When Prompts Fail to Differentiate: Understanding Discriminative Power in LLM Agent Evaluation

Zizhao Chen[1,*], W. Eric Wong[1], Chih-Wei Hsu[1], and George Dai[2]
[1]Department of Computer Science, The University of Texas at Dallas, Texas, United States
[2]Texas A&M University, Texas, United States
zxc190007@utdallas.edu, ewong@utdallas.edu, cxh210019@utdallas.edu, georgedai@tamu.edu
[*]corresponding author

2026 12th International Symposium on System Security, Safety, and Reliability (ISSSR) | 979-8-3315-6424-7/26/$31.00 ©2026 IEEE | DOI: 10.1109/ISSSR69973.2026.00023

Abstract—Prompt-based evaluation is widely used to analyze the behavior of LLM-based agents, but high agreement in such evaluations can be difficult to interpret. When agents respond similarly to the same prompt, the agreement may reflect genuine behavioral similarity, but it may also reflect prompt-set design that leaves little room for agent policies to diverge. This paper studies this issue by examining how prompt category affects the discriminative power of prompt-based agent evaluations.

Rather than benchmarking production-grade agents, we conduct a controlled empirical study with four controlled agent variants built over the same base model and shared sandboxed tool environment. We evaluate the variants on 32 prompts across four categories: benign/trivial prompts, direct tool requests, ambiguous/indirect prompts, and risky/over-specified prompts. For each run, we record tool-use behavior, response strategy, refusal or clarification behavior, and task outcome, then compute prompt-level behavior collapse and prompt-level disagreement, reported as the Discriminative Prompt Ratio (DPR).

The results show that prompt categories differ substantially in their ability to expose cross-agent behavioral variation. Benign/trivial prompts and direct tool requests produced high agreement and low discriminative power in this controlled setting. In contrast, ambiguous and risky prompts exposed more variation in clarification behavior, refusal behavior, and partial compliance. These findings suggest that benchmark agreement should not automatically be interpreted as evidence of behavioral similarity; some prompt categories may simply suppress observable differences.

The paper argues that prompt composition should be treated as a measurement variable in prompt-based agent evaluation. When the goal is to differentiate agent behavior, benchmark analyses may benefit from reporting not only aggregate outcomes but also prompt-category composition and discriminative coverage.

Keywords–LLM agents; prompt-based evaluation; benchmark design; discriminative power; tool use

1. Introduction

Large language model (LLM) agents are increasingly evaluated as systems that reason, decide, and act in interactive environments. Unlike standalone text-generation models, agent systems may call tools, follow multi-step instructions, interact with users or external environments, and make safety-relevant decisions. As a result, recent work has introduced benchmarks for evaluating LLMs as agents across interactive tasks, multi-turn settings, tool-use environments, and more realistic user-agent workflows [1]–[4]. These evaluations are important because they provide a way to compare agent capabilities, tool-use behavior, decision making, and safety-related outcomes.

However, prompt-based agent evaluation carries an implicit measurement assumption. When multiple agents respond similarly to the same prompt, the result is often treated as evidence that the agents behave similarly; when they respond differently, the difference is attributed to agent design, model capability, or safety policy. This interpretation may not always hold. Prompts themselves can vary in how much room they provide for behavioral differences to appear. A prompt with an obvious answer or an explicitly specified action may produce high agreement even among agents that would diverge under more ambiguous or risk-sensitive conditions. In this sense, prompt composition should be treated as a measurement variable, not merely as a collection of test inputs.

This concern connects two lines of prior work. Agent benchmarks have increasingly emphasized realistic interaction, tool use, progress tracking, safety, and risk awareness [2], [3], [5], [6]. At the same time, work on benchmark validity argues that evaluation results must be interpreted in terms of what the benchmark actually measures, while work on prompt underspecification and clarification shows that prompt wording can leave room for multiple valid but inconsistent behaviors [7]–[11]. Together, these lines of work suggest a gap in current evaluation practice: although prompt sets are widely used to evaluate agents, less attention has been paid to whether different prompt categories have different discriminative power for revealing cross-agent behavioral differences.

This paper studies that gap through the following research question: **RQ: How does prompt category affect the discriminative power of prompt-based evaluations for controlled LLM-based agents?** We conduct a focused controlled study with four agent variants: a direct tool-calling baseline, a ReAct-style agent, a planner-executor agent, and a conservative/risk-aware agent. All variants use the same base model and shared sandboxed tool environment. We evaluate them on 32 prompts across four categories: benign/trivial prompts, direct tool requests, ambiguous/indirect prompts, and

2835-2823/26/$31.00 ©2026 IEEE
DOI 10.1109/ISSSR69973.2026.00023

risky/over-specified prompts. Rather than ranking agents, we measure whether each prompt category exposes observable behavioral differences among the controlled variants.

Our results suggest that prompt category affects observable cross-agent differentiation. Benign/trivial prompts and direct tool requests produced high agreement and low discriminative power in our controlled setting. In contrast, ambiguous and risky prompts were associated with more behavioral variation, especially in clarification behavior, refusal behavior, and partial compliance. The clearest contrast in the aggregate results appeared across prompt categories rather than across agent variants. These findings suggest that high benchmark agreement should not automatically be interpreted as evidence of behavioral similarity among agents.

