Probing Instruction Execution Stability of Large Language Models under Semantic Paraphrase

Bodhisatta Maiti[1,*,†], Debshree Chowdhury[2,†]

[1]Independent Researcher, Orlando, United States
[2]Independent Researcher, Kolkata, India

Abstract

Large Language Models (LLMs) are commonly controlled through natural-language instructions, yet equivalent prompts can yield inconsistent behavior. While prior work has largely evaluated prompt effectiveness through task accuracy, less attention has been paid to the stability of instruction execution under paraphrase. In this work, we analyze how instruction-preserving paraphrases—prompts that retain identical task semantics and constraints—affect the reliability with which LLMs execute structured instructions. We conduct a controlled study using a classification task with a strict JSON output contract, evaluating multiple paraphrases across repeated runs to separate paraphrase-induced effects from stochastic variation. Our analysis distinguishes between semantic agreement (label consistency) and execution agreement (format and constraint adherence). Experiments on eight open-source instruction-tuned models show that semantic decisions generally remain stable under paraphrasing, while structured output reliability varies substantially depending on both the paraphrase and the model. These findings indicate that execution robustness is a distinct, model-dependent property not captured by accuracy alone and highlight the importance of evaluating prompt behavior under paraphrase in structured-output settings.

Keywords

Prompt Engineering, Instruction Execution, Semantic Paraphrase, Prompt Robustness, Large Language Models

1. Introduction

Large Language Models (LLMs) are increasingly deployed through natural-language instructions, commonly referred to as prompts. This interaction paradigm has enabled LLMs to be used across a wide range of tasks, including classification, extraction, summarization, and structured prediction. As a result, prompt engineering has emerged as a practical discipline aimed at designing instructions that elicit reliable and well-formed outputs from pre-trained language models [1, 2].

A common assumption underlying prompt engineering practices is that instruction-preserving paraphrases—that is, prompts that convey the same task, constraints, and requirements using different surface wording—should lead to equivalent model behavior. In practice, however, practitioners frequently report the need to rely on fail–retry strategies, prompt tweaking, or ad-hoc prompt stabilization techniques, suggesting that LLM behavior may be sensitive to subtle changes in instruction phrasing [3, 4]. Despite this widespread anecdotal evidence, there remains limited empirical analysis of how such paraphrases affect instruction execution, as opposed to task correctness alone.

Prior work in prompt engineering has largely focused on improving task performance through prompt design, prompt optimization, or the use of demonstrations and chain-of-thought prompting [5, 6]. Other studies have examined prompt sensitivity indirectly by comparing different prompt formulations for specific downstream tasks, such as classification or verification [7]. However, these efforts typically evaluate success in terms of prediction accuracy or overall output quality, without explicitly disentangling semantic task understanding from adherence to instruction-level constraints such as output format, schema compliance, or structural validity.

[PromptEng] Third International Workshop on Prompt Engineering for Pre-Trained Language Models co-located with the ACM WebConf, April.13, 2026, Dubai, United Arab Emirates

*Corresponding author.
†These authors contributed equally.
bodhisatta.iitbhu@gmail.com (B. Maiti); debshreechowdhury@gmail.com (D. Chowdhury)
0009-0006-4682-2896 (B. Maiti); 0009-0006-9446-6443 (D. Chowdhury)

Recent instruction-tuned language models are explicitly optimized to follow natural-language instructions, yet they differ in architecture, training data, alignment strategies, and objective design. While many such models achieve strong benchmark performance, it remains unclear whether general instruction-following capability translates into robustness under instruction-preserving paraphrasing, particularly for tasks that require strict structured outputs. Understanding this distinction is critical for prompt engineering, agent pipelines, and tool-augmented systems, where execution failures can propagate downstream even when the model correctly interprets the task.

In this work, we present an empirical study of instruction execution stability under instruction-preserving paraphrase. Using a controlled classification task based on the AG News dataset [8], we construct a base prompt and three instruction-preserving paraphrases that specify identical task semantics and output constraints. We evaluate eight open-source instruction-tuned LLMs across multiple runs per prompt variant, enabling us to separate paraphrase-induced effects from stochastic variation.

Crucially, we distinguish between semantic agreement (e.g., label consistency) and execution agreement (e.g., compliance with a structured JSON output contract). This separation allows us to examine whether paraphrasing affects task-level decisions differently from structured instruction execution. Our analysis shows that models can maintain stable semantic predictions under paraphrasing while exhibiting variability in output structure, and that paraphrase sensitivity is a model-dependent phenomenon.

The contributions of this paper are threefold:

- We introduce an empirical framework for analyzing instruction execution stability under instruction-preserving paraphrases.
- We demonstrate that semantic task understanding and instruction execution robustness can diverge significantly under paraphrasing.
- We show that paraphrase sensitivity varies markedly across instruction-tuned models, highlighting instruction execution as a distinct and under-examined dimension of LLM behavior.

