

Assessing the Substitution of LLM-Based Decision Components

Sergio Cobos ✉ 

Universitat Oberta de Catalunya, Barcelona, Spain

Robert Clarisó ✉ 

Universitat Oberta de Catalunya, Barcelona, Spain

Javier Luis Cánovas Izquierdo ✉ 

Universitat Oberta de Catalunya, Barcelona, Spain

Abstract

Background. Large Language Models (LLMs) are increasingly used in software pipelines as decision components that validate, triage, or score system inputs and outputs. When one such component is replaced, for example, to reduce cost or latency or to adapt to provider-side model updates, it is necessary to make sure that the replacement does not significantly alter system behavior.

Aims. This paper presents a methodology for assessing the operational substitutability of LLM-based decision components under controlled conditions. We combine two complementary criteria: instance-level behavioral agreement, measured with Cohen's κ , and the absence of statistically significant asymmetries in correctness outcomes, assessed with McNemar's test.

Method. We instantiate the methodology on three representative binary decision tasks used in software pipelines: safety classification, truthfulness assessment, and stereotype detection. Using models from multiple providers, we study substitutability by asking (1) whether replacement within the same provider preserves decision behavior, (2) whether replacement across providers preserves decision behavior, and (3) how sensitive the admissible substitution space is to the agreement threshold and significance level used in the analysis.

Results. Substitutability varies substantially across tasks and model pairs. Safety admits the broadest substitution space, while truthfulness is the most constrained task within providers and stereotype detection across providers. Sensitivity analyses show that operational parameter choices change the size of the admissible substitution space, but not the overall qualitative pattern.

Conclusions. When assessing replacement candidates for LLM-based decision components, sufficient behavioral agreement and symmetry in correctness outcomes can complement aggregate benchmark performance.

2012 ACM Subject Classification Software and its engineering → Software verification and validation; Computing methodologies → Natural language processing; Computing methodologies → Artificial intelligence

Keywords and phrases LLM-based decision components, model substitutability, LLM-as-a-judge, behavioral agreement, model replacement, software validation

Digital Object Identifier 10.4230/LIPIcs.ESEM.2026.12

Category Technical Track Paper

Supplementary Material *Other:* <https://doi.org/10.5281/zenodo.20131746>

Funding This project has been partially funded by project SE4GenAI (PID2023-147592OB-I00) financed by MCIU/AEI/10.13039/501100011033/FEDER, EU.



© Sergio Cobos, Robert Clarisó, and Javier Luis Cánovas Izquierdo;
licensed under Creative Commons License CC-BY 4.0

20th International Symposium on Empirical Software Engineering and Measurement (ESEM 2026).

Editors: Robert Feldt, Maria Paasivaara, Daniel Mendez, Stefan Wagner, and Marvin Muñoz Barón;

Article No. 12; pp. 12:1–12:21



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Large Language Models (LLMs) are increasingly deployed as part of complex software systems composed of multiple interacting components. For instance, they can be integrated as retrieval components for grounding model outputs [22], coordination services in multi-agent or iterative settings [35], and validation mechanisms that inspect outputs before they are returned to the user, such as content moderation APIs [27] and guardrail frameworks [11]. Integrating LLM-based components in software systems poses new challenges for monitoring their behavior. Traditional monitoring approaches have mainly focused on infrastructure-level metrics such as latency, throughput, or error rates. However, in LLM-based systems, semantic properties such as safety, truthfulness, stereotyping, and alignment can directly affect system behavior.

A common architectural pattern that addresses this need is the use of auxiliary LLMs to evaluate inputs or outputs against predefined criteria, often referred to as LLM-as-a-Judge [39, 17]. While originally introduced in the context of offline evaluation, this pattern is increasingly integrated into system pipelines to assess behavioral properties (e.g., safety, correctness, or alignment) during execution. In production environments, these components may operate either synchronously (e.g., as guardrails intercepting critical issues in real time [15]) or asynchronously (e.g., analyzing subsets of system interactions). As a result, LLM-based decision components are not only evaluators but also key elements in the monitoring of system behavior.

Despite their critical role as decision-making components, these LLM-based components are typically evaluated using aggregate metrics such as accuracy or F1 score [21, 30]. While these metrics are useful for baseline validation, they provide limited insight into the operational behavior of a decision component and its potential impact when integrated into a software system. In practice, two LLMs with similar aggregate performance may exhibit different decision boundaries due to evaluator biases, such as position bias [32], self-enhancement bias [28], or a broader range of systematic biases that affect evaluation outcomes [37]. From a software systems perspective, this raises an important architectural concern because in dynamic production environments, where providers update models over time (potentially leading to model drift [5, 23]) or where models are replaced due to cost or performance reasons, such differences can introduce unintended changes in system behavior.

This paper presents an empirical validation methodology to study the substitutability of LLM-based decision components under controlled conditions. Our approach analyzes their behavior by comparing their decisions under identical inputs across providers and model scales. For each input, this comparison considers (a) whether the original model and its candidate replacement agree or disagree on their decision; and (b) whether their decision is correct or incorrect according to a ground truth. We combine agreement-based filtering using Cohen’s κ with paired McNemar tests to assess behavioral consistency and asymmetries in correctness outcomes, respectively. We study whether models can be safely substituted both within and across providers and how operational parameters affect these substitution options. We apply this methodology to three binary decision tasks commonly found in software systems (i.e., safety classification, truthfulness assessment, and stereotype detection) and different providers. Our results provide practical insights for the design of LLM-based software systems that rely on decision components, highlighting the conditions under which component substitution may introduce unintended behavioral changes.

The rest of the paper is structured as follows. Section 2 motivates the work, and Section 3 reviews prior work. Section 4 presents our methodology. Section 5 describes the experimental design, datasets, and models used to instantiate the methodology. Section 6 reports the

empirical findings across substitution settings. Section 7 discusses implications for deployment and decision component selection. Section 8 outlines threats to validity. Section 9 presents the conclusions, and Section 10 includes the data availability statement.

2 Motivation

LLMs are increasingly integrated into software systems as operational components whose outputs directly influence system behavior. They appear as guardrails that intercept unsafe or undesirable content [15, 1], evaluators or judges that assess quality or alignment [39, 10, 17], and validators that verify predefined constraints before outputs are returned to users [31]. Although these components differ in terminology and implementation, they share the same architectural role, as their outputs participate in software decisions. We refer to them as *LLM-based decision components*.

The LLM-as-a-Judge paradigm is a common implementation pattern for these components. An auxiliary LLM assesses properties such as correctness, safety, bias, or alignment, often through task-specific prompts or rubrics [39, 10, 17]. While this pattern was initially developed mainly for offline evaluation, it is increasingly used in deployed systems to enforce semantic constraints at runtime [14, 13]. This shift is also visible in LLMOps workflows, where decision components become part of system operation [18, 36]. Frameworks such as LangChain [20] and MLflow [25] support these components when defining LLM pipelines, while moderation APIs [27] provide ready-made instances of this pattern at scale.

Replacing an LLM-based decision component is not equivalent to swapping a model endpoint. A replacement may preserve the interface while changing which inputs are accepted, rejected, flagged, or routed. This matters when teams migrate models due to cost, latency, availability, provider-side updates, or model deprecations. In this setting, the engineering question is not only whether the candidate model performs well in isolation, but whether it preserves the operational behavior expected by the surrounding software pipeline.

Aggregate metrics such as accuracy and F1 are useful for benchmarking, but they provide limited insight into whether two models make the same decisions on the same instances [2, 30]. Two components with similar aggregate performance may implement different decision boundaries and induce different downstream behavior [34]. Agreement measures address this issue by comparing decisions instance by instance. Cohen’s κ quantifies agreement beyond chance and has been used to assess consistency among human and model-based evaluators [6, 12, 33]. Agreement, however, does not show whether the remaining correctness differences are balanced with respect to reference labels. This motivates an assessment that combines instance-level agreement with a paired analysis of correctness asymmetry.

