RCScore: Quantifying Response Consistency in Large Language Models

Dongjun Jang and **Youngchae Ahn** and **Hyopil Shin**
Department of Linguistics
Seoul National University
{qwer4107, estelle1026, hpshin}@snu.ac.kr

Abstract

Current LLM evaluations often rely on a single instruction template, overlooking models' sensitivity to instruction style—a critical aspect for real-world deployments. We present RCScore, a multi-dimensional framework quantifying how instruction formulation affects model responses. By systematically transforming benchmark problems into multiple instruction styles, RCScore reveals performance variations undetected by conventional metrics. Our experiments across ten LLMs on four reasoning benchmarks demonstrate that instruction style can shift accuracy by up to 16.7% points. We introduce Cross-Response Similarity (CRS), a method applying RCScore metrics to measure stylistic self-consistency, and establish its strong correlation with task accuracy, suggesting consistency as a valuable proxy for model reliability. Additional findings show that deterministic decoding produces more stylistically stable outputs, and model scale correlates positively with cross-style consistency. RCScore offers a principled approach to assess instruction robustness.

1 Introduction

Large language models (LLMs) exhibit remarkable proficiency in complex reasoning, multi-step problem-solving, and creative generation, significantly extending their utility beyond fundamental text generation (Brown et al., 2020; Touvron et al., 2023). While these models demonstrate advanced cognitive capabilities in areas such as mathematical reasoning, logical deduction, and code generation, current evaluation benchmarks inadequately capture the subtleties of these sophisticated tasks.

Current evaluation practices for LLMs predominantly rely on traditional metrics like accuracy and F1 scores. Although mathematically validated, these metrics fall short in assessing the quality, consistency, and multifaceted aspects of open-ended LLM responses, where diverse valid outputs are common. Despite impressive scores on reasoning-intensive benchmarks like MMLU (Hendrycks et al., 2020), GPQA (Rein et al., 2024), and GSM8K (Cobbe et al., 2021), this reliance creates a fundamental disconnect between the sophisticated outputs of LLMs and the simplistic nature of their evaluation.

To address these limitations, one strategy that has been explored is the use of LLMs as evaluators (Zheng et al., 2023; Xia et al., 2025). While offering flexibility and scalability, this approach faces challenges including domain knowledge constraints requiring costly fine-tuning and inherent model biases that potentially undermine evaluation objectivity. Critically, it introduces a circular evaluation paradigm, where LLMs assess other LLMs, compromising objectivity and reliability as identified by Wang et al. (2023).

A particularly significant yet underexplored evaluation gap concerns the impact of instruction formulation on model performance. Research has demonstrated that LLM performance varies considerably depending on prompt design and instruction style (Liu et al., 2023). Beyond methods such as Chain-of-Thought (Wei et al., 2022) and Tree-of-Thoughts (Yao et al., 2023), recent studies have highlighted the sensitivity of language models to instruction phrasing. Ajith et al. (2023) and Cao et al. (2024) analyze performance variation under prompt perturbations, while Mizrahi et al. (2024) show that evaluations based on a single template can yield unreliable metrics across paraphrased instructions. This sensitivity extends to subtle linguistic variations as shown by Wahle et al. (2024) and Leidinger et al. (2023). Following this, recent studies propose metrics for prompt sensitivity, including rephrasing-based evaluation (Lu et al., 2024; Errica et al., 2024) and embedding-based coherence scoring (Lauriola et al., 2025). However, these approaches rely on ad-hoc rewriting, incur high computational costs, and offer limited insight into

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 5690–5708
November 4-9, 2025

model behavior.

To address this evaluation gap, we introduce RCScore[1], a comprehensive metric that methodically assesses how LLMs respond to different instruction styles. Through structured variation of instructional cues across clause types, RCScore evaluates responses across three key dimensions—Structurality, Lexicality, and Coherence—providing deeper insights into model capabilities. Our metric delivers three important contributions: (1) it reveals performance variations across instruction styles that traditional accuracy-focused benchmarks often miss; (2) it introduces Cross-Response Similarity (CRS) to measure stylistic self-consistency, showing that higher consistency correlates with better task accuracy, suggesting consistency as a reliability indicator; and (3) it provides an easy-to-reproduce, efficient protocol for assessing model robustness.

2 Related Work

2.1 Traditional LLM Evaluation Methods

As LLMs have advanced rapidly, a variety of evaluation methodologies have emerged to assess their capabilities. However, most mainstream benchmarks—such as MMLU (Hendrycks et al., 2020), BIG-Bench (Srivastava et al., 2022), GSM8K (Cobbe et al., 2021), GPQA (Rein et al., 2024), HellaSwag (Zellers et al., 2019), MATH (Hendrycks et al., 2021), and DROP (Dua et al., 2019)—continue to rely on traditional metrics such as exact match accuracy and F1 score. This simplification is further reflected in the technical reports of many recent LLMs, which predominantly emphasize accuracy-based performance results (Achiam et al., 2023; Grattafiori et al., 2024; Team et al., 2023; Yang et al., 2024; Team et al., 2025; Anthropic, 2024; Guo et al., 2025). Mondorf and Plank (2024) argue that LLMs' reasoning abilities remain unclear due to an overreliance on shallow accuracy-based metrics. Other studies similarly critique evaluations based solely on final answers, and propose alternative methods (Mahdavi et al., 2025; Nguyen et al., 2024; Golovneva et al., 2023).

2.2 Beyond Accuracy-Centric Evaluation

To address the limitations of accuracy-centric evaluation, recent studies have explored alternative strategies that go beyond exact match or lexical overlap. One prominent direction is the *LLM-as-a-judge* paradigm, where strong models are used to assess the quality of generated responses directly (Zheng et al., 2023; Xia et al., 2025). Yet, studies show that such methods can diverge from human judgments and suffer from structural flaws (Wang et al., 2023, 2025), casting doubt on their reliability. In parallel, other studies assess different facets of LLM output quality. FactScore (Min et al., 2023) verifies generations by checking atomic factual units against external sources. SelfCheckGPT (Manakul et al., 2023) evaluates factuality through consistency across sampled outputs, and ARES (Saad-Falcon et al., 2023) scores the quality of individual components in retrieval-augmented generation pipelines.

2.3 Stylistic Variation in Prompting: Effects and Evaluation

Prompt formulation plays a critical role in shaping LLM behavior and performance, often enabling substantial gains without parameter updates (Liu et al., 2023). The field evolved from zero-shot (Radford et al., 2019) and few-shot approaches (Brown et al., 2020) to more sophisticated techniques like Chain-of-Thought (Wei et al., 2022), Tree of Thoughts (Yao et al., 2023), self-refine prompting (Madaan et al., 2023), and Automatic Prompt Engineer (Zhou et al., 2023), demonstrating prompt design's significant impact on LLM capabilities.

As prompting techniques developed, research shifted toward understanding how stylistic variations across prompts affect model performance. He et al. (2024) demonstrated that surface-level formatting changes (e.g., JSON, YAML) affect model outputs. On the linguistic side, Wahle et al. (2024) examined the impact of morphological, syntactic, and lexical paraphrases, while Leidinger et al. (2023) highlighted LLMs' sensitivity to subtle linguistic variations. Using prompt variants that are semantically equivalent, these studies demonstrated that models are highly sensitive to even subtle stylistic cues.

Motivated by these findings, several studies have sought to formalize the effect of prompt variability by designing dedicated evaluation metrics. Lu et al. (2024) proposed sensitivity-based metrics to assess how prompt formulation influences model performance. Errica et al. (2024) introduced sensitivity and consistency metrics, using paraphrased benchmark questions generated via LLaMA3. Lau-

[1]RCScore github link https://github.com/Junmaij/RCScore

riola et al. (2025) focused on coherence, measuring the average cosine similarity between answers to equivalent questions.

3 Experiment on Instruction Style Sensitivity

Large language models often exhibit varying performance depending on how instructions are phrased, yet traditional benchmarks rarely account for this variability. Prior work shows that subtle prompt variations affect how models respond (He et al., 2024; Wahle et al., 2024; Leidinger et al., 2023), and Mizrahi et al. (2024) further demonstrate that evaluations relying on a single instruction template are inadequate. Building on these findings, we conducted comprehensive evaluations across diverse reasoning tasks and multiple model families to quantify how different instruction formulations influence model accuracy and response characteristics.

3.1 Benchmark

We selected four benchmarks covering diverse reasoning tasks in mathematical and scientific domains. The American Invitational Mathematics Examination (AIME 2024)[2] contains competitive mathematics problems that require precise numerical reasoning. GSM8K (Cobbe et al., 2021) includes diverse grade school math word problems designed to test multi-step arithmetic reasoning. MATH-500 is a curated subset of 500 problems from the MATH benchmark (Hendrycks et al., 2021), selected by OpenAI as part of the *Let's Verify Step by Step* study (Lightman et al., 2023). For advanced scientific reasoning, we use the GPQA-Diamond subset of the GPQA (Rein et al., 2024), which is originally a multiple-choice benchmark but is evaluated in our setting without answer options to better assess pure reasoning ability. These benchmarks were chosen due to their prominent role in recent LLM technical reports (Achiam et al., 2023; Grattafiori et al., 2024; Team et al., 2023; Yang et al., 2024; Team et al., 2025; Anthropic, 2024; Guo et al., 2025).