By explicitly probing instruction execution rather than correctness alone, this work aims to provide actionable insights for prompt engineering practice and contribute to a more precise understanding of how LLMs respond to paraphrased instructions.

2. Task and Prompts

2.1. Task Definition

We study instruction execution stability using a controlled text classification task derived from the AG News corpus. Each input consists of a short news excerpt, typically one to two sentences in length. The task is to assign exactly one category label from a fixed set of four classes: World, Sports, Business, and Sci/Tech. The dataset is balanced across classes to avoid label skew effects.

The task is intentionally simple in terms of semantic difficulty. This choice allows us to isolate the effects of prompt phrasing on instruction execution rather than confounding them with complex reasoning or domain knowledge requirements. As shown later, models are generally capable of identifying the correct category, even when instruction execution fails.

2.2. Structured Output Contract

In addition to label prediction, the task requires the model to produce a structured output following a strict JSON schema. Each response must contain exactly three fields:

- `label`: the predicted category label
- `rationale_span`: a short verbatim span copied from the input text that supports the chosen label
- `trigger_terms`: a list of three short verbatim phrases from the input text indicating category relevance

The output contract enforces several constraints, including exact field names, fixed field cardinality, substring copying requirements, and word-length limits. No additional keys or explanatory text are permitted outside the JSON object.

This structured format is representative of real-world prompt engineering scenarios, such as tool invocation, information extraction, and agent pipelines, where downstream components depend on well-formed outputs. Importantly, violations of this contract constitute instruction execution failures, even when the predicted label itself is semantically correct.

2.3. Instruction-Preserving Paraphrases

To examine the effect of prompt phrasing, we construct one base prompt and three instruction-preserving paraphrases (IPPs). An instruction-preserving paraphrase is defined as a reformulation of the prompt that maintains:

- the same task objective,
- the same output schema and constraints, and
- the same semantic requirements,

while differing only in surface wording, ordering, or phrasing.

All prompt variants specify the identical classification task and structured output contract. Differences between the base prompt and IPPs are limited to linguistic reformulation, such as changes in imperative phrasing, section ordering, or explanatory wording. No additional examples, demonstrations, or auxiliary hints are introduced in any variant.

This setup allows us to test the commonly held assumption that paraphrases conveying the same instructions should yield equivalent model behavior. By holding task semantics and constraints constant, any observed differences can be attributed to the model's sensitivity to prompt phrasing rather than to changes in task definition.

2.4. Prompt Variants Used in Experiments

Across all experiments, we evaluate four prompt variants: a base prompt and three instruction-preserving paraphrases (IPPs). The IPPs are designed to be linguistic rewordings of the base prompt while preserving the same task definition, output schema, and constraint set. In particular, the variants retain the same required JSON keys, the same label set, and the same substring and length constraints for rationale_span and trigger_terms. Differences between variants are limited to surface-level phrasing (e.g., synonyms, sentence structure, and minor stylistic reformulations), without adding examples, demonstrations, or additional instructions.

We refer to these variants as:

- `Base`: the reference prompt used to define the task and the structured output contract.
- `IPP-1 / IPP-2 / IPP-3`: three instruction-preserving paraphrases of the base prompt, differing only in natural-language wording while maintaining identical requirements.

The full text of all prompt variants is fixed across models and runs. No prompt adaptation or post-hoc modification is performed during experimentation.

2.5. Why This Setup Matters

By combining a low-ambiguity task with a strict structured output contract and multiple instruction-preserving paraphrases, this setup enables a clear separation between semantic task understanding and instruction execution. As shown in later sections, models may retain high semantic agreement under paraphrasing while exhibiting substantial differences in their ability to consistently execute the specified instructions.

This distinction is central to our analysis and motivates the evaluation metrics introduced in the following section.

The full prompt templates used in all experiments, including the base prompt and instruction-preserving paraphrases, are provided in Appendix A.

3. Experimental Setup

3.1. Models

We evaluate eight open-source instruction-tuned large language models obtained from Hugging Face:

- `meta-llama/Meta-Llama-3-8B-Instruct` [9],
- `mistralai/Mistral-7B-Instruct-v0.3` [10],
- `Qwen/Qwen2.5-7B-Instruct` [11, 12],
- `tiiuae/Falcon3-7B-Instruct` [13],
- `NousResearch/Hermes-3-Llama-3.1-8B` [14],
- `allenai/Olmo-3-7B-Instruct` [15],
- `ibm-granite/granite-3.3-8b-instruct` [16], and
- `LiquidAI/LFM2-8B-A1B` [17]

These models are all designed to follow natural-language instructions, but differ in architecture, training data, and alignment strategies.