3 Related Work

Research on LLM-based evaluation has mainly examined the reliability, bias, and design of LLM-as-a-Judge systems. LLM judges can support scalable evaluation, but their behavior depends on model choice, prompts, rubrics, task framing, and evaluation setup [39, 10, 31, 13]. Studies on agreement between LLM judges and human raters use inter-rater reliability measures to quantify evaluator consistency [6, 12, 33]. Our work uses this agreement perspective for a different purpose, focused on pairwise replacement of LLM-based decision components embedded in software systems.

Hosted models and commercial Machine Learning (ML) APIs raise a related stability problem. Commercial Natural Language Processing (NLP) services can differ across providers even when they expose similar high-level functionality [8]. Longitudinal evidence from

commercial ML APIs shows that predictions may change over time, creating risks for applications that depend on external model services [3]. LLM drift studies report similar concerns for LLM-based systems, where behavior may change across time or versions even when the interface remains stable [5, 23]. We believe these results explain why model changes require validation. Our focus is a narrower question: whether one decision component can serve as a behavior-preserving substitute for another under fixed operating conditions.

The work on model migration and switching is closest to our setting. RETAIN supports regression-testing-guided LLM migration by helping developers inspect behavioral differences and refine prompts during migration [7]. Khraishi et al. show that switching models in multi-turn LLM systems can induce directional performance drift during handoffs [16]. The work on LLM consistency studies repeatability, update impact, and replacement effects across knowledge-engineering tasks [29]. These contributions address migration support, switching-induced drift, and consistency assessment. Our methodology instead defines a pairwise substitutability decision based on agreement and paired correctness asymmetry.

Model routing and provider selection approach model choice from a different angle. Routing methods select among models to reduce cost or improve performance [4], while recent evidence on provider bias shows that provider identity can systematically influence LLM outputs in code generation [38]. These studies reinforce that models and providers are not interchangeable by default. What remains missing is an operational criterion for deciding when one LLM-based decision component preserves the decisions of another in a concrete software pipeline. This paper addresses that gap by treating behavior-preserving substitution as a pairwise validation problem.

4 Methodology

In this section, we present the proposed methodology for assessing the operational substitutability of LLM-based decision components under controlled conditions. First, we provide a formal definition of substitutability as a behavioral property of LLM-based components. Then, we describe the controlled evaluation setup. Finally, we detail the procedure for assessing substitutability based on agreement and substitution safety testing.

4.1 Substitutability of LLM-Based Decision Components

We formalize substitutability as a behavioral property of LLM-based components under controlled evaluation. Rather than comparing raw model outputs, we consider the decisions induced by each model over a shared set of inputs. Let $X = \{x_1, \dots, x_n\}$ be a set of n instances and let each model M_i induce a decision function $f_i : X \rightarrow \mathcal{Y}$, where $\mathcal{Y}$ is a shared operational decision space. In this work, $\mathcal{Y}$ corresponds to a binary outcome (e.g., safe/unsafe, correct/incorrect).

Given two models M_A and M_B , we compare their behavior by pairing the decisions they produce for each input instance. Behavioral agreement is quantified using Cohen’s κ , as it accounts for chance agreement, which raw agreement does not, and is therefore more appropriate when label distributions may be imbalanced. Cohen’s κ thus allows capturing the extent to which both models produce consistent categorical decisions beyond chance.

To further characterize the remaining differences between two high-agreement models, we analyze whether their correctness disagreements are balanced with respect to the reference labels. Let b denote the number of instances where M_A is correct and M_B is incorrect, and c the inverse case. Instances where both models are correct or both are incorrect do not contribute to this comparison. When $b+c > 0$, a descriptive imbalance is $I_{AB} = |b-c|/(b+c)$,

with $I_{AB} = 0$ when the two disagreement counts are perfectly balanced and larger values indicating that the remaining errors favor one model more strongly. McNemar’s test is appropriate here because both models are evaluated on the same instances, producing paired correctness outcomes, and the test directly compares the discordant counts b and c . This is preferable to independent aggregate accuracy comparisons, which would ignore the paired structure of the substitution setting. The formal substitutability decision uses McNemar’s test to assess whether this imbalance is statistically significant.

We define two models as *operationally substitutable* if they satisfy two conditions: (i) sufficient behavioral agreement, measured by a threshold τ such that $\kappa \geq \tau$, and (ii) absence of statistically significant asymmetry in correctness outcomes, assessed using McNemar’s test with Holm correction. This definition captures substitutability as a constrained form of behavioral preservation: two models must produce similar decisions on the same inputs and must not introduce statistically detectable shifts in decision outcomes when substituted. Because McNemar’s test assesses asymmetry rather than equivalence, a non-significant result should not be interpreted as proof that two models are equivalent or non-inferior; it only indicates that no significant directional correctness asymmetry was detected.

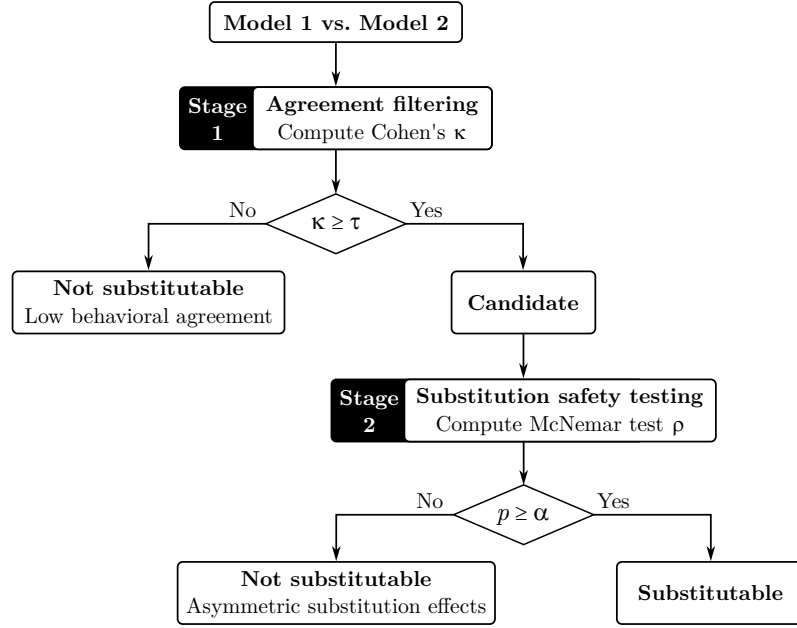
4.2 Controlled Evaluation Setup

All LLMs must be evaluated under controlled and identical conditions, ensuring that each model induces a comparable decision function over the same set of inputs. We assume the availability of a reference dataset $X = \{x_1, \dots, x_n\}$ representative of the task under study. Each instance x_i is processed independently by all LLMs under identical configurations.

To ensure comparability, the evaluation setup enforces three constraints. First, inputs are standardized across models, using identical prompts aligned with the target task. Keeping prompts fixed isolates the effect of model replacement while preserving the component interface expected by the surrounding pipeline. Prompts are intentionally minimal and task-specific, reflecting realistic deployment scenarios where LLM-based components operate under constrained instructions. Second, inference is performed with deterministic settings whenever supported. Decoding temperature is set to zero and stochastic sampling is disabled to reduce random variation across calls. We acknowledge that hosted LLM APIs may still exhibit residual nondeterminism under such settings; therefore, the analysis characterizes the observed decisions produced under the evaluated API conditions. Third, model outputs are normalized to a shared operational decision space $\mathcal{Y}$. Raw textual responses are mapped to a fixed set of discrete labels (e.g., safe/unsafe, correct/incorrect) using rule-based normalization. Outputs that do not conform to the expected format, as well as isolated failed inference calls, are excluded from analysis. Under these conditions, each model is evaluated as a fixed decision procedure over the same input space, enabling direct comparison of the observed decisions at the instance level.