3.2 Models

Our evaluation covers ten open-source instruction-tuned LLMs spanning three major model families. This includes the Gemma 3 series from Google—Gemma 3 4B, 12B, and 27B Instruct variants (Team et al., 2025); the Qwen 2.5 Instruct models from Alibaba at 3B, 7B, 32B, and 72B scales (Yang et al., 2024); and Meta's Llama 3 family—Llama 3.2 3B, Llama 3.1 8B, and Llama 3.3 70B Instruct models (Grattafiori et al., 2024).

[2] https://huggingface.co/datasets/HuggingFaceH4/aime_2024

3.3 Instruction Style Variations

Instruction style variations for Style Sensitivity Experiment

Style	Characteristics and Example
Declarative	*Statements presenting facts or information* "The problem should be solved step by step. The answer is to be suggested in the following format."
Interrogative	*Direct questions challenging reasoning* "Could you solve the problem step by step? Would you suggest the answer in the following format?"
Exclamative	*Expressions emphasizing importance or difficulty* "How important it is to solve the problem step by step! What a necessity it is to suggest the answer in the following format!"
Imperative	*Command-based instructions directing processes* "Solve the problem step by step. Suggest the answer in the following format."

Figure 1: Instruction Styles Employed in Experiments. Detailed prompt templates are provided in Figure 8.

To probe stylistic sensitivity without introducing semantic drift, we adopted a minimal intervention strategy. Rather than altering the benchmark question itself, we applied controlled stylistic variations solely to the instruction prefix. This design enables targeted analysis of how stylistic differences in prompts influence model responses, while ensuring that the core task input remains unchanged. Our typology follows the syntactic clause classification of Huddleston et al. (2002), comprising Declarative, Interrogative, Exclamative, and Imperative forms. We ensured semantic consistency by fixing main verbs (e.g., solve, suggest) and maintaining comparable lexical complexity across styles (Type Token Ratio: 0.75–0.78). The prompt structure used in our experiments is illustrated in Figure 8.

3.4 Experimental Settings

We evaluated models using two decoding strategies: beam search and greedy search, both with `max_new_tokens = 2048`. For beam search, we

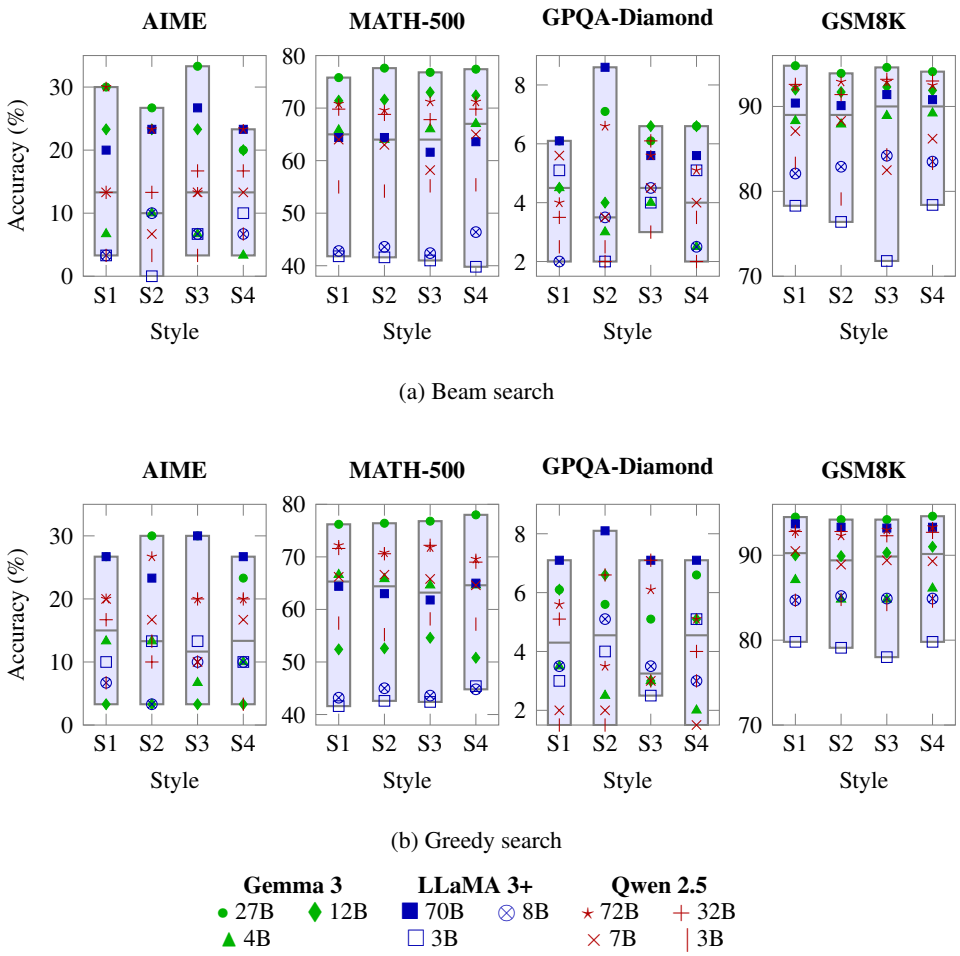


Figure 2: Instruction style sensitivity: Accuracy ranges (S1-S4) by model family and size for Beam Search (a, T=1.0) and Greedy Search (b, T=0.0), showing performance impact of instruction style.

used `temperature = 1.0`, `top-k = 50`, and `top-p = 0.9`. Greedy search involved selecting the highest probability token at each step. We included both decoding strategies to comprehensively examine model behavior across a broad range of generation settings.

3.5 Result

Our experiments demonstrate performance variations across instruction styles and decoding strategies (Figure 2). Models exhibit accuracy fluctuations when identical problems are framed using different instruction styles (S1: Declarative, S2: Interrogative, S3: Exclamative, S4: Imperative, as per Figure 1). Detailed Style Sensitivity Index scores across all benchmarks are provided in Table 4, with comprehensive benchmark-specific analyses available in Appendix D.

Under beam search (Figure 2a), accuracy variations were pronounced. For instance, on AIME, Gemma 3-27B and Qwen 2.5-72B exhibited gaps of 13.3% and 16.7%, respectively, while LLaMA 3-70B showed a 3.0% gap on GPQA-Diamond. Under greedy search (Figure 2b), these variations remained with reduced magnitude—for example, LLaMA 3-70B and Qwen 2.5-72B each shifted by 6.7% on AIME. This confirms that even deterministic decoding remains susceptible to instruction style.

These findings underscore that single-style benchmarks offer an incomplete view of model capabilities. We posit that instruction-induced accuracy fluctuations reflect differences in *response consistency*—the degree to which a model produces structurally, lexically, and logically stable outputs across prompt styles.

4 RCScore: Quantifying Response Consistency Across Instruction Styles

The observed accuracy fluctuations, as detailed in Section 3.5, highlight the limitations of relying solely on task performance metrics. To enable a

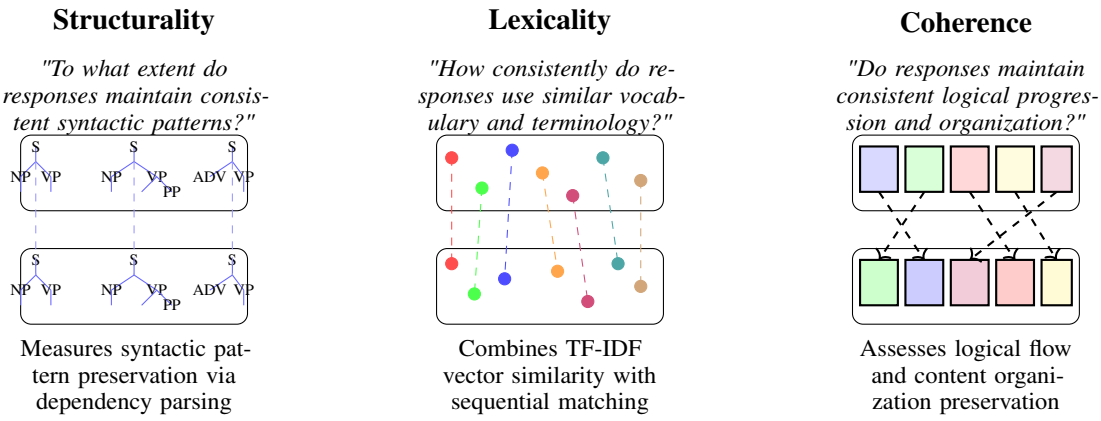


Figure 3: **RCScore**: A multi-dimensional metric for quantifying response consistency across instruction styles.

more precise and multi-faceted analysis of how instruction style influences the fundamental qualities of text generated by LLMs, we introduce RCScore, a comprehensive framework that measures response variation across stylistic dimensions. Unlike conventional metrics focused purely on correctness, RCScore captures how different prompt formulations impact model behavior through three complementary dimensions—Structurality, Lexicality, and Coherence—each quantifying a core aspect of stylistic variation (Figure 3). This enables structured evaluation of instruction sensitivity by examining stylistic consistency and assessing fidelity to reference outputs under diverse prompt formulations.

