LET AGENTS KNOW HOW HARD THIS WORK IS

Anonymous authors
Paper under double-blind review

ABSTRACT

In LLM-based agent system, effective task planning is essential for accurate and successful execution. This requires the agent to accurately understand the complexity of current task and decompose it appropriately. Most of the existing agent systems rely on the inherent capabilities of large language models to address the problem of understanding task complexity. However, because LLMs lack an explicit and measurable task-complexity signal for determining an appropriate planning depth, agent systems may produce inappropriate planning. In this paper, we propose a measurable notion, termed "Task Complexity", for evaluating task difficulty. We define task complexity as: the minimum number of planning steps required for a fixed agent system to complete a given task with a success rate threshold. This notion of Task Complexity provides an agent-dependent characterization of task difficulty and offers empirical insight into the capability limits of an agent system across different tasks. Experiments across multiple tasks and agents show that Task Complexity produces stable task-difficulty rankings, predicts execution success under different planning steps, and reveals systematic differences between the capability boundaries of agent systems. We further demonstrate that Task Complexity can guide task decomposition by adapting planning steps to the demands of individual tasks. These results establish Task Complexity as a measurable foundation for complexity-aware planning and agent-system evaluation.

1 INTRODUCTION

Effective execution in LLM-based agent systems depends not only on selecting appropriate actions but also on decomposing tasks at an appropriate level(Xie et al., 2024a; Qiao et al., 2024b). Existing planning benchmarks show that LLM performance generally declines as task complexity increases (Jansen et al., 2024; Cao et al., 2024). To generate an appropriate plan, an agent must understand not only the task objective but also how difficult the task is relative to its own capabilities. However, existing agent systems typically rely on the implicit reasoning abilities of LLMs to estimate this difficulty. Because LLMs are not provided with an explicit notion of task complexity and are not specifically trained to be complexity-aware, their planning decisions may be poorly calibrated (Erdogan et al., 2025). Consequently, they may under-plan difficult tasks or over-plan simple ones, leading to unreliable execution, unnecessary token consumption, increased latency, and excessive tool usage(Wang et al., 2024).

In this paper, we argue that task difficulty in agentic systems should be understood as an agent-relative and empirically measurable parameter. We introduce the notion of Task Complexity, defined as the minimum number of planning steps required for a fixed agentic system to complete a given task with a specified success rate.

$$C_(A, x) = \min \{k \in N_0 \mid p_A(\text{success} \mid x, k) \geq \tau\} \tag{1}$$

where k denotes the planning steps allocated to the task, success probability $p_A(\text{success} \mid x, k)$ is estimated through repeated independent executions and τ is the target success rate. Our definition transform the study of task difficulty assessment from a subjective or model internal judgment to an operational measurement based on actual agent performance. Under this view, the complexity of a task is not an inherent property of the task alone, but depends on the capabilities, tools, memory, and execution strategy of the agent system(Yang et al., 2024; Trivedi et al., 2024; Zhao et al., 2024).

This definition rests on two assumptions. First, once the agent generates a task plan, each planned step is passed to the executor for actual execution. Second, throughout the complexity evaluation,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

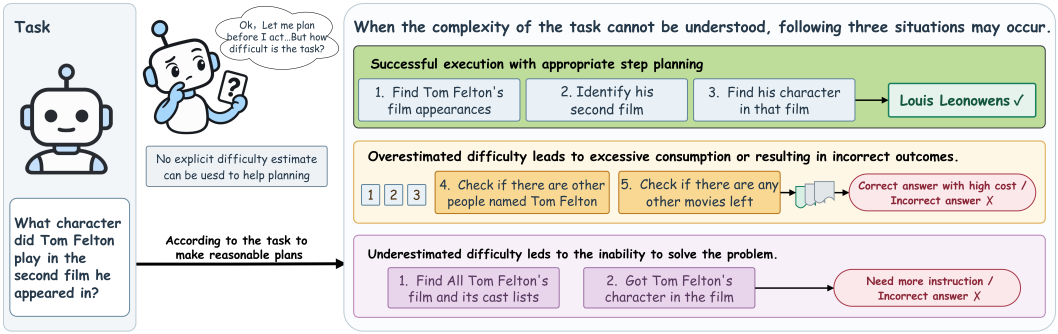

Figure 1: Motivation for Task Complexity. Agent can generate and execute a plan to solve a task, but it lacks an explicit measure of how difficult the task is relative to its own capabilities. This may lead to excessive decomposition or insufficient decomposition.

the agent system is held fixed: its model parameters, configuration, tools, persistent memory remain unchanged across the whole trials. Limitations in long-context utilization (Liu et al., 2024), memory, and tool-use competence (Qin et al., 2024) can therefore produce different complexity values for the same task across different agent systems(Xie et al., 2024b; Huang et al., 2024). This agent-relative formulation enables Task Complexity to characterize and compare system capability boundaries under controlled conditions.

