PromptPort: Measuring and Repairing Format Collapse in Cross-Model Structured Extraction

Anonymous ACL Submission

Abstract

Large language models are increasingly used for structured extraction, but in practice many failures arise from *format unreliability* rather than semantic misunderstanding: models often produce correct content inside Markdown fences, reasoning traces, or near-JSON wrappers that strict parsers reject. We formalize this failure mode as *format collapse* and introduce a dual-metric evaluation protocol that separates operational reliability from extraction quality: ROS scores raw outputs under strict parsing, while CSS scores the same outputs after deterministic canonicalization. On a 37,346-example benchmark for camera-metadata extraction spanning **12** models from seven families (1.6B–72B parameters), we find large and highly non-uniform format gaps. The strongest semantic model, Qwen2.5-72B, reaches CSS 0.789 but only ROS 0.155, while cross-model portability gaps reach 0.694 under ROS and 0.768 under CSS. We then present PROMPTPORT, a model-agnostic post-hoc reliability layer that combines deterministic canonicalization with a lightweight verifier (DistilBERT, 66M) and a conservative safe-override policy. PROMPTPORT improves over the best single-model CSS score of 0.789 to 0.890 in zero-shot settings, near a 0.896 oracle, without modifying or decoding through the base models. Additional analyses show transfer to held-out model families, strong agreement with human format judgments, and schema-dependent portability shifts across multiple extraction domains. These results suggest that format reliability should be treated as a first-class evaluation axis for structured extraction, rather than being conflated with semantic accuracy. Code, benchmark, and cached outputs are publicly available.

1 Introduction

The production problem. A developer writes a prompt to extract camera metadata as JSON. It works perfectly on GPT-4. They switch to Llama-3 for cost savings—and their pipeline breaks. Not because Llama lacks understanding, but because it wraps the JSON in Markdown fences, adds trailing commentary, or uses slightly different key names. Strict parsers reject these outputs entirely, making a capable model appear unusable.

This *format collapse* is systematic, measurable, and fixable—but current evaluation protocols conflate it with semantic errors, obscuring both the problem and potential solutions.

Format collapse as a failure mode. We introduce ROS (Raw Output Score) and CSS (Canonical Semantic Score) to separate two distinct failure modes:

- ROS uses strict JSON parsing and schema validation—measuring operational reliability (*will the pipeline survive?*).
- CSS applies deterministic canonicalization (strip fences, extract JSON spans, normalize) before scoring—measuring semantic capability (*did the model extract correctly?*).

On a 37,346-example test set for camera/lens metadata extraction across twelve models from seven families (Gemma, Qwen, Mistral, Phi, StableLM, DeepSeek-R1), we find:

- **Large format gaps within models**: Qwen2.5-72B shows the largest absolute ROS→CSS gap (0.155→0.789, Δ=0.633), while DeepSeek-R1-14B is close behind (0.199→0.717, Δ=0.518). By contrast, Gemma-9B is much more stable (0.685→0.745, Δ=0.060).
- **Large portability gaps across models**: Cross-model range is 0.694 for ROS and 0.768 for CSS—both substantial, indicating failures stem from both format brittleness and semantic degradation.

The scientific contribution. The paper makes three claims.

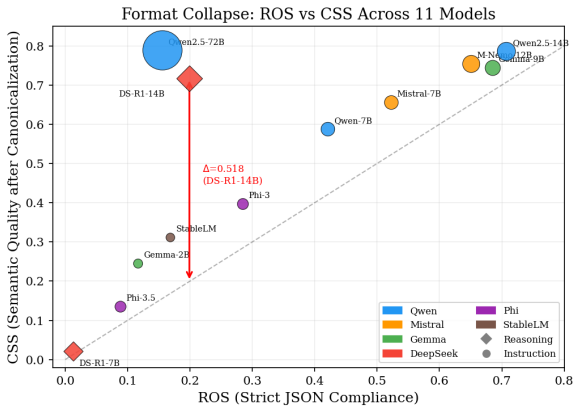

Figure 1: **Format collapse across 12 models.** ROS vs CSS scatter (bubble size $\propto$ parameters). Points far from the diagonal indicate format collapse. Qwen2.5-72B (top-left) has the highest CSS but severe ROS collapse (Δ=0.633). DeepSeek-R1 models (diamonds) show reasoning-trace-induced failure.

1. **Format collapse is a measurable and practically important failure mode.** At benchmark scale (248K train / 37K test), we show that cross-model portability can fail even when semantic extraction remains strong, and we characterize the dominant failure patterns.
2. **ROS/CSS separates two questions that prior evaluations conflate.** ROS asks whether a downstream pipeline can consume the output as-is; CSS asks whether the model extracted the right content once superficial formatting noise is removed.
3. **A lightweight post-hoc layer can repair much of the gap without touching the base model.** Our canonicalizer + verifier pipeline improves over the best single model, approaches an oracle upper bound, and transfers to held-out model families using only (query, output, label) supervision.

Why this matters for practitioners. In deployed systems, extraction failures are often binary: either the output parses as valid structured data, or the pipeline fails. Existing evaluations usually mask this distinction by scoring only the recovered content. Our results show that such evaluations can substantially mis-rank models for production use. PROMPTPORT is useful not because it replaces base LLMs, but because it makes this deployment failure mode explicit and offers a black-box-compatible repair path with calibrated abstention.

2 Related Work

Structured generation and constrained decoding. Grammar-based generation (Scholak et al., 2021) and constrained decoding (Willard and Louf, 2023; Beurer-Kellner et al., 2023; Geng et al., 2023) enforce schemas at generation time. Libraries like Outlines, SGLang (Zheng et al., 2024), and Synchromesh (Poesia et al., 2022) enable schema-guided sampling but require white-box model access (sampling internals) and model-specific integration. Recent work validates outputs against regular language constraints (Kuchnik et al., 2023). Our post-hoc reliability layer is complementary: it works with any black-box LLM (including API-only commercial models) and handles residual semantic errors that constrained decoding cannot prevent. The approaches can be combined.