Structurality Structurality quantifies the syntactic similarity between text samples, addressing the research question: *"To what extent do responses maintain consistent syntactic patterns and grammatical structures?"* This dimension is formalized as:

$$S(D_a, D_b) = \frac{1}{|M|} \sum_{(s_i,t_i)\in M} J(P(s_i), P(t_i)) \quad (1)$$

Here, M denotes semantically aligned sentence pairs (identified via BERTScore[3]), $P(s)$ extracts syntactic patterns of the form $\langle pos_t, dep_r, pos_h \rangle$, capturing part-of-speech tags and dependency relations, and J computes the Jaccard similarity between the resulting pattern sets.

Lexicality Lexicality captures the degree of lexical overlap and vocabulary similarity between generated responses, aiming to answer the question: *"How consistently do responses employ similar vocabulary, terminology, and phrasing?"* We define this dimension as follows:

$$L(D_a, D_b) = w_{\text{TF}} \cdot S_{\text{TF}} + w_{\text{RL}} \cdot S_{\text{RL}} \quad (2)$$

Here, S_{TF} denotes the cosine similarity between TF-IDF vectors, which reflects the global term distribution across documents. In addition, S_{RL} measures sequential lexical overlap via ROUGE-L, capturing local word order. By combining these complementary perspectives, Lexicality provides a more comprehensive measure of textual similarity. We assign equal weights ($w_{\text{TF}} = w_{\text{RL}} = 0.5$) to balance global and local contributions.

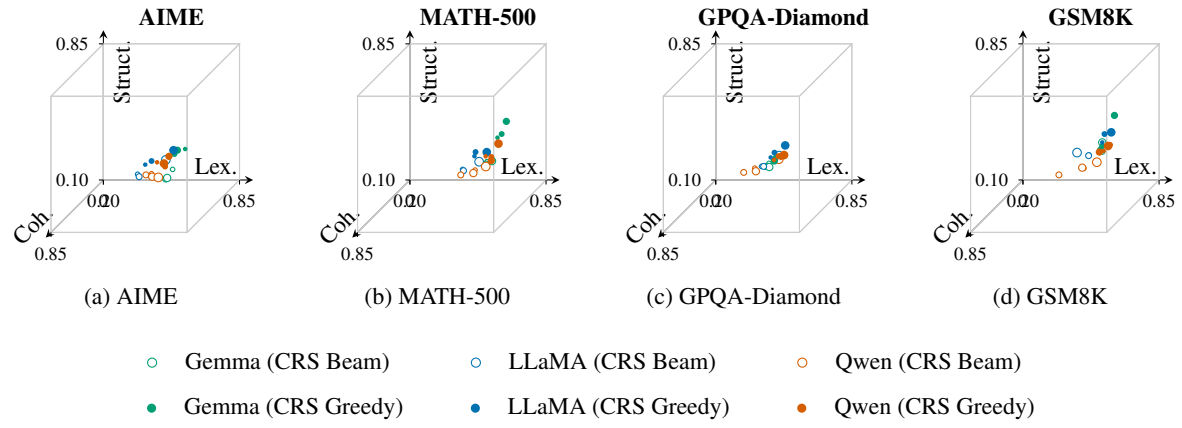
Coherence Coherence assesses the logical flow and organizational similarity between model outputs, addressing the question: *"To what extent do responses maintain consistent logical progression and content organization?"* We define the coherence score as:

$$C(D_a, D_b) = S(D_a, D_b) \cdot C_w(D_a, D_b) \quad (3)$$

The primary component $S(D_a, D_b)$ aggregates four distinct aspects of organizational similarity: order correlation, position matching, sequential continuity, and semantic alignment. Each aspect captures a different facet of how well the content structure is preserved between responses. To ensure comparisons remain semantically meaningful, we apply a content-weighted penalty term $C_w(D_a, D_b)$ that down-weights alignments over irrelevant or trivial content.

A final RCScore is then computed as a weighted average of these three dimensions, with Structurality, Lexicality, and Coherence each assigned

[3]We use the RoBERTa-Large model (Liu et al., 2019) for our BERTScore implementation.



(a) AIME (b) MATH-500 (c) GPQA-Diamond (d) GSM8K

Figure 4: CRS values (RCScore dimensions: Lexicality, Structurality, Coherence) per benchmark. Beam Search (hollow circles) vs. Greedy Search (filled circles). Marker size reflects model parameter count.

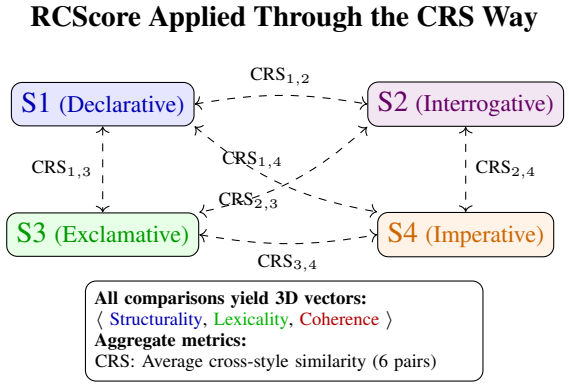


Figure 5: The Cross-Response Similarity (CRS) way applies RCScore dimensions to measure consistency across all possible combinations of instruction styles (6 pairwise comparisons).

a weight of 0.33. For practical examples of how these dimensions quantify similarity, refer to Table 5 (high similarity) and Table 6 (low similarity), which demonstrate the metrics' application through concrete cases. A comprehensive mathematical treatment is provided in Appendix A.

4.1 Quantification of Cross-Style Response Consistency using RCScore

To assess how instruction styles affect model response characteristics beyond accuracy, we introduce the Cross-Response Similarity (CRS) way of applying RCScore dimensions. This approach (Figure 5), detailed in Appendix A.2, quantifies the stylistic consistency among a model's own responses when presented with the same underlying problem framed by different instruction styles. CRS thus highlights a model's tendency to produce stylistically similar or divergent responses based on variations in prompt formulation. A key aspect of our methodology is the computation of CRS values from the three-dimensional RCScore vector—⟨Structurality, Lexicality, Coherence⟩—as defined in Appendix A. This approach retains the multi-dimensional nature of response style and enables fine-grained analysis of which linguistic aspects (syntactic structure, lexical choice, or logical organization) are most affected by instruction variations. For each problem instance, CRS values are computed by aggregating all pairwise comparisons among responses generated from different instruction styles. The detailed aggregation procedure is outlined in Algorithm 1.

4.2 Analyzing Cross-Response Similarity (CRS) with RCScore Dimensions

We analyze model consistency across instruction styles using the CRS way defined in Section 4.1. CRS applies RCScore's three dimensions (Structurality, Lexicality, and Coherence) to quantify response stability. Tables (1a, 1b) present these values for Beam and Greedy Search respectively, with scores ranging from 0 to 1 (higher values indicating greater consistency).

Our analysis reveals three key patterns. First, decoding strategy significantly affects consistency - Greedy Search consistently yields higher CRS values than Beam Search (e.g., LLaMA 3.3-70B's GSM8K RCScore increases from 0.44 to 0.67). Second, larger models generally demonstrate higher consistency, as seen with LLaMA 3.3-70B outperforming LLaMA 3.2-3B on AIME (0.44 vs. 0.31 with Beam Search). Third, task complexity influences consistency - Gemma 3-27B with

Model	AIME				MATH-500				GPQA-Diamond				GSM8K			
	CRS				CRS				CRS				CRS			
	Struct.	Lex.	Coh.	RCScore	Struct.	Lex.	Coh.	RCScore	Struct.	Lex.	Coh.	RCScore	Struct.	Lex.	Coh.	RCScore
Gemma 3-4B	0.25	0.61	0.27	0.38	0.33	0.68	0.38	0.46	0.29	0.54	0.29	0.37	0.42	0.70	0.45	0.52
Gemma 3-12B	0.20	0.58	0.28	0.36	0.32	0.68	0.39	0.46	0.27	0.52	0.28	0.36	0.41	0.71	0.45	0.52
Gemma 3-27B	0.21	0.59	0.29	0.37	0.34	0.71	0.40	0.48	0.28	0.55	0.32	0.38	0.47	0.72	0.48	0.56
LLaMA 3.2-3B	0.22	0.44	0.26	0.31	0.26	0.54	0.30	0.36	0.28	0.51	0.31	0.37	0.37	0.63	0.39	0.46
LLaMA 3.1-8B	0.21	0.45	0.27	0.31	0.26	0.55	0.32	0.38	0.28	0.52	0.31	0.37	0.38	0.64	0.43	0.48
LLaMA 3.3-70B	0.34	0.61	0.38	0.44	0.35	0.66	0.44	0.48	0.37	0.62	0.40	0.46	0.38	0.57	0.38	0.44
Qwen 2.5-3B	0.25	0.53	0.34	0.37	0.28	0.62	0.37	0.42	0.25	0.46	0.25	0.32	0.29	0.60	0.37	0.42
Qwen 2.5-7B	0.24	0.50	0.33	0.35	0.24	0.54	0.33	0.37	0.23	0.41	0.26	0.30	0.23	0.46	0.30	0.33
Qwen 2.5-32B	0.23	0.53	0.33	0.36	0.25	0.60	0.33	0.39	0.25	0.48	0.30	0.34	0.28	0.58	0.33	0.39
Qwen 2.5-72B	0.25	0.58	0.40	0.41	0.31	0.68	0.40	0.46	0.34	0.61	0.37	0.44	0.31	0.65	0.33	0.43

(a) Beam Search CRS Values (temperature = 1.0).

