

# Hidden Non-Determinism in Large Language Model APIs: A Lightweight Provenance Protocol for Reproducible Generative AI Research

LUCAS ROVER\*, UTFPR – Universidade Tecnológica Federal do Paraná, Brazil

YARA DE SOUZA TADANO, UTFPR – Universidade Tecnológica Federal do Paraná, Brazil

**Background:** Generative AI models produce non-deterministic outputs that vary across runs, even under nominally identical configurations. This variability threatens the reproducibility of studies that rely on large language model (LLM) outputs, yet most existing experiment-tracking tools were not designed for the specific challenges of text-generation workflows.

**Objectives:** We propose a lightweight, open-standard protocol for logging, versioning, and provenance tracking of generative AI experiments. The protocol introduces two novel documentation artifacts—Prompt Cards and Run Cards—and adopts the W3C PROV data model to create auditable, machine-readable provenance graphs linking every output to its full generation context.

**Methods:** We formalize the protocol and evaluate it empirically through 1,864 controlled experiments. These experiments employ two models—LLaMA 3 8B (locally deployed) and GPT-4 (cloud API)—on two Natural Language Processing (NLP) tasks (scientific summarization and structured extraction) across 30 scientific abstracts and five experimental conditions that systematically vary the seed, temperature, and decoding strategy. We measure output variability using Exact Match Rate, Normalized Edit Distance, ROUGE-L, and BERTScore, and quantify the protocol’s own overhead in terms of time and storage.

**Results:** Under greedy decoding ( $t=0$ ), LLaMA 3 achieves near-perfect reproducibility on extraction (EMR = 0.987) and summarization (EMR = 0.947). By contrast, GPT-4 under identical greedy settings achieves only EMR = 0.443 for extraction and EMR = 0.230 for summarization, consistent with substantial server-side non-determinism that is invisible without systematic logging. Increasing temperature to 0.7 eliminates exact matches for both models. The protocol adds a mean overhead of 25.43 ms per run (0.545% of inference time) and approximately 4.1 KB per run record, totaling 19.52 MB for all 1,864 runs.

**Conclusions:** Our results demonstrate that (1) local inference is substantially more reproducible than API-based inference even under nominally identical parameters, (2) structured output tasks are inherently more reproducible than open-ended generation, (3) temperature is the dominant *user-controllable* factor affecting variability, and (4) comprehensive provenance logging can be achieved with negligible overhead. The protocol, reference implementation, and all experimental data are publicly available.

**CCS Concepts:** • Software and its engineering → Software testing and debugging; Documentation; • Computing methodologies → Machine learning.

**Additional Key Words and Phrases:** reproducibility, large language models, non-determinism, provenance, generative AI, experiment tracking, W3C PROV, scientific methodology

**JAIR Associate Editor:** Insert JAIR AE Name

\*Corresponding Author.

Authors’ Contact Information: Lucas Rover, ORCID: [0000-0001-6641-9224](#), lucasrover@utfpr.edu.br, UTFPR – Universidade Tecnológica Federal do Paraná, Programa de Pós-Graduação em Engenharia Mecânica, Ponta Grossa, Paraná, Brazil; Yara de Souza Tadano, ORCID: [0000-0002-3975-3419](#), yaratadano@utfpr.edu.br, UTFPR – Universidade Tecnológica Federal do Paraná, Programa de Pós-Graduação em Engenharia Mecânica, Ponta Grossa, Paraná, Brazil.



This work is licensed under a Creative Commons Attribution International 4.0 License.

© 2026 Copyright held by the owner/author(s).