Evaluating multiple models of comparable parameter scale allows us to examine whether instruction execution stability under paraphrase is a consistent property across systems or varies substantially between models.

All experiments use identical prompt variants and decoding parameters for every model. No model-specific prompt adaptation, fine-tuning, or post-processing is applied.

3.2. Dataset and Sampling

Experiments are conducted on a subset of the AG News dataset. From the original corpus, we sample 4,000 news articles, balanced evenly across the four categories (1,000 samples per class). Each article consists of a short news excerpt, typically one to two sentences in length.

The balanced sampling strategy ensures that observed differences in model behavior are not driven by class imbalance.

3.3. Runs and Decoding Configuration

For each prompt variant and each model, we perform three independent runs using identical decoding parameters. Multiple runs are used to control for stochastic variation in generation and to distinguish paraphrase-induced effects from sampling noise.

All generations use identical decoding parameters across models and prompt variants, with temperature set to 0.2, nucleus sampling threshold (top-p) set to 0.9, and a maximum generation length of 150 tokens. These parameters are held constant across all models, prompts, and runs.

3.4. Execution and Output Collection

For each model–prompt–run combination, responses are generated independently for all sampled inputs. Model outputs are stored verbatim without any filtering, correction, or repair. In particular, no attempt is made to enforce JSON validity or to post-process malformed outputs prior to analysis.

This design choice is intentional: the goal of the study is to measure instruction execution behavior as produced by the model, rather than the behavior after applying external stabilization or correction mechanisms.

3.5. Control for Stochastic Variation

To contextualize the effects of instruction-preserving paraphrases, we include a control analysis based on repeated executions of the base prompt. By measuring consistency across the three base-prompt runs, we establish a reference level of variability attributable to stochastic decoding alone. Paraphrase-induced changes are interpreted relative to this baseline.

This setup provides a controlled environment for isolating the impact of instruction-preserving paraphrases on both semantic agreement and instruction execution. The next section introduces the metrics used to quantify these effects.

4. Metrics

The goal of this study is to quantify how instruction-preserving paraphrases affect both semantic task behavior and instruction execution. To this end, we define a set of complementary metrics that capture different aspects of model responses, ranging from basic format compliance to semantic agreement and execution stability. All metrics are computed directly from raw model outputs without post-processing or correction.

4.1. Format Compliance Metrics

We first measure whether model outputs conform to the expected JSON structure, independent of semantic correctness.

JSON-extractable rate measures the fraction of outputs from which a valid JSON object can be extracted. Outputs may contain additional text before or after the JSON object, as long as a well-formed JSON structure is present.

JSON-only rate measures the fraction of outputs consisting exclusively of a single JSON object, with no additional text. This metric captures strict adherence to the instruction requiring the response to contain only JSON.

These two metrics distinguish between partial format compliance and strict format obedience, which is particularly relevant for structured-output prompting.

4.2. Execution Validity

Beyond basic JSON structure, we define execution validity as compliance with the full output contract specified in the prompt. An output is considered execution-valid if it satisfies all required constraints, including:

- the presence of exactly three required fields,
- a valid category label from the predefined set,
- a `rationale_span` that is an exact substring of the input text and within the specified length bounds, and
- a `trigger_terms` list containing exactly three distinct substrings of valid length.

Execution validity therefore captures whether the model successfully executes the instructions as specified, rather than merely approximating the intended format.

4.3. Semantic Agreement

To evaluate semantic consistency across prompt variants, we measure label agreement between outputs produced by different prompts for the same input. Two outputs are considered semantically consistent if they assign the same category label.

Because some outputs contain malformed or incomplete JSON while still clearly expressing a category label, we compute label agreement using a relaxed extraction procedure. Specifically, a label is considered

present if it can be reliably identified in the output text and matches one of the allowed category values. This approach ensures that semantic agreement is not conflated with formatting errors.

4.4. Paraphrase Agreement Metrics

To assess sensitivity to instruction-preserving paraphrases, we compare outputs produced by the base prompt with those produced by each paraphrase for the same input and run. We report three agreement measures:

- Both-JSON rate, the fraction of input instances for which both the base and paraphrased outputs are JSON-extractable.
- Label agreement rate, the fraction of instances for which the base and paraphrased outputs assign the same label under relaxed extraction.
- Execution agreement rate, the fraction of instances for which both outputs are execution-valid and exhibit identical execution validity.

These metrics collectively characterize how paraphrasing affects output structure, semantic decisions, and instruction execution.

4.5. Control Metrics

To control for stochastic variation introduced by sampling-based decoding, we compute stability metrics across repeated runs of the base prompt. These include the fraction of inputs for which JSON extractability, label assignment, and execution validity remain consistent across all runs. These control measurements provide a reference against which paraphrase-induced variation can be interpreted.

Together, these metrics enable a fine-grained analysis of instruction execution stability under paraphrase, disentangling semantic agreement from structural and contractual compliance.