Model	AIME				MATH-500				GPQA-Diamond				GSM8K			
	CRS				CRS				CRS				CRS			
	Struct.	Lex.	Coh.	RCScore	Struct.	Lex.	Coh.	RCScore	Struct.	Lex.	Coh.	RCScore	Struct.	Lex.	Coh.	RCScore
Gemma 3-4B	0.42	0.72	0.44	0.53	0.52	0.78	0.55	0.61	0.32	0.57	0.31	0.40	0.46	0.73	0.54	0.57
Gemma 3-12B	0.44	0.71	0.58	0.57	0.54	0.80	0.55	0.63	0.33	0.58	0.32	0.41	0.46	0.73	0.51	0.57
Gemma 3-27B	0.41	0.68	0.43	0.51	0.63	0.84	0.61	0.69	0.32	0.58	0.34	0.41	0.69	0.84	0.69	0.74
LLaMA 3.2-3B	0.30	0.50	0.34	0.38	0.36	0.62	0.38	0.45	0.36	0.58	0.40	0.44	0.47	0.72	0.48	0.56
LLaMA 3.1-8B	0.32	0.53	0.34	0.40	0.40	0.64	0.43	0.49	0.40	0.61	0.44	0.48	0.55	0.76	0.58	0.63
LLaMA 3.3-70B	0.42	0.67	0.46	0.52	0.43	0.72	0.52	0.56	0.45	0.67	0.47	0.53	0.58	0.81	0.64	0.67
Qwen 2.5-3B	0.34	0.58	0.42	0.45	0.39	0.70	0.47	0.52	0.34	0.59	0.38	0.44	0.45	0.75	0.56	0.59
Qwen 2.5-7B	0.33	0.63	0.46	0.48	0.39	0.73	0.48	0.53	0.37	0.62	0.41	0.47	0.46	0.76	0.49	0.57
Qwen 2.5-32B	0.40	0.66	0.50	0.52	0.37	0.73	0.48	0.52	0.39	0.65	0.47	0.50	0.41	0.70	0.46	0.53
Qwen 2.5-72B	0.33	0.61	0.41	0.45	0.47	0.77	0.50	0.58	0.38	0.65	0.42	0.49	0.43	0.73	0.42	0.52

(b) Greedy Search CRS Values (temperature = 0.0).

Table 1: Cross-Response Similarity (CRS) values for AIME, MATH-500, GPQA-Diamond, and GSM8K benchmarks across models, comparing Beam Search (temperature = 1.0) and Greedy Search (temperature = 0.0) generation methods.

Greedy Search achieves 0.51 on AIME but 0.74 on GSM8K. Figure 4 visually maps these multi-dimensional relationships across models and benchmarks.

5 Is Cross-Style Consistency a Reliable Proxy for Accuracy?

Metric	Pearson r	p-val	Spearman ρ	p-val
Beam Search (temperature = 1.0)				
$\text{RCScore}_{\text{Struct}}$	0.57	1.4e-04	0.49	0.001
$\text{RCScore}_{\text{Lex}}$	0.65	5.0e-06	0.68	1.7e-06
$\text{RCScore}_{\text{Coh}}$	0.64	9.0e-06	0.65	6.9e-06
$\text{RCScore}_{\text{Overall}}$	0.66	3.1e-06	0.65	6.9e-06
Greedy Search (temperature = 0.0)				
$\text{RCScore}_{\text{Struct}}$	0.675	1.74e-06	0.684	1.13e-06
$\text{RCScore}_{\text{Lex}}$	0.790	1.39e-09	0.786	1.88e-09
$\text{RCScore}_{\text{Coh}}$	0.656	4.38e-06	0.641	8.32e-06
$\text{RCScore}_{\text{Overall}}$	0.733	7.43e-08	0.725	1.20e-07

Table 2: Pearson's r and Spearman's ρ correlations between mean task accuracy and RCScore dimensions (Structurality, Lexicality, Coherence, and Overall) applied through the Cross-Response Similarity (CRS) way across all model-benchmark pairs (N=40) for Beam Search and Greedy Search

To investigate whether a model's consistency across varied instruction styles correlates with its task-solving accuracy, we analyzed the relationship between our CRS values and conventional accuracy metrics. For each of the ten models across four benchmarks (AIME, GSM8K, MATH-500, GPQA), we computed CRS values using RCScore's three dimensions (Structurality, Lexicality, Coherence) and the aggregated RCScore. Concurrently, we calculated a mean accuracy score for each of the 40 model-benchmark pairs by averaging accuracies achieved under each of the four instruction styles. We then measured Pearson's r and Spearman's ρ correlations between these mean accuracies and each dimension of CRS to assess both linear and monotonic relationships.

The results, presented in Table 2, reveal a significant positive correlation between a model's ability to maintain stylistic consistency and its overall task performance. Notably, the aggregated RCScore and its Lexicality dimension when applied through the CRS way showed strong correlations with mean accuracy. For instance, Lexicality exhibited a Pearson's $r > 0.65$ under beam search and $r \approx 0.79$ under greedy search. This suggests that models

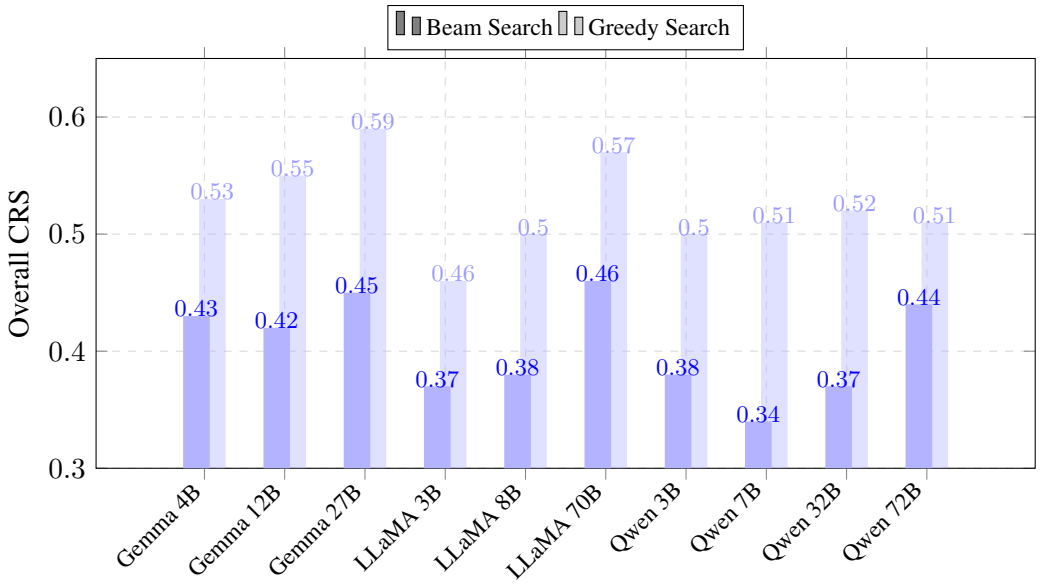


Figure 6: Overall CRS values for Gemma, LLaMA, and Qwen model variants. Solid bars correspond to beam search (T=1.0), and semi-transparent bars to greedy search (T=0.0).

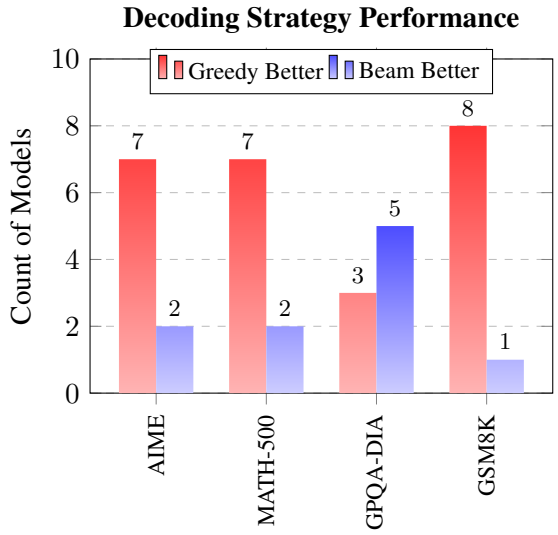


Figure 7: Decoding strategy performance: Number of models where Greedy Search or Beam Search achieved higher average accuracy per benchmark. Greedy Search generally performed better, aligning with its higher CRS values (Figure 6). Detailed average accuracy scores are available in Table 7.

producing lexically consistent explanations across different instruction styles tend to achieve higher accuracy. More broadly, all dimensions demonstrated statistically significant positive correlations with task accuracy when measured via the CRS approach. This consistent pattern indicates that higher cross-style self-similarity in a model's responses is a robust indicator of better task performance. These findings underscore the utility of RCScore dimensions applied through CRS in capturing nuanced behavioral traits of LLMs—specifically, the ability to generate consistent outputs irrespective of instruction style—that are meaningfully aligned with their fundamental task-solving capabilities. Thus, cross-style consistency, as quantified through RCScore dimensions, emerges not just as a measure of stylistic stability but as a valuable proxy for a model's underlying accuracy and reliability.

