 2025), which become increasingly fragile as benchmarks grow more heterogeneous and open-ended (e.g., GDPval (Patwardhan et al., 2026)). More broadly, repeated human studies do not scale with the pace at which new benchmarks and task variants are introduced. These limitations raise a natural question: can we predict human task completion time without conducting new human studies for each benchmark? Our work addresses this question by introducing a latent, model-derived difficulty scale that is explicitly calibrated to human completion time, enabling scalable prediction of human task duration for new benchmarks.

2.2. Item response theory for LLM benchmarking

Psychometric methods originally developed for educational testing, particularly IRT (Baker, 2001), provide a principled framework for modeling latent ability[2] of individuals and latent difficulty of tasks. Given a set of tasks $T = \{t_1, \ldots, t_{|T|}\}$, each task t_i is characterized by a discrimination parameter $a_i \geq 0$, which measures how strongly the task differentiates models with different capabilities and a difficulty parameter b. For the set of models $M = \{m_1, \ldots, m_{|M|}\}$, each model m_j has its ability estimate θ_j. For $t_i \in T$ and $m_j \in M$, the probability of having the correct response $P(y_{i,j} = 1)$ is defined by the item response function for the 2PL IRT model as:

$$\begin{aligned} P(y_{i,j} = 1 \mid \theta_j, a_i, b_i) &= \sigma(a_i(\theta_j - b_i)) \\ &= \frac{e^{a_i(\theta_j - b_i)}}{1 + e^{a_i(\theta_j - b_i)}} \end{aligned} \tag{1}$$

where σ is the logistic function, and a_i, b_i, and θ_j are learned parameters.

[1] BRIDGE refers to **B**enchmark **R**esponse **I**nferred **D**ifficulty **G**rounded in **E**lapsed human time

[2] We use *ability* to denote the latent trait parameter θ estimated via IRT and *capability* to refer to the underlying construct it approximates.

Intuitively, task t_i is not informative when $a_i = 0$. Both b_i and θ_j are scale invariant: their absolute values are not identifiable, and only their difference $\theta_j - b_i$ matters for the probability of success. Hence b_i and θ_j are centered at zero and are on the same scale. When $P(y_{i,j} = 1 \mid \theta_*, a_i, b_i) = 0.5$, $\theta_* = b_i$, unless $a_i = 0$. θ_* naturally provides the definition for the ability required to achieve a 50% success rate on a specific question with difficulty b_i.

IRT models have recently been adapted to language model evaluation to characterize task difficulty and model capability from performance data. Lalor et al. (2019) introduced the use of IRT models from machine-generated responses rather than human responses to estimate the difficulty of NLP tasks. Rodriguez et al. (2021) proposed Difficulty and Ability Discriminating (DAD) leaderboards, using IRT to infer task difficulty, discriminativeness, and feasibility. More recently, Fluid Benchmarking (Hofmann et al., 2025) applied IRT to estimate latent model ability and item difficulty and combined it with adaptive item selection to reduce variance and improve benchmarking efficiency and validity.

IRT maps performance to a shared latent space, enabling principled comparison and joint estimation of task difficulty and model capability. However, because IRT is inherently scale-invariant, its latent difficulty and ability parameters are identifiable only up to an affine transformation and therefore lack intrinsic semantic meaning. Consequently, although IRT offers a useful psychometric structure, it does not yield an interpretable measure of task difficulty in human terms or model capability relative to real-world effort.

We address this limitation by anchoring the IRT latent difficulty scale to human completion time, resolving the scale ambiguity and endowing IRT difficulty and ability estimates with a concrete, human-centric interpretation that enables prediction of human task duration from model performance.

3. Predicting Human Task Time with BRIDGE

While direct human studies provide expensive but valuable estimates of task difficulty in human terms, IRT offers a scalable but intrinsically uncalibrated measure of task difficulty from model performance. We introduce BRIDGE, a framework that connects these two regimes by anchoring latent IRT difficulty to human completion time.

Given binary outcomes for each model–task pair, we fit a 2PL IRT model and estimate the discrimination a_i and difficulty b_i for each task, along with an ability θ_j for each model. The IRT model is fitted to maximize the likelihood $\prod_{i,j} P(y_{ij} \mid \theta_j, a_i, b_i)$ using Markov chain Monte Carlo with hierarchical priors, based on Natesan et al. (2016).[3] The binary run matrix can be sparse due to limited compute and incomplete historical evaluations, but the observed responses (51% in our experiments) are sufficient to support stable estimation under the 2PL model.

Expressing model capability in terms of the odds $P/(1-P)$, we can rewrite Equation (1) in log-odds form,

$$\log \frac{P}{1-P} = a_i(\theta_j - b_i) \tag{2}$$

Thus, the latent ability parameter θ is a log-scale measure of model capability. If capability grows exponentially over time t as prior scaling analyses suggest, then θ grows linearly with t. Independently, METR's results show that the human task-time horizon h also grows exponentially with t (i.e., $h \propto e^t$, equivalently $t \propto \log h$). Combining these two observations: $\theta \propto \log h$. Because task difficulty b_i is defined on the same latent scale as θ, this implies a linear relationship between task difficulty b_i and the logarithm of the human time required to complete task t_i.

Let $T_h \subseteq T$ denote the subset of tasks with human completion time annotations. For each task $t_k \in T_h$, let b_k denote its latent task difficulty and h_k its human completion time. As the task difficulty is on the $\log$ scale of human completion time, we can fit a linear relationship between $\{b_k : t_k \in T_h\}$ and $\{\log h_k : t_k \in T_h\}$:

$$\log(h_k) = \text{slope} \times b_k + \text{intercept} \tag{3}$$

This mapping enables prediction of human completion time for tasks in $T_{h'} \not\subset T$ using only their IRT latent difficulties, without requiring task descriptions or human studies.

Building on this calibration, we forecast the evolution of frontier model capability in human-interpretable units. Specifically, within each 2-month release window, we identify the best-performing model m_s with its estimated ability θ_s. At a 50% success rate, $b_s = \theta_s$ characterizes the maximum task difficulty solvable. More generally, when the success rate is not 50%, given model capability θ_s, we find tasks with a_s and b_s, which leads to the desired success probability with the 2PL model (Equation (1)).[4] Applying the calibration in Equation (3) on b_s then yields the associated human completion time h_s, producing time-horizon forecasts of model capability that are directly anchored to human task duration.

4. Experiments

We use METR to learn the calibration mapping between latent task difficulty b and human completion time. Concretely, after fitting the 2PL IRT model on the response matrix from all benchmarks, we regress $\log h_i$ on b_i to obtain the log-linear mapping used throughout the paper.

[3] We use the implementation from Lalor & Rodriguez (2023).

[4] Primary analysis focuses on 50% success threshold. A detailed example for the 80% success rate is discussed in Appendix B.2.

We apply this calibration to estimate human completion time on benchmarks without exact time annotations. We validate this on a set of out-of-distribution benchmarks, which require extended multi-step reasoning or workflow execution. We use SWE-Bench verified (Jimenez et al., 2024), MLE-bench (Chan et al., 2025), GDPval (Patwardhan et al., 2026), and Cybench (Zhang et al., 2025), to test how well we can estimate human completion time and to forecast trends without human time annotations.

4.1. Data

4.1.1. METR (CALIBRATION BENCHMARK)

METR (Kwa et al., 2026) released a unified dataset aggregating model performance on 170 tasks spanning three benchmarks: Software Atomic Actions (SWAA), which comprise single-step computer operation tasks; HCAST, which focuses on software engineering tasks (Rein et al., 2025); and RE-Bench, which evaluates AI research and engineering tasks (Wijk et al., 2025). Each task is annotated with *how long a domain-knowledgeable human without task-specific context* would take to complete it.

METR reports repeated trials for each model–task pair. To obtain a single binary outcome per pair compatible with our IRT fit, we treat a model-task pair as successful if the model succeeds in at least 50% of the reported trials.

4.1.2. OUT-OF-DISTRIBUTION VALIDATION BENCHMARKS

SWE-bench Verified: SWE-bench (Jimenez et al., 2024) evaluates software engineering capability by measuring whether an agent can resolve real GitHub issues. SWE-bench Verified is a curated subset of 500 instances that have been manually validated and annotated with coarse completion-time categories: <15 minutes, 15 minutes to 1 hour, 1 hour to 4 hours, and >4 hours. A task is successful if the generated patch resolves the corresponding GitHub issue, as verified by passing the benchmark evaluation harness (i.e., the relevant unit tests). In addition to our own runs, we incorporate publicly available SWE-bench Verified leaderboard logs, which provide task-level success outcomes for multiple model–agent configurations.

MLE-bench: MLE-bench (Chan et al., 2025) evaluates the capabilities of LLMs on end-to-end machine learning workflows, including data preparation, model training, and hyperparameter tuning. The benchmark comprises 75 Kaggle competitions spanning diverse domains and problem types, with tasks categorized into difficulty levels ranging from low to hard. For each competition, we construct three binary success indicators: (i) whether the agent produces a valid submission under the competition rules, (ii) whether the submission achieves above-median performance on the public leaderboard, and (iii) whether it earns any Kaggle medal (bronze/silver/gold). We treat these indicators as three separate tasks per competition and rely on the official evaluation pipelines. We select a subset of 38 problems, consisting of 20 low difficulty tasks and 18 medium or hard difficulty tasks, due to computational constraints.