DOI: [10.1613/jair.1.xxxxx](#)

**JAIR Reference Format:**

Lucas Rover and Yara de Souza Tadano. 2026. Hidden Non-Determinism in Large Language Model APIs: A Lightweight Provenance Protocol for Reproducible Generative AI Research. *Journal of Artificial Intelligence Research* (2026), 25 pages. doi: 10.1613/jair.1xxxxx

## 1 Introduction

When a researcher queries GPT-4 with the same prompt and temperature zero, one would reasonably expect identical outputs. Our experiments show otherwise: across five controlled seeds under greedy decoding, GPT-4 produces the same extraction result only 44% of the time. This hidden non-determinism exemplifies a fundamental challenge introduced by the rapid adoption of large language models (LLMs) in scientific research: how to ensure that studies relying on generative AI outputs are reproducible, auditable, and scientifically rigorous. Unlike traditional computational experiments, in which deterministic algorithms produce identical results given identical inputs, LLMs exhibit inherent variability in their outputs due to stochastic sampling, floating-point non-determinism, and opaque model-versioning practices (Y. Chen et al. 2023; Zhu et al. 2023).

This reproducibility challenge is not merely theoretical. Baker (2016) reported that over 70% of researchers have failed to reproduce another scientist’s experiment, a crisis that extends to AI research (Gundersen and Kjensmo 2018; Hutson 2018; Kapoor and A. Narayanan 2023; Stodden et al. 2016). For generative AI specifically, the problem is compounded by several factors unique to text-generation workflows: (1) the same prompt can yield semantically similar yet textually distinct outputs across runs; (2) API-based models may undergo silent updates that alter behavior; (3) temperature and sampling parameters create a high-dimensional space of possible outputs; and (4) no established standard exists for documenting the full context needed to understand, audit, or reproduce a generative output.

Existing experiment-tracking tools such as MLflow (Zaharia et al. 2018), Weights & Biases (Biewald 2020), and DVC (Kuprieiev et al. 2024) were designed primarily for training pipelines and numerical metrics. Although valuable for their intended purposes, these tools lack features critical for generative AI studies: structured prompt versioning, cryptographic output hashing for tamper detection, provenance graphs linking outputs to their full generation context, and environment fingerprinting specific to inference-time conditions.

In this paper, we make three contributions:

- (1) **A lightweight protocol** for logging, versioning, and provenance tracking of generative AI experiments. The protocol introduces *Prompt Cards* and *Run Cards* as structured documentation artifacts, and adopts the W3C PROV data model (Moreau and Missier 2013) for machine-readable provenance graphs.
- (2) **An empirical evaluation** of both the protocol’s effectiveness and the reproducibility characteristics of LLM outputs. Through 1,864 controlled experiments with LLaMA 3 8B (local) and GPT-4 (API) across two tasks, 30 abstracts, and five conditions, we quantify output variability using four complementary metrics and measure the protocol’s overhead. Our results reveal a striking reproducibility gap between local and API-based inference that is invisible without systematic logging.
- (3) **A reference implementation** in Python that demonstrates the protocol’s practical applicability, together with all experimental data, to facilitate adoption and independent verification.

The remainder of this paper is organized as follows. Section 2 reviews related work on reproducibility in AI and experiment tracking. Section 3 formalizes the protocol design. Section 4 describes the experimental methodology. Section 5 presents the empirical results. Section 6 discusses findings, limitations, and practical implications. Section 7 concludes with directions for future work.

## 48 2 Related Work

### 49 2.1 Reproducibility in AI Research

50 The reproducibility crisis in AI has been documented extensively. [Gundersen and Kjensmo \(2018\)](#) surveyed 400  
 51 AI papers and found that only 6% provided sufficient information for full reproducibility. [Pineau et al. \(2021\)](#)  
 52 reported on the NeurIPS 2019 Reproducibility Program, which introduced reproducibility checklists and found  
 53 significant gaps between reported and actual reproducibility. More recently, [Gundersen, Helmert, et al. \(2024\)](#)  
 54 described four institutional mechanisms adopted by JAIR—reproducibility checklists, structured abstracts, badges,  
 55 and reproducibility reports—establishing a community standard for what should be documented in AI research.  
 56 [Gundersen, Gil, et al. \(2018\)](#) identified three levels of reproducibility in AI—method, data, and experiment—and  
 57 argued that all three are necessary for scientific progress. [Belz et al. \(2021\)](#) conducted a systematic review of 601  
 58 NLP papers and confirmed pervasive under-reporting of experimental details, while [Dodge et al. \(2019\)](#) proposed  
 59 improved reporting standards for ML experiments, including confidence intervals and significance tests across  
 60 multiple runs. More broadly, [Kapoor and A. Narayanan \(2023\)](#) identified data leakage as a widespread driver of  
 61 irreproducible results across 17 scientific fields that use ML-based methods.