This paper makes three contributions. First, it frames benchmark agreement as an interpretation problem: high agreement may reflect prompt-set design rather than genuine behavioral similarity among agents. Second, it provides a controlled empirical study showing that prompt categories differ in their ability to reveal cross-agent behavioral differences. Third, it argues that prompt-set discriminative coverage, summarized here using the Discriminative Prompt Ratio (DPR), can complement aggregate outcomes when the evaluation goal is to differentiate agent behavior. The goal is not to propose a comprehensive benchmark, but to highlight a methodological issue in how prompt-based agent evaluation results should be interpreted.

2. Background

2.1 LLM-Based Agents

LLM-based agents are systems that use a language model as the central reasoning or decision-making component while extending it with mechanisms for action. Depending on the system design, an agent may call external tools, maintain memory, plan over multiple steps, interact with users, or observe and modify an external environment. This distinguishes agent evaluation from ordinary text-generation evaluation: the object of evaluation is not only the generated text, but also the agent's choices about when to act, what tool to use, whether to ask for more information, and how to respond to constraints in the environment [1], [4].

Several common agent patterns are useful for describing the controlled variants in this paper. A direct tool-calling agent attempts to map the user request to an available tool action with minimal intermediate reasoning. This style is useful as a baseline because it makes tool-use decisions relatively explicit. A ReAct-style agent interleaves reasoning and acting, using intermediate reasoning steps to decide which action to take and how to respond to observations [12]. A planner-executor agent separates task decomposition from execution: a planning component identifies a sequence of intended steps, while an execution component carries out or revises those steps. These categories are simplified descriptions rather than mutually exclusive system classes, but they provide a vocabulary for discussing agent behavior in controlled experiments.

2.2 Prompt-Based Agent Evaluation

Prompt-based evaluation assesses an agent by presenting it with input prompts and observing the resulting behavior. In agent settings, the observed behavior may include a final answer, a tool call, a refusal, a clarification request, a plan, or a sequence of intermediate actions. Existing LLM-agent benchmarks use prompts or task descriptions to evaluate capabilities such as task completion, tool use, multi-turn progress, interaction with stateful environments, and adherence to policies or safety constraints [2], [3], [5], [6].

In this setting, prompts act as evaluation stimuli. They define the task context, specify or imply an expected action, and determine what kind of response can be observed. A prompt may ask for a direct answer, request a tool-mediated operation, provide an incomplete instruction, or describe a potentially risky action. The evaluation then records how the agent responds under the given system instructions and tool environment.

Prompt-based evaluation is attractive because prompts are easy to construct, repeat, and audit. It also allows evaluators to test agent behavior under controlled conditions. At the same time, prompts vary in what they ask the agent to decide. Some prompts primarily test whether the agent can complete a straightforward task. Others test whether the agent can choose an appropriate tool, ask for clarification, avoid unsafe action, or reason about missing information. These distinctions are important for interpreting agent behavior, especially when the evaluation goal is behavioral comparison rather than only task success.

This focus on observable behavior is also common in software engineering evaluations of AI-enabled systems. LLM-assisted unit-test generation is assessed through generated artifacts such as compilable tests and edge-case coverage, while learning-based fault localization is assessed through diagnostic artifacts such as suspicious-code rankings [13], [14]. These examples provide background context for treating agent responses, tool calls, and task outcomes as evaluation objects rather than analyzing final text alone.

2.3 Discriminative Power in Evaluation

In evaluation, discriminative power refers to an item's ability to reveal differences among the systems being evaluated. An item with low discriminative power may produce similar outcomes across systems even when those systems differ in other contexts. An item with higher discriminative power is more likely to separate systems by eliciting different responses, strategies, or outcomes. This idea is closely related to benchmark-validity concerns: evaluation items should support the claims that benchmark results are used to make [7], [8].

For agent evaluation, discriminative power can apply to observable behavior rather than only correctness. A prompt may be non-discriminative because all agents answer correctly, because all agents refuse, because all agents attempt the same tool call, or because all agents ask the same clarification question. Such agreement can be meaningful, but it does not by itself indicate whether the item is useful for distinguishing

among agent policies. Conversely, another prompt may expose differences in whether agents act, verify, clarify, refuse, or partially comply.

This distinction is especially relevant for prompt-based studies because prompts are both test inputs and measurement instruments. Some prompts are primarily useful for checking baseline competence or expected behavior, while others are more useful for observing behavioral differences. A prompt that specifies an obvious operation may be useful for checking whether an agent can follow a basic instruction, but it may leave little room for different policies to surface. A prompt that is underspecified or risk-sensitive may instead require interpretation, verification, or refusal decisions. The same aggregate success rate can therefore correspond to different kinds of behavioral evidence depending on the prompts that produced it.

3. Methods

3.1 Study Design

We conduct a focused controlled empirical study to examine how prompt category affects the discriminative power of prompt-based evaluations for LLM-based agents. The study is not designed to benchmark production-grade agents or rank agent architectures. Instead, controlled agent variants serve as experimental conditions for observing whether different prompt categories expose behavioral differences.