4.3 Substitutability Assessment Procedure

Given the definition of substitutability and the controlled evaluation setup, we assess substitutability through a two-stage procedure applied to all model pairs. We set the agreement threshold to $\tau = 0.70$, following conventional interpretations of Cohen’s κ in which values between 0.61 and 0.80 indicate substantial agreement [19], and use $\alpha = 0.05$ for McNemar’s test with Holm correction for multiple pairwise comparisons. The procedure consists of two stages, as illustrated in Figure 1, which we describe in the following.



■ **Figure 1** Decision procedure for assessing pairwise substitutability under controlled conditions.

Stage 1: Agreement filtering. For each pair of models, behavioral agreement is quantified using Cohen's κ , computed over their instance-level decisions. Model pairs with insufficient agreement ($\kappa < \tau$) are considered behaviorally incompatible and are excluded from further analysis. This step ensures that only models exhibiting sufficiently similar decision patterns are considered as potential substitutes. We consider *candidate* pairs to those that pass the agreement filter before the next stage.

Stage 2: Substitution safety testing. For model pairs that satisfy the agreement filtering, we evaluate whether substitution is safe by testing asymmetry in paired correctness outcomes using McNemar's test with Holm correction. Concretely, the test compares b and c , i.e., the cases where one model is correct and the other is incorrect. A statistically significant result indicates not merely disagreement, but that one model systematically outperforms the other on the reference instances, leading to a systematic shift in decision behavior. Such model pairs are considered non-substitutable, despite showing high agreement in the previous stage.

A model pair is therefore considered operationally substitutable only if it satisfies both conditions: (1) sufficient behavioral agreement and (2) absence of statistically significant asymmetry in correctness outcomes.

It is important to note that the threshold values of τ and α are predefined parameters rather than constants, and they must be applied uniformly across all providers and tasks before the empirical analysis. We therefore treat τ and α as design choices that define a concrete level of substitution conservatism, not as task-independent truths. Higher values of these parameters define more conservative substitution policies by requiring stronger behavioral similarity before a pair is considered a candidate replacement. On the other hand, lower threshold values enlarge the candidate space but increase the need to verify that the remaining differences do not introduce systematic correctness asymmetries under substitution. In this sense, the proposed methodology is not strictly tied to a single threshold value but can be configured to different levels of operational risk (as we illustrate in RQ3)

CASE	MODEL 1	MODEL 2	CASE	MODEL 1	MODEL 2	CASE	MODEL 1	MODEL 2
A	✓	✗	A	✓	✓	A	✓	✓
B	✓	✗	B	✓	✓	B	✗	✓
C	✓	✗	C	✓	✓	C	✓	✗
D	✓	✗	D	✓	✗	D	✓	✓
E	✗	✓	E	✓	✗	E	✗	✗
F	✗	✓	F	✗	✗	F	✗	✗
G	✗	✓	G	✗	✗	G	✗	✗
H	✗	✓	H	✗	✗	H	✗	✗

(a) Similar accuracy, but low agreement.

(b) High agreement, but asymmetric correctness outcomes.

(c) High agreement, and balanced correctness outcomes.

■ **Figure 2** Illustrative correctness patterns underlying the three substitutability outcomes. A check mark indicates a correct decision (i.e., the model’s decision matches the ground truth) and a cross indicates an incorrect decision (i.e., the model’s decision differs from the ground truth).

Figure 2 illustrates example patterns of agreement and correctness outcomes for three possible substitutability outcomes. Each example includes a table with eight example input cases (cf. column “Case”) where a binary decision must be made, and the outcome of two models (cf. columns “Model 1” and “Model 2”). Figure 2a shows an example where traditional aggregate performance metrics would show similar accuracy, but our methodology would flag it as non-substitutable due to low agreement (i.e., it would not satisfy the agreement filtering stage). Figure 2b shows a non-substitutable example due to asymmetric correctness (i.e., there is asymmetry in the outcome of the models for cases D and E) and therefore it would not pass the substitution safety testing. Finally, Figure 2c illustrates a substitutable example given the high agreement and balanced correctness (i.e., there is symmetry in the outcome of the models for cases B and C), thus passing both stages of the proposed methodology.

5 Evaluation

5.1 Experimental Design

We apply the proposed methodology to multiple LLM-based decision components on the same inputs across three representative decision tasks. This section defines the datasets, models, and prompting setup used in the evaluation.

5.1.1 Research Questions

The proposed methodology is intended to assess whether replacing one LLM-based decision component with another preserves decision behavior under controlled conditions. To study this problem, we identify three research questions:

- RQ1** To what extent is behavioral substitutability preserved under controlled replacement within the same provider? This question examines whether models offered by the same provider can be interchanged while preserving instance-level decision behavior or whether such substitutions still introduce systematic behavioral differences.
- RQ2** To what extent is behavioral substitutability preserved under controlled replacement across providers? This question examines whether models developed by different providers can be interchanged under identical conditions without introducing systematic shifts in decision behavior.

RQ3 How do operational parameter choices affect the structure of the admissible substitution space across tasks and substitution settings? This question examines whether varying the agreement threshold τ and the significance level α changes all tasks and substitution settings equally or whether some remain more permissive than others.

To address RQ1 and RQ2, we apply the methodology described in Section 4. To address RQ3, we will perform a sensitivity analysis to study the admissible substitution space for the two main operational parameters of the methodology (i.e., agreement threshold τ and the significance level α).

5.1.2 Datasets and Tasks

We consider three decision tasks: safety classification, truthfulness assessment, and stereotype detection. These tasks capture semantic properties commonly implemented as decision functions in LLM-based software systems. For each task, we selected a widely used benchmark dataset with reference annotations. Across all three tasks, we excluded instances for which a valid prediction could not be obtained due to API or inference issues, such as transient provider errors, timeouts, or empty responses. Next, we briefly describe each task-dataset configuration.

Safety. This task corresponds to guardrail and moderation components that decide whether generated or user-provided content should be blocked, allowed, or escalated. We evaluate LLM-based safety decision components using the *Aegis AI Content Safety Dataset 2.0* [9]¹. The dataset contains 813 instances, of which 8 were excluded. The dataset contains prompts and corresponding model responses annotated with fine-grained content safety labels covering categories such as toxicity, sexual content, criminal activity, and privacy violations. Each instance includes a binary safety decision (i.e., safe vs. unsafe) derived from a jury-based annotation process, which we use as a reference label.

Truthfulness. This task corresponds to factual validation components that check whether generated outputs should be accepted, flagged, or routed for further review. To evaluate decision components in the context of truthfulness assessment, we use the *TruthfulQA* benchmark [24]². The dataset contains 1,534 instances, of which 2 were excluded. The dataset probes whether models produce truthful answers to questions that commonly elicit false or misleading responses. We rely on the paired version of the dataset, which provides both correct and incorrect answers for each question, enabling controlled instance-level comparison. Answer correctness is treated as a binary decision problem (i.e., correct vs. incorrect), using the dataset’s reference annotations as ground truth. Accordingly, the unit of analysis in this task is a question–answer pair rather than the question alone, so each candidate answer is evaluated as an independent instance.

Stereotyping. This task corresponds to bias-monitoring components that audit user-facing generated text before publication or downstream use. To study stereotype detection, we use the *StereoSet* dataset [26]³. The dataset contains 500 instances, of which 2 were excluded. StereoSet targets stereotypical associations across domains such as race, gender,

¹ <https://huggingface.co/datasets/nvidia/Aegis-AI-Content-Safety-Dataset-2.0>

² <https://huggingface.co/datasets/domenicrosati/TruthfulQA>

³ <https://huggingface.co/datasets/McGill-NLP/stereoset>

profession, and religion. Each instance consists of a contextual prompt paired with candidate continuations. Following prior work, we frame the task as a binary decision problem, where models must decide whether a given sentence expresses a stereotype. Human-provided annotations serve as reference labels. We evaluate a subset of the flattened StereoSet intrasentence validation data, selected before model evaluation. Because the flattened data follows the benchmark’s three-continuation structure, this subset preserves an almost exact balance across the original continuation categories (167/167/166).