62 For generative AI specifically, [Y. Chen et al. \(2023\)](#) demonstrated that ChatGPT’s outputs on NLP benchmarks  
 63 exhibit non-trivial variability across identical queries, even with temperature set to zero. [Zhu et al. \(2023\)](#)  
 64 showed that reproducibility degrades further when tasks involve subjective judgment, such as social computing  
 65 annotations. Most recently, [Atil et al. \(2024\)](#) systematically measured the non-determinism of five LLMs under  
 66 supposedly deterministic settings across eight tasks, finding accuracy variations up to 15% across runs and  
 67 introducing the Total Agreement Rate (TAR) metric. [Ouyang et al. \(2024\)](#) confirmed that temperature zero  
 68 does not guarantee determinism in ChatGPT code generation. Most recently, [Yuan et al. \(2025\)](#) traced such  
 69 non-determinism to numerical precision issues in GPU kernels and proposed LayerCast as a mitigation strategy.  
 70 Our work complements these studies in three specific ways. First, whereas prior studies (including Atil et al.’s  
 71 five-model, eight-task study) measure variability post hoc, we provide a structured provenance protocol that  
 72 enables *prospective* documentation and audit—answering not only “how much variability?” but also “why did  
 73 these outputs differ?” through cryptographic hashing and W3C PROV graphs. Second, we directly compare  
 74 local and API-based inference on identical tasks with identical prompts, isolating the deployment paradigm  
 75 as a variable—a comparison absent from prior work. Third, we quantify the overhead of systematic logging,  
 76 demonstrating that the “cost of knowing” is negligible.

### 77 2.2 Experiment Tracking Tools

78 Several tools exist for tracking machine learning experiments, although none was designed specifically for  
 79 generative AI text-output workflows:

80 **MLflow** ([Zaharia et al. 2018](#)) provides experiment tracking, model packaging, and deployment. It logs parameters,  
 81 metrics, and artifacts, but focuses on training pipelines and numerical outcomes rather than text-generation  
 82 provenance.

83 **Weights & Biases** ([Biewald 2020](#)) offers experiment tracking with visualization dashboards. It supports prompt  
 84 logging but lacks structured prompt versioning, cryptographic output hashing, and provenance graph generation.

85 **DVC** ([Kuprieiev et al. 2024](#)) provides data versioning through git-like operations. While effective for dataset  
 86 management, it does not address run-level provenance or prompt documentation.

87 **OpenAI Eval** ([OpenAI 2023](#)) is a framework for evaluating LLM outputs against benchmarks. It provides  
 88 structured evaluation but is tightly coupled to OpenAI’s ecosystem and does not generate interoperable provenance  
 89 records.

90 **LangSmith** ([LangChain 2023](#)) offers tracing and evaluation for LLM applications. It captures detailed execution  
 91 traces but uses a proprietary format and requires cloud connectivity.

95 Table 1. Comparison of our protocol with existing reproducibility tools and frameworks for GenAI experiments. Checkmarks  
 96 ( $\checkmark$ ) indicate full support; tildes (~) indicate partial support; dashes (–) indicate no support.