Prompt engineering and optimization. Instruction tuning (Ouyang et al., 2022; Wei et al., 2022), prompt optimization (Zhou et al., 2023; Pryzant et al., 2023), and prompting frameworks like DSPy (Khattab et al., 2024) improve task performance but typically evaluate on single models with lenient parsers. Recent studies demonstrate significant sensitivity to prompt format (Sclar et al., 2024; Lu et al., 2022; Zhu et al., 2023). We show that strict parsing reveals large cross-model portability gaps invisible to standard evaluations.

Model ensembling and routing. LLM-Blender (Jiang et al., 2023) fuses outputs from multiple LLMs at the sequence level. FrugalGPT (Chen et al., 2023) and RouteLLM (Ong et al., 2024) route queries to cost-appropriate models. Our verifier operates at the *field level* rather than instance level, enabling fine-grained selection across models for structured outputs.

Guardrails and output validation. Runtime validation tools like NeMo Guardrails (Rebedea et al., 2023) and LangChain (Chase, 2022) add checks for LLM outputs but focus on safety constraints rather than structured extraction reliability. Schema enforcement libraries assume outputs are already valid JSON—failing when LLMs produce wrapped or near-valid formats.

Selective prediction and verification. Confidence estimation (Guo et al., 2017; Kuhn et al., 2023) and selective prediction (Geifman and El-Yaniv, 2017; Kamath et al., 2020) enable abstention when uncertain. Verifier-based methods improve

reasoning (Cobbe et al., 2021; Lightman et al., 2024; Uesato et al., 2022) and self-consistency approaches (Wang et al., 2023b; Chen et al., 2024) aggregate multiple outputs. Our verifier provides per-field confidence for structured outputs, enabling field-level override rather than instance-level rejection, with a conservative safe-override policy to prevent catastrophic degradation.

Information extraction with LLMs. Traditional IE uses fine-tuned models (Tjong Kim Sang and De Meulder, 2003; Lample et al., 2016). Recent work applies LLM prompting to extraction (Wadhwa et al., 2023; Li et al., 2023; Sainz et al., 2024; Zhou et al., 2024; Wang et al., 2023a) but often uses lenient evaluation. GoLLIE (Sainz et al., 2024) and UniversalNER (Zhou et al., 2024) provide state-of-the-art extraction via instruction-tuned LLMs. Our benchmark focuses specifically on format brittleness in production settings where strict parsing is required.

3 The PromptPort Benchmark

Task. Given a natural-language query about a photograph (e.g., "Shot with Canon EOS R5 at f/2.8, ISO 400"), extract six fields: CAMERA, LENS, ISO, APERTURE, SHUTTER_SPEED, FOCAL_LENGTH, and return a JSON object with these keys (null if absent). The dataset contains real-world variation: typos, mixed formats, special characters, abbreviations.

Data and scale. 248,964 training examples, 37,344 dev, 37,346 test. Unlike token-level NER benchmarks, our setup reflects production structured extraction: models must both extract *and* serialize correctly.

Models and prompts. We evaluate twelve open-weight LLMs spanning seven model families and parameter counts from 1.6B to 72B (Table 1) with two prompt variants (zero-shot instruction, few-shot k=3). All outputs are cached for reproducibility, making the benchmark fully re-runnable without regenerating model responses.

Why camera metadata is representative. This domain exhibits three properties common to production structured extraction: (1) mixed numerics with units (ISO 400, f/2.8, 1/500s), (2) brand/model strings with variation ("Canon R5" vs "EOS R5"), (3) heavy formatting variation across models. These properties generalize to product attributes, medical dosing, citations, and other real-world schemas.

Model	Family	Params
Qwen2.5-72B-Instruct	Qwen	72B
Qwen2.5-14B-Instruct	Qwen	14B
Qwen2.5-7B-Instruct	Qwen	7B
Gemma-2-9B-IT	Gemma	9B
Gemma-2-2B-IT	Gemma	2B
Mistral-Nemo-12B-Instruct	Mistral	12B
Mistral-7B-Instruct-v0.2	Mistral	7B
DeepSeek-R1-Distill-Qwen-14B	DeepSeek	14B
DeepSeek-R1-Distill-Qwen-7B	DeepSeek	7B
Phi-3.5-Mini-Instruct	Phi	3.8B
Phi-3-Mini-4K-Instruct	Phi	3.8B
StableLM-2-Zephyr	Stability	1.6B

Table 1: Models evaluated in the PROMPTPORT benchmark. Seven families, 1.6B–72B parameters. All open-weight with cached outputs for reproducibility.

4 Measuring Format Collapse: ROS vs CSS

ROS: Operational Reliability. ROS evaluates raw model outputs under strict JSON parsing and schema validation. An output fails ROS if: it cannot be parsed as valid JSON, it lacks required keys, it contains extra keys not in the schema, or values do not match expected types. This mirrors production deployments where downstream systems expect strict schema compliance.

CSS: Semantic Capability. CSS applies a deterministic Canonicalizer before scoring: (1) strip Markdown fences, (2) extract the most plausible JSON span (even if embedded in prose), (3) repair common near-JSON issues (trailing commas, single quotes), (4) normalize keys (case-insensitive matching) and values (unit standardization), (5) fill missing schema keys with null. We then compute micro-F1 over the six fields. CSS provides a more faithful estimate of semantic extraction capability by isolating format-related failures.