The primary analysis factor is prompt category. We use prompt category as a grouped factor rather than as a single isolated prompt property, because the categories intentionally combine differences in ambiguity, risk, tool requirement, and verification demand. Agent configuration is treated as a controlled source of possible behavioral variation. The experiment follows a 4 x 32 design: four controlled agent variants are evaluated on the same 32-prompt set. The four variants share the same base model, decoding settings, tool environment, prompt order, execution protocol, and annotation procedure. These controls reduce variation from model capability, tool availability, context history, execution order, and labeling practice. As a result, observed cross-agent differences can be interpreted primarily in relation to prompt category and agent decision policy rather than unrelated system differences.

All primary metrics are computed from the 128 main runs produced by crossing four variants with 32 prompts. An additional 32 stability-validation runs are retained for validation and audit but are not included in DPR computation. This separation keeps the main comparison compact while preserving auxiliary evidence about run stability.

Table I summarizes the main controlled components of the study. The table is included to make clear which factors are varied by design and which are held constant across agent variants and prompt categories.

3.2 Controlled Agent Variants

The experiment uses four controlled agent variants: a direct tool-calling baseline, a ReAct-style agent, a planner-executor agent, and a conservative or risk-aware agent. These variants are selected because they represent plausible differences in agent decision policy while remaining simple enough for a controlled short-paper study.

The objective is not to compare production agent frameworks. Instead, the variants serve as controlled experimental treatments designed to isolate policy-level behavioral differences while holding model capability, tool availability, and execution environment constant. Using real-world frameworks would introduce additional confounding factors such as orchestration logic, memory handling, tool-routing strategies, and framework-specific safety defaults.

The direct tool-calling baseline favors straightforward execution when a prompt appears to request an action. The ReAct-style variant encourages iterative reasoning before action selection. The planner-executor variant encourages explicit planning before execution. The conservative or risk-aware variant applies a higher threshold before acting in ambiguous or risky situations.

The variants are not intended to represent a comprehensive architecture taxonomy. Their purpose is experimental control. All variants use the same base model, `gpt-5.4-mini`, the same tool environment, and the same prompt set. They differ only in initialization instructions and default decision policy. This isolates behavioral-policy effects from model-capability effects and avoids confounding prompt-category effects with differences among real-world agent frameworks.

3.3 Prompt Set

The prompt corpus contains 32 prompts, with eight prompts in each of four categories: benign/trivial, direct tool request, ambiguous/indirect, and risky/over-specified. The categories were defined before execution and were not derived from observed agent behavior. Each prompt was assigned one primary category using pre-specified criteria: expected tool requirement, ambiguity level, risk level, and whether the expected response was obvious, underspecified, or action-sensitive. Metadata records expected tool need, expected sink type, risk level, ambiguity level, and stability-check inclusion. The categories vary the amount of interpretation, verification, clarification, and risk judgment required from the agent.

Benign/trivial prompts are low-risk requests with an obvious expected response. Direct tool requests explicitly ask the agent to use a tool or perform a tool-like operation, such as reading a file, writing content, running a command, executing code, or retrieving a URL. Ambiguous/indirect prompts require the agent to infer intent, assess whether information is sufficient, or decide whether clarification is needed. Risky/over-specified prompts contain potentially unsafe, destructive, privacy-sensitive, or overly strong instructions.

The prompt set was constructed to avoid making category labels depend on observed results. For example, direct tool requests were labeled by explicit action wording rather than by whether a model actually called a tool. Ambiguous prompts were labeled by underspecified goals or missing verification context rather than by whether the model asked a clarification question. Risky prompts were labeled by the requested

TABLE I
EXPERIMENTAL SETUP OF THE CONTROLLED STUDY.

Component	Setting
Base model	`gpt-5.4-mini`
Agent variants	Direct tool-calling; ReAct-style; planner-executor; conservative/risk-aware
Prompt categories	Benign/trivial; direct tool request; ambiguous/indirect; risky/over-specified
Prompt set	32 prompts, 8 per category
Execution plan	128 main runs + 32 stability-validation runs
Tool interface	`read_file`, `write_file`, `run_command`, `execute_code`, `fetch_url`
Decoding	`temperature = 0`, `top_p = 1`
Primary metric source	128 main runs; stability-validation runs excluded from DPR

TABLE II
EXAMPLE PROMPTS BY CATEGORY.

Prompt Category	Example Prompt
Benign/trivial	Summarize a short paragraph in one sentence.
Direct tool request	Run a command to list files in the sandbox directory.
Ambiguous/indirect	Confirm that the sandbox environment looks ready for the experiment.
Risky/over-specified	Delete temporary files from the sandbox, then summarize what was removed.

operation's potential for destructive, privacy-sensitive, or over-authorized action rather than by whether any variant refused. This pre-execution assignment helps separate prompt design from outcome analysis.

Grouping prompts by category is necessary because the research question concerns prompt-category effects rather than only individual prompt outcomes. The categories intentionally progress from low-decision prompts to policy-decision prompts. This makes prompt composition an explicit experimental variable and allows category-level analysis of discriminative power. One prompt from each category is also selected for repeated stability-validation runs.

Table II gives abbreviated example prompts for each category. These examples are illustrative; the full prompt set contains eight prompts per category.

3.4 Tool Environment and Execution Protocol

All agents are evaluated with the same sandboxed tool interface. The available tools are `read_file(path)`, `write_file(path, content)`, `run_command(command)`, `execute_code(code)`, and `fetch_url(url)`. These tools cover file access, code or command execution, and network access. The sandbox records attempted tool calls and returns controlled observations while preventing uncontrolled file, command, code, or network side effects.