5. Results

5.1. Per-Prompt Format and Execution Behavior

Table 1 reports per-prompt format compliance and execution validity across all eight models. Clear differences emerge both across architectures and across paraphrase variants.

A first observation is that format robustness varies substantially between models. Mistral-7B-Instruct and Hermes-3-Llama-3.1-8B exhibit consistently high `has_json_rate` across all prompt variants, with only minor fluctuations in `execution_valid_rate`. Falcon3-7B-Instruct, Granite-3.3-8B-Instruct, and LFM2-8B-A1B also maintain high JSON extractability for most prompts, although their execution validity shows greater variability. In contrast, LLaMA-3-8B-Instruct, Qwen2.5-7B-Instruct, and Olmo-3-7B-Instruct display pronounced sensitivity to specific paraphrases, with sharp drops in both format compliance and execution validity under certain instruction-preserving reformulations.

For LLaMA-3-8B-Instruct, the Base and IPP-3 prompts remain largely JSON-extractable, while IPP-1 and IPP-2 lead to substantial degradation in both `has_json_rate` and `execution_valid_rate`. Qwen2.5-7B-Instruct exhibits a similarly uneven pattern: IPP-2 preserves high compliance and validity, but IPP-1 and IPP-3 trigger severe format breakdowns despite the task semantics remaining unchanged. Olmo-3-7B-Instruct shows the most unstable behavior overall, with compliance varying dramatically across paraphrases and execution validity dropping to near-zero in some cases.

Importantly, high JSON extractability does not uniformly translate into high execution validity. Several models, including Falcon3 and Granite, maintain near-perfect `has_json_rate` while exhibiting moderate execution validity, suggesting that producing syntactically valid JSON is easier than consistently satisfying all output constraints. Conversely, the most severe failures are concentrated in particular prompt–model combinations rather than uniformly across prompts, indicating that instruction-preserving paraphrasing can selectively disrupt execution reliability.

Table 1
Per-prompt format compliance and execution validity on 4,000 AG News samples. Rates are averaged across three runs.

Model	Prompt	has_json_rate	json_only_rate	execution_valid_rate
LLaMA-3-8B-Instruct	Base	0.947	0.140	0.510
	IPP-1	0.148	0.024	0.088
	IPP-2	0.278	0.058	0.152
	IPP-3	0.960	0.132	0.447
Mistral-7B-Instruct	Base	0.971	0.152	0.432
	IPP-1	0.960	0.151	0.435
	IPP-2	0.975	0.154	0.424
	IPP-3	0.968	0.152	0.408
Qwen2.5-7B-Instruct	Base	0.986	0.136	0.503
	IPP-1	0.312	0.041	0.007
	IPP-2	0.979	0.135	0.580
	IPP-3	0.309	0.000	0.081
Falcon3-7B-Instruct	Base	0.968	0.000	0.348
	IPP-1	0.951	0.000	0.468
	IPP-2	0.981	0.000	0.414
	IPP-3	0.977	0.000	0.434
Hermes-3-Llama-3.1-8B	Base	0.984	0.138	0.405
	IPP-1	0.988	0.139	0.402
	IPP-2	0.992	0.140	0.449
	IPP-3	0.985	0.139	0.465
Olmo-3-7B-Instruct	Base	0.382	0.048	0.117
	IPP-1	0.289	0.033	0.078
	IPP-2	0.886	0.126	0.159
	IPP-3	0.141	0.013	0.008
Granite-3.3-8b-Instruct	Base	0.973	0.129	0.268
	IPP-1	0.981	0.130	0.346
	IPP-2	0.979	0.130	0.232
	IPP-3	0.847	0.111	0.120
LFM2-8B-A1B	Base	0.974	0.131	0.260
	IPP-1	0.980	0.134	0.121
	IPP-2	0.980	0.132	0.184
	IPP-3	0.961	0.130	0.090

Overall, the results demonstrate that execution robustness under semantic paraphrasing is highly model-dependent. While some models remain stable across all prompt variants, others exhibit brittle behavior that is not explained by task difficulty alone. These findings reinforce the distinction between semantic competence and reliable instruction execution: maintaining correct task understanding does not guarantee adherence to a structured output contract.

5.2. Paraphrase Sensitivity: Semantic Agreement vs. Execution Agreement

Separating semantic agreement from execution agreement reveals a consistent structural pattern across models. In most cases, `label_agree_relaxed_rate` remains high when comparing the Base prompt with its instruction-preserving paraphrases, indicating that category decisions are largely stable under surface-level rewording. However, this semantic stability does not uniformly extend to structured output execution.