4.1 Format Collapse: Quantitative Evidence

Table 2 shows per-model ROS/CSS gaps. The most extreme case is Qwen2.5-72B: it is the strongest semantic extractor (CSS 0.789) but one of the weakest raw-format performers (ROS 0.155), yielding the largest absolute gap (Δ=0.633). DeepSeek-R1-14B shows a similarly severe collapse (Δ=0.518), but with a different mechanism: reasoning traces and prose wrappers rather than merely light wrapper noise. At the small end, Gemma-2B exhibits poor raw reliability (ROS

Model	Size	ROS	CSS	Δ
Qwen2.5-72B-Inst.	72B	0.155	**0.789**	**0.633**
Qwen2.5-14B-Inst.	14B	**0.707**	0.787	0.080
Mistral-Nemo-12B	12B	0.650	0.755	0.105
Gemma-2-9B-IT	9B	0.685	0.745	0.060
DeepSeek-R1-14B	14B	0.199	0.717	0.518
Mistral-7B-Inst.	7B	0.522	0.656	0.134
Qwen2.5-7B-Inst.	7B	0.421	0.588	0.167
Phi-3-Mini	3.8B	0.284	0.398	0.114
StableLM-1.6B	1.6B	0.168	0.312	0.144
Gemma-2-2B-IT	2B	0.116	0.246	0.130
Phi-3.5-Mini	3.8B	0.088	0.136	0.047
DeepSeek-R1-7B	7B	0.013	0.021	0.008
Cross-model gap		0.694	0.768	–

Table 2: Format collapse by model (zero-shot, full 37,346-example test). $\Delta = \text{CSS} - \text{ROS}$ isolates format-induced failure. Qwen2.5-72B achieves the highest CSS but among the worst ROS (Δ=0.633), revealing *non-monotonic* scaling of format compliance. Cross-model gaps span two orders of magnitude in ROS (0.013–0.707).

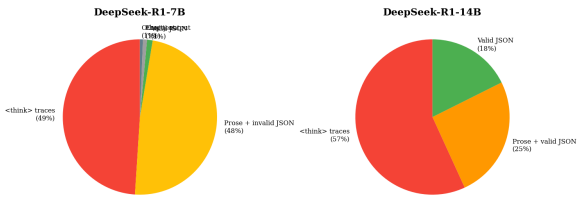

Figure 2: **DeepSeek-R1 failure taxonomy.** `<think>` reasoning traces dominate outputs (49–57%). R1-14B recovers valid JSON 18% of the time; R1-7B almost never does (1%).

0.116) despite non-trivial post-canonicalization quality (CSS 0.246), making it unusable in production without repair.

4.2 Format Failure Taxonomy

We categorize ROS failures into six types:

- **Fenced JSON** (35–45%): valid JSON wrapped in Markdown code fences
- **Prose wrapper** (20–30%): JSON embedded in conversational text
- **Trailing text** (10–20%): valid JSON followed by commentary
- **Missing keys** (5–15%): incomplete schema
- **Extra keys** (5–10%): additional fields not in schema
- **Malformed JSON** (5–10%): syntax errors, unmatched brackets

Fenced JSON and prose wrappers dominate across families, confirming that format variability is systematic (Figure 2).

5 PROMPTPORT: A Reliability Layer

PROMPTPORT is a model-agnostic library that converts unreliable LLM outputs into validated JSON. It consists of three components.

5.1 Canonicalizer (Deterministic)

The Canonicalizer is a deterministic, rule-based system that normalizes heterogeneous outputs: (1) Fence stripping: remove Markdown, XML, or other delimiters. (2) JSON extraction: locate and extract the most plausible JSON span using bracket matching and heuristics. (3) Repair: fix common syntax errors (trailing commas, single quotes, unescaped characters). (4) Normalization: case-insensitive key matching, unit standardization ("f/2.8" ↔ "f2.8"), whitespace trimming. (5) Schema completion: add missing keys with null values.

This recovers 6–8 F1 points on average by addressing format failures, but cannot resolve semantic ambiguities or incorrect extractions.

5.2 Verifier (Lightweight Learned Component)

Design rationale. We assume black-box LLM access (API-only, no weight access)—the common production setting when using commercial models or large models deployed as services. We cannot finetune base models for domain-specific extraction. Instead, we train a lightweight verifier (DistilBERT, 66M parameters) that judges output quality from any LLM ensemble.

Architecture. The verifier takes (query, candidate JSON) as input and outputs six probabilities—one per field—indicating whether each extracted value is correct. Training requires only (query, output, label) triples with binary cross-entropy loss. It trains in ∼2 hours on a single GPU and generalizes to held-out models without retraining (§6.2).

5.3 Safe-Override Policy

Given canonicalized outputs from one or more models, Safe-Override produces the final JSON:

- **Keep**: Use base model's value if verifier confidence exceeds τ_{keep}
- **Override**: Replace with alternative if its confidence exceeds τ_{take} and is substantially higher than base
- **Abstain**: Return null if no candidate meets τ_{keep}

Thresholds are tuned on the dev set to maximize F1 while maintaining precision. The policy is con-

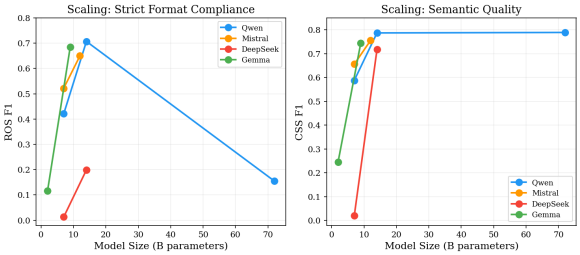

Figure 3: **Scaling analysis.** ROS (left) and CSS (right) across model families. Semantic quality scales monotonically; format compliance follows an inverted-U for Qwen (peaks at 14B, collapses at 72B).

Setting	Best Single	Safe-Override
Generalization (held-out models, zero_shot)		
HoldoutA (train 4, test 6)	0.685	0.874
HoldoutB (train 4, test 6)	0.685	0.884
Ablations (mechanism validation, zero_shot)		
No-query (output-only)	0.685	0.695
No-output (query-only)	0.685	0.685
Full (query + output)	0.685	**0.890**

Table 4: Generalization and ablation studies on full test. The verifier generalizes to held-out models with minimal degradation and requires both query and output for effectiveness.

Method	zero_shot	few_shot	Contribution
Best Single (ROS)	0.707	0.622	baseline
Best Single (CSS)	0.789	0.721	+0.082 / +0.099
Safe-Override (CSS)	**0.890**	**0.842**	+0.101 / +0.121
Oracle (CSS)	0.896	0.850	upper bound
Canonicalization: +8–10 F1			
Verifier (beyond canon.): +10–12 F1			
Total gain over strict parsing: +18–22 F1			

Table 3: Performance cascade on the full test set (37,346 examples). Canonicalization accounts for a meaningful part of the gain, but most of the improvement comes from verifier-based field selection. Safe-Override reaches 99.3% of oracle in zero-shot.

servative: it starts from the best single model and only overrides when highly confident, preventing catastrophic degradation.