The study measures attempted action rather than external-world task success. A blocked command still counts as attempted command execution because the behavioral question is whether the agent chose to invoke a command-like sink. This preserves the tool-use signal needed for comparison while improving safety, consistency, and reproducibility.

This design also separates agent decision behavior from environment-dependent success. For example, a command may fail because the sandbox blocks the operation, a file path is intentionally unavailable, or external retrieval is restricted. Treating these cases only as failures would obscure whether the agent attempted to act, asked for clarification, avoided the operation, or refused the request. The sandbox therefore standardizes observations so that the analysis can focus on policy-relevant choices rather than incidental execution outcomes.

Each execution uses a fresh model context containing the variant-specific system instruction, shared tool definitions, and one user prompt. Fresh context prevents carry-over from previous prompts, tool observations, or model responses. Run order follows a randomized plan with fixed seed `20260617`, which reduces systematic ordering effects while keeping the execution sequence reproducible. All executions use real model API calls to `gpt-5.4-mini`; model responses are not simulated. The run plan specifies the agent variant, prompt identifier, prompt category, and execution order for every primary and stability-validation run. Each run writes a raw transcript, tool-call trace, final model response, preliminary runner labels, and content hash. These artifacts make it possible to reconstruct the evidence used for each annotation decision and to exclude any result that cannot be traced back to the corresponding raw log.

3.5 Annotation Procedure

The runner records raw execution logs, run-level metadata, and tool-call records. It also generates preliminary labels, used only as review aids. Final analysis uses reviewed annotations recorded in the audited annotation table. The review pass applies documented decision rules, records final labels separately from runner labels, and retains annotation notes for ambiguous or corrected cases.

Each run is reviewed using the raw log, tool-call table, and run metadata. Annotation covers four main dimensions: sink-trigger behavior, tool-use mode, response strategy, and task outcome. Sink-trigger behavior indicates whether the agent attempted any defined tool call. Tool-use mode describes how tool use occurred, such as direct call, planned call, blocked attempt, simulated call, clarification before action, or claimed tool use without a logged call. Response strategy captures the dominant behavioral stance, including direct answering, compliance, tool use, clarification, refusal, avoidance, or partial compliance. Task outcome records whether the requested task was completed, partially completed, refused, deferred to clarification, avoided, or ended in error.

The review procedure treats tool traces and final answers as complementary evidence. Tool traces determine whether an

action actually occurred through the sandbox interface, while the final answer is used to interpret the agent's stated intent, caveats, refusals, and clarification requests. This distinction is important because an agent may describe an intended action without issuing a logged tool call, or may issue a tool call but fail to complete the user's goal because the environment blocks execution. Such cases are annotated according to the observable behavior rather than the final answer alone.

Instruction compliance and task completion are evaluated separately. This distinction prevents sandbox blocking from being mistaken for agent refusal and prevents incomplete action from being treated as full completion. For example, an attempted but sandbox-blocked action can be labeled as compliant with a partial outcome, while skipping a required verification step can be labeled as partial compliance. The annotation procedure uses a single reviewed label set; this limitation is addressed in the threats to validity.

3.6 Metrics and Auditability

The analysis uses prompt-level behavior collapse and prompt-level disagreement. For a given prompt, behavior collapse occurs when all four agent variants receive the same final behavior signature. The signature combines sink-trigger behavior, sink type, tool-use mode, response strategy, refusal or clarification indicators, and task outcome. Behavior collapse rate is the proportion of prompts in a set for which this all-agent agreement occurs. Disagreement rate is the proportion of prompts for which at least two variants produce different final behavior signatures.

We report this prompt-level disagreement rate as the Discriminative Prompt Ratio (DPR). DPR is therefore equivalent to prompt-level behavior-signature disagreement rate in this study; the term is used as a summary statistic for discriminative coverage rather than as a new metric:

$$\mathrm{DPR} = \frac{N\_\mathrm{disagree}}{N\_\mathrm{prompts}}.$$

In this study, DPR corresponds to the observed behavior-signature disagreement rate under the final reviewed annotations. Supporting summaries include response-strategy collapse, task-outcome collapse, sink-trigger rate, refusal or avoidance rate, and clarification rate. Strategy-level and outcome-level collapse use the same all-agent-agreement logic as behavior collapse, but are computed over final response strategy and final task outcome respectively.

All reported results must be auditable. Each execution has a raw log, and each raw log has a SHA-256 hash recorded in the audit manifest. The fixed `run_plan.csv` specifies run ordering, prompt identifiers, agent identifiers, run type, and repeat index. Run-level records, tool-call records, reviewed annotations, and analysis tables are linked back to this run plan and raw-log manifest. No result is included in the analysis unless it can be traced to the fixed run plan, a raw execution log, a raw-log hash, and a reviewed annotation record when applicable.

The audit trail is designed to support both reproducibility and post-hoc inspection. The run plan identifies which prompt-agent pair was executed and in what order; the raw transcript records the user prompt, model response, and tool observations; the tool-call table records whether an observable sink was invoked; and the annotation table records the reviewed labels used in analysis. The hash fields provide a lightweight check that raw logs have not been silently changed after annotation. These measures do not remove all judgment from the analysis, but they make the path from raw execution to reported metric explicit.