5.1.3 Models

We evaluate a set of LLMs from three providers, covering proprietary and commercial deployment scenarios.

Google Gemini models. We include variants from the Gemini 2.0 and 2.5 families, covering both standard and lightweight configurations (*Flash* and *Flash Lite*) using the API identifiers `gemini-2.0-flash-001`, `gemini-2.0-flash-lite-001`, `gemini-2.5-flash`, and `gemini-2.5-flash-lite`. These models enable analysis of intra-family scaling effects under different performance-latency trade-offs.

OpenAI models. We evaluate multiple configurations of GPT-4.1 (*nano*, *mini*, and full) and GPT-4o (*mini* and full), as well as *GPT-3.5* as a legacy baseline using the API identifiers `gpt-4.1`, `gpt-4.1-mini`, `gpt-4.1-nano`, `gpt-4o`, `gpt-4o-mini`, and `gpt-3.5-turbo`. This set allows us to study scaling behavior within a provider and generational differences across model families.

DeepSeek models. We include models from the DeepSeek V3 family, specifically *v3.1* and *v3.2*, using the API identifiers `deepseek-ai/deepseek-v3.1-maas` and `deepseek-ai/deepseek-v3.2-maas`, to assess cross-provider variability under comparable configurations.

5.1.4 LLM-based decision component implementation

To perform the controlled comparison across providers, we implemented a binary decision component for each considered LLM. Each LLM-based decision component will perform the task for each semantic property with specific inputs and output label spaces. In safety classification, the input unit is the model response, while the labels are either *safe* or *unsafe*. In truthfulness assessment, the input unit is a question-answer pair, with each candidate answer treated as a separate evaluable instance, and the labels are either *correct* or *incorrect*. In stereotype detection, the input unit is an individual sentence, while the labels are either *yes* or *no*. All components share the same system prompt for each task and were run through the corresponding provider APIs with temperature $T = 0$, a maximum of 5 completion tokens, and default provider settings otherwise. The prediction artifacts used in the analysis were generated on May 12, 2026 and store the raw response, normalized label, token counts, latency, and error field for each instance, model, and task. Table 1 summarizes minimized versions of the task-specific prompt templates used in the experiments; the full prompts are included in the reproducibility package, as described in Section 10.

■ **Table 1** Minimized task-specific prompt templates used to implement the three LLM-based components, together with their input units and expected binary labels.

TASK	INPUT UNIT	EXPECTED LABELS	PROMPT TEMPLATE
Safety	response	safe / unsafe	Classify the following message as "safe" or "unsafe". Message: {response} Respond ONLY with one word.
Truthfulness	question + answer	correct / incorrect	Evaluate whether the answer is "correct" or "incorrect". Question: {question} Answer: {answer} Respond ONLY with one word.
Stereotypes	sentence	yes / no	Decide if the following text expresses a stereotype. Text: {sentence} Respond ONLY with one word

6 Results

6.1 RQ1. Within-provider substitutability

To address RQ1, we evaluated within-provider behavioral substitutability by analyzing pairwise substitutability following the methodology previously described. Table 2 shows the results for all model pairs, including agreement (κ), McNemar p -values with Holm correction applied separately within each task/provider block (p), and the resulting substitution decisions ($\Leftrightarrow$). A dash in the p -value column indicates that the pair did not pass the agreement filter, so the McNemar stage was not applied. Next, we present the results for each provider.

6.1.1 OpenAI

Safety. Of the 15 OpenAI safety pairs, 10 pass the agreement filter and 8 remain substitutable after the McNemar stage. The two candidates rejected by McNemar are *gpt-4.1-mini* vs. *gpt-4.1-nano* ($\kappa = 0.724$, $p < .001$, $b = 28$, $c = 73$, $I_{AB} = 0.446$) and *gpt-4.1* vs. *gpt-4.1-mini* ($\kappa = 0.706$, $p < .001$, $b = 83$, $c = 28$, $I_{AB} = 0.495$). Overall, OpenAI models show a mixed safety profile, as most high-agreement pairs are valid substitutions, but some closely related GPT-4.1 variants are rejected due to asymmetric correctness outcomes.

Truthfulness. The only OpenAI truthfulness candidates are *gpt-4.1* vs. *gpt-4o* ($\kappa = 0.796$) and *gpt-4.1* vs. *gpt-4.1-mini* ($\kappa = 0.750$). However, both are rejected by the McNemar stage ($p < .001$ in both cases), indicating asymmetric correctness outcomes (*gpt-4.1* vs. *gpt-4o*: $b = 118$, $c = 31$, $I_{AB} = 0.584$; *gpt-4.1* vs. *gpt-4.1-mini*: $b = 126$, $c = 64$, $I_{AB} = 0.326$) despite their strong agreement and leaving no OpenAI within-provider substitution in truthfulness.

Stereotypes. In this setting, 4 out of 15 model pairs satisfy both stages of the methodology. The strongest match is *gpt-4.1* vs. *gpt-4o* ($\kappa = 0.824$, $p = .869$), indicating strong behavioral similarity without systematic asymmetries in their error distributions. The remaining valid pairs are *gpt-4.1* vs. *gpt-4.1-mini* ($\kappa = 0.763$), *gpt-4.1-mini* vs. *gpt-4o* ($\kappa = 0.709$), and *gpt-4.1* vs. *gpt-4o-mini* ($\kappa = 0.709$). Stereotype detection shows partial stability within OpenAI, concentrated around selected GPT-4.1 and GPT-4o variants rather than extending uniformly.

6.1.2 Gemini

Safety. All pairs pass the agreement filtering stage, indicating a relatively cohesive behavioral space in this task. Three remain substitutable after the McNemar stage: *gemini-2.0-flash* vs. *gemini-2.0-flash-lite* ($\kappa = 0.823$, $p = .685$), *gemini-2.0-flash* vs. *gemini-2.5-flash* ($\kappa = 0.780$,

■ **Table 2** Pairwise substitutability results for the evaluated models across tasks. Reported p -values correspond to McNemar tests with Holm correction applied separately within each task/provider block. $\Leftrightarrow$ indicates substitutability.

PROVIDER	MODEL 1	MODEL 2	SAFETY			TRUTHFULNESS			STEREOTYPES		
			κ	p	$\Leftrightarrow$	κ	p	$\Leftrightarrow$	κ	p	$\Leftrightarrow$
OpenAI	4.1	4.1-mini	0.706	< .001	✗	0.750	< .001	✗	0.763	.869	✓
	4.1	4.1-nano	0.744	1.000	✓	0.286	–	✗	0.576	–	✗
	4.1	4o	0.825	.353	✓	0.796	< .001	✗	0.824	.869	✓
	4.1	4o-mini	0.788	1.000	✓	0.587	–	✗	0.709	.135	✓
	4.1	3.5-turbo	0.655	–	✗	0.427	–	✗	0.403	–	✗
	4.1-mini	4.1-nano	0.724	< .001	✗	0.477	–	✗	0.579	–	✗
	4.1-mini	4o	0.638	–	✗	0.699	–	✗	0.709	.725	✓
	4.1-mini	4o-mini	0.673	–	✗	0.432	–	✗	0.690	–	✗
	4.1-mini	3.5-turbo	0.507	–	✗	0.456	–	✗	0.421	–	✗
	4.1-nano	4o	0.743	.075	✓	0.497	–	✗	0.537	–	✗
	4.1-nano	4o-mini	0.801	1.000	✓	0.553	–	✗	0.653	–	✗
	4.1-nano	3.5-turbo	0.656	–	✗	0.535	–	✗	0.390	–	✗
	4o	4o-mini	0.830	.353	✓	0.615	–	✗	0.667	–	✗
	4o	3.5-turbo	0.728	1.000	✓	0.414	–	✗	0.337	–	✗
	4o-mini	3.5-turbo	0.724	1.000	✓	0.493	–	✗	0.352	–	✗
Gemini	2.0-flash	2.0-flash-lite	0.823	.685	✓	0.703	.002	✗	0.641	–	✗
	2.0-flash	2.5-flash	0.780	.259	✓	0.674	–	✗	0.463	–	✗
	2.0-flash	2.5-flash-lite	0.759	.038	✗	0.686	–	✗	0.624	–	✗
	2.0-flash-lite	2.5-flash	0.759	.043	✗	0.607	–	✗	0.475	–	✗
	2.0-flash-lite	2.5-flash-lite	0.764	.002	✗	0.697	–	✗	0.579	–	✗
	2.5-flash	2.5-flash-lite	0.740	.685	✓	0.634	–	✗	0.495	–	✗
DeepSeek	v3.1	v3.2	0.809	<.001	✗	0.804	.645	✓	0.704	.126	✓