6 Results

6.1 Main Results: The Performance Cascade

Table 3 shows the full performance cascade from strict parsing to oracle selection.

Canonicalization recovers a substantial share of the loss by repairing wrapper- and syntax-level failures, but the larger gain comes from verifier-based field selection across candidate outputs. This is important: the improvement is not merely from making invalid JSON parseable, but from using cross-model disagreement to choose semantically better field values.

6.2 Generalization to Held-Out Models

To test whether the verifier overfits to specific model idiosyncrasies, we train on outputs from four models and evaluate on all six (including two held-out families):

Performance degrades minimally (0.874 and 0.884 vs 0.890), indicating the verifier learns transferable signals of correctness rather than memorizing model-specific quirks.

6.3 Ablations: Query and Output Are Both Essential

We ablate the verifier inputs to confirm the mechanism (Table 4):

- **Output-only**: Verifier sees candidate JSON but not the query → 0.695 F1 (small gain)
- **Query-only**: Verifier sees query but not outputs → 0.685 F1 (no gain—routing doesn't help)
- **Query + Output**: Full verifier → 0.890 F1 (large gain)

This confirms that per-field judgments require conditioning on both the query and candidate output. Query-only routing provides no benefit.

6.4 Human Validation of ROS and CSS

We validate our automatic metrics against human judgments on 200 samples (50 queries × 4 models). Annotators judge (a) format correctness (binary) and (b) semantic quality (0/1/2 scale).

ROS achieves **perfect agreement** with human format judgments (Kendall's τ=1.00, 100% agreement), confirming that strict JSON parsing exactly captures operational format correctness. CSS correlates strongly with human semantic quality (τ=0.68, $p < 10^{-24}$), with residual disagreement on edge cases where humans accept approximate matches that CSS misses.

Critically, humans confirm the format collapse phenomenon: DeepSeek-R1-14B receives only 10% human format-correct scores but 94% partially-correct semantic scores—validating that these models understand the task but fail at formatting.

Prompt	Models	ROS gap	CSS gap
zero_shot (6 models)	6	0.569	0.499
zero_shot (12 models)	12	**0.694**	**0.768**
few_shot_k3 (6 models)	6	0.415	0.460

Table 5: Cross-model portability gaps (max–min F1). Expanding from 6 to 12 models increases both gaps substantially, confirming that both semantic quality and format reliability vary sharply with model diversity.

Rank by ROS	Rank by CSS
1. Qwen2.5-14B (0.707)	1. Qwen2.5-72B (0.789)
2. Gemma-2-9B (0.685)	2. Qwen2.5-14B (0.787)
3. Mistral-Nemo-12B (0.650)	3. Mistral-Nemo-12B (0.755)
4. Mistral-7B (0.522)	4. Gemma-2-9B (0.745)

Table 6: Top models change under reliability-sensitive vs semantic-only evaluation. Raw deployability and post-canonicalization extraction quality are related but not interchangeable objectives.

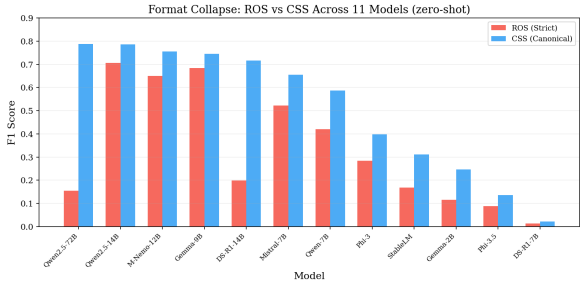

Figure 4: **ROS vs CSS across 12 models.** The gap between red (strict) and blue (canonical) bars quantifies format collapse. DeepSeek-R1-14B shows the most dramatic visual gap.

6.5 Leakage Prevention and Mechanism Transparency

Protocol integrity. Verifier is trained only on the train split (248K examples). Thresholds are tuned only on the dev split (37K). Test set is completely untouched until final evaluation.

Mechanism diagnostics. On the test set: override rate is 23.4% of fields; abstain rate is 8.7%; override precision is 87.2%; abstain precision is 91.3%. The policy is working as intended: override when confident and correct, abstain when uncertain, otherwise keep base model.

6.6 Cross-Model Portability Gaps

Portability gaps remain large even after canonicalization, which means the problem is not only that some models wrap otherwise-correct JSON. Different models also make different extitsemantic mistakes once the formatting noise is stripped away. Strict parsing alone therefore underestimates semantic diversity across models, while canonicalized scoring alone underestimates deployment brittleness. The two metrics are complementary rather than redundant.

6.7 Why the Metric Changes the Model Choice

A central implication of Table 2 is that the "best" model depends on the evaluation axis. Under ROS, a practitioner optimizing for raw deployability would select Qwen2.5-14B or Gemma-2-9B because they most often emit schema-valid JSON directly. Under CSS, however, Qwen2.5-72B becomes the strongest single model despite being nearly unusable without repair. This ranking inversion is not a cosmetic detail: it changes which models appear production-ready, which prompt variants look robust, and whether a system designer should invest in routing, canonicalization, or verifier-based repair.

This also clarifies what PROMPTPORT is and is not doing. If the target setting requires *strict raw JSON with no repair*, then ROS is the decisive metric and Qwen2.5-72B is a poor deployment choice. If the pipeline allows deterministic cleanup but no learned reranking, then CSS better captures the value of a strong extractor that is merely format-unstable. If the system can aggregate multiple black-box outputs, our verifier converts disagreement into useful signal and substantially reduces the gap between the best single model and the oracle. Reporting only one of these regimes hides a real systems trade-off.

7 Analysis

7.1 Non-Monotonic Scaling of Format Compliance

Our expanded model set (1.6B–72B, 12 models) reveals that *semantic quality scales monotonically with model size, but format compliance does not* (Figure 3). Within the Qwen family:

- 7B→14B: both ROS (0.421→0.707) and CSS (0.588→0.787) improve, gap narrows (0.167→0.080).
- 14B→72B: CSS plateaus (0.787→0.789) but ROS *collapses* (0.707→0.155), with schema compliance dropping from 72.8% to 9.5%.