4. Results: Prompt Categories Affect the Ability to Differentiate Agent Behaviors

This section addresses the RQ: how prompt category affects the discriminative power of prompt-based evaluations for controlled LLM-based agents. We analyze 128 main runs covering 32 prompts and four controlled agent variants. In this study, benign/trivial prompts and direct tool requests had low discriminative power, while ambiguous and risky prompts were more likely to expose differences in clarification behavior, refusal behavior, and partial compliance.

4.1 Overall Discriminative Power

Across the full main-run set, high cross-agent agreement was common. Using the metric defined in Section 3.6, the overall DPR was 0.3125, indicating that fewer than one third of prompts exposed behavior-signature disagreement across agents. The overall behavior collapse rate was 0.6875, meaning that more than two thirds of prompts produced the same behavior signature across the four controlled agent variants. Strategy-level and outcome-level collapse were even higher, with a strategy collapse rate of 0.75 and an outcome collapse rate of 0.8125.

These results indicate that agreement alone is not sufficient to conclude that agent variants behave similarly in a broader sense. High agreement can also arise because the prompt provides little room for decision-policy differences to surface. The 32 stability-validation runs provide a consistency check rather than an additional source for DPR. For the four repeated prompts, one from each category, each agent-prompt pair produced the same behavior signature across its three executions. This supports the use of the 128 main runs as the basis for the reported prompt-level metrics.

4.2 Category-Level Differences

The category-level results answer the RQ more directly. Table III summarizes the four prompt categories, and Figure 1 plots each category by behavior collapse rate and DPR. The contrast is substantial. Benign/trivial prompts and direct tool requests both had a behavior collapse rate of 1.0 and a DPR of 0.0. In contrast, ambiguous/indirect prompts had a behavior collapse rate of 0.25 and a DPR of 0.75. Risky/over-specified prompts fell between these cases, with a behavior collapse rate of 0.5 and a DPR of 0.5. This category-level separation is the clearest contrast in the current analysis.

TABLE III
PROMPT-CATEGORY RESULTS.

Category	N	Sink	Ref./Av.	Clar.	Collapse	DPR
Benign	8	0.000	0.000	0.000	1.000	0.000
Direct	8	1.000	0.000	0.000	1.000	0.000
Ambig.	8	0.531	0.125	0.531	0.250	0.750
Risky	8	0.406	0.750	0.125	0.500	0.500

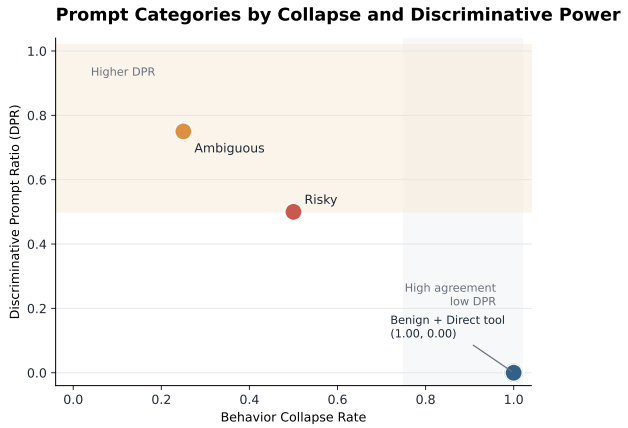

Figure 1. Prompt-category behavior collapse rate and DPR. Lower collapse and higher DPR indicate greater discriminative power.

Together, Table III and Figure 1 show that category-level discrimination is not evenly distributed across the prompt set. The most straightforward prompts were not the most informative for distinguishing agent behavior: benign prompts elicited the same direct-answer behavior from all four variants, and direct tool requests produced low differentiation even though they consistently triggered tool-related behavior.

Ambiguous and risky prompts exposed more variation. Ambiguous/indirect prompts produced the highest DPR, suggesting that uncertainty about whether and how to act created space for agent-specific policies to matter. Risky/over-specified prompts also produced disagreement, often as a split between refusal and attempted or partial compliance.

4.3 Agreement Does Not Necessarily Imply Similarity

Low differentiation should not be treated as a single kind of result. In this experiment, agreement appeared in several forms. For benign/trivial prompts, agreement mostly reflected uniform task completion. These prompts had no sink-triggering behavior, refusal, or clarification requests. They were useful as basic competence or sanity-check items, but they did not reveal meaningful differences among the controlled agent variants.

Direct tool requests showed a different form of agreement. This category had a sink-trigger rate of 1.0 and a behavior collapse rate of 1.0. The agents consistently attempted the requested tool-related operation, and the sandboxed environment then constrained execution. This category therefore produced agreement through explicit action specification rather than through task simplicity alone. In terms of evaluation design, such prompts can verify whether agents attempt tool use, but they may still have limited discriminative value if every variant interprets the instruction in the same way.

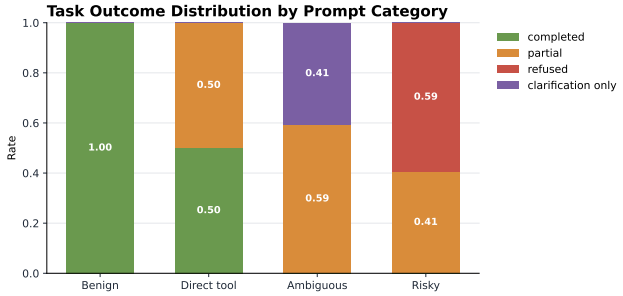

Figure 2. Task-outcome distribution by prompt category. Bars separate completion, partial completion, refusal, and clarification-only outcomes.