Feature	Ours	MLflow	W&B	DVC	OpenAI Eval	LangSmith
Prompt versioning (Prompt Card)	$\checkmark$	–	~	–	~	~
Run-level provenance (W3C PROV)	$\checkmark$	–	–	–	–	–
Cryptographic output hashing	$\checkmark$	–	–	$\checkmark$	–	–
Seed & param logging	$\checkmark$	$\checkmark$	$\checkmark$	–	$\checkmark$	$\checkmark$
Environment fingerprinting	$\checkmark$	~	~	~	–	–
Model weights hashing	$\checkmark$	–	~	$\checkmark$	–	–
Overhead <1% of inference	$\checkmark$	~	~	N/A	N/A	~
Designed for GenAI text output	$\checkmark$	–	–	–	$\checkmark$	$\checkmark$
Open standard (PROV-JSON)	$\checkmark$	–	–	–	–	–
Local-first (no cloud dependency)	$\checkmark$	$\checkmark$	–	$\checkmark$	–	–

113 More broadly, [Bommasani et al. \(2022\)](#) identified reproducibility as a key risk for foundation models, and [Liang](#)  
 114 [et al. \(2023\)](#) proposed the HELM benchmark for holistic evaluation of language models, including robustness and  
 115 fairness dimensions that complement our reproducibility focus. In the provenance space, [Padovani et al. \(2025\)](#)  
 116 recently introduced yProv4ML, a framework that captures ML provenance in PROV-JSON format with minimal  
 117 code modifications; our protocol shares the commitment to W3C PROV but targets the specific challenges of  
 118 stochastic text generation rather than training pipelines.

119 Table 1 provides a systematic feature-by-feature comparison of our protocol with these tools. The key distinction  
 120 is not merely one of tooling but of *scientific capability*: existing tools log what happened during training (parameters,  
 121 metrics, artifacts), whereas our protocol enables answering questions that these tools cannot—specifically,  
 122 whether two generative outputs are provably derived from identical configurations, which exact factor caused  
 123 a divergence between non-identical outputs, and whether an output has been tampered with post-generation.  
 124 These capabilities require the combination of cryptographic hashing, structured prompt documentation, and  
 125 W3C PROV provenance graphs that no existing tool provides. In short, our contribution is not an alternative  
 126 experiment tracker but a *reproducibility assessment framework* designed for the unique challenges of stochastic  
 127 text generation.

### 129 2.3 Provenance in Scientific Computing

130 Data provenance—the lineage of data through transformations—has a rich history in database systems and  
 131 scientific workflows ([Herschel et al. 2017](#)). The W3C PROV family of specifications ([Moreau and Missier 2013](#))  
 132 provides a standardized data model for representing provenance as directed acyclic graphs of *entities*, *activities*,  
 133 and *agents*. [Samuel and König-Ries \(2022\)](#) applied provenance tracking to computational biology workflows,  
 134 demonstrating its value for reproducibility. However, to our knowledge, no prior work has applied W3C PROV  
 135 specifically to generative AI experiment workflows, in which the challenge involves not only tracking data  
 136 lineage but also capturing the stochastic generation context that determines output variability.

137 Taken together, these gaps point to a clear need: a lightweight, standards-based protocol that bridges generative  
 138 AI inference with the provenance infrastructure already established in scientific computing. The next section  
 139 presents our design for such a protocol.

### 142 3 Protocol Design

143 Our protocol addresses the question: *What is the minimum set of metadata that must be captured for each generative*  
 144 *AI run to enable auditing, reproducibility assessment, and provenance tracking?* We address this question through  
 145 four complementary components.

146

#### 147 3.1 Scope and Design Principles

148 The protocol is designed around three principles:

149

- 150 (1) **Completeness:** Every factor that can influence a generative output must be captured—prompt text, model  
 151 identity and version, inference parameters, environment state, and timestamps.
- 152 (2) **Negligible overhead:** The logging process must not materially affect the experiment. We target <1%  
 153 overhead relative to inference time.
- 154 (3) **Interoperability:** All artifacts are stored in open, machine-readable formats (JSON, PROV-JSON), aligned  
 155 with the FAIR (Findable, Accessible, Interoperable, Reusable) principles ([Wilkinson et al. 2016](#)), to enable  
 156 tool integration and long-term preservation.