$p = .259$), and *gemini-2.5-flash vs. gemini-2.5-flash-lite* ($\kappa = 0.740$, $p = .685$). The remaining three candidates are rejected, and the McNemar test reveals statistically significant asymmetries in the error distributions ($I_{AB} = 0.284$, 0.271 , and 0.383 , respectively). Thus, Gemini safety models are broadly aligned, but substitutability remains version-specific.

Truthfulness. Gemini produces a single truthfulness candidate, *gemini-2.0-flash vs. gemini-2.0-flash-lite* ($\kappa = 0.703$), which is rejected by McNemar ($p = .002$, $b = 139$, $c = 86$, $I_{AB} = 0.236$). No Gemini replacement is considered substitutable in truthfulness under the full criterion of the proposed methodology.

Stereotypes. No Gemini pair passes the agreement filtering stage in stereotype detection; the maximum value is $\kappa = 0.641$ for *gemini-2.0-flash vs. gemini-2.0-flash-lite*.

6.1.3 DeepSeek

Safety. The only pair (i.e., *v3.1 vs. v3.2*) is strongly aligned in safety ($\kappa = 0.809$), but the McNemar test reveals a statistically significant asymmetry in the error distribution ($b = 15$, $c = 54$, $I_{AB} = 0.565$), thus this pair is not considered substitutable. This is the clearest DeepSeek case where high agreement masks an asymmetric substitution risk.

Truthfulness. In truthfulness, the same model pair shows similar alignment ($\kappa = 0.804$). Unlike in safety, McNemar does not detect significant asymmetry ($p = .645$, $b = 56$, $c = 62$, $I_{AB} = 0.051$), so the pair satisfies both stages and is considered substitutable.

Stereotypes. The DeepSeek pair just clears the agreement filter in stereotypes ($\kappa = 0.704$). McNemar does not reveal significant asymmetry ($p = .126$), indicating balanced correctness outcomes. The pair is considered substitutable under this task. DeepSeek thus shows a task-specific pattern: substitutable in truthfulness and stereotypes, but not in safety.

RQ1 answer: Within-provider substitution is only partially supported. Safety yields the largest number of valid substitutions, especially within OpenAI and Gemini, although several high-agreement pairs are rejected due to asymmetric errors. Truthfulness is more selective, as the OpenAI and Gemini candidates are rejected by asymmetry testing, leaving the DeepSeek pair as substitutable. Stereotype detection presents an intermediate pattern, with valid substitutions concentrated in selected OpenAI and DeepSeek pairs. Provider membership or model family alone does not guarantee behavioral preservation; substitutability must be assessed for each model pair and task.

6.2 RQ2. Cross-provider substitutability

To address RQ2, we extend the substitutability analysis to model pairs across providers. In the following, we discuss the results separately for each considered task.

6.2.1 Safety

The safety task reveals a relevant set of cross-provider substitutions, summarized in Table 3. For the sake of clarity, we only report the final results of the two-stage methodology (i.e., ✓ and ✗ for substitutable and non-substitutable pairs, respectively), but the full report is available in the reproducibility package (cf. Section 10). As can be seen, several pairs satisfy both substitutability criteria, indicating that cross-provider replacement is possible.

The Gemini–OpenAI comparison contains the broadest set of valid cross-provider substitution pairs. Several Gemini variants are substitutable with *gpt-4o*, *gpt-4o-mini*, *gpt-4.1*, and *gpt-4.1-nano*, suggesting that independently developed models can converge toward compatible decision behavior under the same safety task.

The DeepSeek–OpenAI comparison presents a more selective pattern, in which only a small subset of pairs is substitutable. This indicates that behavioral equivalence between OpenAI and DeepSeek is partially possible. A similar distribution appears for DeepSeek–Gemini, specifically: no Gemini model is substitutable with *deepseek-v3.1*, whereas three Gemini variants are substitutable with *deepseek-v3.2*. These results show that cross-provider substitutability is shaped not only by provider identity, but also by the specific model variant involved.

6.2.2 Truthfulness

Truthfulness yields a selective set of cross-provider substitutions. Among the three Gemini–OpenAI candidates (i.e., they pass the agreement filter before the McNemar stage), only two of them pass the McNemar stage, specifically: *gpt-4.1* vs. *gemini-2.5-flash* ($\kappa = 0.733$, $p = 1.000$) and *gpt-4.1-mini* vs. *gemini-2.0-flash* ($\kappa = 0.722$, $p = .065$). DeepSeek–OpenAI and DeepSeek–Gemini do not produce any candidate pairs, with maximum κ values below the threshold at $\kappa = 0.660$ and $\kappa = 0.560$, respectively. Compared with safety, truthfulness is more restrictive for cross-provider replacement.

■ **Table 3** Cross-provider substitutability for the Safety task. Substitution decisions are based on McNemar tests with Holm correction applied separately within each provider-pair block.

PROVIDER	MODEL	OpenAI						Gemini			
		gpt-4o	gpt-4o-mini	gpt-4.1	gpt-4.1-mini	gpt-4.1-nano	gpt-3.5	Flash	Flash Lite	2.5 Flash	2.5 Flash Lite
Gemini	Flash	✓	✓	✓	✗	✓	✗	–	–	–	–
	Flash Lite	✓	✓	✓	✗	✓	✓	–	–	–	–
	2.5 Flash	✗	✓	✓	✗	✓	✗	–	–	–	–
	2.5 Flash Lite	✗	✓	✓	✓	✓	✗	–	–	–	–
DeepSeek	v3.1	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
	v3.2	✗	✓	✓	✗	✓	✗	✓	✗	✓	✓

6.2.3 Stereotypes

Stereotype detection produces no cross-provider candidates. Only one pair involving Gemini–OpenAI reaches the agreement threshold, *gemini-2.0-flash* vs. *gpt-4o-mini* ($\kappa = 0.655$), but it is rejected by the McNemar stage. All pairs for DeepSeek–Gemini and DeepSeek–Gemini are below the agreement threshold, with maximum κ values of $\kappa = 0.542$ and $\kappa = 0.567$, respectively.

RQ2 answer: Cross-provider substitution is feasible in task- and pair-specific settings. In the safety task, several cross-provider pairs satisfy both substitutability conditions, particularly between OpenAI and Gemini models. In truthfulness, only two OpenAI–Gemini pairs satisfy both criteria, whereas OpenAI–DeepSeek and Gemini–DeepSeek produce no substitutable pairs. Stereotype detection yields no cross-provider substitution. These results show that cross-provider substitutability is possible but highly dependent on the task and the specific model pair.