5.1 Parameter-Scale and Decoding Strategy Effects on CRS

Figure 6 displays CRS values, highlighting the influence of model scale and decoding strategy on stylistic consistency. Larger models (>70B), such as LLaMA 3.3-70B and Qwen 2.5-72B, generally demonstrate higher CRS, suggesting that increased parameter scale improves stylistic stability against varied instruction phrasings. Decoding strategy also plays a critical role: greedy search ($T = 0.0$) consistently yields 7-13% higher CRS scores than beam search ($T = 1.0$) across all models, indicating deterministic decoding produces more stylistically stable outputs. Notably, this enhanced consistency with greedy search correlates with improved task performance. As illustrated in Figure 7, greedy decoding led to higher average accuracy for a majority of models on the AIME, MATH-500, and GSM8K benchmarks. While Beam Search appears to have a slight edge on GPQA-Diamond for more models, the overall low accuracy scores on this benchmark (typically below 10% for most models,

as seen in Table 3) suggest that these particular differences in decoding strategy performance might be less indicative of a general trend. Overall, the deterministic approach of greedy search also narrows the CRS performance gap between smaller and larger models, particularly benefiting smaller models in maintaining consistency.

6 Conclusion

We introduce RCScore, a multi-dimensional metrics designed to assess LLM sensitivity to instruction style, extending traditional accuracy-focused evaluations. RCScore quantifies response variations along three core dimensions: Structurality, Lexicality, and Coherence. Through experiments applying varied instruction styles to established benchmarks, we demonstrate that LLM performance can fluctuate based on instruction phrasing. Our findings introduce Cross-Response Similarity (CRS) as a measure of stylistic self-consistency, establishing a strong positive correlation between CRS and task accuracy. Additionally, we observe that larger models exhibit greater stylistic stability, and deterministic decoding produces more consistent outputs. These insights suggest that cross-style consistency serves as a valuable indicator of model reliability, offering deeper perspectives on LLM robustness.

7 Limitations

While RCScore offers a more nuanced evaluation of instruction style sensitivity, its current implementation relies on a predefined set of four linguistic styles and specific English-language NLP tools for dimensional analysis (e.g., dependency parsing, TF-IDF). The framework's applicability to other languages or a broader range of stylistic variations (e.g., code-switching, highly informal language) remains to be explored, and the computational cost, though minimized, might still be a factor for extremely large-scale evaluations across numerous model-benchmark pairs.

Additionally, our focus on mathematical and reasoning tasks, while methodologically sound, may not generalize to more subjective domains like creative writing or emotional support, where style sensitivity might manifest differently. The current RCScore dimensions also lack validation through human evaluation to confirm their alignment with human perceptions of consistency or quality. Furthermore, the weights for aggregating RCScore dimensions are currently uniform, and future work could investigate task-dependent or empirically derived weighting schemes.

8 Ethical Consideration

The primary ethical considerations for RCScore relate to its potential application and interpretation. While designed to improve LLM evaluation by revealing sensitivities, there is a risk that findings could be used to "over-optimize" models for specific instruction styles, potentially at the expense of generalizability or by inadvertently encoding biases present in the chosen stylistic variations.

The datasets used (AIME, MATH, GPQA, GSM8K) are standard academic benchmarks, and the instruction variations are syntactically derived without introducing new factual content, minimizing the risk of generating harmful or biased test material. We acknowledge that inconsistent performance across instruction styles raises accessibility concerns, as users with different cultural backgrounds or neurodivergent conditions might naturally prefer specific formulation patterns, potentially creating uneven experiences with AI systems.

Additionally, the increased computational requirements for multi-style evaluation introduce environmental considerations that should be balanced against the benefits of more comprehensive assessment. We encourage users of RCScore to be mindful of these aspects and to employ the framework as a tool for understanding and improving model robustness broadly, rather than for narrow stylistic optimization.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Anirudh Ajith, Chris Pan, Mengzhou Xia, Ameet Deshpande, and Karthik Narasimhan. 2023. Instructeval: Systematic evaluation of instruction selection methods. *arXiv preprint arXiv:2307.00259*.

Anthropic. 2024. Introducing the next generation of claude.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Bowen Cao, Deng Cai, Zhisong Zhang, Yuexian Zou, and Wai Lam. 2024. On the worst prompt performance of large language models. *arXiv preprint arXiv:2406.10248*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*.

Federico Errica, Giuseppe Siracusano, Davide Sanvito, and Roberto Bifulco. 2024. What did i do wrong? quantifying llms' sensitivity and consistency to prompt engineering. *arXiv preprint arXiv:2406.12334*.

Olga Golovneva, Moya Peng Chen, Spencer Poff, Martin Corredor, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. 2023. ROSCOE: A suite of metrics for scoring step-by-step reasoning. In *The Eleventh International Conference on Learning Representations*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Jia He, Mukund Rungta, David Koleczek, Arshdeep Sekhon, Franklin X Wang, and Sadid Hasan. 2024. Does prompt formatting have any impact on llm performance? *arXiv preprint arXiv:2411.10541*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

R.D. Huddleston, G.K. Pullum, and L. Bauer. 2002. *The Cambridge Grammar of the English Language*. The Cambridge Grammar of the English Language. Cambridge University Press.

Ivano Lauriola, Stefano Campese, and Alessandro Moschitti. 2025. Analyzing and improving coherence of large language models in question answering. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 11740–11755.

Alina Leidinger, Robert Van Rooij, and Ekaterina Shutova. 2023. The language of prompting: What linguistic properties make a prompt successful? *arXiv preprint arXiv:2311.01967*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM computing surveys*, 55(9):1–35.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *Preprint*, arXiv:1907.11692.

Sheng Lu, Hendrik Schuff, and Iryna Gurevych. 2024. How are prompts different in terms of sensitivity? In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5833–5856.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36:46534–46594.

Hamed Mahdavi, Alireza Hashemi, Majid Daliri, Pegah Mohammadipour, Alireza Farhadi, Samira Malek, Yekta Yazdanifard, Amir Khasahmadi, and Vasant Honavar. 2025. Brains vs. bytes: Evaluating llm proficiency in olympiad mathematics. *arXiv preprint arXiv:2504.01995*.

Potsawee Manakul, Adian Liusie, and Mark JF Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *arXiv preprint arXiv:2305.14251*.

Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. 2024. State of what art? a call for multi-prompt llm evaluation. *Transactions of the Association for Computational Linguistics*, 12:933–949.

Philipp Mondorf and Barbara Plank. 2024. Beyond accuracy: Evaluating the reasoning behavior of large language models–a survey. *arXiv preprint arXiv:2404.01869*.

Minh-Vuong Nguyen, Linhao Luo, Fatemeh Shiri, Dinh Phung, Yuan-Fang Li, Thuy-Trang Vu, and Gholamreza Haffari. 2024. Direct evaluation of chain-of-thought in multi-hop reasoning with knowledge graphs. *arXiv preprint arXiv:2402.11199*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Jon Saad-Falcon, Omar Khattab, Christopher Potts, and Matei Zaharia. 2023. Ares: An automated evaluation framework for retrieval-augmented generation systems. *arXiv preprint arXiv:2311.09476*.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Jan Philip Wahle, Terry Ruas, Yang Xu, and Bela Gipp. 2024. Paraphrase types elicit prompt engineering capabilities. *arXiv preprint arXiv:2406.19898*.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. 2023. Large language models are not fair evaluators. *arXiv preprint arXiv:2305.17926*.

Ruiqi Wang, Jiyu Guo, Cuiyun Gao, Guodong Fan, Chun Yong Chong, and Xin Xia. 2025. Can llms replace human evaluators? an empirical study of llm-as-a-judge in software engineering. *arXiv preprint arXiv:2502.06193*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. 2025. Evaluating mathematical reasoning beyond accuracy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 27723–27730.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Shunyu Yao, Diszzzzan Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. In *The Eleventh International Conference on Learning Representations*.

A RCScore Computation Details

A.1 Dimension Formulations

All three metrics employ a unified computational approach where similarity scores are calculated between document pairs, normalized to the [0,1] range, with higher values indicating greater similarity along the respective dimension.

A.1.1 Structurality

We compute Structurality through syntactic pattern comparison between aligned sentence pairs:

$$Struct(D_a, D_b) = \frac{1}{|M|} \sum_{(s_i,t_i)\in M} J(P(s_i), P(t_i)) \tag{4}$$

where D_a, D_b denote the source and target documents, $M = \{(s_1, t_1), (s_2, t_2), \ldots, (s_n, t_n)\}$ represents the set of semantically aligned sentence pairs determined via BERTScore similarity, $P(s)$ extracts the set of syntactic patterns from sentence s, and $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$ calculates the Jaccard similarity between pattern sets. Each syntactic pattern has the form $\langle pos_t, dep_r, pos_h \rangle$ representing the part-of-speech tag, dependency relation, and head token's part-of-speech.

This approach captures fine-grained syntactic relationships while abstracting away from specific lexical choices, allowing us to measure structural similarity even when vocabulary differs substantially.

A.1.2 Lexicality

We define Lexicality as the weighted sum of two complementary similarity measures:

$$\text{Lex}(D_a, D_b) = w_{\text{TF}} \cdot S_{\text{TF}} + w_{\text{RL}} \cdot S_{\text{RL}} \quad (5)$$

where S_{TF} represents the TF-IDF cosine similarity between document vectors, and S_{RL} quantifies the ROUGE-L F-measure based on longest common subsequence. w_{TF} and w_{RL} are weights with default values $w_{\text{TF}} = w_{\text{RL}} = 0.5$. This combined approach balances global vocabulary distribution similarity (TF-IDF) with local sequential word matching (ROUGE-L).