Within a fixed planning interface, Task Complexity can be interpreted as the smallest reliable planning budget for a task. A plan containing too few steps may place excessive actions, dependencies, or reasoning requirements within each step, exceeding the executor's single-step capabilities(Qiao et al., 2024b; Xie et al., 2024a). Conversely, an unnecessarily detailed plan may increase execution cost without providing a corresponding improvement in reliability. The minimum step count that reaches the target success rate therefore identifies the coarsest decomposition that remains reliable for the given agent system.

Our main contributions are as follows:

- We formulate Task Complexity as an operational, objective measure in the planning phase.
- We introduce an empirical measurement for characterizing task difficulty.
- We evaluate the proposed measure in terms of measurement stability, capability-boundary characterization, and complexity-prediction accuracy, and its utility for complexity-aware planning.

2 Related Work

Planning plays a central role in LLM-based agent systems, particularly for tasks involving long execution horizons. A prominent line of recent research tends to explicitly separates planning from execution. Plan-and-Act employs a planner to generate structured high-level plans and an executor to translate them into environment-specific actions, and further improves plan generation through synthetic training data (Erdogan et al., 2025). LLMCompiler further modularizes the agent workflow by explicitly separating task planning from function execution. (Kim et al., 2024).

Other approaches enrich the planning process through more deliberate search or external knowledge. Language Agent Tree Search integrates Monte Carlo Tree Search, value estimation, self-reflection, and environmental feedback to jointly support reasoning, acting, and planning (Zhou et al., 2024), while the World Knowledge Model supplies global task knowledge and dynamically updated state knowledge to improve agent planning (Qiao et al., 2024a). Domain-specific systems further demonstrate the importance of structured planning and execution. (Jiang et al., 2025; Fan et al., 2025; He et al., 2024). These studies establish planning as a critical component of agent systems, but primarily focus on improving plan content, action selection, tool orchestration, or execution efficiency.

Some methods further adapt task decomposition according to execution feedback. ADaPT recursively decomposes a subtask when the executor is unable to complete it, thereby adapting decom-

position depth to both complexity and the capability of the executor model (Prasad et al., 2024). However, complexity in ADaPT remains implicit in the executor's failure and the resulting recursive decomposition process. It is used as a behavioral trigger rather than formulated as an independently measurable quantity. Consequently, it does not provide a complexity value that can be consistently evaluated across tasks or used to characterize the capability boundary of a fixed agent system under a specified reliability requirement.

A parallel line of research develops benchmarks and evaluation frameworks for analyzing agent capabilities(Ma et al., 2024; Shen et al., 2024; Li et al., 2024; Koh et al., 2024). These benchmarks provide increasingly detailed assessments of where an agent succeeds or fails. Nevertheless, they primarily evaluate agent performance using success, progress, stage-specific accuracy, or error categories, rather than defining the difficulty of a task relative to the evaluated agent.

NATURAL PLAN more directly investigates the relationship between task difficulty and planning performance (Zheng et al., 2024). It constructs natural-language planning tasks in trip planning, meeting planning, and calendar scheduling, and controls difficulty through externally specified attributes such as the number of cities, people, attendees, or days. Its results show that model performance generally deteriorates as these structural variables increase. Such benchmark-centric attributes provide interpretable proxies for difficulty and are useful for constructing controlled evaluation sets. However, their complexity are selected by benchmark designers and remain tied to particular task domains.

Task difficulty has also been studied as a signal for adaptive reasoning and resource allocation. Adaptive-RAG trains a classifier to select among no-retrieval, single-step retrieval, and iterative retrieval strategies according to query complexity (Jeong et al., 2024). Adaption-of-Thought estimates question difficulty and uses it to adapt demonstration construction and retrieval for reasoning tasks (Xu et al., 2024). More recently, inference-budget-aware optimization has been used to encourage reasoning models to allocate shorter computations to simple questions and larger budgets to difficult ones (Yu et al., 2025). These methods demonstrate the practical value of complexity-aware computation, but their notions of difficulty are operationalized for choosing among predefined inference or retrieval strategies. They do not aim to establish a general evaluation criterion for the planning requirements of an agent system.