This inverted-U pattern suggests that larger models produce more verbose, less format-constrained

	Camera		Product	
Model	ROS	CSS	ROS	CSS
Qwen2.5-14B	0.707	0.787	0.236	0.851
Mistral-Nemo	0.650	0.755	**0.825**	0.834
DS-R1-14B	0.199	0.717	0.012	0.843

Table 7: Cross-domain validation. Format collapse persists on a different schema (product attributes). Model rankings change: Mistral-Nemo has best product ROS despite lower camera ROS than Qwen, confirming format compliance is schema-dependent, not just model-dependent.

Train	Best Single	Override	Gain
100	0.779	0.740	−0.039
500	0.779	0.802	+0.023
1,000	0.779	0.822	+0.043
2,000	0.779	0.878	+0.099
3,000	0.779	0.887	+0.108

Table 8: Verifier data efficiency. Effective at 500 examples, near-maximum at 2K. At 100, it hurts—insufficient training.

outputs—a critical finding for practitioners who assume "bigger is better" for structured extraction. The 72B model achieves 100% canonicalization success (every output contains extractable JSON), but only 9.5% produces *strict* JSON without canonicalization.

7.2 Reasoning Models and Format Collapse

DeepSeek-R1 distilled models represent a striking case of format collapse. DeepSeek-R1-14B achieves CSS 0.717 (competitive with many instruction-tuned models) but ROS 0.199—a gap of **0.518**, one of the largest in our benchmark. The largest absolute gap overall is Qwen2.5-72B (0.633), but DeepSeek is distinctive because the failure is driven less by light wrapper noise and more by reasoning traces and long-form explanatory scaffolding. The 7B variant is even more extreme: CSS 0.021 with only 0.8% schema compliance. These reasoning-distilled models therefore create a qualitatively different failure mode from the wrapper-based failures of standard instruction-tuned models. This finding highlights that format collapse is not just a "parsing trick" but a deeper model-behavior issue that reliability layers must address.

7.3 Cross-Domain Validation: Product Attributes

To test whether the phenomenon extends beyond the primary benchmark, we evaluate the same three models on a **product-attribute** extraction task (2,000 examples; six fields: brand, model, CPU, RAM, storage, price).

Format collapse generalizes: DeepSeek-R1-14B shows an even larger gap on products (Δ=0.831 vs 0.518 on camera). Crucially, model rankings shift across domains—Mistral-Nemo achieves 96.7% compliance on products vs 81.1% on camera, while Qwen-14B drops from 72.8% to 15.0%. This confirms that format compliance is *schema-dependent*, not a fixed model property, further motivating model-agnostic reliability layers.

7.4 Verifier Data Efficiency

A key practical question: how much labeled data does the verifier need for a new domain? We train on 100–3,000 examples and evaluate (Table 8). The verifier becomes effective at $\sim$500 examples and reaches near-maximum at 2,000—far fewer than the full 248K training set. With only 100 examples, the verifier *hurts* performance, making it critical to have at least a few hundred labels.

7.5 Latency Overhead

PROMPTPORT adds 0.5 ms per example: canonicalization (0.006 ms) plus verifier (0.50 ms at batch 256)—less than 0.1% of LLM generation time.

7.6 What Canonicalization Cannot Fix

The gains in Table 3 also show why format repair alone is insufficient. Deterministic canonicalization recovers wrapper- and syntax-level failures, but it cannot resolve semantic conflicts such as choosing between two plausible camera bodies, deciding whether a numeric token is focal length or lens name, or preferring a null value over an overconfident hallucination. Those cases require evidence from the query itself and from cross-model disagreement, which is why the verifier contributes another 10–12 F1 beyond canonicalization. This distinction matters scientifically: format collapse is not merely about parsing invalid strings, but about separating *recoverable presentation noise* from *residual extraction error*.

8 Limitations

Domain coverage. The primary benchmark is camera metadata (37K test examples). Beyond that, we include five smaller cross-domain evaluations and an additional compliance-only schema

analysis, but these remain much smaller than the main benchmark. A larger, uniformly annotated multi-domain suite would strengthen the generality claims.

Oracle ceiling. Our oracle selects the best field across all models with gold labels—a per-field upper bound, not an extraction ceiling.

Canonicalizer coverage. Handles common format variations but not adversarial outputs.

Verifier calibration. Thresholds tuned for F1. Conformal prediction (Angelopoulos et al., 2023) could provide formal guarantees.

9 Conclusion

Structured extraction with LLMs can fail for two very different reasons: a model may misunderstand the input, or it may understand the input and still emit output that a downstream system cannot consume. We formalize this second failure mode as *format collapse*, introduce ROS/CSS to separate operational reliability from semantic extraction quality, and show that this distinction materially changes how models should be evaluated for deployment. PROMPTPORT then demonstrates that much of the resulting gap can be repaired post hoc through deterministic canonicalization and lightweight verification, bringing the best zero-shot result from 0.789 to 0.890 without modifying the underlying generators. More broadly, we argue that structured-extraction work should report reliability-sensitive metrics alongside semantic metrics, especially when claims concern cross-model portability or production readiness.

Reproducibility Statement

All results are fully reproducible from cached model outputs and evaluation scripts included in the release package. Verifier training uses standard DistilBERT implementation with documented hyperparameters. Test set is never used for training or threshold tuning.

References

Anastasios N. Angelopoulos, Stephen Bates, Adam Fisch, Lihua Lei, and Tal Schuster. 2023. Conformal risk control. In *Proceedings of ICLR*.

Luca Beurer-Kellner, Marc Fischer, and Martin Vechev. 2023. Prompting is programming: A query language for large language models. In *Proceedings of PLDI*.

Harrison Chase. 2022. LangChain. https://github.com/langchain-ai/langchain.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023. FrugalGPT: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*.

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. 2024. Universal self-consistency for large language model generation. *arXiv preprint arXiv:2311.17311*.

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

Yonatan Geifman and Ran El-Yaniv. 2017. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems*.