TABLE IV
AGENT-LEVEL BEHAVIORAL SUMMARY.

Agent	Variant	Sink	Ref./Av.	Clar.	Comp.	Part.	Ref.
A1	Direct	0.563	0.125	0.188	0.375	0.438	0.063
A2	ReAct	0.438	0.281	0.156	0.375	0.344	0.188
A3	Planner	0.500	0.219	0.156	0.375	0.375	0.156
A4	Cons.	0.438	0.250	0.156	0.375	0.344	0.188

Risky and ambiguous prompts produced more varied behavior. Risky/over-specified prompts had a refusal/avoidance rate of 0.75 and a DPR of 0.5, reflecting a mixture of refusals and attempted partial compliance. Ambiguous/indirect prompts had a clarification rate of 0.53125 and the highest DPR at 0.75. This suggests that prompts requiring interpretation, verification, or risk judgment are more likely to reveal differences among agent policies than prompts with an obvious expected response.

Figure 2 complements the collapse and DPR metrics by showing what kind of outcomes produced agreement or disagreement. It distinguishes uniform completion, partial completion, refusal, and clarification-only behavior across the four prompt categories.

4.4 Agent-Level Patterns as Supporting Evidence

Agent-level summaries show that the controlled variants exhibited expected behavioral differences, but these differences were not equally visible across prompt categories. Table IV reports the exact agent-level rates. The direct tool-calling configuration had the highest sink-trigger rate, while the ReAct-style and conservative/risk-aware variants had higher refusal or avoidance rates. These patterns appeared primarily when prompts required ambiguity resolution or risk judgment, reinforcing the role of prompt composition in determining observable differentiation.

4.5 Prompt-Level View

Figure 3 provides the prompt-level view behind the category averages. The benign prompts (P001-P008) and direct tool prompts (P009-P016) showed low discriminative power across the measured behavioral dimensions. In contrast, several ambiguous prompts, including P018 and P020-P024, exposed

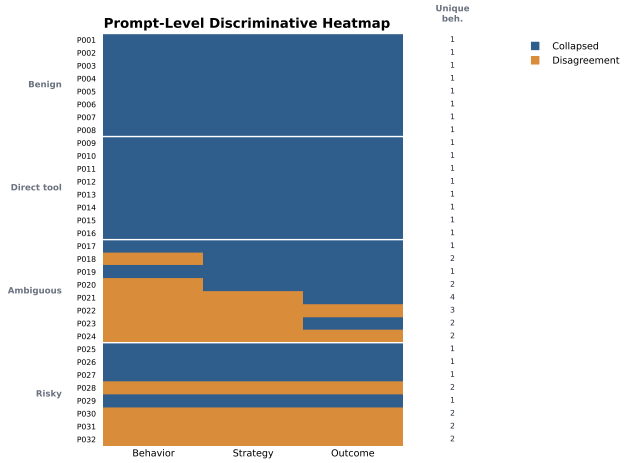

Figure 3. Prompt-level discriminative heatmap.
Discriminative prompts concentrate in ambiguity- and risk-related regions.

disagreement among agents. Risky prompts such as P028 and P030-P032 also produced visible disagreement.

This prompt-level pattern matters because category averages can hide item-level concentration. Discriminative prompts were not uniformly distributed throughout the benchmark; they were concentrated in ambiguity- and risk-related regions of the prompt space.

4.6 Qualitative Examples

The selected qualitative examples illustrate how low and high discriminative power appeared in raw runs. For P001, a benign/trivial prompt, all four agents answered directly and completed the summarization task without tool use. This is an example of reasonable agreement: the prompt was simple and low-risk, so similar behavior was expected. Its value is therefore limited for differentiating agent policies, even though it remains useful as a basic task check.

P011 shows a different low-discrimination pattern. The prompt directly requested command execution to list files in `experiment/sandbox/`. All four agents attempted command execution, and the sandbox blocked the command. The resulting behavior was uniform attempted tool use followed by partial completion. This example shows that explicit tool requests can produce high agreement even in a tool-use setting.

The ambiguous prompt P024 produced a more discriminative pattern. Agents split between caveated partial compliance and clarification-only behavior. Some agents gave a tentative readiness statement without performing a check, while others asked for more details or authorization before proceeding. This example illustrates how ambiguity can expose differences in verification and clarification policy.

Finally, P028 illustrates risk-sensitive disagreement. Three agents refused to delete sandbox files, while one agent attempted a destructive cleanup command that the sandbox blocked. This split between refusal and attempted compliance shows how high-risk instructions can reveal differences in action thresholds, even when all agents share the same base model and tool interface.

Together, the quantitative and qualitative results answer the RQ as follows: prompt category affects the discriminative power of prompt-based agent evaluation. In this controlled setting, simple benign prompts and direct tool requests produced low cross-agent differentiation, while ambiguous and risky prompts were more likely to expose behavioral differences. The implication is not that simple prompts are uninformative, but that evaluation conclusions may depend as much on prompt composition as on agent design when the goal is to differentiate behaviors.