The LLM Already Knows considers difficulty from the internal perspective of a target language model (Zhu et al., 2025). It models token-level generation as a Markov chain and learns a value function over hidden representations to estimate expected output quality from the initial hidden state. This formulation provides an efficient estimate of model-perceived question difficulty and can guide adaptive inference without generating an answer in advance. However, it measures the expected response quality of an individual LLM and depends on access to model-internal representations. In contrast, the difficulty of an agentic task may also depend on the planner, executor, tools, memory, interaction protocol, and environment. Evaluating only the underlying LLM therefore does not fully characterize the difficulty experienced by the complete agent system.

3 METHODOLOGY

3.1 TASK COMPLEXITY DEFINITION AND ASSUMPTIONS

Let A denote a fixed agent system and x a task instance. The agent system includes the underlying LLM, planner, executor, tools, memory mechanism, system prompt, inference parameters, and execution policy. Given a target success-rate threshold τ, we define the Task Complexity of x with respect to A as

$$C_\tau(A, x) = \min \{k \in N_0 \mid S_A(x, k) \geq \tau\}, \tag{2}$$

where k denotes the number of planning steps and

$$S_A(x, k) = P(\text{success} \mid A, x, k) \tag{3}$$

is the probability that agent system A successfully completes task x when using a k-step plan.

A planning step is defined as a non-empty and executable subgoal generated by the planner. Each planning step is passed to the executor and corresponds to one bounded executor invocation. The executor operates under fixed token, action, tool-use, and time budgets. This operational definition prevents an arbitrarily complex sequence of actions from being represented as a single planning step.

The definition rests on two principal assumptions. First, after the planner generates a task plan, every planned step is passed to the executor for actual execution. Second, the evaluated agent system is held fixed throughout complexity measurement. In particular, its model parameters, prompts, tools, persistent memory, planning policy, and execution policy remain unchanged across evaluation trials. Transient states and observations may evolve within an individual execution, but the environment and transient agent state are reset before each repeated trial.

Task Complexity is therefore not the minimum number of steps associated with a single successful execution. Agent behavior may vary because of model sampling, tool uncertainty, or environmental stochasticity. Instead, $C_\tau(A, x)$ represents the smallest planning granularity under which the fixed system achieves the required level of execution reliability.

In an LLM-based agent system, each additional planning step may introduce further prompts, tool calls, execution traces, and summarization context. These things will increase token consumption and consequently add latency and monetary cost. We also conduct a controlled experiment using the CAMEL agent framework on the GSM8K task to examine these token cost. We execute a same task with planning budgets ranging from one to six steps while keeping the agent configuration fixed. Every run produces the correct final answer. We then record the token consumption, calculated as the sum of input and output tokens used during planning, all execution steps, and final summarization.

Table 1: Token consumption in a CAMEL pilot experiment on GSM8K. Token usage includes planning, all execution steps, and the final summary.

Steps	Input Cost	Output Cost	Total Cost
1	12,107	731	12,838
2	14,490	466	14,956
3	21,806	749	22,555
4	28,601	895	29,496
5	39,162	1,054	40,216
6	42,797	1,200	43,997

As shown in Table 1, total token consumption increases monotonically from 12,838 tokens with one planning step to 43,997 tokens with six planning steps. Because the task and final-answer correctness are held constant, this comparison shows that using more planning steps can substantially increase execution cost without improving the outcome performance. Thus, when multiple planning budgets satisfy the same target success-rate threshold, preferring the smallest qualifying budget is both effective and efficient.

3.2 Complexity Measurement

Since the true success probability $S_A(x, k)$ is generally unknown, we estimate it through repeated executions. For each candidate planning-step count k, we independently execute the agent N times and compute

$$\widehat{S}_A(x, k) = \frac{1}{N} \sum_{i=1}^{N} I\left[\mathcal{E}(A_i(x, k)) = 1\right], \tag{4}$$

where $\mathcal{E}$ is a fixed task-success evaluator and $I[\cdot]$ is the indicator function. The empirical Task Complexity is then

$$\widehat{C}_\tau(A, x) = \min\left\{k \in \mathcal{K} \mid \widehat{S}_A(x, k) \geq \tau\right\}, \tag{5}$$

where $\mathcal{K} = \{1, 2, \ldots, K_{\max}\}$ is the evaluated range of planning-step counts.

The measurement procedure evaluates every candidate k in $\mathcal{K}$ and selects the smallest value that satisfies the success-rate requirement. This procedure does not assume that success probability increases monotonically with the number of planning steps. Although additional decomposition may help an executor handle a difficult task, excessive decomposition may also introduce longer contexts, redundant dependencies, or additional opportunities for execution errors.

If no candidate value reaches the target threshold, the task is recorded as

$$\widehat{C}_\tau(A, x) > K_{\max}, \tag{6}$$

rather than being assigned an arbitrary finite complexity value. The task is therefore considered unsolved by A under the evaluated planning budgets and protocol.