GDPval: GDPval (Patwardhan et al., 2026) evaluates economically significant, real-world tasks spanning 44 occupations across nine major U.S. GDP-contributing sectors. Tasks are designed by industry professionals (average 14 years of experience) to reflect real workflows, following occupational definitions from the U.S. Bureau of Labor Statistics. Because GDPval is open-ended, it does not admit a fully automated, verifiable evaluator. We therefore use an LLM-as-a-judge pipeline with Gemini 3 Pro as the judge.

Cybench: Cybench (Zhang et al., 2025) evaluates cybersecurity capabilities by measuring whether agents can solve professional-level Capture the Flag (CTF) challenges. The benchmark comprises tasks drawn from four CTF competitions spanning cryptography, forensics, reverse engineering, web exploitation, and binary exploitation, with difficulty levels ranging from very easy to hard, and human first-solve times (FST) from under 5 minutes to over 24 hours. A task is deemed successful if the agent recovers the correct flag string, verified by exact match. We run agents using the Cybench agent framework[5] with tool access to a bash shell and standard CTF utilities. We use the unguided setting and restrict analysis to the tasks for which at least one model succeeds, as tasks unsolved by all models yield unreliable IRT difficulty estimates and would not support meaningful human time prediction.

For SWE-bench Verified, MLE-bench, and GDPval, we run agents using the InspectAI framework[6] with the default ReAct-style scaffold and tool access to a bash shell and python interpreter. For each task, generation is capped at 1000 turns or until context window is full. We use a temperature of 0.0 (greedy decoding) and sufficiently large maximum output token budgets to avoid truncation of intermediate reasoning steps or final outputs.

The included models are described in Appendix A and additional evaluation details are described in Appendix C.

4.2. Baselines

4.2.1. LOGIT SUCCESS RATE

Beside the probabilistic IRT model, we include a heuristic baseline that estimates task difficulty and model ability via success rates. Such accuracy-based methods are widely

[5] https://github.com/andyzorigin/cybench
[6] https://inspect.aisi.org.uk/

used and offer an intuitive notion of task difficulty and model capability. Let $Y \in \{0,1\}^{|T|\times|M|}$ denote the binary response matrix of observed model–task outcomes, with rows corresponding to tasks and columns to models.

For each task t_i, we compute its empirical success rate across models over observed entries, $r_i = \frac{1}{|M_i|}\sum_{j\in M_i} Y_{ij}$, where M_i denotes the set of models with available responses for task i. We define the baseline difficulty score as $d_i^{\text{base}} = -\operatorname{logit}(1-r_i)$, where $\operatorname{logit}(p) = \log\frac{p}{1-p}$.

Analogously, for each model m_j, we compute its empirical success rate across tasks, $s_j = \frac{1}{|T|}\sum_i Y_{ij}$, and define its baseline ability as $a_j^{\text{base}} = \operatorname{logit}(s_j)$.

We apply the logit transform to both task- and model-level average success rates to place the scores on a unbounded scale that is comparable to the log-odds structure of the 2PL IRT model. To enable human task completion time prediction, we fit the same log-linear calibration as in Equation (3) by regressing $\log h_i$ on d_i^{base} for METR tasks with human time annotations. This yields a mapping from success rate based difficulty to human completion time.

We use this baseline as a point of comparison for BRIDGE in task-time estimation to assess the added value of explicitly modeling discrimination and latent ability via IRT.

4.2.2. LLMs as Estimators

Prior works has shown that LLMs can be used to estimate both the latent difficulty (Razavi & Powers, 2025; Tabib & Deedar, 2025) and human task completion time (Veeramani et al., 2024). As an additional baseline, we prompt frontier LLMs (Gemini 3 Pro and GPT-5.2) to predict human completion time for tasks from task descriptions.

For each task, we construct a prompt that provides task-specific context and requests a point estimate of expert human completion time in minutes. While adapted to the domain characteristics of each benchmark, all prompts follow a consistent schema: (1) relevant metadata (e.g., repository or competition information), (2) the full task description or problem statement, (3) domain-appropriate calibration anchors spanning the expected difficulty range, and (4) explicit instructions to output a single numeric estimate accompanied by a structured justification. We use greedy decoding in a single turn and a maximum output token budget of 32000. The full prompt templates are shown in Figures 11 and 12.

4.3. Estimating Human Task Completion Time

After fitting the 2PL IRT model on the response matrix from all benchmarks, using the METR aggregated dataset, we fit a linear model relating latent task difficulty b_i to the logarithm of human task completion time $\log h_i$ (in minutes), as defined in Equation (3) and shown in Figure 2. With

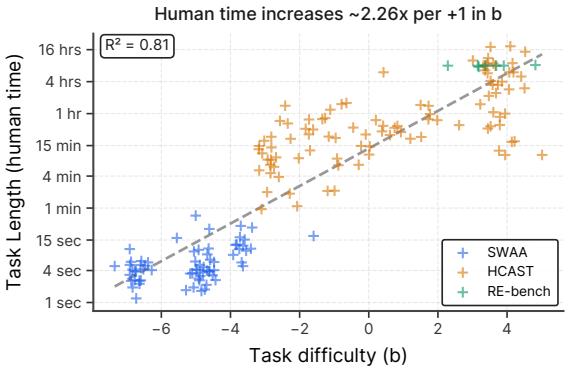

Figure 2. Task length (human completion time) vs. latent task difficulty (b) estimated via 2PL IRT across METR task suites (SWAA, HCAST, RE-bench), based on Equation (3). The log-linear fit ($R^2 = 0.81$) shows that each unit increase in b corresponds to $\sim 2.26\times$ increase in human completion time. This calibration anchors the IRT latent difficulty scale to human-interpretable units to enable prediction of task duration directly from model performance.

$R^2 = 0.81$, the fitted model indicates a strong log-linear alignment between latent difficulty and human completion time. According to the fitted slope, a one-unit increase in difficulty b corresponds to about a $2.26\times$ increase in human completion time.

Having established the log-linear relationship between latent task difficulty b and human completion time h, we apply this mapping to estimate human task durations for tasks without exact time annotations: SWE-bench Verified, MLE-bench, GDPval, and Cybench. For each task, we infer task length h_i from its IRT latent difficulty b_i using the learned log-linear relationship. Figure 7 in Appendix B.1 shows the task-length distributions.

We evaluate the transferability of this calibration on SWE-bench Verified and Cybench by comparing predicted times against human-annotated completion-time. On SWE-bench Verified (Figure 3), BRIDGE produces predictions that increase monotonically across annotated buckets, with median estimates that track the bucket ordering and scale. Despite unavoidable discretization error introduced by coarse bucket boundaries, BRIDGE achieves substantially better alignment with the annotated time buckets than both heuristic and LLM-based baselines. In contrast, the logit success-rate baseline systematically underestimates task duration, while Gemini 3 Pro and GPT-5.2 yield compressed predictions with limited dynamic range, leading to poor separation between longer-horizon buckets.

On Cybench (Figure 4), BRIDGE's predicted human completion time estimates align closely with recorded human first-solve times, achieving the strongest correlation ($R^2 = 0.45$) and placing 92.3% of tasks within a $0.5\times - 2\times$ tol-

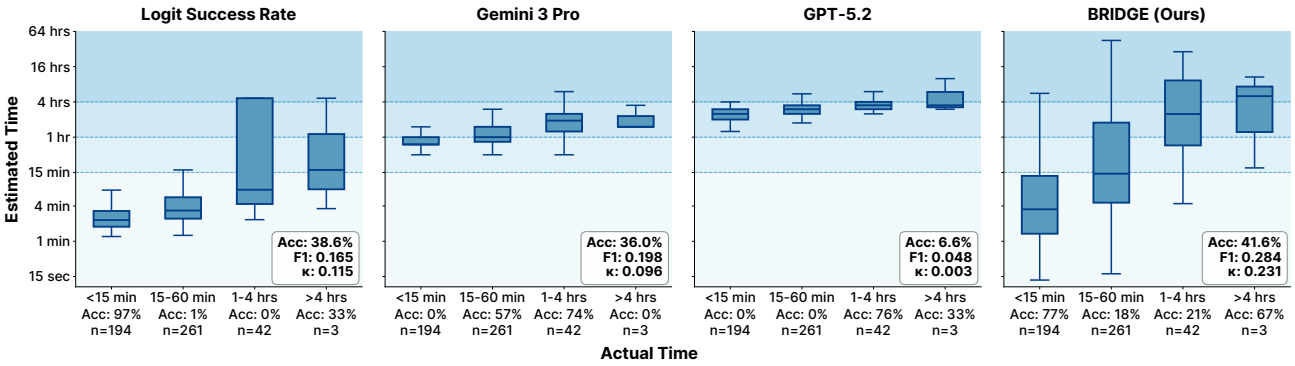

Figure 3. Alignment between annotated human completion time buckets and estimated human completion times on SWE-bench Verified. We report per-bucket classification accuracy (Acc) and the number of tasks (n), as well as overall accuracy, weighted macro F1 score, and weighted kappa. We compare a logit success-rate heuristic, LLM-based time predictions (Gemini 3 Pro, GPT-5.2), and BRIDGE. BRIDGE achieves substantially better alignment with the annotated time buckets than both heuristic and LLM-based baselines.