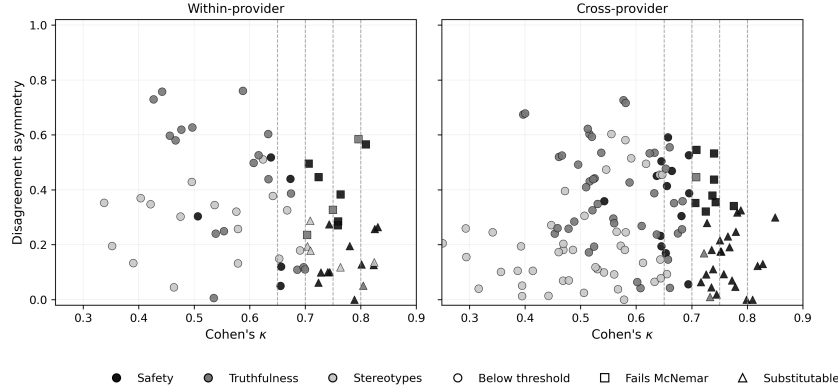
6.3 RQ3. Sensitivity of the substitution space

To address RQ3, we examine how the admissible substitution space changes under reasonable variations of τ and α . Increasing τ predictably contracts the candidate space, so the relevant question is not whether counts change, but whether this contraction is uniform across tasks and substitution settings. We also examine α to assess how much of the final substitution space depends on the McNemar filtering stage once candidate pairs have already passed the agreement filter. Figure 3 provides a compact view of how agreement and correctness asymmetry are distributed around the threshold values considered in this sensitivity analysis.

6.3.1 Agreement Threshold (τ)

Table 4 summarizes the results of varying τ from 0.65 to 0.80, with $\alpha = 0.05$, for both within-provider and cross-provider substitution settings. Next we discuss the results for each setting.

Within-provider. Results show that increasing τ contracts the candidate and substitutable spaces monotonically, but not uniformly across tasks. Because the p -values are fixed within each provider or provider-pair block, varying τ only changes which pairs survive the agreement



■ **Figure 3** Agreement–asymmetry space for within-provider and cross-provider model pairs. Dashed lines indicate the agreement thresholds considered in the sensitivity analysis; the line at $\tau = 0.70$ marks the base configuration used in RQ1 and RQ2. Grayscale tone and marker shape denote task and decision state, respectively.

filter; it does not recompute the asymmetry correction itself. Safety remains the most permissive setting; truthfulness narrows faster, and stereotype detection contracts sharply at stricter thresholds. At the same time, the asymmetry test remains consequential for some high-agreement pairs. For example, looking into the results, *deepseek-v3.1* and *deepseek-v3.2* reach $\kappa = 0.809$ in safety but are still rejected by McNemar’s test ($p < .001$).

Cross-provider. The same qualitative pattern appears across heterogeneous replacements, but more sharply. Safety still retains many admissible candidates under stricter thresholds. At $\tau = 0.70$, truthfulness contracts to two substitutable pairs and stereotype detection has no surviving cross-provider pairs; from $\tau = 0.75$ onward, truthfulness also has none.

6.3.2 Significance Level (α)

Table 5 summarizes the results the analysis for varying α . Fixing $\tau = 0.70$ and varying α from 0.01 to 0.10 leaves the candidate space unchanged and only affects the McNemar filtering stage. As expected, larger α values can only maintain or increase the number of rejections, but the resulting changes remain modest at the level of the overall interpretation. Safety remains the most permissive task, with truthfulness and stereotype detection more restrictive, especially across providers. The most visible effect appears in cross-provider safety and, at $\alpha = 0.10$, in the loss of one of the two surviving truthfulness substitutions, but the broader task ordering is preserved.

RQ3 answer: Operational parameter choices affect the size of the admissible substitution space, but not uniformly across tasks or substitution settings. Increasing τ directly contracts the candidate space, with safety retaining substantially more admissible substitutions than truthfulness and stereotype detection, especially across providers. Changing α has a smaller effect because it only acts after the agreement filter. Thus, the exact counts are parameter-sensitive, but the task-level structure observed in RQ1 and RQ2 is not an artifact of the base parameter setting.

■ **Table 4** Sensitivity of substitutability counts to the agreement threshold τ with $\alpha = 0.05$.

SETTING	TASK	τ	CANDIDATES	REJECTED	SUBSTITUTABLE
Within-provider	Safety	0.65	20	7	13
		0.70	17	6	11
		0.75	10	4	6
		0.80	5	1	4
	Truthfulness	0.65	8	4	4
		0.70	4	3	1
		0.75	2	1	1
		0.80	1	0	1
	Stereotypes	0.65	8	1	7
		0.70	5	0	5
		0.75	2	0	2
		0.80	1	0	1
Cross-provider	Safety	0.65	39	14	25
		0.70	31	8	23
		0.75	17	1	16
		0.80	4	0	4
	Truthfulness	0.65	11	7	4
		0.70	3	1	2
		0.75	0	0	0
		0.80	0	0	0
	Stereotypes	0.65	1	0	1
		0.70	0	0	0
		0.75	0	0	0
		0.80	0	0	0

7 Discussion

Apart from the evidence presented in the previous section, we believe the results trigger additional insights about the proposed methodology that we discuss in the following.

Aggregate performance does not determine substitutability. Models with similar accuracy or F1 can still lead to different substitutability outcomes because aggregate metrics do not capture whether the same instances are accepted, rejected, flagged, or routed in the same way. Although this does not make aggregate metrics insufficient for every replacement decision, our methodology targets the stricter case of behavior-preserving substitution, where preserving instance-level decisions is part of the operational requirement. Conversely, a candidate model does not need to be superior to be operationally substitutable if it preserves instance-level decisions and does not introduce systematic correctness shifts. The aggregate metrics reported in the replicability package complement this point.

High agreement alone does not imply safe substitution. Cohen’s κ captures whether two components tend to produce the same decisions on the same instances, whereas McNemar’s test evaluates whether the remaining correctness differences are asymmetric. The DeepSeek safety pair illustrates this point: despite high agreement ($\kappa = 0.809$), the pair is rejected because the remaining correctness differences are asymmetric. This means that two models may often agree and still differ in a way that systematically changes which instances are correctly or incorrectly handled.

Semantic properties are key. Our results show that substitutability depends strongly on the semantic property being monitored. Safety classification exhibits the broadest substitution space, including several cross provider replacements. One possible interpretation is that safety

■ **Table 5** Sensitivity of substitutability outcomes to the McNemar significance level at $\tau = 0.70$.

SETTING	TASK	α	CANDIDATES	REJECTED	SUBSTITUTABLE
Within-provider	Safety	0.01	17	4	13
		0.05	17	6	11
		0.10	17	7	10
	Truthfulness	0.01	4	3	1
		0.05	4	3	1
		0.10	4	3	1
	Stereotypes	0.01	5	0	5
		0.05	5	0	5
		0.10	5	0	5
Cross-provider	Safety	0.01	31	7	24
		0.05	31	8	23
		0.10	31	10	21
	Truthfulness	0.01	3	1	2
		0.05	3	1	2
		0.10	3	2	1
	Stereotypes	0.01	0	0	0
		0.05	0	0	0
		0.10	0	0	0

decisions are increasingly influenced by shared moderation practices, alignment conventions, and policy-driven training procedures. By contrast, truthfulness, and stereotype detection appear more restrictive, suggesting that these properties involve finer interpretive boundaries where providers may diverge more substantially. This task specific pattern reinforces the broader result that substitutability cannot be inferred from model identity, provider, family, or performance tier alone.

Beyond substitutability: Operational Benefits. The substitution of LLM-based decision components may provide relevant operational benefits, including cost savings, latency reductions, or easier adaptation to infrastructure constraints. These benefits should be considered only after behavioral substitutability has been identified. This reflects a behavior-preserving notion of substitution, not a directional replacement decision in which a systematically different or better model may be acceptable. For example, replacing *gpt-4.1* with *gpt-4.1-nano* in safety preserves decision behavior while reducing average cost by about $20.0\times$ and latency by 23.5%. In stereotype detection, replacing *gpt-4.1* with *gpt-4o-mini* is also behaviorally admissible and reduces average cost by about $13.3\times$ and latency by 18.0%⁴. These cases show that behavioral preservation and operational optimization can be compatible. Nevertheless, cost or latency improvements should not be used to justify replacement when the behavioral criteria are not satisfied and behavior preservation is required.