3.3 System-Level Interpretation and Comparison

Task Complexity is an agent-relative rather than a task-intrinsic quantity. For two agent systems A_1 and A_2, the same task may satisfy

$$C_\tau(A_1, x) \neq C_\tau(A_2, x), \tag{7}$$

because the systems may differ in their underlying LLMs, tools, memory mechanisms, planners, executors, or interaction protocols. A lower complexity value indicates that the corresponding system can reliably complete the task with fewer planning steps, whereas a higher value indicates that finer task decomposition is required to remain within the system's execution capability.

Complexity values are directly comparable only when the evaluated systems share the same task set, success evaluator, success-rate threshold. Under these controlled conditions, Task Complexity provides an outcome-grounded basis for evaluating task difficulty and comparing the planning capabilities of different agent systems.

Under these controlled conditions, if

$$C_\tau(A_1, x) < C_\tau(A_2, x), \tag{8}$$

then A_1 reaches the required success rate using fewer planning steps than A_2. Therefore, A_1 exhibits greater planning efficiency than A_2 on task x under the specified evaluation protocol. Conversely, if

$$C_\tau(A_1, x) > C_\tau(A_2, x), \tag{9}$$

then A_2 exhibits greater planning efficiency on that task. If the two complexity values are equal, the systems are indistinguishable under this metric for the given task and threshold.

This comparison is task-specific and should not be interpreted as evidence that one system is universally superior. A system-level comparison must additionally consider the distribution of Task Complexity over a shared task set.

4 Experiments

Our experiments investigate two primary research questions:

Q1: Does a fixed agent system exhibit a measurable minimum planning budget for reliably completing individual tasks?

Q2: Can Task Complexity characterize the relative capability and task coverage of different agent systems?

4.1 Experimental setup.

We evaluate two representative Plan-and-Act agent frameworks, CAMEL and MetaGPT, on 25 tasks from GSM8K and 10 tasks from HotpotQA. To construct these evaluation subsets, we first perform an unconstrained preliminary run on the GSM8K and HotpotQA dataset and record the number of planning steps generated by the agent system for each task. We then group the tasks according to their observed planning steps and proportionally sample from each group. This stratified sampling procedure preserves the distribution of agent-generated planning lengths observed during the preliminary run in the resulting evaluation subsets. Then, both frameworks are evaluated on the same task instances within each dataset. For GSM8K, the number of planning steps is varied over $k \in \{1, \ldots, 6\}$, where for HotpotQA, it is varied over $k \in \{1, \ldots, 8\}$. For each combination of agent system, task, and planning-step count, we conduct 20 repeated executions. The planner is constrained to generate exactly k executable steps, each of which is passed to the executor for actual execution. The main evaluation alone comprises 9,200 independent task executions. Additional experiments evaluate the repeatability of Task Complexity measurements and its utility for complexity-guided planning. Overall, the complete empirical study involves more than 20,000 task executions.

Table 2: Mean success rate (%) across tasks at each planning-step count. Dashes denote step counts outside the evaluated range. Bold indicates the highest mean for each agent–dataset combination.

Framework	Dataset	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8
CAMEL	GSM8K	67.4	**67.8**	66.6	62.0	43.0	31.0	–	–
MetaGPT	GSM8K	66.8	**74.0**	66.8	51.2	27.6	13.4	–	–
CAMEL	HotpotQA	17.0	37.0	54.0	**59.5**	52.0	37.5	21.0	10.0
MetaGPT	HotpotQA	24.5	45.5	63.0	**67.5**	58.5	43.5	26.0	12.5

Both frameworks use GPT-4o-mini as their backbone language model and are equipped with only two types of tools: a script-execution tool and a web-interaction tool. The agent state is reset between trials. All other components, including the underlying LLM, tool configurations, memory mechanism, system prompt, decoding parameters, and success evaluator, are held fixed when varying k.

4.2 Evaluation criterion.

We set the target success-rate threshold to $\tau = 0.65$. This threshold was selected after examining the collected results to retain as many tasks as possible for which a finite empirical Task Complexity could be estimated within the tested budget range. For each task, its empirical Task Complexity is the smallest tested k at which the agent achieves a success rate of at least 0.65. If the threshold is not reached under any tested k, the task is reported as *Not reached* and its complexity is considered greater than $K_{\max}$.