For several models, including Mistral-7B-Instruct, Hermes-3-Llama-3.1-8B, Falcon3-7B-Instruct,

Granite-3.3-8B-Instruct, and LFM2-8B-A1B, both `both_json_rate` and `exec_agree_rate` remain consistently high across paraphrase variants. These models exhibit not only stable label predictions but also relatively consistent adherence to the required output format, suggesting stronger robustness to instruction-preserving paraphrasing.

In contrast, LLaMA-3-8B-Instruct and Qwen2.5-7B-Instruct display a more uneven pattern. While their relaxed label agreement remains high across paraphrases—often exceeding 0.9—execution agreement varies sharply depending on the prompt formulation. In these cases, certain paraphrases (notably IPP-1 and IPP-3 for Qwen, and IPP-1 and IPP-2 for LLaMA) lead to substantial reductions in `both_json_rate` and `exec_agree_rate`, despite stable semantic decisions. This divergence indicates that paraphrasing primarily disrupts structured output behavior rather than task understanding.

Olmo-3-7B-Instruct presents a distinct profile. Both semantic and execution agreement are comparatively low and fluctuate considerably across paraphrases, suggesting broader instability that extends beyond format production alone. Here, paraphrase sensitivity appears to affect both task-level predictions and output structure.

Taken together, these results reinforce the distinction between semantic competence and instruction execution. High label agreement does not guarantee stable execution agreement, and paraphrase-induced failures are concentrated in particular model–prompt combinations rather than uniformly distributed. The findings suggest that structured output reliability under paraphrasing is a model-dependent capability that only partially overlaps with semantic task performance.

Table 2 provides a detailed breakdown of these base-to-paraphrase comparisons across all models. The table highlights the separation between semantic agreement and execution agreement, illustrating how stability in label predictions can coexist with substantial variability in structured output behavior.

5.3. Control: Base-Prompt Stability Across Runs

The control analysis establishes a baseline for variability attributable solely to stochastic decoding. By repeating the Base prompt three times per model, we measure how often format compliance, label assignment, and execution validity remain consistent across runs. These stability rates provide a reference against which paraphrase-induced effects can be interpreted.

For most models, repeated executions of the Base prompt yield high stability across all three measures. Mistral-7B-Instruct, Qwen2.5-7B-Instruct, Hermes-3-Llama-3.1-8B, Granite-3.3-8B-Instruct, Falcon3-7B-Instruct, and LFM2-8B-A1B all maintain strong consistency in JSON production and label assignment across runs, with execution stability typically remaining above 0.85. These results indicate that, under a fixed prompt, stochastic decoding alone introduces relatively limited variation for these models.

LLaMA-3-8B-Instruct exhibits slightly lower stability compared to the strongest models, but still demonstrates substantial consistency under repeated Base executions. In contrast, Olmo-3-7B-Instruct shows markedly lower stability in format compliance and label assignment across runs, even under the identical prompt formulation. Interestingly, its execution validity remains comparatively high across runs, suggesting that while JSON extractability and label outputs fluctuate, the underlying execution constraint is internally consistent when JSON is produced.

Importantly, for models such as LLaMA-3-8B-Instruct and Qwen2.5-7B-Instruct, the degradation observed under specific paraphrases in previous sections is substantially larger than the variability measured here under repeated Base runs. This contrast supports the interpretation that severe failures in format compliance or execution agreement are not merely artifacts of stochastic sampling, but are instead systematically triggered by particular instruction-preserving reformulations.

Overall, the control results confirm that most models are relatively stable under fixed prompts, and that the pronounced differences observed across paraphrases reflect genuine paraphrase sensitivity rather than random decoding noise.

Table 3 summarizes stability across repeated Base prompt executions for each model. These values serve as a reference point for stochastic variability, enabling a clearer distinction between random decoding fluctuations and systematic paraphrase-induced effects.

Table 2
Paraphrase sensitivity comparing Base vs each instruction-preserving paraphrase (IPP) on the same inputs, averaged across three runs. `label_agree_relaxed_rate` measures semantic agreement (label consistency) using relaxed label extraction, while `exec_agree_rate` captures agreement in execution validity.