A.1.3 Coherence

We formulate Coherence as the product of structural alignment and content similarity:

$$\text{Coherence}(D_a, D_b) = S(D_a, D_b) \cdot C_w(D_a, D_b) \quad (6)$$

where $S(D_a, D_b) = w_O \cdot O + w_P \cdot P + w_N \cdot N + w_C \cdot C_s$ represents the structural similarity score combining four components: (1) $O = \frac{\tau+1}{2}$, the normalized Kendall's Tau correlation (τ) of chunk order; (2) P, position matching with emphasis on document endpoints; (3) N, sequential continuity between matched chunks; and (4) C_s, semantic similarity between aligned chunks. The weights w_O, w_P, w_N, w_C have default values of 0.25 each. The term $C_w(D_a, D_b) = C_s^2$ applies a quadratic content-weighted penalty to ensure that structural similarities are only meaningful when the compared chunks contain related information.

The computation involves segmenting documents into semantic chunks of adaptive size k, aligning chunks using a combined BERTScore and TF-IDF similarity matrix, and analyzing the sequence and position of matched chunk pairs.

Algorithm 1 Applying RCScore Dimensions through the CRS Way

Require: Style-variant responses $\{R_s\}$ where $s \in S$

Ensure: Three-dimensional CRS values using RCScore dimensions

function RCSCORE(D_1, D_2)
  return $\langle$Structurality, Lexicality, Coherence$\rangle$ ▷ 3D similarity vector
end function
$\text{CRS}_{\text{vectors}} \leftarrow \emptyset$
$n \leftarrow \binom{|S|}{2}$ ▷ Number of style pairs
for $(s_i, s_j) \in \binom{S}{2}$ **do** ▷ All style pairs
  $\vec{sim}_{i,j} \leftarrow \text{RCScore}(R_{s_i}, R_{s_j})$
  $\text{CRS}_{\text{vectors}} \leftarrow \text{CRS}_{\text{vectors}} \cup \{\vec{sim}_{i,j}\}$
end for
$\text{CRS} \leftarrow \frac{1}{n} \sum_{\vec{v} \in \text{CRS}_{\text{vectors}}} \vec{v}$ ▷ Dimension-preserving average
return $\text{CRS} = \langle \text{CRS}_{\text{Struct}}, \text{CRS}_{\text{Lex}}, \text{CRS}_{\text{Coh}} \rangle$

A.2 Cross-Response Similarity (CRS) Way of Applying RCScore

To quantify how instruction styles influence model responses, we apply RCScore dimensions through the Cross-Response Similarity (CRS) way (Algorithm 1):

- **Cross-Response Similarity (CRS)** - measures the consistency across $\binom{|S|}{2}$ pairs of responses generated under different instruction formulations using RCScore dimensions.

The key innovation in our approach is preserving RCScore's three-dimensional nature when calculating cross-style consistency. Each comparison produces a vector $\langle struct, lex, coh \rangle$ representing Structurality, Lexicality, and Coherence dimensions. This allows for a detailed analysis of which specific aspects of reasoning style are most affected by instruction variations.

For each problem, we compute:

$$\text{CRS}_p = \text{Aggregate}\{\vec{sim}_{i,j} | (s_i, s_j) \in \binom{S}{2}\} \quad (7)$$

Model	AIME				MATH-500				GPQA-Diamond				GSM8K			
	S1	S2	S3	S4	S1	S2	S3	S4	S1	S2	S3	S4	S1	S2	S3	S4
Gemma 3-4B	6.7	10.0	6.7	3.3	65.8	64.0	66.0	67.0	4.5	3.0	4.0	2.5	88.3	87.9	88.9	89.2
Gemma 3-12B	23.3	23.3	23.3	20.0	71.4	71.6	73.0	72.4	4.5	4.0	6.6	6.6	92.0	91.7	92.3	91.9
Gemma 3-27B	30.0	26.7	33.3	20.0	75.8	77.6	76.8	77.4	6.1	7.1	6.1	6.6	94.8	93.9	94.6	94.1
LLaMA 3.2-3B	3.3	0.0	6.7	10.0	41.8	41.6	41.0	39.8	5.1	2.0	4.0	5.1	78.3	76.4	71.8	78.4
LLaMA 3.1-8B	3.3	10.0	6.7	6.7	42.8	43.6	42.4	46.4	2.0	3.5	4.5	2.5	82.1	82.9	84.2	83.5
LLaMA 3.3-70B	20.0	23.3	26.7	23.3	64.4	64.4	61.6	63.6	6.1	8.6	5.6	5.6	90.4	90.1	91.4	90.8
Qwen 2.5-3B	3.3	3.3	3.3	6.7	55.0	54.2	55.2	55.4	2.5	2.5	3.0	3.5	83.3	79.1	84.3	83.3
Qwen 2.5-7B	13.3	6.7	13.3	13.3	64.0	63.0	58.2	65.0	5.6	3.5	4.5	4.0	87.1	88.3	82.5	86.2
Qwen 2.5-32B	13.3	13.3	16.7	16.7	69.8	68.8	67.8	69.8	3.5	2.0	6.1	2.0	92.6	91.4	93.2	93.0
Qwen 2.5-72B	30.0	23.3	13.3	23.3	70.8	69.6	71.2	71.2	4.0	6.6	5.6	5.1	92.3	92.9	92.9	92.5

(a) Beam search (temperature = 1.0).

Model	AIME				MATH-500				GPQA-Diamond				GSM8K			
	S1	S2	S3	S4	S1	S2	S3	S4	S1	S2	S3	S4	S1	S2	S3	S4
Gemma 3-4B	13.3	13.3	6.7	10.0	66.6	65.8	64.6	64.6	3.5	2.5	3.0	2.0	87.1	84.8	84.8	86.1
Gemma 3-12B	3.3	3.3	3.3	3.3	52.4	52.6	54.6	50.8	6.1	6.6	3.0	5.1	90.0	89.9	90.3	91.0
Gemma 3-27B	26.7	30.0	30.0	23.3	76.2	76.4	76.8	78.0	6.1	5.6	5.1	6.6	94.5	94.2	94.2	94.6
LLaMA 3.2-3B	10.0	13.3	13.3	10.0	41.6	42.6	42.4	45.4	3.0	4.0	2.5	5.1	79.8	79.1	78.0	79.8
LLaMA 3.1-8B	6.7	3.3	10.0	10.0	43.2	45.0	43.6	44.8	3.5	5.1	3.5	3.0	84.7	85.2	84.9	84.9
LLaMA 3.3-70B	26.7	23.3	30.0	26.7	64.4	63.0	61.8	65.0	7.1	8.1	7.1	7.1	93.7	93.3	93.2	93.3
Qwen 2.5-3B	6.7	13.3	10.0	3.3	57.4	55.2	58.2	57.2	1.5	1.5	3.0	3.0	84.7	84.8	84.2	84.6
Qwen 2.5-7B	20.0	16.7	10.0	16.7	66.2	66.6	65.8	64.6	2.0	2.0	3.0	1.5	90.5	88.9	89.4	89.3
Qwen 2.5-32B	16.7	10.0	20.0	20.0	71.6	70.6	72.2	69.0	5.1	6.6	7.1	4.0	92.8	92.8	92.3	92.7
Qwen 2.5-72B	20.0	26.7	20.0	20.0	72.2	70.8	71.8	69.6	5.6	3.5	6.1	5.1	92.8	92.3	93.0	93.3

(b) Greedy search (temperature = 0.0).

Table 3: Accuracy by Instruction Style (S1: Declarative, S2: Interrogative, S3: Exclamative, S4: Imperative) on four benchmarks (AIME, MATH-500, GPQA-Diamond, and GSM8K) for different decoding strategies.

where S is the set of instruction styles, and $\vec{sim}_{i,j}$ is the similarity vector between responses generated under instruction styles s_i and s_j, calculated using RCScore dimensions.

B Experimental Implementation Details

RCScore evaluates model performance across different instruction styles while maintaining consistent semantic content. Figure 8 shows our implementation approach for applying these style variations.

Rather than completely rephrasing benchmark problems, which would be computationally expensive and potentially introduce unintended semantic shifts, we adopted a lightweight approach that appends style-specific instructions to the original problem text. This preserves the core problem content while varying only the instruction component.

As shown in Figure 8, our implementation defines four instruction styles—Declarative, Interrogative, Exclamative, and Imperative—following linguistic clause type classifications. Each style maintains consistent main verbs (solve, suggest) while varying syntactic structure. We carefully controlled the Type-Token Ratio (0.75-0.78) across all styles to ensure lexical complexity remained comparable.

This implementation approach enables efficient evaluation of style sensitivity at scale across multiple benchmarks and models, with minimal disruption to the original benchmark problems.

C Detailed Performance Results

C.1 Performance with Beam Search (Temperature = 1.0)

Table 3a provides comprehensive accuracy results for all evaluated models across the four instruction styles (S1: Declarative, S2: Interrogative, S3: Exclamative, S4: Imperative) and four reasoning tasks when using beam search with **temperature = 1.0**. These detailed measurements complement the visualization in Figure 2 (a), providing exact numerical values.