4.3 Results for Q1.

Existence of a minimum effective planning budget. Table 2 shows that execution success changes systematically with the number of planning steps. On GSM8K, both systems achieve their highest mean success rate at $k = 2$, reaching 67.8% for CAMEL and 74.0% for MetaGPT. On HotpotQA, both systems reach their highest mean at $k = 4$, obtaining 59.5% and 67.5%, respectively. The different peak locations indicate that the planning granularity required for reliable execution depends on the task domain.

At $\tau = 0.65$, a finite Task Complexity is identified for 24 of the 25 GSM8K tasks and eight of the ten HotpotQA tasks for both systems. The remaining tasks are recorded as having complexity greater than $K_{\max}$. Therefore, every evaluated agent–task pair receives an operationally defined outcome: either a minimum effective planning budget or an explicit position beyond the evaluated capability boundary.

As shown in Table 2, the success-rate profiles are non-monotonic, which may result from redundant execution or excessive task decomposition (Wu et al., 2026; Cemri et al., 2025). Meanwhile, Table 1 shows that smaller planning budgets incur lower token consumption. Taken together, allocating more planning steps does not necessarily improve task success, whereas using fewer steps generally reduces token consumption. Therefore, for a fixed agent system and a specified reliability threshold, there exists a minimum effective planning budget: the smallest number of planning steps that achieves the required success rate while avoiding unnecessary token consumption.

Repeatability of the complexity estimate. To examine whether empirical Task Complexity can be reproduced under independent measurements, we repeat the complete measurement procedure 10 times for tasks at each complexity level. For every repetition, each planning budget is evaluated over 20 executions using $\tau = 0.65$. We record the accuracy with respect to the nominal complexity and the sample variance of the independently estimated complexity values.

As shown in Table 3, Task Complexity is fully consistent across the ten independent measurements for tasks with complexity levels 1–3, resulting in zero observed variance. At complexity levels 4 and 5, one of the ten measurements differs from the nominal complexity by one planning step, reducing the exact-match accuracy to 90% and increasing the sample variance to 0.100. At complexity level 6, three measurements produce an adjacent estimate of 5, reducing accuracy to 70% and increasing the variance to 0.233.

Table 3: Repeat results over 10 independent measurements. Accuracy is the proportion of measurements that recover the nominal complexity, and variance is computed over the 10 estimated complexity values.

Task complexity	Accuracy	Variance of $\hat{C}_\tau$
1	1	0.000
2	1	0.000
3	1	0.000
4	0.9	0.100
5	0.9	0.100
6	0.7	0.233

The increasing variance is primarily associated with the success rate margin around τ, rather than with the numerical complexity level alone. Lower-complexity tasks generally have success rates well separated from τ, whereas the success rates of harder tasks may lie closer to the threshold, making their complexity estimates more sensitive to sampling variation.

4.4 Results for Q2.

We next examine the task-level complexity distributions. The quantity n_k in Tables 4 and 5 denotes the number of tasks whose smallest planning budget satisfying $\tau = 0.65$ is exactly k.

Table 4: Success-rate statistics and Task Complexity distribution on GSM8K. Here, n_k is the number of tasks satisfying $\widehat{C}_{0.65}(A, x) = k$. Mean, Max., and Min. are percentages across all 25 tasks at the corresponding k.

k	CAMEL				MetaGPT			
	n_k	Mean	Max.	Min.	n_k	Mean	Max.	Min.
1	14	67.4	100.0	35.0	15	66.8	100.0	30.0
2	7	**67.8**	100.0	5.0	7	**74.0**	100.0	45.0
3	2	66.6	100.0	5.0	1	66.8	100.0	20.0
4	0	62.0	100.0	10.0	1	51.2	100.0	20.0
5	1	43.0	100.0	0.0	0	27.6	80.0	5.0
6	0	31.0	95.0	0.0	0	13.4	90.0	0.0
Not reached	1	–	–	–	1	–	–	–

Table 5: Success-rate statistics and Task Complexity distribution on HotpotQA. Here, n_k is the number of tasks satisfying $\widehat{C}_{0.65}(A, x) = k$. Mean, Max., and Min. are percentages across all 10 tasks at the corresponding k.

k	CAMEL				MetaGPT			
	n_k	Mean	Max.	Min.	n_k	Mean	Max.	Min.
1	0	17.0	55.0	5.0	1	24.5	65.0	5.0
2	1	37.0	85.0	10.0	2	45.5	95.0	10.0
3	4	54.0	75.0	20.0	2	63.0	90.0	20.0
4	1	**59.5**	80.0	35.0	2	**67.5**	95.0	35.0
5	2	52.0	70.0	30.0	1	58.5	85.0	35.0
6	0	37.5	75.0	20.0	0	43.5	75.0	20.0
7	0	21.0	60.0	10.0	0	26.0	55.0	10.0
8	0	10.0	35.0	5.0	0	12.5	35.0	5.0
Not reached	2	–	–	–	2	–	–	–

On GSM8K, MetaGPT reaches the reliability threshold on 15 tasks with one planning step, compared with 14 for CAMEL. Within two steps, MetaGPT covers 22 tasks, whereas CAMEL covers 21 tasks. Both systems eventually reach the threshold on 24 tasks of the 25 tasks, giving the same

Table 6: Predicted and independently measured Task Complexity for five held-out target tasks.