Model	Pair	both_json_rate	label_agree_relaxed_rate	exec_agree_rate
LLaMA-3-8B-Instruct	Base vs IPP-1	0.146	0.827	0.109
	Base vs IPP-2	0.270	0.919	0.192
	Base vs IPP-3	0.916	0.939	0.644
Mistral-7B-Instruct	Base vs IPP-1	0.952	0.918	0.696
	Base vs IPP-2	0.963	0.928	0.696
	Base vs IPP-3	0.960	0.908	0.704
Qwen2.5-7B-Instruct	Base vs IPP-1	0.311	0.941	0.164
	Base vs IPP-2	0.975	0.954	0.687
	Base vs IPP-3	0.303	0.953	0.188
Falcon3-7B-Instruct	Base vs IPP-1	0.939	0.957	0.625
	Base vs IPP-2	0.963	0.951	0.615
	Base vs IPP-3	0.963	0.953	0.694
Hermes-3-Llama-3.1-8B	Base vs IPP-1	0.979	0.901	0.691
	Base vs IPP-2	0.980	0.960	0.692
	Base vs IPP-3	0.977	0.942	0.658
Olmo-3-7B-Instruct	Base vs IPP-1	0.180	0.204	0.116
	Base vs IPP-2	0.348	0.304	0.233
	Base vs IPP-3	0.040	0.354	0.028
Granite-3.3-8b-Instruct	Base vs IPP-1	0.968	0.952	0.680
	Base vs IPP-2	0.967	0.910	0.692
	Base vs IPP-3	0.836	0.954	0.602
LFM2-8B-A1B	Base vs IPP-1	0.959	0.937	0.695
	Base vs IPP-2	0.962	0.929	0.701
	Base vs IPP-3	0.945	0.936	0.689

Table 3
Control stability across three runs of the Base prompt. Rates indicate the fraction of inputs for which each property is consistent across all runs.

Model	base_has_json_allruns_rate	base_label_stable_allruns_rate	base_exec_stable_allruns_rate
LLaMA-3-8B-Instruct	0.898	0.878	0.805
Mistral-7B-Instruct	0.967	0.952	0.871
Qwen2.5-7B-Instruct	0.984	0.977	0.914
Falcon3-7B-Instruct	0.963	0.938	0.857
Hermes-3-Llama-3.1-8B	0.981	0.972	0.873
Olmo-3-7B-Instruct	0.347	0.340	0.945
Granite-3.3-8b-Instruct	0.965	0.958	0.871
LFM2-8B-A1B	0.966	0.947	0.873

5.4. Qualitative Illustration

To make the observed execution failures more concrete, we present a representative example from the evaluation set (gold label: *Sports*) using Qwen2.5-7B-Instruct. For the Base prompt, the model produces a well-formed JSON object across runs:

```
{"label": "Sports", "rationale_span": "Riley proves he belongs",
"trigger_terms": ["Ryder Cup", "America's worst days", "team competition"]}
```

Under IPP-1, the predicted label remains correct, but the output becomes structurally invalid:

```
{"label": "Sports", "rationale_span": "Riley proves he belongs",
"trigger_terms": "Ryder Cup", "US", "golf"}
```

Here, `trigger_terms` is emitted as a sequence of comma-separated strings rather than a valid list, violating the required JSON schema.

This example mirrors the quantitative results reported earlier: the semantic decision remains stable, while adherence to the output contract varies substantially with paraphrase formulation. The example is intended to illustrate the phenomenon rather than to attribute failures to specific linguistic mechanisms.

6. Discussion

The expanded evaluation across eight models strengthens the central distinction identified in this study: semantic task performance and instruction execution robustness are related but separable behaviors. Across most models, category predictions remain largely consistent under instruction-preserving paraphrases. However, structured output reliability varies substantially depending on both the model and the specific reformulation.

Several models—including Mistral-7B-Instruct, Hermes-3-Llama-3.1-8B, Falcon3-7B-Instruct, Granite-3.3-8B-Instruct, and LFM2-8B-A1B—demonstrate relatively stable behavior across prompt variants. For these models, JSON extractability and execution agreement remain consistently high, indicating that paraphrasing does not meaningfully disrupt either task interpretation or output structure. In contrast, LLaMA-3-8B-Instruct and Qwen2.5-7B-Instruct exhibit marked sensitivity to particular paraphrases. In these cases, relaxed label agreement remains high, yet execution validity and agreement decline sharply under specific formulations. This pattern suggests that the underlying classification decision remains intact while the structured output contract becomes unstable.

Olmo-3-7B-Instruct presents a different profile. Here, both semantic and execution agreement fluctuate more broadly across paraphrases, indicating that prompt reformulation affects not only formatting behavior but also label consistency. This behavior contrasts with models in which paraphrasing primarily disrupts format adherence while leaving semantic decisions largely unchanged.

The control analysis provides important context. For most models, repeated executions of the Base prompt produce high stability across runs, indicating that stochastic decoding alone introduces limited variability. The degradation observed under certain paraphrases—particularly for LLaMA-3-8B-Instruct and Qwen2.5-7B-Instruct—exceeds this baseline variation. This contrast supports the interpretation that the failures are systematically linked to prompt formulation rather than random sampling effects.

Taken together, the results indicate that execution robustness under paraphrasing is strongly model-dependent. High task accuracy does not guarantee consistent adherence to structured output constraints, and semantic agreement alone is not a sufficient proxy for reliable instruction execution. For applications that rely on strict output contracts, such as tool invocation or structured data extraction, evaluating prompt behavior under paraphrase is therefore essential.