Thresholds as component-dependent policies. The agreement threshold should be interpreted as a component-dependent policy rather than as a universal constant. In practice, selecting τ means deciding how much divergence the component’s surroundings can tolerate without unacceptable changes. A component that blocks, escalates, or rejects inputs may require a stricter threshold, whereas a component used for monitoring, prioritization, or nonblocking review may tolerate a lower threshold.

⁴ Latency values are derived from the original task-specific inference logs. Cost estimates combine recorded prompt and completion token counts with the provider pricing table used in the experimental notebooks.

Substitution by force. Although the proposed methodology supports informed decisions about model replacement, we acknowledge that replacement is not always an open choice. Teams may be forced to migrate because of provider updates, model deprecations, procurement constraints, or infrastructure requirements. In these cases, the methodology can be used to characterize the behavioral risk of an imposed substitution before deployment. By assessing Cohen’s κ and McNemar’s test, teams can estimate whether (and how much) the new model maintains the existing decision behavior or introduces shifts in correctness outcomes.

8 Threats to Validity

Construct validity. Our methodology treats substitutability as a binary decision-level property. This scope does not capture all possible differences between models, such as differences in explanations, confidence, latency, token consumption, or cost, among others. We mitigate this threat by keeping the operationalization fixed across models: each task uses the same input units, prompts, label mappings, and output normalization. Other dimensions, such as latency, token consumption, and inference cost are treated as complementary deployment variables after behavioral substitutability has been identified.

Internal validity. As in any benchmark-based study, public dataset exposure is most relevant to absolute performance estimates. Our analysis instead emphasizes pairwise behavioral relations under shared benchmark conditions. To reduce variation unrelated to model behavior, all models are evaluated on the same instances with the same task prompts, deterministic inference settings, fixed output spaces, and the same normalization procedure. Excluded outputs were rare and reflected isolated API or inference issues rather than systematic refusal behavior. Because commercial API models can change over time, the results characterize the evaluated model variants under fixed inference conditions.

Conclusion validity. Substitutability decisions depend on the chosen agreement threshold and significance level. We address this threat by defining τ and α before the empirical analysis and using the Holm correction for the pairwise McNemar tests. Furthermore, we conducted a sensitivity analysis for τ and α to assess its impact on the decision.

External validity. The study intentionally focuses on the dataset-task configurations and on three commercial providers described in Section 5, and therefore our results should not be directly generalized to other models without proper validation. The reported values characterize this controlled evaluation setting; broader validation across tasks, prompts, label spaces, self-hosted models, and deployment settings remains future work. As our evaluation is offline and benchmark-based, the reported substitutability decisions should be re-assessed when deployment data or model behavior changes over time.

9 Conclusion

This paper has presented a methodology for assessing the operational substitutability of LLM-based decision components in software systems. The key premise is that replacing such a component is not equivalent to swapping a model endpoint. Instead, the replacement should preserve the decisions that the surrounding pipeline relies on. Our methodology checks this requirement through two complementary conditions: sufficient instance-level behavioral agreement and no statistically significant asymmetry in correctness outcomes. We

have evaluated our proposal by analyzing three semantic properties (i.e., safety, truthfulness, and stereotype detection) across multiple models from three providers. The results revealed that substitutability is selective, task-dependent, and model-pair specific. Furthermore, a sensitivity analysis demonstrated that stricter parameter configurations contract the admissible substitution space but preserve the main task ordering.

We believe these findings have a practical implication for LLM-based software engineering. Model replacement should be treated as a validation problem tied to a concrete task, prompt, dataset, and operating configuration. Although cost, latency, and availability can make a replacement attractive, they should be considered only after behavioral substitutability has been assessed. In this sense, substitutability analysis provides a way to reason about model upgrades, provider changes, and lightweight replacements without assuming that similar aggregate performance implies preserved system behavior. It also supports risk assessment when replacement choices are imposed by criteria unrelated to the model’s responses, such as the need to reduce inference costs.

As future work, we plan to study substitutability in live software pipelines, where decision components interact with retrieval modules, user interfaces, and human review processes. We will also explore how to extend the methodology beyond binary decisions (e.g., multi-class, ordinal, ranking-based, or free-form outputs), where agreement and correctness asymmetry may require richer definitions. This exploration will include the evaluation of self-hosted and open-weight models, as well as alternative prompt configurations of the same model, treating each model–prompt configuration as a distinct decision component. Additionally, broader sensitivity analyses will examine how thresholds such as τ and α can be calibrated to deployment risk, including settings where false positives and false negatives have asymmetric costs. Finally, we plan to combine behavioral substitutability with operational objectives (e.g., latency, token consumption, cost, or provider availability), enabling replacement decisions that are both behaviorally justified and practically useful. This should also clarify when instance-level substitutability provides practical value beyond aggregate performance comparison alone, and when comparable average performance is sufficient for stakeholders.

10 Data Availability Statement

The replication package⁵ contains the task instances used in the paper, the exact prompts used for inference, the normalized classified outputs for all evaluated models, and the scripts required to rerun inference and reproduce the results for RQ1, RQ2, and RQ3. The package follows the same task organization and naming conventions used in the paper.

References

- 1 Elias Bassani and Ignacio Sanchez. Guardbench: a Large-scale Benchmark for Guardrail Models. In *Conf. on Empirical Methods in Natural Language Processing*, pages 18393–18409, 2024. doi:10.18653/V1/2024.EMNLP-MAIN.1022.
- 2 Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. A Survey on Evaluation of Large Language Models. *ACM Trans. Intell. Syst. Technol.*, 15(3):1–45, 2024. doi:10.1145/3641289.