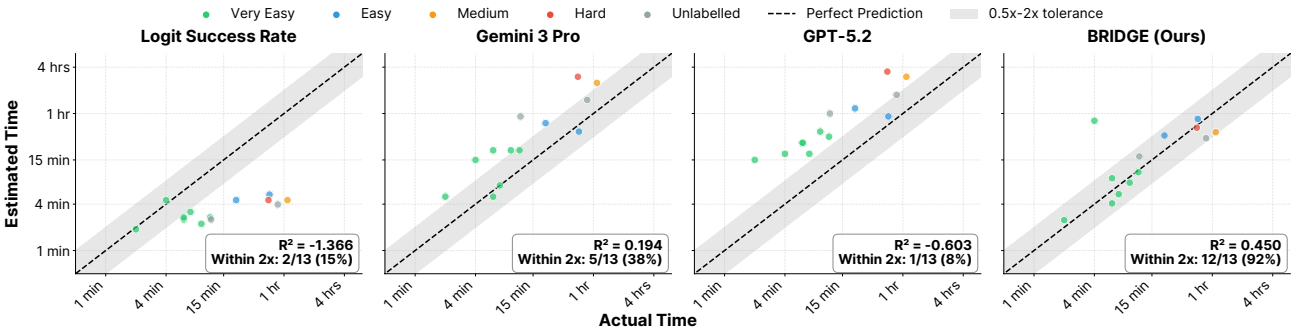

Figure 4. Alignment between actual human completion time (first-solve time) and estimated completion times on Cybench. The logit success-rate baseline substantially underestimates task duration, while LLM-based estimates consistently overestimate it. In contrast, BRIDGE aligns closely with actual human times, with 92.3% of tasks falling within a $0.5\times$–$2\times$ tolerance band.

erance band. The logit success rate baseline substantially underestimates task duration, reflecting its inability to account for heterogeneity in model capability. Gemini 3 Pro and GPT-5.2 capture the correct qualitative ordering of tasks but consistently overestimate absolute completion times, resulting in markedly lower tolerance-band coverage.

On MLE-bench (Figure 7), tasks yielding only valid submissions are estimated to require substantially shorter completion times than achieving above median performance on the leaderboard or earning any medal.

Overall, these results show that BRIDGE yields human task completion time estimates that are quantitatively accurate where ground truth is available and qualitatively well calibrated across benchmarks, consistently outperforming success rate based and LLM based human completion time prediction baselines.

4.4. Forecasting Capabilities Without Human Studies

To track the evolution of frontier model capabilities, we analyze task success probability as a function of estimated human completion time on a logarithmic scale. We partition models into 5-month release windows and, within each window, select the model with the highest estimated ability, denoted $m_{\max}$ with ability $\theta_{\max}$. For tasks in METR, SWE-bench Verified, MLE-bench, and GDPval, we use the IRT-estimated latent parameters (a_i, b_i) together with $\theta_{\max}$ to compute task success probabilities p_i via Equation (1). Each task's latent difficulty b_i is then mapped to estimated human completion time h_i using Equation (3). We plot p_i against h_i across tasks in Figure 5.

Frontier models released in 2025 achieve 50% success on tasks estimated to require $\sim$ 1–2.5 hours of human effort. Raising the success threshold from 50% to 80% substantially contracts the range of solvable task lengths: as shown in Figure 8, even the most recent models are largely limited to tasks requiring less than one hour of human completion time. The plotted probability–time relationships are not perfectly smooth inverse sigmoids due to heterogeneity in task discrimination a_i and difficulty b_i even within the same benchmark, underscoring the importance of task-level granularity. We further observe markedly steeper curves for METR and SWE-bench than for MLE-bench and GDPval,

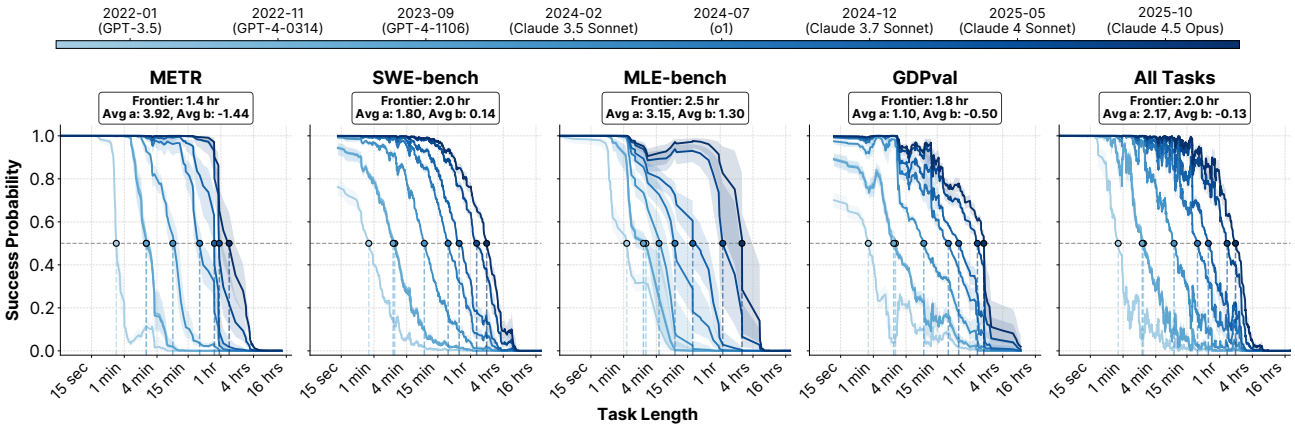

Figure 5. Success probability versus estimated human task completion time for different models, smoothed with a window of 15 tasks. Solvable task lengths at the 50% success threshold are indicated across model release dates, with darker blue denoting more recent models. SOTA models achieve 50% success on tasks estimated to require ∼1.4–2.5 hours of human effort. Steeper curves reflect higher task discrimination parameters a. Non-smoothness arises from heterogeneity in task-level difficulty and discrimination (a_i, b_i), highlighting the importance of task-level granularity.

consistent with their higher discrimination parameters a.

Finally, we use BRIDGE to forecast how the human completion time horizon of frontier models evolves over model release dates without human annotations. We partition models into 2-month release windows and select the best-performing model m_s in each window with estimated ability θ_s. For 50% success rate, $b_s = \theta_s$. b_s is then mapped to human completion time h_s with Equation (3). Figure 6 visualizes task length horizon h_i vs. model release date. At a 50% success rate, frontier models exhibit exponential growth in solvable task length over time, with capabilities doubling approximately every 6 months. The current state-of-the-art (SOTA) model reaches a task-length horizon of roughly two hours at 50% success rate. We observe analogous behavior at an 80% success threshold (Figure 9), albeit with a reduced achievable task length and a similar doubling time.

Overall, these results show that BRIDGE enables forecasting of frontier model capabilities in human-interpretable units using model performance alone. The resulting growth rates closely align with estimates obtained from direct human studies, as in Kwa et al. (2026), supporting the use of model performance-derived latent difficulty as a scalable and reliable proxy for human time annotations when tracking long-horizon capability progress.

4.5. Sensitivity to model pool and sparsity

As models are not evaluated on all tasks, we assess robustness to incomplete model-task coverage via a sparsity ablation, randomly removing 10% to 70% of observed entries and refitting IRT. Sparsity (%) denotes the fraction of originally observed model-task responses removed, and Observed (%) denotes the fraction of the full model-task matrix that remains observed after ablation. $\rho(b)$ is the Spearman correlation between ablated-data task difficulty estimates and the full-data estimates, and $\rho(\theta)$ is the corresponding correlation for model ability.

Table 1. Robustness of IRT parameter estimates under different sparsity levels in model-task binary success indicator response matrix. The actual estimates use 51% observed values in the response matrix.

Sparsity (%)	Observed (%)	$\rho(b)$	$\rho(\theta)$
10	45	0.9947	0.9980
20	40	0.9913	0.9970
30	35	0.9869	0.9955
40	30	0.9810	0.9947
50	25	0.9735	0.9903
60	20	0.9614	0.9872
70	15	0.9424	0.9798

The latent difficulty parameter b_i used for time calibration remains highly stable. Its correlation with the full-data estimate stays above 0.97 after removing 50% of observations and above 0.94 even at 70% removal. The latent model ability θ_j is even more stable, with consistently higher correlations across all sparsity levels. This suggests that the current 51% observed values in the model-task response matrix is sufficient for reliable estimation, and BRIDGE is robust to different model-task coverages.