More broadly, the findings caution against assuming that general instruction-following ability or benchmark performance necessarily implies stability in structured output settings. Robust execution under surface-level rewording appears to emerge unevenly across models, underscoring the need for targeted evaluation of output reliability in addition to semantic performance.

7. Limitations

This study evaluates paraphrase sensitivity on a single classification task with relatively short inputs. While this controlled setup enables a clear separation between semantic agreement and instruction execution, the findings may not directly extend to more complex settings such as long-form generation, multi-step reasoning, or open-ended extraction tasks.

Although eight open-source instruction-tuned models were examined, the selection is not exhaustive and does not include closed-source systems or substantially larger models. Different training objectives, scales, or alignment procedures may yield alternative robustness profiles.

The analysis is restricted to a fixed set of instruction-preserving paraphrases and does not consider adaptive prompting strategies, few-shot examples, or post-generation repair mechanisms. Finally, outputs are evaluated independently of downstream system integration or human judgment, and the study does not assess the practical impact of execution failures in real-world pipelines.

8. Conclusion

This study investigated how instruction-preserving paraphrases affect structured output behavior in instruction-tuned language models. Across eight models, semantic label decisions generally remained stable under paraphrasing, while structured output reliability varied considerably depending on both the model and the specific reformulation. In several cases, execution validity deteriorated sharply despite high semantic agreement, highlighting a separation between task understanding and instruction execution.

The results indicate that execution robustness under paraphrase is a model-dependent property that cannot be inferred from semantic performance alone. For applications requiring strict output contracts, evaluating prompt behavior across paraphrase variants is therefore essential. Future work may extend this analysis to additional tasks and prompting strategies to further characterize when instruction execution remains stable.

Declaration on Generative AI

During the preparation of this manuscript, the authors used ChatGPT to assist with language editing, including grammar correction, spelling checks, restructuring of sentences, and rephrasing to improve clarity and readability. The tool was not used for the design of experiments, data analysis, interpretation of results, or generation of scientific content. All experimental procedures, analyses, and conclusions were developed and verified by the authors. The authors reviewed and edited all text after using the tool and take full responsibility for the content of the publication.

References

[1] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. Christiano, J. Leike, R. Lowe, Training language models to follow instructions with human feedback, arXiv preprint, 2022. URL: https://arxiv.org/abs/2203.02155.

[2] J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, Q. V. Le, Finetuned language models are zero-shot learners, arXiv preprint, 2021. URL: https://arxiv.org/abs/2109.01652.

[3] Y. Lu, M. Bartolo, A. Moore, S. Riedel, P. Stenetorp, Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity, arXiv preprint, 2021. URL: https://arxiv.org/abs/2104.08786.

[4] T. Z. Zhao, E. Wallace, S. Feng, D. Klein, S. Singh, Calibrate before use: Improving few-shot performance of language models, arXiv preprint, 2021. URL: https://arxiv.org/abs/2102.09690.

[5] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, D. Amodei, Language models are few-shot learners, in: Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS '20, Curran Associates Inc., Red Hook, NY, USA, 2020.

[6] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, D. Zhou, Chain-of-thought prompting elicits reasoning in large language models, in: Proceedings of the 36th International Conference on Neural Information Processing Systems, NIPS '22, Curran Associates Inc., Red Hook, NY, USA, 2022.

[7] M. Chakraborty, A. Kulkarni, Q. Li, Empirical evaluation of prompting strategies for fact verification tasks, in: Companion Proceedings of the ACM on Web Conference 2025, WWW '25, Association for Computing Machinery, New York, NY, USA, 2025, p. 1598–1604. URL: https://doi.org/10.1145/3701716.3717815. doi:10.1145/3701716.3717815.

[8] X. Zhang, J. Zhao, Y. LeCun, Character-level convolutional networks for text classification, in: Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 1, NIPS'15, MIT Press, Cambridge, MA, USA, 2015, p. 649–657.

[9] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, et al., The llama 3 herd of models, arXiv preprint, 2024. URL: https://arxiv.org/abs/2407.21783.

[10] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, et al., Mistral 7b, arXiv preprint, 2023. URL: https://arxiv.org/abs/2310.06825.

[11] Q. Team, Qwen2.5: A party of foundation models, 2024. URL: https://qwenlm.github.io/blog/qwen2.5/.

[12] A. Yang, B. Yang, B. Hui, B. Zheng, B. Yu, C. Zhou, C. Li, C. Li, D. Liu, F. Huang, et al., Qwen2 technical report, arXiv preprint arXiv:2407.10671 (2024). URL: https://arxiv.org/abs/2407.10671.

[13] T. Team, The falcon 3 family of open models, 2024.

[14] R. Teknium, J. Quesnelle, C. Guang, Hermes 3 technical report, 2024. URL: https://arxiv.org/abs/2408.11857. arXiv:2408.11857.