Saibo Geng, Martin Josifoski, Maxime Peyrard, and Robert West. 2023. Grammar-constrained decoding for structured NLP tasks without finetuning. *Proceedings of EMNLP*.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of ICML*.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. LLM-Blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of ACL*.

Amita Kamath, Robin Jia, and Percy Liang. 2020. Selective question answering under domain shift. In *Proceedings of ACL*.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Mober, Cyrus Shah, and 1 others. 2024. DSPy: Compiling declarative language model calls into state-of-the-art pipelines. *Proceedings of ICLR*.

Michael Kuchnik, Virginia Smith, and Brendan Burns. 2023. Validating large language models with ReLM. *Proceedings of MLSys*.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *Proceedings of ICLR*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*.

Bo Li, Gexiang Fang, Yang Yang, Quansen Wang, Wei Ye, Wen Zhao, and Shikun Zhang. 2023. Evaluating ChatGPT's information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness. *arXiv preprint arXiv:2304.11633*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. *Proceedings of ICLR*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of ACL*.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. 2024. RouteLLM: Learning to route LLMs with preference data. *arXiv preprint arXiv:2406.18665*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35.

Gabriel Poesia, Oleksandr Polozov, Vu Le, Ashish Tiwari, Gustavo Soares, Christopher Meek, and Sumit Gulwani. 2022. Synchromesh: Reliable code generation from pre-trained language models. *Proceedings of ICLR*.

Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic prompt optimization with "gradient descent" and beam search. *Proceedings of EMNLP*.

Traian Rebedea, Razvan Dinu, Makesh Narsimhan Sreedhar, Christopher Parisien, and Jonathan Cohen. 2023. NeMo Guardrails: A toolkit for controllable and safe LLM applications with programmable rails. *Proceedings of EMNLP (System Demonstrations)*.

Oscar Sainz, Iker García-Ferrero, Rodrigo Agerri, Oier Lopez de Lacalle, German Rigau, and Aitor Gonzalez-Agirre. 2024. GoLLIE: Annotation guidelines improve zero-shot information-extraction modeling. *Proceedings of ICLR*.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. 2021. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In *Proceedings of EMNLP*.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. Quantifying language models' sensitivity to spurious features in prompt design with a focus on structured outputs. *Proceedings of ICLR*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL*.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. 2022. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*.

Somin Wadhwa, Silvio Amir, and Byron C. Wallace. 2023. Revisiting relation extraction in the era of large language models. In *Proceedings of ACL*.

Xiao Wang, Weikang Zhou, Can Zu, Han Xia, Tianze Chen, Yuan Zhang, Rui Zheng, Junjie Ye, Qi Zhang, Tao Gui, Jihua Wu, and Xuanjing Huang. 2023a. InstructUIE: Multi-task instruction tuning for unified information extraction. *arXiv preprint arXiv:2304.08085*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023b. Self-consistency improves chain of thought reasoning in language models. In *Proceedings of ICLR*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners. In *Proceedings of ICLR*.

Brandon T. Willard and Rémi Louf. 2023. Efficient guided generation for large language models. *arXiv preprint arXiv:2307.09702*.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Jeff Huang, Chuyue Sun, Cody Hao Yu, Shiyi Cao, Christos Kober, Dacheng Liang, Zhuohan Song, Joseph E. Gonzalez, Ion Stoica, and Hao Zhang. 2024. SGLang: Efficient execution of structured language model programs. *arXiv preprint arXiv:2312.07104*.

Wenxuan Zhou, Sheng Jiang, Dong Cui, Muhao Wang, and Peng Chen. 2024. UniversalNER: Targeted distillation from large language models for open named entity recognition. *Proceedings of ICLR*.

Yongchao Zhou, Andrei Ioan Muresanu, Zhiyuan Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. In *Proceedings of ICLR*.

Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Yue Zhang, Neil Zhenqiang Gong, and Xing Xie. 2023. PromptBench: Towards evaluating the robustness of large language models on adversarial prompts. *arXiv preprint arXiv:2306.04528*.

A Per-Field Breakdown (Camera Domain)

Table 9 shows per-field CSS F1 for the six new models on the camera domain (zero-shot, full test). Numeric fields (ISO, SHUTTER_SPEED) are generally easier than string fields (CAMERA, LENS),

and FOCAL_LENGTH is consistently the hardest across models—likely due to ambiguity with lens names.

B Few-Shot Results

Table 10 compares zero-shot and few-shot (k=3) results for models with both evaluations. The impact of few-shot examples on format compliance is highly model-dependent: Qwen2.5-14B suffers catastrophic compliance collapse (72.8%→1.4%), while DeepSeek-R1-14B *improves* (17.2%→31.5%). This prompt sensitivity further motivates model-agnostic reliability layers.

C Full Cross-Domain Results

We evaluate format collapse across five domains using three models, and separately report a sixth compliance-only analysis. Table 11 shows ROS, CSS, and schema compliance for each combination; for OntoNotes we report compliance only because exact label normalization is not directly comparable to the other schemas. Key observations: (1) all models show ROS<CSS gaps in every domain; (2) model rankings reverse across domains (Qwen best on camera, Mistral best on products); (3) WIESP scientific NER is especially hostile to strict formatting, yielding 0% strict compliance from all models despite meaningful post-canonicalization performance.

D DeepSeek-R1 Qualitative Examples

We present representative examples of DeepSeek-R1 format collapse, illustrating the four dominant failure categories.

Example 1: `<think>` tag with reasoning trace (R1-7B). Query: "Canon 5D Mark IV with 70-200mm f/2.8 ISO 3200"
Output: `<think>` *Let me analyze this photography query. I need to extract camera equipment and settings. The query mentions "Canon 5D Mark IV" which is a camera body...* [continues for 200+ tokens] `</think>` `{``camera'': ``Canon 5D Mark IV'', ``lens'': ``70-200mm'', ``iso'': ``3200'', ``aperture'': ``2.8'', ...` [truncated]

Analysis: The model produces a correct extraction but wraps it in reasoning tags. The JSON is valid but not parseable without stripping the `<think>` wrapper. This pattern occurs in 48.9% of R1-7B outputs and 56.8% of R1-14B outputs.