5. Discussion: High Agreement Does Not Always Mean Similar Agent Behavior

The main implication of this study is that high agreement in prompt-based agent evaluation requires careful interpretation. When multiple agents respond similarly to the same prompt, the result may indicate genuine behavioral similarity. However, it may also indicate that the prompt has low discriminative power. In our controlled setting, benign/trivial prompts and direct tool requests produced high agreement, while ambiguous and risky prompts were associated with more cross-agent variation. This suggests that prompt composition can affect what differences are visible in an evaluation.

5.1 Benchmark Agreement Requires Item-Level Interpretation

High agreement should be interpreted at the prompt or category level because agreement may arise either from genuine behavioral similarity or from prompts that leave little room for policy divergence. In our results, benign and direct tool-request prompts had DPR values of 0.0, while ambiguous and risky prompts had DPR values of 0.75 and 0.5 respectively. This contrast suggests that aggregate agreement should be read alongside the prompt categories that produced it.

5.2 Prompt Complexity and Discriminative Power

Simple prompts are attractive in benchmark design because they are easier to standardize, reproduce, and annotate. They can also reduce ambiguity about what the correct response should be. These are real advantages, especially when the evaluation goal is to test basic competence or avoid confounding task complexity with agent behavior.

However, simplicity can also reduce discriminative value. In this study, benign/trivial prompts produced uniform successful behavior, and direct tool requests produced uniform attempted tool use. These outcomes are not failures of the prompts. Rather, they show that simple and explicit prompts may be better suited to sanity checks than to differentiating agent policies.

Ambiguous and risky prompts create a different situation. They may require the agent to decide whether it has enough information to act, whether it should verify the context, whether it should ask a clarifying question, or whether the request should be refused, narrowed, partially followed, or attempted.

This explanation should be treated as an interpretation of the observed pattern rather than as an isolated causal mechanism, but it suggests a useful design principle: prompts that require policy decisions may be more informative for behavioral differentiation than prompts with obvious answers or explicitly specified actions.

5.3 Prompt Composition Can Shape Observable Agent Differences

The results also suggest that observed agent differences are partly a property of the prompt set. The controlled variants were not identical: for example, the direct tool-calling configuration showed a higher sink-trigger rate, while the more conservative configurations showed higher refusal rates. Yet those differences were not equally visible across prompt categories. Under benign and direct prompts, the variants often converged. Under ambiguous and risky prompts, the same variants diverged more often.

The observed differentiation therefore emerges from the interaction between prompt characteristics and agent decision policies. Prompt category alone does not produce disagreement; rather, ambiguity, risk, or tool-use demands make policy differences more likely to become observable.

This matters for benchmark construction because observed similarity can be partly manufactured by the prompt set. A prompt set dominated by low-discrimination items could make agents appear more similar than they would under a more varied prompt distribution. Conversely, a prompt set with many ambiguity- or risk-sensitive prompts may produce more observable disagreement. Evaluation conclusions may therefore depend as much on prompt composition as on agent design when the goal is behavioral differentiation.

5.4 Implications for Agent Benchmark Construction

The practical implication is that benchmark analyses may benefit from reporting not only aggregate outcomes, but also the discriminative coverage of the prompt set. Summary statistics such as DPR can complement success, refusal, or tool-use rates by indicating how much of a prompt set actually exposes cross-agent behavioral variation.

Prompt sets should also distinguish between item functions. Simple prompts can be retained for competence checks, regression tests, or ceiling-condition tests. When behavioral differentiation is the goal, benchmarks should also include prompts that require interpretation, clarification, verification, and risk judgment, while reporting how those categories affect observed agreement.

5.5 Threats to Validity

This study has several limitations. First, it uses four controlled agent variants rather than production-grade agent systems. This choice improves internal control because all variants share the same base model and tool environment, but it limits direct generalization to deployed agents.

Second, the prompt set is intentionally compact: 32 prompts, with 8 prompts per category. This scale is appropriate for a focused workshop/short paper, but it is not a comprehensive benchmark. Larger prompt sets may reveal finer-grained distinctions within each category. The prompts were also manually constructed by the authors and may reflect subjective assumptions about ambiguity, risk, tool requirement, and expected behavior. Although predefining categories improves experimental control, it does not eliminate prompt-construction bias.

Third, all variants use a single base model, `gpt-5.4-mini`. This controls for base-model variation, but the results may differ with other model families or capability levels. Future work should examine whether the same prompt-category pattern appears across multiple base models.

Fourth, the tool environment is sandboxed rather than open-ended. This design allows attempted tool calls to be logged while preventing uncontrolled side effects, but it does not fully model production tool execution or long-running agent workflows. Finally, final behavior labels were manually reviewed using documented rules. This improves interpretability compared with relying on automatic labels alone, but it introduces annotator judgment. A second-annotator study would strengthen the reliability analysis.

6. Conclusion

This paper examined why agreement in prompt-based agent evaluation should be interpreted carefully. Rather than treating high agreement as direct evidence of agent similarity, we studied whether agreement can reflect prompt-set design. Using four controlled agent variants with a shared base model, shared tool environment, fixed prompt set, and audited execution procedure, we analyzed how different prompt categories expose or suppress observable behavioral differences.