Task group	Measured complexity	Predicted complexity
1	2	2
2	1	1
3	3	3
4	2	3
5	4	4

final capability coverage of 96%. Among these covered tasks, the mean Task Complexity is 1.50 for MetaGPT and 1.63 for CAMEL. MetaGPT therefore reaches the same capability coverage with slightly fewer planning steps.

The success-rate statistics provide complementary evidence. At $k = 2$, MetaGPT achieves a higher mean success rate than CAMEL (74.0% versus 67.8%) and a substantially higher minimum success rate (45.0% versus 5.0%). This indicates more consistent execution across GSM8K tasks at the most effective planning budget.

On HotpotQA, MetaGPT reaches the threshold on three tasks within two planning steps, whereas CAMEL reaches it on only one. Within four steps, MetaGPT covers seven tasks and CAMEL covers six. Both systems eventually reach the threshold on eight of the ten tasks, corresponding to the same final coverage of 80%. However, the mean Task Complexity among covered tasks is 3.00 for MetaGPT and 3.50 for CAMEL. MetaGPT therefore achieves the same task coverage using fewer planning steps. It also obtains a higher mean success rate at every tested value of k on HotpotQA.

Across both datasets, MetaGPT reaches the target reliability on at least as many tasks as CAMEL under the same planning budget. However, the systems have identical final coverage on each dataset. These results demonstrate that Task Complexity captures two complementary properties: the complexity distribution reflects planning efficiency, whereas the proportion of tasks reaching the threshold reflects the capability boundary of the agent system.

4.5 Complexity-guided planning case study.

In addition to evaluating Task Complexity as a measurement criterion, we conduct a controlled case study to examine whether historical complexity measurements can provide useful guidance for planning a new task. This experiment is intended as a feasibility study rather than the development of a sophisticated complexity-prediction model.

We construct five groups of GSM8K-style math tasks. Each group contains five historical tasks and one target task. We first measure the complexity of every historical task using the same repeated-execution protocol described above. The five historical tasks and their measured complexity values are then treated as a small complexity database. For each target task, we calculate its cosine similarity to the five historical tasks in the corresponding group. These similarity values are used as weights to compute a weighted average of the historical complexities. The target task is excluded from the database during prediction. Finally, we independently measure its Task Complexity by evaluating all candidate planning-step counts, allowing the predicted and measured values to be compared.

As shown in Table 6, the predicted complexity exactly matches the independently measured value for four of the five target tasks, corresponding to an exact-match accuracy of 80%. The only prediction error occurs in Group 4, where the complexity is overestimated by one planning step. The mean absolute error is therefore 0.2 steps, and all predictions fall within one step of the measured complexity.

These results provide initial evidence that historical Task Complexity measurements can be transferred to structurally similar tasks. Even a simple similarity-weighted estimation method can provide an approximate planning-step budget before executing a new task. This experiment does not establish the general effectiveness of a particular prediction model; rather, it demonstrates that Task Complexity can serve as a planning signal and motivates future work on more general complexity-prediction mechanisms.

4.6 DISCUSSION.

The results answer both experimental questions. For Q1, repeated execution under a fixed system configuration produces an identifiable minimum effective planning budget for most tasks. Tasks that do not reach the threshold remain explicitly represented as being beyond the evaluated capability boundary. For Q2, comparing complexity distributions reveals differences in planning efficiency even when systems have the same final task coverage.

These findings support our main contribution: Task Complexity provides an outcome-grounded and agent-relative evaluation criterion. It measures task difficulty through the observed reliability of the complete agent system rather than through human-designed task attributes or an LLM's subjective difficulty judgment. It therefore supports both task-level complexity measurement and system-level capability comparison under a shared evaluation protocol.

The controlled planning case study further illustrates the practical utility of the proposed measure. Across five independently constructed task groups, the similarity-based method achieves an exact-match accuracy of 80% and a mean absolute error of 0.2 planning steps. Although the experiment is deliberately lightweight, it shows that measured complexity values can be stored and transferred to similar tasks to provide an explicit prior for planning-step allocation.