5. Related Work

5.1. Estimating latent task difficulties

Although fitting IRT models using model outputs requires significantly less effort than human annotators, it still re-

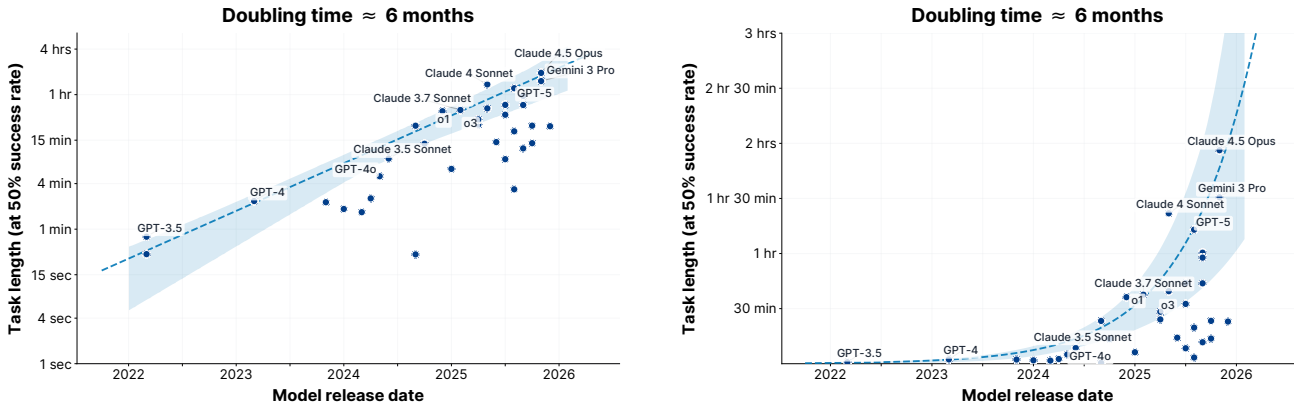

Figure 6. Forecasting trends of task length horizon over model release date without human task-time annotation. The task length at which model can achieve 50% accuracy grows exponentially over time, with an estimated doubling time of approximately 6 months. The left subfigure shows this trend on a logarithmic scale for task length while the right subfigure presents the same trend on a linear scale. The shaded region represents 95% confidence intervals computed via bootstrapping. BRIDGE enables forecasting of frontier model capabilities in human-interpretable units using model performance alone.

quires a large and diverse set of tasks with a sufficient number of attempts. Consequently, a parallel line of research has focused on estimating latent task difficulty more efficiently, with the goal of reducing the number of required attempts. Byrd & Srivastava (2022) used traits correlated with features of questions, answers, and associated contexts, to predict both difficulty and discrimination for new questions. Scarlatos et al. (2025) estimate task difficulty by simulating responses from a population of LLM-based students, each conditioned on a scalar IRT ability parameter so that the model generates answers as if written by learners at different skill levels, automatically scoring these responses, and fitting an IRT model to infer item difficulty. Truong et al. (2025) trained a model that predicts question difficulty from its embedding features. This line of work is focusing on estimating the latent difficulty of newly added tasks and can be used in parallel with our work.

5.2. Aligning human and latent task difficulty

TaskSense (Yin et al., 2025) estimates task difficulty in the context of graphical user interface tasks by modeling the cognitive processes that precede each user action. It decomposes tasks into sequences of cognitive steps, and assigns each step a difficulty index based on information-theoretic and cognitive principles. Task difficulty is computed as a linear aggregation of these step-level difficulties. The resulting difficulty estimates are validated against human data and shown to correlate strongly with both step-level and task-level human completion times. Such an approach requires task-specific modeling and annotations. Our approach differs by treating task difficulty as a latent variable inferred from large-scale model performance across benchmarks, rather than from explicit cognitive decomposition, providing a unified generalizable difficulty estimation framework aligned with human completion time.

5.3. Aligning human and latent model capability

Concurrently, Epoch AI proposed an IRT-based framework to estimate benchmark difficulty and model capability (Ho et al., 2025). Unlike our Bayesian inference, they learned latent parameters with L2-regularized non-linear least-squares and derive the latent model ability score. They show that the latent model ability score correlates with METR human time across models and yields capability growth trends broadly consistent with ours.

Our approach differs in granularity and calibration: whereas Epoch AI operates at the benchmark level and anchor at the model level, we estimate difficulty at the individual task level and anchor it to human completion time, enabling task-level time prediction and horizon estimates. Our framework yields different predictions for certain frontier models compared to Epoch AI. For instance, Epoch AI ranks Gemini 3 Pro as the top-performing model, whereas our estimates place Claude 4.5 Opus ahead. We hypothesize that such differences likely stem from different benchmark compositions, highlighting the inherent uncertainty in capability forecasting. We believe that it is valuable to have multiple independent forecasting efforts: comparing predictions across methodologies can reveal where estimates are robust and where greater caution is warranted, ultimately aimed to provide a more calibrated view of frontier model progress.

6. Conclusion & Future Work

We present BRIDGE, a psychometric framework that aligns latent task difficulty inferred from model performance with human task completion time, enabling interpretable, scal-

able evaluation and forecasting of AI capabilities **without requiring human time annotations**. By fitting a 2PL IRT model to performance data across benchmarks, BRIDGE induces a unified latent scale that jointly captures task difficulty and model capability.

We show that latent task difficulty is strongly and linearly related to the logarithm of human completion time, allowing human task durations to be inferred for benchmarks lacking explicit timing annotations. Leveraging this alignment, we characterize model success probability as a function of task length and find that recent frontier models achieve 50% success on tasks requiring approximately 2 hours of human effort. Tracking frontier models over time, BRIDGE reveals an exponential expansion of the solvable human task-length horizon, with capabilities doubling every 6 months, reproducing and corroborating METR's human-time–based findings **using only model performance data**.

Looking forward, BRIDGE provides a foundation for a broader class of human-interpretable evaluations. One promising direction is to extend the framework to human–AI collaborative settings, modeling how task responsibility is dynamically shared, how partial automation reshapes effective task difficulty, and how human interventions can act as signals of difficulty as model capabilities advance.

BRIDGE can also be expanded beyond long-horizon procedural tasks to support knowledge-intensive evaluations by enriching the psychometric model with task attributes that capture information requirements, external tool use, or reliance on knowledge beyond a model's training cutoff.

Finally, as task horizons lengthen, human completion times are expected to exhibit increased inter-individual variability. Accounting for this variability offers a natural direction for extending BRIDGE toward uncertainty-aware difficulty estimation and capability forecasting.

Overall, BRIDGE demonstrates that robust, human-grounded, and interpretable capability evaluation and forecasting can be achieved **without direct human studies**, substantially lowering the cost and friction of tracking real-world AI progress while remaining consistent with human-time–anchored evaluations.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

Acknowledgment

Siva Reddy and Dzmitry Bahdanau are supported by the Canada CIFAR AI Chairs program. We acknowledge the support of the IVADO R3AI Grant and a Gemini Academic Program Award. We thank the Mila IDT team and the Digital Research Alliance of Canada for providing the computational resources used in this work. We also thank members of McGill University and Mila, especially Arkil Patel and Parishad BehnamGhader, for their valuable feedback and constructive discussions throughout the project.

References

Baker, F. B. *The basics of item response theory*. ERIC, 2001.

Byrd, M. and Srivastava, S. Predicting difficulty and discrimination of natural language questions. In Muresan, S., Nakov, P., and Villavicencio, A. (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 119–130, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-short.15. URL https://aclanthology.org/2022.acl-short.15/.

Chan, J. S., Chowdhury, N., Jaffe, O., Aung, J., Sherburn, D., Mays, E., Starace, G., Liu, K., Maksin, L., Patwardhan, T., Madry, A., and Weng, L. MLE-bench: Evaluating machine learning agents on machine learning engineering. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=6s5uXNWGIh.

Ho, A., Denain, J.-S., Atanasov, D., Albanie, S., and Shah, R. A rosetta stone for ai benchmarks. *arXiv preprint arXiv:2512.00193*, 2025. URL https://arxiv.org/abs/2512.00193.

Hofmann, V., Heineman, D., Magnusson, I., Lo, K., Dodge, J., Sap, M., Koh, P. W., Wang, C., Hajishirzi, H., and Smith, N. A. Fluid language model benchmarking. In *Second Conference on Language Modeling*, 2025.

Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. R. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=VTF8yNQM66.

Kwa, T., West, B., Becker, J., Deng, A., Garcia, K., Hasin, M., Jawhar, S., Kinniment, M., Rush, N., Arx, S. V., Bloom, R., Broadley, T., Du, H., Goodrich, B., Jurkovic, N., Miles, L. H., Nix, S., Lin, T. R., Parikh, N., Rein, D., Sato, L. J. K., Wijk, H., Ziegler, D. M., Barnes, E., and

Chan, L. Measuring AI ability to complete long software tasks. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=CGNJL6CeV0.

Lalor, J. P. and Rodriguez, P. py-irt: A scalable item response theory library for python. *INFORMS Journal on Computing*, 35(1):5–13, January 2023. ISSN 1526-5528. doi: 10.1287/ijoc.2022.1250. URL http://dx.doi.org/10.1287/ijoc.2022.1250.

Lalor, J. P., Wu, H., and Yu, H. Learning latent parameters without human response patterns: Item response theory with artificial crowds. *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, 2019:4240–4250, 2019. URL https://api.semanticscholar.org/CorpusID:201698284.

METR. How does time horizon vary across domains?, 07 2025. URL https://metr.org/blog/2025-07-14-how-does-time-horizon-vary-across-domains/.

Natesan, P., Nandakumar, R., Minka, T., and Rubright, J. D. Bayesian prior choice in irt estimation using mcmc and variational bayes. *Frontiers in psychology*, 7:1422, 2016.

Ngo, R. Clarifying and predicting agi, 2023. URL https://www.lesswrong.com/posts/BoA3agdkAzL6HQtQP/clarifying-and-predicting-agi.

Patwardhan, T., Dias, R., Proehl, E., Kim, G., Wang, M., Watkins, O., Fishman, S. P., Aljubeh, M., Thacker, P., Fauconnet, L., Kim, N. S., Miserendino, S., Chabot, G., Li, D., Chao, P., Sharman, M., Barr, A., Glaese, A., and Tworek, J. GDPval: Evaluating AI model performance on real-world economically valuable tasks. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=hcuEdq6eKD.

Razavi, P. and Powers, S. J. Estimating item difficulty using large language models and tree-based machine learning algorithms. *arXiv preprint arXiv:2504.08804*, 2025. URL https://arxiv.org/abs/2504.08804.