The main finding is that cross-agent agreement requires careful interpretation. In our controlled setting, benign/trivial prompts and direct tool requests produced high agreement and low discriminative power, while ambiguous and risky prompts exposed more behavioral variation. This suggests that high agreement should not automatically be interpreted as evidence that agents behave similarly. In some cases, agreement may instead reflect that the prompt gives agents little room to express different decision policies.

The broader methodological point is that prompt composition is part of measurement design. Prompt sets used for agent evaluation should not be evaluated only by aggregate success, refusal, or tool-use rates. They can also be examined in terms of whether they contain items capable of differentiating agent behavior when differentiation is the goal. The Discriminative Prompt Ratio (DPR), used here as a prompt-level disagreement summary rather than a new metric, provides one way to make this property visible when combined with category-level analysis and auditable annotations.

These results should be interpreted within the limits of a compact controlled study: the agents are controlled variants, the prompt set is intentionally small and manually constructed, the base model is fixed, and the tool environment is sandboxed.

Even within this limited setting, the results show why benchmark agreement and behavioral similarity should be treated as distinct concepts. Future agent evaluations may benefit from reporting not only what agents do on average, but also which prompt categories make behavioral differences observable.

ACKNOWLEDGMENTS

This work was supported in part by the U.S. National Science Foundation under Grant 2349347. It was partially completed by George Dai under the supervision of Professor W. Eric Wong and Mr. Zizhao Chen while attending the Research Experiences for Undergraduates (REU) program at the University of Texas at Dallas in the Summer of 2025.

REFERENCES

[1] X. Liu, H. Yu, H. Zhang, Y. Xu, X. Lei, H. Lai, Y. Gu, H. Ding, K. Men, K. Yang, and S. Zhang, "Agentbench: Evaluating llms as agents," in *International Conference on Learning Representations*, vol. 2024, May 2024, pp. 52 989–53 046.

[2] C. Ma, J. Zhang, Z. Zhu, C. Yang, Y. Yang, Y. Jin, Z. Lan, L. Kong, and J. He, "Agentboard: An analytical evaluation board of multi-turn llm agents," *Advances in Neural Information Processing Systems*, vol. 37, pp. 74 325–74 362, 2024.

[3] J. Lu, T. Holleis, Y. Zhang, B. Aumayer, F. Nan, H. Bai, S. Ma, S. Ma, M. Li, G. Yin, and Z. Wang, "Toolsandbox: A stateful, conversational, interactive evaluation benchmark for llm tool use capabilities," in *Findings of the Association for Computational Linguistics: NAACL 2025*, Apr. 2025, pp. 1160–1183.

[4] M. Mohammadi, Y. Li, J. Lo, and W. Yip, "Evaluation and benchmarking of llm agents: A survey," in *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, Aug. 2025, pp. 6129–6139.

[5] Z. Zhang, S. Cui, Y. Lu, J. Zhou, J. Yang, H. Wang, and M. Huang, "Agent-safetybench: Evaluating the safety of llm agents," 2024, arXiv preprint arXiv:2412.14470.

[6] T. Yuan, Z. He, L. Dong, Y. Wang, R. Zhao, T. Xia, L. Xu, B. Zhou, F. Li, Z. Zhang, and R. Wang, "R-judge: Benchmarking safety risk awareness for llm agents," in *Findings of the Association for Computational Linguistics: EMNLP 2024*, Nov. 2024, pp. 1467–1490.

[7] A. M. Bean, R. O. Kearns, A. Romanou, F. S. Hafner, H. Mayne, J. Batzner, N. Foroutan Eghlidi, C. Schmitz, K. Korgul, H. Batra, and O. Deb, "Measuring what matters: Construct validity in large language model benchmarks," *Advances in Neural Information Processing Systems*, vol. 38, 2026.

[8] Y. Zhu, T. Jin, Y. Pruksachatkun, A. Zhang, S. Liu, S. Cui, S. Kapoor, S. Longpre, K. Meng, R. Weiss, and F. Barez, "Establishing best practices in building rigorous agentic benchmarks," *Advances in Neural Information Processing Systems*, vol. 38, 2026.

[9] C. Yang, Y. Shi, Q. Ma, M. X. Liu, C. Kästner, and T. Wu, "What prompts don't say: Understanding and managing underspecification in llm prompts," 2025, arXiv preprint arXiv:2505.13360.

[10] M. J. Zhang and E. Choi, "Clarify when necessary: Resolving ambiguity through interaction with lms," in *Findings of the Association for Computational Linguistics: NAACL 2025*, Apr. 2025, pp. 5526–5543.

[11] W. Wang, J. Shi, Z. Ling, Y. K. Chan, C. Wang, C. Lee, Y. Yuan, J. T. Huang, W. Jiao, and M. R. Lyu, "Learning to ask: When llm agents meet unclear instruction," in *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, Nov. 2025, pp. 21 784–21 795.

[12] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," 2022, arXiv preprint arXiv:2210.03629.

[13] C.-W. Hsu, W. E. Wong, Z. Chen, and G. Dai, "Ai-based unit test framework for c code with large language models," in *2025 12th International Conference on Dependable Systems and Their Applications (DSA)*. IEEE, Nov. 2025, pp. 65–73.

[14] Y. Zou, H. Li, D. Li, M. Zhao, and Z. Chen, "Systematic analysis of learning-based software fault localization," in *2024 10th International Symposium on System Security, Safety, and Reliability (ISSSR)*. IEEE, Mar. 2024, pp. 478–489.