5 CONCLUSION

In this paper, we introduced Task Complexity, an agent-relative and execution-grounded argument for evaluating task difficulty in agent systems. For a fixed agent system and reliability threshold, Task Complexity is defined as the minimum number of executable planning steps required to achieve the target success rate. Unlike difficulty measures based on manually selected task attributes or an LLM's subjective judgment, our definition derives task difficulty from the observed execution behavior of the complete agent system, including its planner, executor, tools, and memory.

We further presented an experiment for measuring Task Complexity through real execution. Our experiments with CAMEL and MetaGPT on GSM8K and HotpotQA show that different agent systems can require different planning budgets for the same task set. The resulting complexity distributions distinguish planning efficiency from final task coverage and therefore provide a system-level basis for capability comparison. Task Complexity does not require manually annotated difficulty labels. Our controlled case study further shows that even a simple similarity-weighted method can use historical complexity measurements to predict similar tasks' complexity. Although this experiment is not intended to establish a general complexity-prediction mechanism, it demonstrates that Task Complexity can serve as a lightweight prior for planning-step allocation.

Future work may investigate more sample-efficient measurement procedures, uncertainty-aware complexity estimation, and generalization across broader task domains and agent configurations. Overall, Task Complexity provides a abstraction for characterizing agent-specific task difficulty, identifying capability boundaries, and supporting complexity-aware planning without constraining the agent to imitate historical execution trajectories.

6 AI USE STATEMENT

In this work, we used generative AI tools for edit a research paper to improve readability. We have not used generative AI tools for such as problem formulation, experimental design and methodology. All AI-assisted edits were reviewed by the authors. The authors take responsibility for the final content of this work.

REFERENCES

Ruisheng Cao, Fangyu Lei, Haoyuan Wu, Jixuan Chen, Yeqiao Fu, Hongcheng Gao, Xinzhuang Xiong, Hanchong Zhang, Yuchen Mao, Wenjing Hu, Tianbao Xie, Hongshen Xu, Danyang Zhang, Sida Wang, Ruoxi Sun, Pengcheng Yin, Caiming Xiong, Ansong Ni, Qian Liu, Victor Zhong, Lu Chen, Kai Yu, and Tao Yu. Spider2-V: How far are multimodal agents from automating data science and engineering workflows? In *Advances in Neural Information Pro-*

cessing Systems, volume 37. Curran Associates, Inc., 2024. doi: 10.52202/079017-3421. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/c2f71567cd53464161cab3336e8fc865-Abstract-Datasets_and_Benchmarks_Track.html.

Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei A Zaharia, Joseph Gonzalez, and Ion Stoica. Why do multi-agent llm systems fail? In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference. Curran Associates, Inc., 2025. doi: 10.52202/085713-4082. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/b1041e52d3be19f0a9bc491657488e4a-Paper-Datasets_and_Benchmarks_Track.pdf.

Lutfi Eren Erdogan, Nicholas Lee, Sehoon Kim, Suhong Moon, Hiroki Furuta, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. Plan-and-act: Improving planning of agents for long-horizon tasks. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 15419–15462, 2025.

Meihao Fan, Ju Fan, Nan Tang, Lei Cao, Guoliang Li, and Xiaoyong Du. AutoPrep: Natural language question-aware data preparation with a multi-agent framework. *Proceedings of the VLDB Endowment*, 18(10):3504–3517, 2025.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. WebVoyager: Building an end-to-end web agent with large multimodal models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6864–6890. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.371. URL https://aclanthology.org/2024.acl-long.371/.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. MLAgentBench: Evaluating language agents on machine learning experimentation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 20271–20309. PMLR, 2024. URL https://proceedings.mlr.press/v235/huang24y.html.

Peter Jansen, Marc-Alexandre Côté, Tushar Khot, Erin Bransom, Bhavana Dalvi Mishra, Bodhisattwa Prasad Majumder, Oyvind Tafjord, and Peter Clark. DiscoveryWorld: A virtual environment for developing and evaluating automated scientific discovery agents. In *Advances in Neural Information Processing Systems*, volume 37. Curran Associates, Inc., 2024. doi: 10.52202/079017-0324. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/13836f251823945316ae067350a5c366-Abstract-Datasets_and_Benchmarks_Track.html.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 7036–7050, 2024.

Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, Yang Song, Chen Zhu, Hengshu Zhu, and Ji-Rong Wen. KG-Agent: An efficient autonomous agent framework for complex reasoning over knowledge graph. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pp. 9505–9523, 2025.