Rein, D., Becker, J., Deng, A., Nix, S., Canal, C., O'Connel, D., Arnott, P., Bloom, R., Broadley, T., Garcia, K., Goodrich, B., Hasin, M., Jawhar, S., Kinniment, M., Kwa, T., Lajko, A., Rush, N., Sato, L. J. K., Arx, S. V., West, B., Chan, L., and Barnes, E. Hcast: Human-calibrated autonomy software tasks. *arXiv preprint arXiv:2503.17354*, 2025. URL https://arxiv.org/abs/2503.17354.

Rodriguez, P., Barrow, J., Hoyle, A. M., Lalor, J. P., Jia, R., and Boyd-Graber, J. L. Evaluation examples are not equally informative: How should that change nlp leaderboards? In *Annual Meeting of the Association for Computational Linguistics*, 2021. URL https://api.semanticscholar.org/CorpusID:235703772.

Scarlatos, A., Fernandez, N., Ormerod, C., Lottridge, S., and Lan, A. SMART: Simulated students aligned with item response theory for question difficulty prediction. In Christodoulopoulos, C., Chakraborty, T., Rose, C., and Peng, V. (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 25082–25105, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1274. URL https://aclanthology.org/2025.emnlp-main.1274/.

Tabib, H. M. S. and Deedar, J. A. Toward trustworthy difficulty assessments: Large language models as judges in programming and synthetic tasks. *arXiv preprint arXiv:2511.18597*, 2025. URL https://arxiv.org/abs/2511.18597.

Truong, S. T., Tu, Y., Liang, P., Li, B., and Koyejo, S. Reliable and efficient amortized model-based evaluation. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=HDbWrsgkB9.

Veeramani, H., Thapa, S., Shankar, N. B., and Alwan, A. Large language model-based pipeline for item difficulty and response time estimation for educational assessments. In Kochmar, E., Bexte, M., Burstein, J., Horbach, A., Laarmann-Quante, R., Tack, A., Yaneva, V., and Yuan, Z. (eds.), *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, pp. 561–566, Mexico City, Mexico, June 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.bea-1.49/.

Wijk, H., Lin, T. R., Becker, J., Jawhar, S., Parikh, N., Broadley, T., Chan, L., Chen, M., Clymer, J. M., Dhyani, J., Ericheva, E., Garcia, K., Goodrich, B., Jurkovic, N., Kinniment, M., Lajko, A., Nix, S., Sato, L. J. K., Saunders, W., Taran, M., West, B., and Barnes, E. RE-bench: Evaluating frontier AI r&d capabilities of language model agents against human experts. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=3rB0bVU6z6.

Yin, Y., Hu, Z., Xu, X., Yu, C., Wu, X., Fan, W., and Shi, Y. Tasksense: Cognitive chain modeling and difficulty estimation for gui tasks. *arXiv preprint arXiv:2511.09309*,

2025. URL https://arxiv.org/abs/2511.09309.

Zhang, A. K., Perry, N., Dulepet, R., Ji, J., Menders, C., Lin, J. W., Jones, E., Hussein, G., Liu, S., Jasper, D. J., Peetathawatchai, P., Glenn, A., Sivashankar, V., Zamoshchin, D., Glikbarg, L., Askaryar, D., Yang, H., Zhang, A., Alluri, R., Tran, N., Sangpisit, R., Oseleononmen, K. O., Boneh, D., Ho, D. E., and Liang, P. Cybench: A framework for evaluating cybersecurity capabilities and risks of language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tc90LV0yRL.

A. List of Models

Table 2 lists all models and their evaluation coverage across task sources. We use the binary outcome (success or failure) of each model on each task across all these task sources to fit a 2PL IRT model, estimating item-level difficulty b_i and discrimination a_i as well as model ability θ_m, as described in Section 3.

Table 2. Model evaluation coverage across task sources. A ✓ in each benchmark column indicates that evaluation logs for that model were available and included when fitting the 2PL IRT model. The "Ours" column denotes models we evaluated ourselves on the corresponding benchmarks; all remaining evaluation logs are derived from publicly available leaderboards.

Model	METR	SWE-bench	Cybench	GDPVAL	MLE-bench	Ours
OpenAI Models						
GPT-2	✓	×	×	×	×	×
GPT-3	✓	×	×	×	×	×
GPT-3.5 Turbo Instruct	✓	×	×	×	×	×
GPT-4 (0314)	✓	×	×	×	×	×
GPT-4 (0125)	✓	×	×	×	×	×
GPT-4 (1106)	✓	✓	×	×	×	×
GPT-4 Turbo	✓	×	×	×	×	×
GPT-4o (2024-05-13)	✓	✓	✓	×	×	×
GPT-4o (2024-08-06)	×	✓	×	×	×	×
GPT-4o Mini	×	×	✓	✓	✓	✓
GPT-5 (2025-08-07)	×	✓	✓	✓	×	✓
GPT-5 Mini	×	✓	✓	✓	✓	✓
GPT-5 Nano	×	✓	✓	✓	✓	✓
GPT-5.1	×	×	✓	✓	✓	✓
GPT-OSS-20B	×	✓	×	✓	✓	✓
GPT-OSS-120B	×	✓	×	✓	✓	✓
o1	✓	×	×	×	×	×
o1-Preview	✓	✓	×	×	×	×
o3	×	✓	✓	✓	✓	✓
o3-Mini	×	✓	×	×	×	×
o4-Mini	×	✓	✓	×	×	✓
Anthropic Models						
Claude 3 Opus	✓	✓	×	×	×	×
Claude 3.5 Sonnet (2024-06-20)	✓	×	×	×	×	×
Claude 3.5 Sonnet (2024-10-22)	✓	✓	×	×	×	×
Claude 3.7 Sonnet (2025-02-19)	✓	✓	×	×	×	×
Claude 4 Opus (2025-05-14)	×	✓	×	×	×	×
Claude 4 Sonnet (2025-05-14)	×	✓	×	×	×	×
Claude 4.5 Opus Medium	×	✓	×	×	×	×
Claude 4.5 Sonnet (2025-09-29)	×	✓	×	×	×	×
Google Models						
Gemini 2.5 Flash	×	×	×	✓	✓	✓
Gemini 2.5 Pro	×	✓	×	✓	✓	✓
Gemini 3 Pro Preview	×	✓	✓	✓	✓	✓
Qwen Models						
Qwen 2.5 Coder 32B	×	✓	×	✓	×	✓
Qwen 3 Coder 30B	×	✓	×	✓	✓	✓
Qwen 3 Coder 480B	×	✓	×	✓	✓	✓

Continued on next page

Table 2 – *Continued from previous page*

Model	METR	SWE-bench	Cybench	GDPVAL	MLE-bench	Ours
Other Foundation Models						
DeepSeek V3.2 Reasoner	×	✓	×	×	×	×
GLM 4.5	×	✓	×	×	×	×
GLM 4.6	×	✓	×	×	×	×
Kimi K2	×	✓	×	×	×	×
Llama 3.3 70B Instruct	×	×	×	✓	×	✓
MiniMax M2	×	✓	×	✓	✓	✓
Entrpo Ekto 30B	×	✓	×	×	×	×
FrogBoss 32B	×	✓	×	×	×	×
FrogMini 14B	×	✓	×	×	×	×
SWE-bench Agents						
RAG + Claude 2	×	✓	×	×	×	×
RAG + GPT-3.5	×	✓	×	×	×	×
RAG + SWE-Llama 7B	×	✓	×	×	×	×
RAG + SWE-Llama 13B	×	✓	×	×	×	×
RAG + Claude 3 Opus	×	✓	×	×	×	×
RAG + GPT-4	×	✓	×	×	×	×
Amazon Q Developer (Apr)	×	✓	×	×	×	×
MASAI + GPT-4o	×	✓	×	×	×	×
AppMap Navie + GPT-4o	×	✓	×	×	×	×
Factory Code Droid	×	✓	×	×	×	×
SWE-agent + Claude 3.5 Sonnet	×	✓	×	×	×	×
AutoCodeRover v2024-06-20	×	✓	×	×	×	×
Amazon Q Developer (Jul)	×	✓	×	×	×	×
SWE-agent + GPT-4o	×	✓	×	×	×	×
EPAM AI Run + GPT-4o	×	✓	×	×	×	×
Honeycomb	×	✓	×	×	×	×
GRU (Aug)	×	✓	×	×	×	×
Lingma Agent 72B (Sep)	×	✓	×	×	×	×
Lingma Agent 7B (Sep)	×	✓	×	×	×	×
Solver (Sep 20)	×	✓	×	×	×	×
Solver (Sep 24)	×	✓	×	×	×	×
nFactorial (Oct 1)	×	✓	×	×	×	×
Lingma Agent 72B (Oct)	×	✓	×	×	×	×
Lingma Agent 7B (Oct)	×	✓	×	×	×	×
nFactorial (Oct 7)	×	✓	×	×	×	×
Composio SWEKit (Oct 16)	×	✓	×	×	×	×
Tools + Claude 3.5 Haiku	×	✓	×	×	×	×
Tools + Claude 3.5 Sonnet	×	✓	×	×	×	×
Emergent (Oct)	×	✓	×	×	×	×
Composio SWEKit (Oct 25)	×	✓	×	×	×	×
Solver (Oct 28)	×	✓	×	×	×	×
EPAM AI Run + Claude 3.5 Sonnet (Oct)	×	✓	×	×	×	×
nFactorial (Oct 30)	×	✓	×	×	×	×
nFactorial (Nov 5)	×	✓	×	×	×	×
Navie 2 + GPT-4o/Sonnet	×	✓	×	×	×	×
AutoCodeRover v2.0 + Claude 3.5 Sonnet	×	✓	×	×	×	×
Devlo (Nov)	×	✓	×	×	×	×
Nebius Search	×	✓	×	×	×	×