The results reveal several notable patterns. Within each model family, larger parameter counts generally correlate with higher accuracy. For example, across most tasks, Gemma 3-27B shows higher accuracy than Gemma 3-4B. Different benchmarks exhibit varying levels of style sensitivity; AIME, for instance, shows considerable variance (e.g.,

Model	Style Sensitivity Index (SSI) at temp=1.0				Style Sensitivity Index (SSI) at temp=0.0			
	AIME	MATH	GPQA	GSM8K	AIME	MATH	GPQA	GSM8K
Gemma 3-4B	2.11	0.18	0.95	0.07	1.30	0.14	1.07	0.12
Gemma 3-12B	0.52	0.11	1.08	0.03	0.00	0.28	1.20	0.05
Gemma 3-27B	1.55	0.09	0.43	0.04	0.99	0.09	0.59	0.02
LLaMA 3-3B	4.23	0.19	1.20	0.40	0.92	0.34	1.05	0.10
LLaMA 3-8B	2.39	0.33	1.12	0.10	2.49	0.16	0.81	0.03
LLaMA 3-70B	0.86	0.16	0.65	0.06	0.85	0.18	0.25	0.02
Qwen 2.5-3B	2.26	0.09	0.79	0.25	2.84	0.18	1.78	0.03
Qwen 2.5-7B	1.49	0.43	0.61	0.27	1.54	0.12	1.24	0.08
Qwen 2.5-32B	0.70	0.12	2.67	0.08	1.29	0.15	1.01	0.03
Qwen 2.5-72B	2.42	0.09	0.67	0.03	1.07	0.14	0.85	0.04

Table 4: **Style sensitivity across tasks and temperature settings.** The Style Sensitivity Index (SSI) quantifies variation in model performance across instruction styles, calculated as $\text{SSI} = 5 \cdot (\sigma/\mu) + 0.05 \cdot (\max - \min)$, where σ is population standard deviation and μ is mean accuracy. Higher values indicate greater performance variability. AIME exhibits the highest sensitivity (average SSI: 1.83 at temp=1.0), followed by GPQA-Diamond (average SSI: 1.02), while MATH-500 and GSM8K show lower sensitivity (average SSI: 0.18 and 0.13). While temperature=0.0 generally reduces style sensitivity, three models (Qwen 2.5-3B/32B and LLaMA 3-8B) exhibit increased sensitivity at temperature=0.0 on AIME, demonstrating that deterministic generation can sometimes amplify instruction style effects.

Qwen 2.5-72B ranges from 13.3% with S3 to 30.0% with S1, a 16.7 percentage point (pp) difference). In contrast, on GSM8K, the variation for many models is smaller, though LLaMA 3.2-3B still shows a 6.6% range (71.8% with S3 to 78.4% with S4).

Model families demonstrate distinct patterns of style preference, which can also vary by task. For example, on AIME, Gemma 3-27B achieved its highest accuracy (33.3%) with the Exclamative style (S3), while Qwen 2.5-72B performed best with the Declarative style (S1) at 30.0%. The standard deviation of performance across styles can serve as an indicator of style robustness, with lower deviations suggesting more consistent performance.

C.2 Performance with Greedy Search (Temperature = 0.0)

Table 3b presents accuracy results for the same set of models and tasks when using greedy search (**temperature = 0.0**), providing a deterministic generation scenario. These results are visualized in Figure 2 (b). Similar to beam search results, larger models within each family generally show higher accuracy (e.g., Gemma 3-27B and LLaMA 3.3-70B typically outperform their smaller counterparts).

Benchmarks continue to show varying degrees of sensitivity to instruction styles even in this deterministic setting. AIME and GPQA-Diamond tend to exhibit more pronounced style-based variations compared to GSM8K, where performance is often more consistent. For instance, on AIME, LLaMA 3.1-8B's accuracy ranged from 3.3% (S2) to 10.0% (S3/S4), a 6.7% difference. On GPQA-Diamond, Qwen 2.5-32B varied from 4.0% (S4) to 7.1% (S3), a 3.1% difference. Distinct style preferences persist, with models showing varying optimal styles depending on the task and model size. Standard deviations across styles are generally smaller than at temperature = 1.0, indicating more consistent performance in deterministic generation.

C.3 Comparative Analysis of Temperature Effects

Comparing model performance with beam search (temperature = 1.0) versus greedy search (temperature = 0.0) reveals important insights. Models generally exhibit reduced cross-style variance with greedy search. For example, on AIME, the maximum observed accuracy difference for LLaMA 3.1-8B was 6.7% with greedy search, compared to larger gaps such as 16.7% for Qwen 2.5-72B under beam search.

Despite this reduction in variance, significant style-based performance differences persist even in deterministic generation. As noted, Qwen 2.5-32B on GPQA-Diamond still exhibited a 3.1% difference with greedy search. While greedy search tends to yield more consistent performance across styles for most models, it does not always result in optimal accuracy. In several instances, models achieve their highest accuracy on particular style-task combinations with beam search at temperature = 1.0 (e.g., Gemma 3-27B on AIME with S3: 33.3% at temp=1.0 vs. 30.0% at temp=0.0; LLaMA 3.3-70B

Prompt Template for Instruction Style Variations

Code Implementation:

Instruction Type Definition:

```
sentence_type = [
  # Declarative
  "The problem should be solved step by step.
The answer is to be
  suggested in the following format."
  # Interrogative
  "Could you solve the problem step by step?
Would you suggest
  the answer in the following format?"
  # Exclamative
  "How important it is to solve the problem
step by step! What a
  necessity it is to suggest the answer in
the following format!"
  # Imperative
  "Solve the problem step by step. Suggest
the answer in the
  following format."
]
```

Message Structure:

```
messages = [{
  "role": "user",
  "content": (
    f"{text}\n\n"
    f"{sentence_type}\n"
    f"Solution: [explanation]\n"
    f"Answer: [answer]"
  )
}]
```

Figure 8: **Prompt implementation for RCScore experiments.** We programmatically append different instruction styles to each benchmark problem while maintaining a consistent output structure. This approach enables efficient evaluation of style sensitivity without modifying the core problem content.

on GPQA-Diamond with S2: 8.6% at temp=1.0 vs. 8.1% at temp=0.0), suggesting that some sampling diversity can be beneficial.

These findings indicate that practitioners should consider both instruction style and temperature settings. For applications requiring maximum consistency, greedy search with carefully chosen instruction styles may be preferable. Conversely, applications seeking peak performance might benefit from temperature > 0 with task-specific style optimization. The persistence of style-based variation across temperature settings underscores the importance of comprehensive evaluation frameworks like RCScore.

D Cross-Task Patterns of Style Sensitivity

To quantify how instruction style affects model performance across tasks and temperature settings, we developed the Style Sensitivity Index (SSI):

$$\text{SSI} = 5 \cdot \frac{\sigma}{\mu} + 0.05 \cdot (\max - \min) \tag{8}$$

where σ is the standard deviation of accuracy across styles, μ is mean accuracy, and $(\max - \min)$ captures the absolute performance range. This metric integrates both relative variation and absolute performance differences.

Table 4 presents SSI values for all models across four benchmarks at temperatures 1.0 and 0.0. Style sensitivity patterns persist across all tasks but with task-dependent magnitudes: AIME exhibits highest sensitivity (avg SSI: 1.83 at temp=1.0), followed by GPQA-Diamond (avg SSI: 1.02), with MATH-500 and GSM8K showing substantially lower sensitivity (avg SSI: 0.18 and 0.13). This hierarchy suggests that task complexity and open-endedness amplify instruction phrasing effects.

D.1 Analysis of Style Sensitivity on AIME

AIME demonstrates exceptionally high style sensitivity, with several models showing SSI values exceeding 2.0 at temperature=1.0. LLaMA 3B exhibits the highest sensitivity (SSI: 4.23), followed by LLaMA 8B (2.39), Qwen 72B (2.42), and Qwen 3B (2.26). The competitive nature of AIME problems, requiring advanced mathematical reasoning and precise numerical solutions, appears particularly vulnerable to instruction formulation variations.

While temperature=0.0 generally reduces sensitivity, significant variations persist in deterministic generation. Notably, three models exhibit increased sensitivity at temperature=0.0: Qwen 3B (2.26→2.84), Qwen 32B (0.70→1.29), and LLaMA 8B (2.39→2.49). This contradicts the expectation that deterministic generation would universally stabilize performance across instruction styles.

D.2 Analysis of Style Sensitivity on MATH-500

MATH-500 shows consistently lower style sensitivity than AIME despite both being mathematical reasoning tasks. SSI values predominantly remain below 0.2, with Qwen 7B at temperature=1.0 being the notable exception (SSI: 0.43). The struc-

tured format of MATH-500 problems likely contributes to this reduced sensitivity, providing consistent mathematical notation that models can process regardless of instruction style.

Temperature effects on MATH-500 sensitivity vary by model. Qwen 7B shows significantly higher sensitivity at temperature=1.0 (0.43) compared to temperature=0.0 (0.12), while Gemma 12B exhibits the opposite pattern (0.11→0.28). LLaMA 3B demonstrates significant sensitivity at both temperature settings (0.19→0.34).

D.3 Analysis of Style Sensitivity on GPQA-Diamond

GPQA-Diamond exhibits moderate to high style sensitivity with diverse patterns across model families. Qwen 32B shows exceptionally high sensitivity at temperature=1.0 (SSI: 2.67), followed by LLaMA 3B (1.20) and LLaMA 8B (1.12). The scientific reasoning required by GPQA-Diamond appears particularly susceptible to instruction variations, especially for certain model architectures.