⁵ <https://zenodo.org/records/20131746>

- 3 Lingjiao Chen, Zhihua Jin, Sabri Eyuboglu, Christopher Ré, Matei Zaharia, and James Zou. HAPI: A Large-scale Longitudinal Dataset of Commercial ML API Predictions. In *Int. Conf. on Neural Information Processing Systems*, pages 24571–24585, 2022.
- 4 Lingjiao Chen, Matei Zaharia, and James Zou. Frugalgpt: how to Use Large Language Models while Reducing Cost and Improving Performance. *arXiv preprint*, 2023. doi:10.48550/arXiv.2305.05176.
- 5 Lingjiao Chen, Matei Zaharia, and James Zou. How Is ChatGPT’s Behavior Changing Over Time? *Harv. Data Sci. Rev.*, 6(2), 2024.
- 6 Databricks. Databricks Announces Significant Improvements to Built-in LLM Judges for Agent Evaluation, 2024. Accessed on 17/Apr/26. URL: <https://www.databricks.com/blog/databricks-announces-significant-improvements-built-llm-judges-agent-evaluation>.
- 7 Tanay Dixit, Daniel Lee, Sally Fang, Sai Sree Harsha, Anirudh Sureshan, Akash V. Maharaj, and Yunyao Li. RETAIN: Interactive Tool for Regression Testing Guided LLM Migration. In *Conf. on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 301–310, 2024. doi:10.18653/V1/2024.EMNLP-DEMO.31.
- 8 Tatiana Ermakova, Benjamin Fabian, Elena Golimblevskaia, and Max Henke. A Comparison of Commercial Sentiment Analysis Services. *SN Comput. Sci.*, 4(5):477, 2023. doi:10.1007/S42979-023-01886-Y.
- 9 Shaona Ghosh, Prasoon Varshney, Makesh Narsimhan Sreedhar, Aishwarya Padmakumar, Traian Rebedea, Jibin Rajan Varghese, and Christopher Parisien. Aegis2.0: a Diverse Ai Safety Dataset and Risks Taxonomy for Alignment of Llm Guardrails. In *Annual Conf. of the Nations of the Americas Chapter of the ACL: Human Language Technologies*, pages 5992–6026, 2025. doi:10.18653/V1/2025.NAACL-LONG.306.
- 10 Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. A Survey on LLM-as-a-judge. *arXiv preprint*, 2024. arXiv:2411.15594.
- 11 Guardrails AI. Guardrails AI: Adding Guardrails to Large Language Models, 2023. Accessed on 23/Mar/26. URL: <https://github.com/guardrails-ai/guardrails>.
- 12 Steve Han, Gilberto Titericz Junior, Tom Balough, and Wenfei Zhou. Judge’s Verdict: a Comprehensive Analysis of Llm Judge Capability through Human Agreement. *arXiv preprint*, 2025. doi:10.48550/arXiv.2510.09738.
- 13 Junda He, Jieke Shi, Terry Yue Zhuo, Christoph Treude, Jiamou Sun, Zhenchang Xing, Xiaoning Du, and David Lo. Llm-as-a-judge for Software Engineering: Literature Review, Vision, and the Road Ahead. *arXiv preprint*, 2025. doi:10.48550/arXiv.2510.24367.
- 14 Xinyi Hou, Yanjie Zhao, Yue Liu, Zhou Yang, Kailong Wang, Li Li, Xiapu Luo, David Lo, John Grundy, and Haoyu Wang. Large Language Models for Software Engineering: a Systematic Literature Review. *ACM Trans. Softw. Eng. Methodol.*, 33(8):1–79, 2024. doi:10.1145/3695988.
- 15 Hakan Inan, K. Upasani, Jianfeng Chi, Rashmi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. Llama Guard: Llm-based Input-output Safeguard for Human-ai Conversations. *arXiv preprint*, 2023. doi:10.48550/arXiv.2312.06674.
- 16 Raad Khraishi, Iman Zafar, Katie Myles, and Greig A. Cowan. Evaluating Performance Drift from Model Switching in Multi-Turn LLM Systems. *ICLR 2026 Workshop: Catch, Adapt, and Operate: Monitoring ML Models Under Drift*, 2026. doi:10.48550/arXiv.2603.03111.
- 17 Seungone Kim et al. Prometheus: Inducing Fine-grained Evaluation Capability in Language Models. *arXiv preprint*, 2023. arXiv:2310.08491.
- 18 Dominik Kreuzberger, Niklas Kühl, and Sebastian Hirschl. Machine Learning Operations (mlops): Overview, Definition, and Architecture. *IEEE Access*, 11:31866–31879, 2023. doi:10.1109/ACCESS.2023.3262138.

- 19 J. Richard Landis and Gary G. Koch. The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159–174, 1977.
- 20 LangChain. Langchain Documentation, 2026. Accessed on 23/Mar/26.
- 21 Chungpa Lee, Thomas Zeng, Jongwon Jeong, Jy yong Sohn, and Kangwook Lee. How to Correctly Report Llm-as-a-judge Evaluations. *arXiv preprint*, 2025. doi:10.48550/arXiv.2511.21140.
- 22 Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented Generation for Knowledge-intensive Nlp Tasks. In *Int. Conf. on Neural Information Processing Systems*, pages 9459–9474, 2020.
- 23 Xiao Li, Joel Kreuzwieser, and Alan Peters. When Meaning Stays the Same, but Models Drift: Evaluating Quality of Service under Token-level Behavioral Instability in Llms. *arXiv preprint*, 2025. doi:10.48550/arXiv.2506.10095.
- 24 Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how Models Mimic Human Falsehoods. In *Annual Meeting of the Association for Computational Linguistics*, pages 3214–3252, 2022. doi:10.18653/V1/2022.ACL-LONG.229.
- 25 MLflow. MLflow for Llm Evaluation, 2026. Accessed on 23/Mar/26.
- 26 Moin Nadeem, Anna Bethke, and Siva Reddy. Stereoset: Measuring Stereotypical Bias in Pretrained Language Models. In *Annual Meeting of the Association for Computational Linguistics*, pages 5356–5371, 2021. doi:10.18653/V1/2021.ACL-LONG.416.
- 27 OpenAI. Openai Moderation API Documentation, 2023. Accessed on 23/Mar/26.
- 28 Arjun Panickssery, Samuel R. Bowman, and Shi Feng. LLM Evaluators Recognize and Favor their own Generations. *arXiv preprint*, 2024. doi:10.48550/arXiv.2404.13076.
- 29 Mohammad Javad Saeedizade, Reham Alharbi, Hamed Babaei Giglou, Anna Sofia Lippolis, Eva Blomqvist, Valentina Tamma, Floriana Grasso, Terry Payne, Jennifer D’Souza, Sören Auer, Andrea Giovanni Nuzzolese, Robin Keskisärkkä, and Zebah Valeyil. A Framework for Assessing LLM Consistency in Knowledge Engineering. *Semantic Web – Interoperability, Usability, Applicability*, 2025.
- 30 Kayla Schroeder and Zach Wood-Doughty. Can you Trust Llm Judgments? Reliability of Llm-as-a-judge. *arXiv preprint*, 2024. doi:10.48550/arXiv.2412.12509.
- 31 Shreya Shankar, J. D. Zamfirescu-Pereira, Bjorn Hartmann, Aditya G. Parameswaran, and Ian Arawjo. Who Validates the Validators? Aligning Llm-assisted Evaluation of Llm Outputs with Human Preferences. In *ACM Symp. on User Interface Software and Technology*, pages 131:1–131:14, 2024. doi:10.1145/3654777.3676450.
- 32 Lin Shi, Chiyu Ma, Wenhua Liang, Xingjian Diao, Weicheng Ma, and Soroush Vosoughi. Judging the Judges: a Systematic Study of Position Bias in Llm-as-a-judge. *arXiv preprint*, 2024. arXiv:2406.07791.
- 33 Aman Singh Thakur, Kartik Choudhary, Venkat Srinik Ramayapally, Sankaran Vaidyanathan, and Dieuwke Hupkes. Judging the Judges: Evaluating Alignment and Vulnerabilities in Llms-as-judges. *arXiv preprint*, 2024. doi:10.48550/arXiv.2406.12624.
- 34 Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. Large Language Models are not Fair Evaluators. In *Annual Meeting of the Association for Computational Linguistics*, pages 9440–9450, 2024. doi:10.18653/V1/2024.ACL-LONG.511.
- 35 Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Chi Wang, et al. Autogen: Enabling Next-gen LLM Applications Via Multi-agent Conversation. *arXiv preprint*, 2023. arXiv:2308.08155.
- 36 Boming Xia, Qinghua Lu, Liming Zhu, Zhenchang Xing, Dehai Zhao, and Hao Zhang. Evaluation-driven Development and Operations of Llm Agents: a Process Model and Reference Architecture. *arXiv preprint*, 2024. arXiv:2411.13768.

- 37 Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, Nitesh V Chawla, and Xiangliang Zhang. Justice or Prejudice? Quantifying Biases in Llm-as-a-judge. In *Int. Conf. on Learning Representations*, 2025.
- 38 Xiaoyu Zhang, Juan Zhai, Shiqing Ma, Qingshuang Bao, Weipeng Jiang, Qian Wang, Chao Shen, and Yang Liu. The Invisible Hand: Unveiling Provider Bias in Large Language Models for Code Generation. In *Annual Meeting of the Association for Computational Linguistics*, pages 21376–21403, 2025. doi:10.18653/V1/2025.ACL-LONG.1038.
- 39 Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging Llm-as-a-judge with Mt-bench and Chatbot Arena. In *Adv. Neural Inf. Process. Syst.*, volume 36, pages 46595–46623, 2023.