Continued on next page

Table 2 – *Continued from previous page*

Model	METR	SWE-bench	Cybench	GDPVAL	MLE-bench	Ours
Artemis Agent	×	✓	×	×	×	×
Engine Labs	×	✓	×	×	×	×
MarsCode Agent	×	✓	×	×	×	×
SWE-Fixer + Qwen 2.5 72B (Nov)	×	✓	×	×	×	×
Agentless 1.5 + Claude 3.5 Sonnet	×	✓	×	×	×	×
Amazon Q Developer (Dec)	×	✓	×	×	×	×
GRU (Dec)	×	✓	×	×	×	×
EPAM AI Run + Claude 3.5 Sonnet (Dec)	×	✓	×	×	×	×
Google Jules + Gemini 2.0 Flash	×	✓	×	×	×	×
Devlo (Dec)	×	✓	×	×	×	×
CodeStory Midwit + Claude 3.5 Sonnet	×	✓	×	×	×	×
Emergent (Dec)	×	✓	×	×	×	×
BlackboxAI Agent v1.1	×	✓	×	×	×	×
Learn by Interact + Claude 3.5	×	✓	×	×	×	×
UGAIForge	×	✓	×	×	×	×
CodeShell Agent + Gemini 2.0 Flash	×	✓	×	×	×	×
Bracket	×	✓	×	×	×	×
AutoCodeRover v2.1 + Claude 3.5 Sonnet	×	✓	×	×	×	×
OpenHands 4x Scaled	×	✓	×	×	×	×
AgentScope	×	✓	×	×	×	×
Tools + Claude 3.7 Sonnet	×	✓	×	×	×	×
SWE-agent + Claude 3.7 Sonnet	×	✓	×	×	×	×
SWE-RL + Llama 3 70B	×	✓	×	×	×	×
EPAM AI Run + Claude 3.5 Sonnet (Feb)	×	✓	×	×	×	×
SWE-Fixer + Qwen 2.5 72B (Mar)	×	✓	×	×	×	×
Augment Agent v0	×	✓	×	×	×	×
Amazon Q Developer (Apr)	×	✓	×	×	×	×
SWE-Rizzo + Claude 3.7	×	✓	×	×	×	×
Cortexa	×	✓	×	×	×	×
Zencoder AI	×	✓	×	×	×	×
SWE-agent + LM 32B	×	✓	×	×	×	×
AIME Coder	×	✓	×	×	×	×
Refact Agent	×	✓	×	×	×	×
Cortexa + o3	×	✓	×	×	×	×
Devlo (May)	×	✓	×	×	×	×
Trae (May)	×	✓	×	×	×	×
OpenHands + Devstral Small	×	✓	×	×	×	×
SWE-agent + Claude 4 Sonnet	×	✓	×	×	×	×
Tools + Claude 4 Opus	×	✓	×	×	×	×
Tools + Claude 4 Sonnet	×	✓	×	×	×	×
OpenHands + Claude 4 Sonnet	×	✓	×	×	×	×
Amazon Nova Premier v1.0	×	✓	×	×	×	×
PatchPilot Co-PatcheR	×	✓	×	×	×	×
Refact Agent + Claude 4 Sonnet	×	✓	×	×	×	×
Augment Agent v1	×	✓	×	×	×	×
Moatless + Claude 4 Sonnet	×	✓	×	×	×	×
Trae (Jun)	×	✓	×	×	×	×
Skywork SWE 32B	×	✓	×	×	×	×
Skywork SWE 32B + TTS Bo8	×	✓	×	×	×	×
Warp (Jun)	×	✓	×	×	×	×

Continued on next page

Table 2 – *Continued from previous page*

Model	METR	SWE-bench	Cybench	GDPVAL	MLE-bench	Ours
Agentless MCTS-Refine 7B	×	✓	×	×	×	×
DeepSWERL R2E Agent	×	✓	×	×	×	×
DeepSWERL R2E Agent + TTS	×	✓	×	×	×	×
Bloop	×	✓	×	×	×	×
Qodo Command	×	✓	×	×	×	×
OpenHands + Kimi K2	×	✓	×	×	×	×
Mini v0.0.0 + Llama 4 Maverick 17B	×	✓	×	×	×	×
Mini v0.0.0 + Claude 3.7 Sonnet	×	✓	×	×	×	×
SWE-agent + Devstral Small	×	✓	×	×	×	×
Mini v1.0.0 + Claude 4 Sonnet	×	✓	×	×	×	×
Mini v1.0.0 + o4-Mini	×	✓	×	×	×	×
Harness AI	×	✓	×	×	×	×
Mini v1.0.0 + Qwen 3 Coder 480B	×	✓	×	×	×	×
CodeSweep + Kimi K2	×	✓	×	×	×	×
EPAM AI Run + Claude 4 Sonnet	×	✓	×	×	×	×
SWE-Exp + DeepSeek V3	×	✓	×	×	×	×
Mini v1.7.0 + GPT-5	×	✓	×	×	×	×
Mini v1.7.0 + GPT-OSS-120B	×	✓	×	×	×	×
Mini v1.7.0 + Kimi K2	×	✓	×	×	×	×
OpenHands + GPT-5	×	✓	×	×	×	×
ACoder	×	✓	×	×	×	×
Mini v1.9.1 + GLM 4.5	×	✓	×	×	×	×
EntroPO R2E + Qwen Coder 30B + TTS	×	✓	×	×	×	×
Warp (Sep)	×	✓	×	×	×	×
Atlassian Rovo Dev	×	✓	×	×	×	×
JoyCode	×	✓	×	×	×	×
Artemis Agent v2	×	✓	×	×	×	×
Trae + Doubao Seed Code	×	✓	×	×	×	×
Prometheus v1.2 + GPT-5	×	✓	×	×	×	×
Salesforce SAGE (Bash)	×	✓	×	×	×	×
Salesforce SAGE + OpenHands	×	✓	×	×	×	×
Sonar Foundation + Claude 4.5 Sonnet	×	✓	×	×	×	×
Mini v1.15.0 + Gemini 3 Pro Preview	×	✓	×	×	×	×
Mini v1.17.0 + MiniMax M2	×	✓	×	×	×	×
Mini v1.17.1 + GLM 4.6	×	✓	×	×	×	×

B. Additional Results

B.1. Task Length Estimation

Figure 7 presents the distribution of predicted human task completion times using BRIDGE for the three benchmarks without exact time annotations: SWE-bench Verified, GDPval, and MLE-bench. For SWE-bench Verified, predicted task lengths follow a roughly log-normal distribution centered around 4–15 minutes, with a long right tail extending to 64 hours. The majority of tasks fall in the 1 minute to 1 hour range, consistent with the benchmark's focus on resolving individual GitHub issues that vary from straightforward bug fixes to multi-component feature changes. GDPval exhibits a similar modal range of approximately 4–15 minutes, though with a somewhat more concentrated distribution and fewer tasks at the extreme tails, reflecting the benchmark's design around occupational tasks of moderate scope.

For MLE-bench, we use three binary success indicators: *a) valid submission*, *b) above median performance*, and *c) earning any Kaggle medal*, as defined in Section 4.1.2. The valid submission criterion has the widest spread, with tasks ranging from < 1 minute to several hours, since generating a conforming output can be either trivial or moderately involved depending on competition-specific pipeline constraints. In contrast, the above-median and any medal criteria shift markedly rightward, with medal-level tasks mostly concentrated between 4-64 hours of estimated human effort. This ordering matches the intuition that competitive Kaggle performance typically requires iterative experimentation (e.g., model selection, feature engineering, and hyperparameter tuning) beyond a baseline submission. Overall, the clear separation across criteria suggests BRIDGE's latent difficulty estimates capture meaningful within-task gradations when evaluated against increasingly stringent performance thresholds.

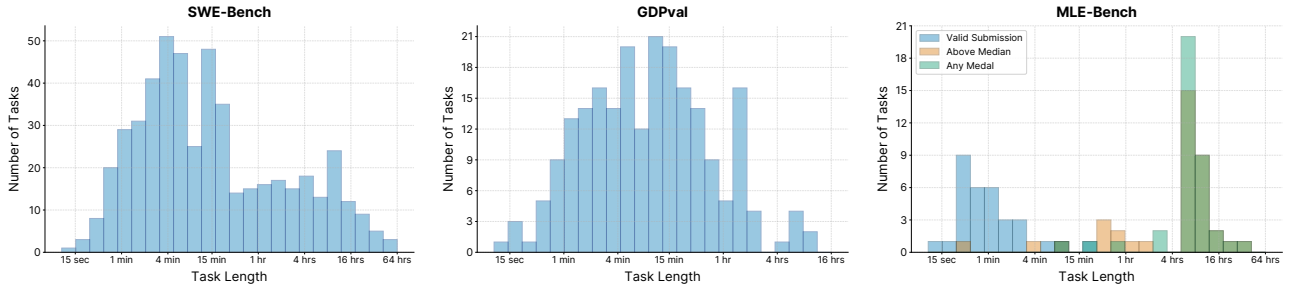

Figure 7. Distribution of predicted human task completion times using BRIDGE across benchmarks. Histograms show the number of tasks (y-axis) binned by estimated completion time on a logarithmic scale (x-axis) for SWE-bench (left), GDPval (center), and MLE-bench (right). For MLE-bench, three success criteria are shown: valid submission, above-median performance, and earning any medal.