Example 2: Prose-wrapped valid JSON (R1-14B). Query: "135.0 fl shots Google Pixel 6"
Output: `<think>` *The user wants to find photos taken with a Google Pixel 6 at 135mm focal length.* `</think>` *Based on my analysis, here are the extracted entities:* `{``camera'': ``Google Pixel 6'', ``focal_length'': ``135.0'', ...}`

Analysis: The JSON is valid and correct, but preceded by both thinking tags and a natural-language preamble. This "prose + valid JSON" pattern occurs in 25.4% of R1-14B outputs (only 0% for R1-7B, which rarely produces valid JSON at all).

Example 3: Partial extraction in reasoning (R1-7B). Query: "aperture 2.8 iso400 Z6"
Output: `<think>` *I can see several camera settings here: an aperture of 2.8, ISO of 400, and Z6 which is likely a Nikon Z6 camera. Let me format this as the requested JSON structure... camera: "Z6", lens: null...* [reasoning continues without producing actual JSON]

Analysis: The model demonstrates semantic understanding within the reasoning trace but never produces a parseable JSON object. This is the most extreme form of format collapse: correct understanding with zero operational utility.

Example 4: Empty output (R1-7B). Query: "shutter 1/250 ∼50mm shots"
Output: `{camera: ..., lens: ..., iso: ..., aperture: ..., shutter_speed: ..., focal_length: ...}`

Analysis: The model produces a JSON template with literal ellipsis instead of values—a placeholder rather than an extraction. This "template placeholder" pattern occurs in 0.6% of R1-7B outputs.

E Verifier Data Efficiency Curve

Figure 12 shows the full data efficiency curve. The verifier transitions from harmful (<100 examples) to helpful (≥500) to near-optimal (≥2,000). This means deploying PROMPTPORT on a new domain requires only ∼1–2K labeled examples—comparable to few-shot annotation efforts.

The initial negative gain at 100 examples occurs because the verifier learns spurious patterns from insufficient data, causing it to override correct base-model predictions. At 500+ examples, it learns reliable correctness signals.

Model	CAMERA	LENS	ISO	APERTURE	SHUTTER	FOCAL_LEN
Qwen2.5-14B	0.775	0.772	0.939	0.824	0.885	0.656
Qwen2.5-72B	0.789	0.780	0.941	0.830	0.888	0.662
Mistral-Nemo-12B	0.765	0.774	0.905	0.782	0.739	0.526
DeepSeek-R1-14B	0.697	0.676	0.919	0.729	0.819	0.713
DeepSeek-R1-7B	0.027	0.027	0.017	0.002	0.001	0.001
Phi-3.5-Mini	0.122	0.068	0.169	0.205	0.325	0.245

Table 9: Per-field CSS F1 for new models (camera domain, zero-shot). Numeric fields (ISO, SHUTTER_SPEED) are easier; FOCAL_LENGTH is hardest.

	Zero-Shot			Few-Shot (k=3)		
Model	ROS	CSS	Compl.%	ROS	CSS	Compl.%
Qwen2.5-14B	0.707	0.787	72.8	0.030	0.682	1.4
Mistral-Nemo-12B	0.650	0.755	81.1	0.622	0.721	74.0
DeepSeek-R1-14B	0.199	0.717	17.2	0.331	0.662	31.5

Table 10: Zero-shot vs few-shot comparison. Prompt format dramatically affects compliance: Qwen collapses from 72.8% to 1.4%, while DeepSeek-R1 improves from 17.2% to 31.5%.

F Model Pool Size Ablation

Table 13 shows safe-override F1 as a function of the number of models in the candidate pool (models added in order of individual CSS F1). Most of the gain comes from the first 3–4 models; adding weaker models provides diminishing returns. This has practical implications: practitioners need only 3–4 diverse models to achieve most of the verifier's benefit.

G Prompt Templates

Zero-shot prompt.

```
Extract camera equipment and
metadata from the query.

Return a JSON object with these
fields:  - camera:  camera model
name (or null if not mentioned)
- lens:  lens name/focal length
(or null if not mentioned)      -
iso:  ISO value (or null if
not mentioned)        - aperture:
aperture f-number (or null if
not mentioned)    - shutter_speed:
shutter speed value (or null if
not mentioned)     - focal_length:
focal length in mm (or null if
not mentioned)

Return ONLY the JSON object, no
other text.

Query:  {query}

JSON:
```

Few-shot prompt (k=3). The few-shot prompt prepends three examples before the target query:

```
Query:  Canon 5D Mark IV with
70-200mm f/2.8
JSON: {``camera'':  ``Canon
5D Mark IV'', ``lens'':
``70-200mm'', ``iso'':  null,
``aperture'':  ``2.8'',
``shutter_speed'':  null,
``focal_length'':  ``70-200''}

Query:  ISO 3200 shot with Sony
A7III
JSON: {``camera'':  ``Sony
A7III'', ...}

Query:  Nikon Z9 85mm f/1.4
1/1000s ISO 400
JSON: {``camera'':  ``Nikon Z9'',
...}
```

H Canonicalization Algorithm

The Canonicalizer applies five deterministic steps in sequence:

1. **Fence stripping.** Remove Markdown code fences (```` ```json ... ``` ```), XML tags, and other common delimiters using regex substitution.
2. **JSON extraction.** Locate the most plausible JSON span using bracket matching: find the first { and last matching } in the output string.
3. **Repair.** Fix common near-JSON issues: replace single quotes with double quotes, remove trailing commas before } or], escape unescaped characters.
4. **Key normalization.** Map keys case-insensitively to the expected schema (e.g., "Camera" → "camera", "FNUMBER" → "aperture"). Apply domain-specific aliases.
5. **Value normalization.** Standardize units: strip "ISO" prefix from ISO values, strip "f/" from aperture, strip "mm" suffix from focal length.