Temperature effects on GPQA sensitivity show model-specific patterns. While LLaMA 70B demonstrates reduced sensitivity at temperature=0.0 (0.65→0.25), several models exhibit increased sensitivity: Qwen 3B (0.79→1.78), Qwen 7B (0.61→1.24), and Gemma 12B (1.08→1.20).

D.4 Analysis of Style Sensitivity on GSM8K

GSM8K demonstrates the lowest style sensitivity among all benchmarks, with most models exhibiting SSI values below 0.1, particularly at temperature=0.0. The well-structured nature of elementary arithmetic word problems likely contributes to this reduced sensitivity, presenting unambiguous reasoning paths regardless of instruction phrasing.

Temperature significantly impacts GSM8K style sensitivity. At temperature=1.0, smaller models show moderate sensitivity: LLaMA 3B (SSI: 0.40), Qwen 7B (0.27), and Qwen 3B (0.25). However, at temperature=0.0, nearly all models demonstrate minimal sensitivity (SSI < 0.05), with only LLaMA 3B (0.10) and Gemma 4B (0.12) showing SSI values above 0.1.

E Discussion

RCScore provides a crucial lens on LLM performance by assessing sensitivity to instruction style—a dimension typically absent in standard benchmarks. Our findings confirm that models exhibit significant accuracy variations and stylistic shifts in their responses (as measured by CRS) when instruction phrasing changes, even if semantic content is preserved. This variability, influenced by task complexity and decoding strategy, highlights the limitations of single-formulation evaluations.

The notable positive correlation between CRS (particularly its lexical and aggregated components) and mean task accuracy suggests a deeper connection: models that maintain stylistic consistency when explaining solutions across varied prompts tend to achieve higher accuracy. This implies that robust problem-solving ability may manifest as more stable expressive patterns. While RCScore does not directly measure internal cognitive processes, the consistency it quantifies in response characteristics, especially when linked to accuracy on reasoning-intensive tasks, offers an empirically grounded proxy for the stability and potential depth of a model's approach. Thus, RCScore moves beyond simple performance metrics, offering diagnostic insights into the reliability and predictability of LLMs in diverse interaction contexts.

F Future Work

A key avenue for future work is to leverage RCScore, especially CRS, as a more direct indicator of robust reasoning capabilities. The hypothesis is that models demonstrating high CRS on complex problems across varied instruction styles are more likely to possess genuine, transferable reasoning skills, as opposed to relying on superficial pattern matching. The observed correlation between CRS and accuracy supports investigating RCScore as a complementary signal for assessing the depth and reliability of a model's problem-solving faculties.

Further applications include using RCScore's dimensional feedback to refine model architectures and training objectives to improve specific aspects of response consistency. Additionally, RCScore can inform optimized prompt engineering practices by identifying instruction styles that elicit not only accurate but also consistently well-formed responses. The framework's principles could also be extended to diverse generative tasks and modalities, and to cross-lingual style sensitivity analyses, ultimately fostering LLMs that are more adaptable and reliable.

Reference	Comparison	Struct	Lexi	Cohe	RCScore	Similarity
The analysis of algorithmic complexity requires a systematic approach. First, we must identify the basic operations within the algorithm. Then, we need to calculate how the number of these operations grows with respect to input size. Many mathematicians confirm that *asymptotic notation* is necessary to express these growth rates. However, there remains debate about the most accurate methods for analyzing algorithms with multiple variables while maintaining practical relevance. Finally, we should consider the trade-offs between time complexity and space complexity when implementing these algorithms in practice.	**Similar 1:** The evaluation of mathematical proofs requires a systematic approach. First, we must identify the basic assumptions within the theorem. Then, we need to calculate how the logical inferences build with respect to axiom use. Many mathematicians confirm that *formal notation* is necessary to express these logical steps. However, there remains debate about the most accurate methods for constructing proofs with multiple lemmas while maintaining intuitive clarity. Finally, we should consider the trade-offs between proof elegance and proof length when presenting these theorems in practice.	0.4163	0.6696	0.8386	0.6415	High
	Similar 2: The formulation of statistical models requires a systematic approach. First, we must identify the basic variables within the dataset. Then, we need to calculate how the relationships between these variables change with respect to sample size. Many mathematicians confirm that *probabilistic notation* is necessary to express these statistical relationships. However, there remains debate about the most accurate methods for building models with multiple parameters while maintaining interpretability. Finally, we should consider the trade-offs between model accuracy and model complexity when applying these techniques in practice.	0.4289	0.7161	0.8474	0.6642	High
	Similar 3: The development of optimization algorithms requires a systematic approach. First, we must identify the basic constraints within the problem. Then, we need to calculate how the solution space changes with respect to parameter values. Many mathematicians confirm that *vector notation* is necessary to express these feasible regions. However, there remains debate about the most accurate methods for solving problems with multiple objectives while maintaining computational efficiency. Finally, we should consider the trade-offs between convergence speed and solution quality when implementing these methods in practice.	0.4225	0.7116	0.8422	0.6588	High

Table 5: RCScore for Reference Paragraph vs. Similar Comparison Paragraphs. This table illustrates RCScore quantification for paragraphs with high stylistic similarity to a reference text, with visual cues for similar concepts (green text for similar concepts, blue text for reference's key phrases).

Reference	Comparison	Struct	Lexi	Cohe	RCScore	Similarity
The analysis of algorithmic complexity requires a systematic approach. First, we must identify the basic operations within the algorithm. Then, we need to calculate how the number of these operations grows with respect to input size. Many mathematicians confirm that *asymptotic notation* is necessary to express these growth rates. However, there remains debate about the most accurate methods for analyzing algorithms with multiple variables while maintaining practical relevance. Finally, we should consider the trade-offs between time complexity and space complexity when implementing these algorithms in practice.	**Different 1:** OMG! Machine learning is SOOO amazing! I tried a neural network yesterday and WOW - it actually worked! Sort of... It got like 85% accuracy which isn't bad for my first try, right?! I think I'm totally gonna be an AI expert now! The code was pretty simple once I figured out all those weird tensor thingies. This stuff is way cooler than boring old algorithms. #AI #MachineLearning #Future	0.0813	0.0141	0.7298	0.2751	Low
	Different 2: Consider a set X with the discrete topology, where every subset of X is open. If X is an infinite set, such as the set of natural numbers N, then X is not compact. This is because the collection of all singleton sets {x} for x in X forms an open cover of X. However, no finite subcollection of this open cover can cover X, as each singleton set only covers one point. Therefore, by the definition of compactness (a space is compact if every open cover has a finite subcover), an infinite set with the discrete topology is not compact. This demonstrates a fundamental concept in point-set topology.	0.2751	0.1428	0.7821	0.4000	Low
	Different 3: To solve this quadratic equation, first, bring all terms to one side to get the form ax^2 + bx + c = 0. Then, identify the coefficients a, b, and c. The solutions for x can be found using the quadratic formula: x = (-b ± sqrt(b^2 - 4ac)) / (2a). Remember that the discriminant, Delta = b^2 - 4ac, determines the nature of the roots. If Delta > 0, there are two distinct real roots. If Delta = 0, there is one real root (a repeated root). If Delta < 0, there are two complex conjugate roots. This method is a cornerstone of algebra and is widely applicable in various scientific and engineering problems.	0.2105	0.1519	0.7953	0.3859	Low

Table 6: Comparison with Semantically and Stylistically Different Paragraphs, highlighting contrasting elements (red text) against the reference's structure (blue text for reference's key phrases).

Model	AIME		MATH-500		GPQA-Diamond		GSM8K	
	Beam	**Greedy**	**Beam**	**Greedy**	**Beam**	**Greedy**	**Beam**	**Greedy**
Gemma 3-4B	6.7	**10.8**	**65.7**	65.4	**3.5**	2.8	**88.6**	85.7
Gemma 3-12B	**22.5**	3.3	**72.1**	52.6	**5.4**	5.2	**92.0**	90.3
Gemma 3-27B	27.5	27.5	**76.9**	76.9	**6.5**	5.9	94.4	**94.4**
LLaMA 3.2-3B	5.0	**11.7**	41.1	**43.0**	**4.1**	3.7	76.2	**79.2**
LLaMA 3.1-8B	6.7	**7.5**	43.8	**44.2**	3.1	**3.8**	83.2	**84.9**
LLaMA 3.3-70B	23.3	**26.7**	63.5	**63.6**	6.5	**7.4**	90.7	**93.4**
Qwen 2.5-3B	4.2	**8.3**	55.0	**57.0**	**2.9**	2.3	82.5	**84.6**
Qwen 2.5-7B	11.7	**15.9**	62.6	**65.8**	**4.4**	2.1	86.0	**89.5**
Qwen 2.5-32B	15.0	**16.7**	69.1	**70.9**	3.4	**5.7**	92.6	**92.7**
Qwen 2.5-72B	**22.5**	21.7	70.7	**71.1**	**5.3**	5.1	92.7	**92.9**
Average	14.5	**15.0**	62.1	**61.1**	4.5	4.4	87.9	**88.8**

Table 7: Average accuracy comparison between Beam Search (temperature = 1.0) and Greedy Search (temperature = 0.0) across four benchmarks. Each value represents the average of S1-S4 instruction style scores. Bold numbers indicate the better-performing decoding strategy for each model and benchmark.