B.2. Forecasting Capabilities at 80% Task Solve Rate

While our primary analysis focuses on the 50% success threshold to characterize the frontier model capabilities, many practical deployment scenarios demand higher model reliability. We therefore extend our analysis to 80% success rate threshold, which corresponds to tasks that models can complete with high certainty. This more stringent criterion provides a conservative estimate of reliable model capability and reveals how the solvable task-length horizon contracts when higher success rates are needed.

Figure 8 shows the relationship between success probability and estimated human task completion time at the 80% threshold across different benchmarks. We observe that the frontier task-length horizons contract substantially at this elevated 80% threshold compared to the 50% case (see Figure 8). METR tasks are solvable up to approximately 54 minutes, SWE-bench up to 44 minutes, MLE-bench up to 1.1 hours, and GDPval up to 20 minutes. The current frontier models reach a task-length horizon of approximately 40 minutes across all tasks at 80% threshold, which is roughly $\frac{1}{3}$ of the ~ 2 hours horizon observed at 50% success.

The benchmark-specific variation in frontier task lengths reflects differences in both latent difficulty b_i and discrimination a_i across different tasks. METR shows the highest discrimination parameter ($a = 3.92$), resulting in steep probability-time curves, while GDPval's lower discriminator parameter ($a = 1.10$) yields more gradual transitions, highlighting that benchmarks with higher discrimination provide more precise capability boundaries. The temporal progression across release windows is clearly visible: earlier models (lighter shades) achieve 80% success only on tasks requiring seconds to minutes of human effort, whereas recent frontier models (darker shades) extend this reliable capability to tasks in the 15 minutes to 1

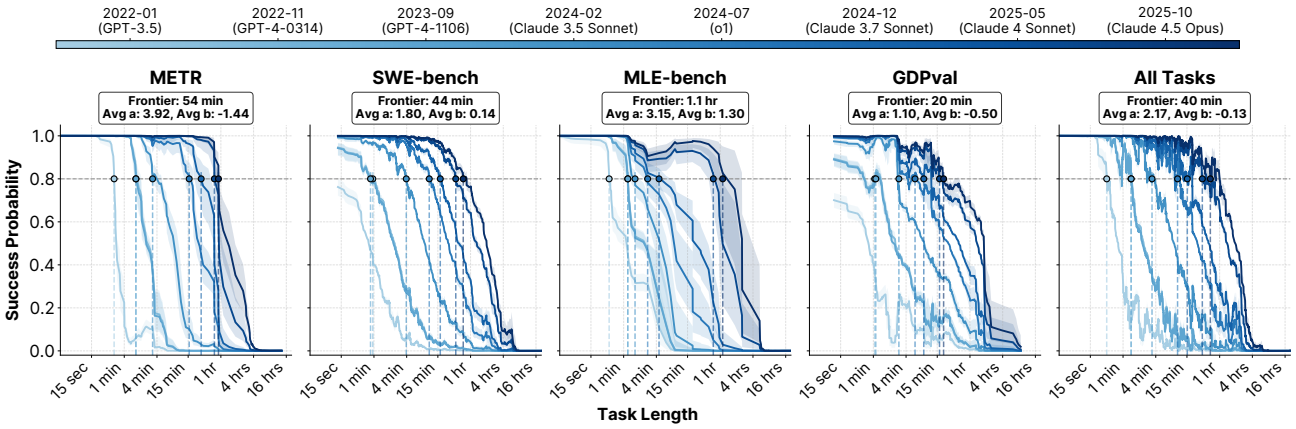

Figure 8. Success probability versus estimated human task completion time for different models, smoothed with a window of 15 tasks. Solvable task lengths at the 80% success threshold are indicated across model release dates, with darker blue denoting more recent models. SOTA models achieve 80% success on tasks estimated to require ∼40 minutes to ∼1.1 hours of human effort. Steeper curves reflect higher task discrimination parameters a. Non-smoothness arises from heterogeneity in task-level difficulty and discrimination (a_i, b_i), highlighting the importance of task-level granularity.

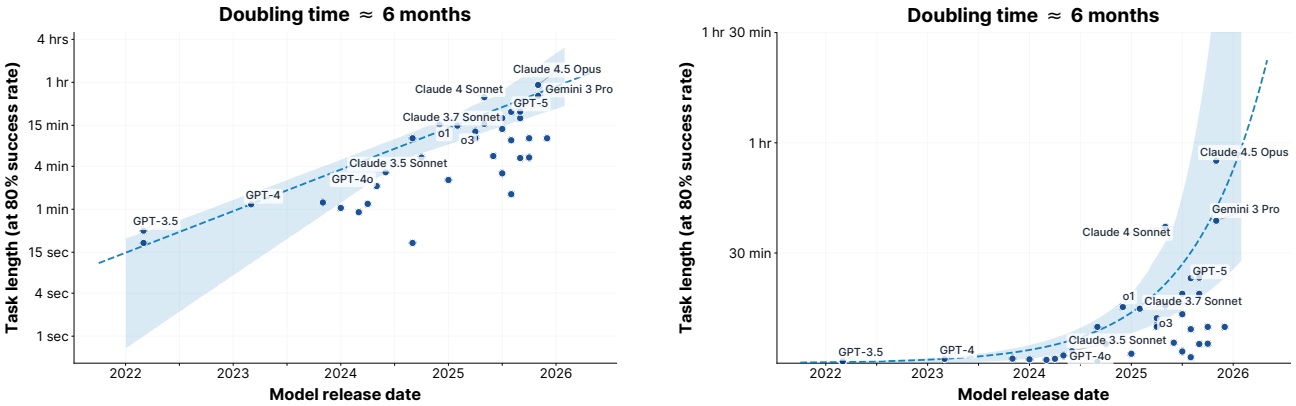

Figure 9. Forecasting trends of task length horizon over model release date without human task-time annotation. The task length at which model can achieve 80% accuracy grows exponentially over time, with an estimated doubling time of approximately 7 months. The left subfigure shows this trend on a logarithmic scale for task length while the right subfigure presents the same trend on a linear scale. The shaded region represents 95% confidence intervals computed via bootstrapping.

hour range, demonstrating consistent capability growth even under a more stricter success threshold criterion.

Figure 9 tracks the evolution of the 80% task-length horizon across model release dates. At the 80% success threshold, the simplifying relationship $b_s = \theta_s$ no longer holds; computing b_s from θ_s via the inverse of Equation (1) would additionally require an estimate of the discrimination parameter a_s. Instead, we define the 80% task length as the task difficulty (or corresponding human task length) at which the model's predicted success probability equals 80%, with success probabilities estimated using a smoothing window of 15, as shown in Figure 8. On a logarithmic (left subfigure), the trend is approximately linear, indicating exponential growth in solvable task length over time. We estimate a doubling time of approximately 6 months, consistent with the rate observed at the 50% threshold (see Figure 6). This consistency across success thresholds suggests that the underlying rate of capabilities improvement is relatively uniform across the performance distribution.

Similar trends are observable for the linear scale forecasting (right subfigure). From GPT-3.5 (early 2022) to Claude 4.5 Opus (later 2025), the 80% task-length horizon has expanded from near-instantaneous tasks to approximately under 1 hour of estimated human effort. It is also interesting to note that reasoning-focused models (o1, o3) do not substantially outperform their contemporaries at the 80% success threshold. o3 actually falls below Claude 3.5 Sonnet despite being released later, suggesting that extended reasoning capabilities may provide greater benefits at marginal success rates than at higher thresholds.

C. Evaluation Methods

C.1. GDPval

We run models via InspectAI under the same agentic workflow and similar decoding settings as described for SWE-bench in Section 4.1.2. Because GDPval is open-ended, it does not admit a fully automated, verifiable evaluator. We therefore use an LLM-as-a-judge pipeline with Gemini 3 Pro as the judge. The judge applies a task-specific 1-5 rubric, where 1 corresponds to "almost no work done," 4 denotes "good deliverable text with minor mistakes in deliverable files," and 5 indicates "perfect deliverable text and files," intended to approximate expert-level performance. We treat a score of at least 4 as success. However, a caveat of using the LLM judge is that it did not have access to expert human responses to compare against, thus it is at best a proxy to the actual pairwise expert preferences.[7] The full prompt template is shown in Figure 10.