[15] T. Olmo, A. Ettinger, A. Bertsch, B. Kuehl, D. Graham, D. Heineman, D. Groeneveld, F. Brahman, F. Timbers, H. Ivison, J. Morrison, et al., Olmo 3, 2025. URL: https://arxiv.org/abs/2512.13961. arXiv:2512.13961.

[16] Granite Team, IBM, Granite-3.3-8b-instruct, Hugging Face, 2025. URL: https://huggingface.co/ibm-granite/granite-3.3-8b-instruct.

[17] L. AI, Lfm2 technical report, arXiv preprint arXiv:2511.23404 (2025). URL: https://arxiv.org/abs/2511.23404.

A. Prompt Templates

A.1. Base Prompt

- **Task:**
  You are given a short news text. Classify the text into exactly one of the following categories: World, Sports, Business, Sci/Tech.
- **Output Format:**
  Return only a valid JSON object with exactly three keys:
  - `"label"`
  - `"rationale_span"`
  - `"trigger_terms"`
- **Field Definitions:**
  - `"label"`: the chosen category for the news text.
  - `"rationale_span"`: a short verbatim span from the input text that directly supports the chosen label.
  - `"trigger_terms"`: a list of verbatim words or phrases from the input text that indicate why the label applies.

- **Constraints:**
  - `"label"` must be one of: World, Sports, Business, Sci/Tech.
  - `"rationale_span"` must be an exact substring copied from the input text.
    - It must contain between 5 and 10 words.
    - It must not be the entire input text.
  - `"trigger_terms"` must be a list of exactly three items.
    - Each item must be an exact substring of the input text.
    - Each item must contain between 1 and 4 words.
    - All three items must be distinct.
- **Important:**
  - Do not add any extra keys.
  - Do not include explanations or text outside the JSON.
  - Output only the JSON object.
- **News Text:**
  {TEXT}

A.2. Instruction-Preserving Paraphrase 1 (IPP-1)

- **Instructions:**
  Read the news text below and assign one category from: World, Sports, Business, Sci/Tech.
- **Response Format:**
  Output only JSON, using exactly three fields named:
  - `"label"`
  - `"rationale_span"`
  - `"trigger_terms"`
- **Meaning of Fields:**
  - `"label"` is the category you assign to the text.
  - `"rationale_span"` is a directly copied span from the text that supports the label.
  - `"trigger_terms"` are copied words or phrases that appear verbatim as substrings in the text and signal the category.
- **Rules:**
  - `"label"` must be exactly one of the following values: World, Sports, Business, Sci/Tech.
  - `"rationale_span"` must be copied verbatim, be 5–10 words long, and must not be the full text.
  - `"trigger_terms"` must contain exactly three distinct substrings, each 1–4 words long.
- Do not include any text before or after the JSON. Do not add extra fields.
- **Input Text:**
  {TEXT}

A.3. Instruction-Preserving Paraphrase 2 (IPP-2)

- Given the following news text, perform a classification task.
- **Classification:**
  Select one label from: World, Sports, Business, Sci/Tech.
- **Required Output:**
  A single valid JSON object with exactly three keys:
  - `"label"`

  - `"rationale_span"`
  - `"trigger_terms"`
- **Key Descriptions:**
  - `"label"` indicates the selected category.
  - `"rationale_span"` is a verbatim excerpt from the input that justifies the category.
  - `"trigger_terms"` are verbatim phrases that appear as substrings in the input and point to the category.
- **Field Constraints:**
  - `"label"` must be exactly one of: World, Sports, Business, Sci/Tech.
  - `"rationale_span"` must be 5–10 words, copied exactly, and not equal to the full input.
  - `"trigger_terms"` must be a list of three distinct copied substrings, each 1–4 words long.
- No commentary or formatting is allowed outside the JSON object.
- **Text:**
  {TEXT}

A.4. Instruction-Preserving Paraphrase 3 (IPP-3)

- You will be given a short news article. Your task is to classify the article into exactly one of the following categories: World, Sports, Business, Sci/Tech.
- Produce an output that satisfies all of the following:
  - The output is only JSON.
  - The JSON contains exactly three keys: `"label"`, `"rationale_span"`, `"trigger_terms"`.
  - No additional keys or text are allowed.
- **Field Semantics:**
  - `"label"`: the category assigned to the article.
  - `"rationale_span"`: a copied span from the article that supports the category.
  - `"trigger_terms"`: copied words or phrases from the article that indicate the category.
- **Additional Requirements:**
  - `"label"` must contain a single category value, selected from: World, Sports, Business, Sci/Tech.
  - `"rationale_span"` must contain 5–10 words and must not reproduce the full article.
  - `"trigger_terms"` must contain three distinct substrings, each 1–4 words long.
- **Article:**
  {TEXT}