Sehoon Kim, Suhong Moon, Ryan Tabrizi, Nicholas Lee, Michael W. Mahoney, Kurt Keutzer, and Amir Gholami. An LLM compiler for parallel function calling. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 24370–24391, 2024.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Russ Salakhutdinov, and Daniel Fried. VisualWebArena: Evaluating multimodal agents on realistic visual web tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 881–905.

Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.50. URL https://aclanthology.org/2024.acl-long.50/.

Manling Li, Shiyu Zhao, Qineng Wang, Kangrui Wang, Yu Zhou, Sanjana Srivastava, Cem Gokmen, Tony Lee, Li Erran Li, Ruohan Zhang, Weiyu Liu, Percy Liang, Li Fei-Fei, Jiayuan Mao, and Jiajun Wu. Embodied agent interface: Benchmarking LLMs for embodied decision making. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. AgentBoard: An analytical evaluation board of multi-turn LLM agents. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. ADaPT: As-needed decomposition and planning with language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 4226–4252, 2024.

Shuofei Qiao, Runnan Fang, Ningyu Zhang, Yuqi Zhu, Xiang Chen, Shumin Deng, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. Agent planning with world knowledge model. In *Advances in Neural Information Processing Systems*, volume 37, 2024a.

Shuofei Qiao, Ningyu Zhang, Runnan Fang, Yujie Luo, Wangchunshu Zhou, Yuchen Jiang, Chengfei Lv, and Huajun Chen. AutoAct: Automatic agent learning from scratch for QA via self-planning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3003–3021. Association for Computational Linguistics, 2024b. doi: 10.18653/v1/2024.acl-long.165. URL https://aclanthology.org/2024.acl-long.165/.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *The Twelfth International Conference on Learning Representations*, 2024.

Yongliang Shen, Kaitao Song, Xu Tan, Wenqi Zhang, Kan Ren, Siyu Yuan, Weiming Lu, Dongsheng Li, and Yueting Zhuang. TaskBench: Benchmarking large language models for task automation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 4540–4574, 2024.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16022–16076. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.850. URL https://aclanthology.org/2024.acl-long.850/.

Jize Wang, Zerun Ma, Yining Li, Songyang Zhang, Cailian Chen, Kai Chen, and Xinyi Le. GTA: A benchmark for general tool agents. In *Advances in Neural Information Processing Systems*, volume 37. Curran Associates, Inc., 2024. doi: 10.52202/079017-2412. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/8a75ee6d4b2eb0b777f549a32a5a5c28-Abstract-Datasets_and_Benchmarks_Track.html.

Yuyang Wu, Yifei Wang, Ziyu Ye, Tianqi Du, Stefanie Jegelka, and Yisen Wang. When more is less: Understanding chain-of-thought length in llms. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 127108–127138, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/ce916251b4fe04f54f99c8d304d68877-Paper-Conference.pdf.

Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. TravelPlanner: A benchmark for real-world planning with language agents. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54590–54613. PMLR, 2024a. URL https://proceedings.mlr.press/v235/xie24j.html.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems*, volume 37. Curran Associates, Inc., 2024b. doi: 10.52202/079017-1650. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/5d413e48f84dc61244b6be550f1cd8f5-Abstract-Datasets_and_Benchmarks_Track.html.

Mayi Xu, Yongqi Li, Ke Sun, and Tieyun Qian. Adaption-of-thought: Learning question difficulty improves large language models for reasoning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 5468–5495, 2024.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems*, volume 37. Curran Associates, Inc., 2024. doi: 10.52202/079017-1601. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/5a7c947568c1b1328ccc5230172e1e7c-Abstract-Conference.html.

Zishun Yu, Tengyu Xu, Di Jin, Karthik Abinav Sankararaman, Yun He, Wenxuan Zhou, Zhouhao Zeng, Eryk Helenowski, Chen Zhu, Sinong Wang, Hao Ma, and Han Fang. Think smarter not harder: Adaptive reasoning with inference aware optimization. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 73304–73325, 2025.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936. URL https://ojs.aaai.org/index.php/AAAI/article/view/29936.

Huaixiu Steven Zheng, Swaroop Mishra, Hugh Zhang, Xinyun Chen, Minmin Chen, Azade Nova, Le Hou, Heng-Tze Cheng, Quoc V. Le, Ed H. Chi, and Denny Zhou. Natural plan: Benchmarking llms on natural language planning, 2024. URL https://arxiv.org/abs/2406.04520.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning, acting, and planning in language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 62138–62160, 2024.

Yubo Zhu, Dongrui Liu, Zecheng Lin, Wei Tong, Sheng Zhong, and Jing Shao. The LLM already knows: Estimating LLM-perceived question difficulty via hidden representations. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 1160–1176, 2025.