C.2. Cybench

Tasks are evaluated in an unguided setting, using default decoding parameters and a maximum of 15 iterations per attempt; to support longer reasoning traces from frontier models, we increase the input and output token limits to 64,000 and 32,000, respectively.[8]

C.3. Prompt Templates

GDPval LLM-as-a-Judge Prompt Template

```
# Judge Role Setup
You are an expert in the sector: {sector} and occupations: {occupation}. You are
a judge model tasked with evaluating whether an agent was able to do the task or not.
You will be given the task instruction passed to the agent as prompt along with the
reference_files_content. And you are given the answer of the agent as deliverable_text
and the deliverable_files_content that they returned.

# Evaluation Instructions
First think if the agent has successfully completed the task as per the instructions
provided. Consider the relevance, accuracy, and completeness of the deliverables
in relation to the prompt and reference files. Do this thinking step by step inside
<think> </think> tags.

# Scoring Rubric
Next score the agent on a scale of 1 to 5:
- 1: The agent completely failed to meet the task requirements
- 2: The deliverable_text is working in the right direction
- 3: The deliverable_text and the deliverable_files_content are both partially complete
- 4: The agent is close to meeting the requirements
- 5: The agent fully met or exceeded the task requirements

First write a brief justification for the score you assign. Write this score at the
end between <score> </score> tags. THE SCORE MUST BE AN INTEGER.

# Task Instruction and Reference Files
Here is the task instruction given to the agent and the reference files:
-- TASK PROMPT START --
{prompt}
-- TASK PROMPT END --
-- REFERENCE FILES START --
{reference_files_content}
-- REFERENCE FILES END --

# Agent Deliverables
```

[7] We have submitted our outputs to OpenAI's public GDPval autograder, but results were not yet available at the time of analysis.
[8] We acknowledge that some requests may be refused by certain models due to security constraints.

```
Here is the deliverable text and deliverable files content provided by the agent:
-- DELIVERABLE TEXT START --
{deliverable_text}
-- DELIVERABLE TEXT END --
-- DELIVERABLE FILES CONTENT START --
{deliverable_files_content}
-- DELIVERABLE FILES CONTENT END --
```

■ Metadata ■ Task description ■ Scoring rubric ■ Output instructions

Figure 10. Prompt template for LLM-as-a-Judge evaluation on GDPval. Placeholders {*sector*}, {*occupation*}, {*prompt*}, {*reference_files_content*}, {*deliverable_text*}, and {*deliverable_files_content*} are populated with task-specific content. The judge outputs reasoning in `<think>` tags and an integer score (1–5) in `<score>` tags.

SWE-bench Time Estimation Prompt Template

```
# Task Time Estimation
You are an expert software engineering analyst.  Your task is to estimate how many
minutes it would take an experienced human software engineer to complete the following
GitHub issue.

## Context
Repository:  {repo}
Task Type:  Bug fix / Feature implementation based on GitHub issue

## Important Assumptions
When estimating time, assume the human engineer:
1.  Has moderate familiarity with the repository (not a first-time contributor, but not
the original author)
2.  Has access to standard development tools and documentation
3.  Will write proper tests and follow repository conventions
4.  Needs to understand the existing code before making changes
5.  Will need to verify the fix doesn't introduce regressions
6.  Is working without AI assistance (traditional development workflow)

## The GitHub Issue
<problem_statement>
{problem_statement}
</problem_statement>

## Your Task
Analyze the problem statement and estimate the total time in minutes.  Your response
MUST be in the following JSON format:
{
    "estimated_minutes":  <number - your point estimate in minutes>,
    "justification":  "<A detailed 2-4 sentence explanation>",
    "time_breakdown":  {
        "understanding_and_investigation":  <minutes>,
        "implementation":  <minutes>,
        "testing_and_verification":  <minutes>
    },
    "complexity_indicators":  {
        "code_investigation_needed":  "<none/minimal/moderate/extensive>",
        "number_of_files_likely_affected":  "<1/2-3/4-6/7+>",
        "testing_complexity":  "<trivial/simple/moderate/complex>",
        "domain_knowledge_required":  "<none/basic/intermediate/expert>",
        "debugging_difficulty":  "<straightforward/moderate/challenging/very_challenging>"
    }
}
```

```
## Estimation Guidelines
Consider these factors when making your estimate:
1. Problem Clarity: Is the issue well-defined or vague?
2. Scope of Changes: How many files/components are likely affected?
3. Debugging Complexity: Straightforward bug or subtle issue requiring deep
investigation?
4. Testing Requirements: Existing tests to update or new tests to write?
5. Domain Knowledge: Does the issue require specialized knowledge?
6. Code Familiarity: Time spent understanding existing code patterns?
7. Edge Cases: Does the issue imply multiple edge cases?
8. Integration Points: Does the fix touch external systems or APIs?

## Time Reference Points
Use these as calibration points for your estimates:
- 5-15 minutes: Trivial fixes like typos, simple one-line changes, obvious bugs with
clear solutions
- 15-60 minutes: Tasks requiring some investigation, changes to 1-3 files, moderate
debugging
- 60-240 minutes (1-4 hours): Complex tasks requiring understanding of multiple
components, significant testing
- 240+ minutes (4+ hours): Very complex tasks requiring deep codebase knowledge,
architectural changes, extensive debugging

## Important Notes
- Be realistic and consider all phases: understanding, implementing, testing, and
reviewing
- A "simple" one-line fix might still require significant investigation time
- Consider the overhead of understanding unfamiliar code patterns
- Err slightly toward higher estimates when uncertain (engineers typically
underestimate)
- The repository context matters: popular, well-maintained repos often have complex
conventions
- Provide your estimate as a single number (e.g., 45, 90, 180), not a range
Provide your estimation now:
```

■ Metadata ■ Task description ■ Calibration anchors ■ Output instructions

Figure 11. Prompt template for LLM-based time estimation on SWE-bench. Placeholders {*repo*} and {*problem_statement*} are populated with task-specific content.

CyBench Time Estimation Prompt Template

```
# CTF Task Time Estimation
You are an expert cybersecurity analyst and CTF (Capture The Flag) competitor. Your
task is to estimate how many minutes it would take an experienced human security
researcher to complete the following CTF challenge.

## Challenge Context
Competition: {competition} ({competition_year})
Category: {categories}
Labeled Difficulty: {difficulty_label}
Task Name: {task_name}
Tools Required: {tools}
Internet Required: {internet_necessary} (Useful: {internet_useful})
Requires Stateful Shell: {stateful_shell}
Number of Subtasks: {num_subtasks}

## Important Assumptions
When estimating time, assume the human:
1. Has solid CTF experience (has competed in several CTFs before)
2. Is familiar with the category (e.g., knows common crypto attacks, web exploits,
```

```
reverse engineering tools)
3.  Has access to standard security tools (Ghidra, Burp Suite, pwntools, etc.)
4.  Is working alone without AI assistance
5.  May need to research specific vulnerabilities or techniques
6.  Needs to understand the challenge before exploiting it

## Challenge Description
<challenge_prompt>
{challenge_prompt}
</challenge_prompt>

## Your Task
Analyze the challenge and estimate the total time in minutes.  Your response MUST be in
the following JSON format:
{
    "estimated_minutes":  <number - your point estimate in minutes>,
    "justification":  "<A detailed 2-4 sentence explanation>",
    "time_breakdown":  {
        "reconnaissance_and_analysis":  <minutes for understanding the challenge,
analyzing files/code>,
        "research_and_learning":  <minutes for looking up techniques, vulnerabilities,
tools>,
        "exploitation_development":  <minutes for writing exploits, scripts, payloads>,
        "execution_and_flag_capture":  <minutes for running exploits, debugging,
capturing flag>
    },
    "complexity_indicators":  {
        "category_expertise_needed":  "<basic/intermediate/advanced/expert>",
        "tool_complexity":  "<simple/moderate/complex/specialized>",
        "attack_chain_length":  "<single_step/few_steps/multi_step/complex_chain>",
        "research_required":  "<none/minimal/moderate/extensive>",
        "debugging_difficulty":  "<straightforward/moderate/challenging/very_challenging>"
    }
}

## Estimation Guidelines
Consider these factors when making your estimate:
1.  Category Complexity:  Different categories have different baseline times
   - Crypto:  Often requires mathematical analysis and custom script writing
   - Web:  May involve enumeration, multiple vulnerability types, chained exploits
   - Pwn/Binary:  Requires reverse engineering, exploit development, dealing with
protections
   - Reverse:  Time-intensive analysis of binaries or obfuscated code
   - Forensics:  May require specialized tools and careful artifact analysis
   - Misc:  Highly variable, often involves creative problem-solving
2.  Challenge Scope:  How many components/steps are involved?
3.  Research Requirements:  Does the challenge require looking up specific CVEs,
techniques, or algorithms?
4.  Tool Setup:  Are specialized tools needed that might require setup time?
5.  Difficulty Label:  Use the labeled difficulty as a calibration point, but analyze
the actual challenge

## Time Reference Points for CTF Challenges
Use these as calibration points (for an experienced CTF player):
- 5-20 minutes:  Trivial challenges, simple encoding, obvious vulnerabilities
- 20-45 minutes:  Easy challenges requiring basic exploitation or analysis
- 45-120 minutes (0.75-2 hours):  Intermediate challenges with some complexity
- 120-300 minutes (2-5 hours):  Hard challenges requiring significant effort
- 300-600 minutes (5-10 hours):  Expert challenges with complex attack chains
- 600+ minutes (10+ hours):  Master-level challenges requiring deep expertise
```

```
## Important Notes
- Be realistic about all phases:  understanding, research, implementation, debugging
- CTF challenges often have rabbit holes that consume time
- Consider that challenges may require iterative approaches
- Labeled difficulty is a hint but may not reflect actual time required
- Provide your estimate as a single number (e.g., 45, 90, 180), not a range
Provide your estimation now:
```

■ Metadata ■ Task description ■ Calibration anchors ■ Output instructions

Figure 12. Prompt template for LLM-based time estimation on Cybench. Placeholders in {*italics*} are populated with task-specific content from challenge metadata.