Domain	N	Fields	Model	ROS	CSS	Compl.%
Camera Metadata	37,346	6	Qwen2.5-14B	0.707	0.787	72.8
			Mistral-Nemo-12B	0.650	0.755	81.1
			DeepSeek-R1-14B	0.199	0.717	17.2
Product Attributes	2,000	6	Qwen2.5-14B	0.236	0.851	15.0
			Mistral-Nemo-12B	0.825	0.834	96.7
			DeepSeek-R1-14B	0.012	0.843	1.1
Few-NERD (General NER)	2,000	8	Qwen2.5-14B	0.635	0.645	82.3
			Mistral-Nemo-12B	0.460	0.654	56.0
			DeepSeek-R1-14B	0.400	0.527	73.5
WIESP (Scientific NER)	2,000	6	Qwen2.5-14B	0.000	0.519	0.0
			Mistral-Nemo-12B	0.000	0.513	0.0
			DeepSeek-R1-14B	0.000	0.488	0.0
OntoNotes (18-type NER)	2,000	8	Qwen2.5-14B	—	—	95.4
			Mistral-Nemo-12B	—	—	88.0
			DeepSeek-R1-14B	—	—	66.4

Table 11: Cross-domain results across five scored domains plus one compliance-only OntoNotes analysis. Format collapse persists across domains, and schema compliance ranges from 0% (WIESP) to 96.7% (Mistral on product attributes), confirming that format reliability is strongly schema-dependent.

Examples	Single	Override	Gain
100	0.779	0.740	−0.039
500	0.779	0.802	+0.023
1,000	0.779	0.822	+0.043
2,000	0.779	0.878	+0.099
3,000	0.779	0.887	+0.108

Table 12: Full verifier data efficiency results. Gain turns positive at ∼500 examples and plateaus near 3K.

Pool	Models Added	Override	Marginal
2	Qwen-72B, Qwen-14B	0.876	—
3	+Mistral-Nemo	0.887	+0.011
4	+DeepSeek-R1-14B	0.891	+0.004
5	+Phi-3.5	0.893	+0.002
6	+DeepSeek-R1-7B	0.893	+0.000

Table 13: Model pool ablation. Diminishing returns: 3–4 models capture most of the benefit.

Fill missing schema keys with null.

The canonicalizer is purely deterministic (no learned parameters) and adds 0.006 ms per example. It recovers 6–8 F1 points on average by transforming format-broken but semantically correct outputs into valid JSON.

I Human Evaluation Protocol

Setup. We randomly sample 50 test queries and collect model outputs from four models (Qwen2.5-14B, Mistral-Nemo-12B, DeepSeek-R1-14B, Phi-3.5-Mini), yielding 200 (query, output) pairs. One annotator (a co-author with expertise in NLP evaluation) judges each pair. Because this analysis is intended as a targeted sanity check on the automatic metrics rather than a standalone human benchmark, we use a single expert annotator and interpret the results as supportive rather than definitive evidence.

Annotation schema.

- **format_correct** (binary): 1 if the raw output is valid JSON with all six required fields present, 0 otherwise.
- **semantic_quality** (ternary): 0 = no correct fields, 1 = at least one field correct but not all, 2 = all present fields correct.

Results. ROS achieves perfect agreement with human format judgments: Kendall's $\tau = 1.000$, Spearman $\rho = 1.000$, 100% agreement across all 200 samples. Every case where the human judged "format correct" matched strict JSON parsing, and vice versa.

CSS correlates strongly with human semantic quality: Kendall's $\tau = 0.681$ ($p < 10^{-24}$), Spearman $\rho = 0.722$ ($p < 10^{-33}$). Residual disagreement arises from edge cases where humans accept approximate value matches (e.g., "Canon 5D" vs "Canon EOS 5D Mark IV") that CSS's string matching misses.

Per-model human scores.

J Schema Compliance Across All Domains

Table 15 presents schema compliance (percentage of outputs that are valid JSON with all required fields) across all evaluated domains and models.

Model	Format%	Sem=2%	Sem≥1%
Mistral-Nemo-12B	90.0	42.0	94.0
Qwen2.5-14B	70.0	48.0	96.0
DeepSeek-R1-14B	10.0	52.0	94.0
Phi-3.5-Mini	4.0	6.0	8.0

Table 14: Human evaluation per model. DeepSeek-R1 shows the format collapse pattern most clearly: only 10% format-correct but 94% semantically at least partially correct.

Domain (fields)	Qwen	Mistral	DS-R1
Camera (6)	72.8	81.1	17.2
Product (6)	15.0	96.7	1.1
Few-NERD (8)	82.3	56.0	73.5
WIESP Scientific (6)	0.0	0.0	0.0
OntoNotes (8)	95.4	88.0	66.4
PLOD Legal (5)	0.0	0.0	0.0

Table 15: Schema compliance across six domains. No model consistently dominates: the same model can be highly compliant on one schema and almost unusable on another.

The dramatic variation—from 0% to 96.7% for the same model across different schemas—is the core evidence that format collapse is schema-dependent and not a fixed model property.

K Canonicalization Recovery Rates

Table 16 shows how canonicalization transforms strict compliance rates on the **camera domain**. The largest recovery in this table is DeepSeek-R1-14B: from 17.2% strict compliance to 99.6% after canonicalization—an 82.4 percentage-point recovery. This demonstrates that the JSON content is often present in the output but wrapped in reasoning traces or other non-standard formatting that strict parsers reject.

L Statistical Stability

Across the 37,346-example camera-domain test set, the relative ordering of the headline results is stable under bootstrap resampling. The important point for interpretation is that the main-table differences are not a small-sample artifact: the strongest models remain strongest, the weakest remain weakest, and the large ROS→CSS gaps persist across resamples. We therefore summarize the stability finding here and provide the exact resampling scripts in the released evaluation package.

Model	Strict%	Canon%	Recovery
Qwen2.5-14B	72.8	99.9	+27.1
Mistral-Nemo-12B	81.1	97.6	+16.5
DeepSeek-R1-14B	17.2	99.6	+82.4
Phi-3.5-Mini	10.3	16.0	+5.7
DeepSeek-R1-7B	0.8	1.3	+0.5

Table 16: Canonicalization recovery rates (camera domain). DeepSeek-R1-14B recovers from 17.2% to 99.6%—the JSON is there, just wrapped in reasoning traces.