157

#### 158 3.2 Prompt Cards

159 A *Prompt Card* is a versioned documentation artifact that captures the design rationale and metadata for a prompt  
 160 template used in experiments. Each Prompt Card contains:

161

- 162 • `prompt_id`: Unique identifier
- 163 • `prompt_hash`: SHA-256 hash of the prompt text, enabling tamper detection
- 164 • `version`: Semantic version number
- 165 • `task_category`: Classification of the task (e.g., summarization, extraction)
- 166 • `objective`: Natural-language description of what the prompt is designed to achieve
- 167 • `assumptions`: Explicit assumptions about inputs and expected behavior
- 168 • `limitations`: Known limitations or failure modes
- 169 • `target_models`: Models for which the prompt was designed and tested
- 170 • `expected_output_format`: Description of the expected output structure
- 171 • `interaction_regime`: Single-turn, multi-turn, or chain-of-thought
- 172 • `change_log`: History of modifications

173

Prompt Cards serve two purposes: they document design intent (supporting human understanding) and they provide a citable, hashable reference for automated provenance tracking. The concept draws inspiration from Model Cards ([Mitchell et al. 2019](#)), Datasheets for Datasets ([Gebru et al. 2021](#)), and model info sheets for reproducibility assessment ([Kapoor and A. Narayanan 2023](#)), extending the structured-documentation paradigm to the prompt layer of the generative AI pipeline.

177

#### 178 3.3 Run Cards

179

A *Run Card* captures the complete execution context of a single generative AI run. Each Run Card records 22 core fields organized into five groups (the complete JSON schema in Appendix B includes these fields plus additional metadata such as `researcher_id`, `affiliation`, `system_logs`, and `errors`):

180

- 181 (1) **Identification:** `run_id`, `task_id`, `task_category`, `prompt_card_ref`
- 182 (2) **Model context:** `model_name`, `model_version`, `weights_hash`, `model_source`
- 183 (3) **Parameters:** `inference_params` (temperature, `top_p`, `top_k`, `max_tokens`, `seed`, `decoding_strategy`),  
 184 `params_hash`
- 185 (4) **Input/Output:** `input_text`, `input_hash`, `output_text`, `output_hash`, `output_metrics`

186

187

188

189 (5) **Execution metadata:** environment (OS, architecture, Python version, hostname), environment\_hash,  
 190 code\_commit, timestamps, execution\_duration\_ms, logging\_overhead\_ms, storage\_kb

191 The separation of logging overhead from execution time is deliberate: it allows researchers to verify that the  
 192 protocol itself does not confound experimental measurements.  
 193

### 194 3.4 W3C PROV Integration

195 Each experimental group (defined by a unique model–task–condition–abstract combination) is automatically  
 196 translated into a W3C PROV-JSON document (Moreau and Missier 2013) that expresses the generation provenance  
 197 as a directed graph. The mapping defines:  
 198

- 199 • **Entities:** Prompt, InputText, ModelVersion, InferenceParameters, Output, ExecutionMetadata
- 200 • **Activities:** RunGeneration (the inference execution)
- 201 • **Agents:** Researcher, SystemExecutor (the execution environment)

202 PROV relations capture the causal structure:

- 203 • used: RunGeneration used Prompt, InputText, ModelVersion, InferenceParameters
- 204 • wasGeneratedBy: Output wasGeneratedBy RunGeneration
- 205 • wasAssociatedWith: RunGeneration wasAssociatedWith Researcher, SystemExecutor
- 206 • wasAttributedTo: Output wasAttributedTo Researcher
- 207 • wasDerivedFrom: Output wasDerivedFrom InputText

208 This standardized representation enables automated reasoning about experiment provenance, including  
 209 detecting when two runs share identical configurations and identifying the specific factors that differ between  
 210 non-identical outputs. An abbreviated example document is given in Appendix C.  
 211

### 212 3.5 Reproducibility Checklist

213 We provide a 15-item checklist organized into four categories—Prompt Documentation, Model and Environment,  
 214 Execution and Output, and Provenance—that researchers can use to self-assess the reproducibility of their  
 215 generative AI studies. The complete checklist is provided in Appendix A.  
 216

### 217 3.6 Extensions for Advanced Workflows

218 While our empirical evaluation focuses on single-turn, single-model inference, the protocol’s field schema is  
 219 designed to accommodate more complex workflows through optional extension fields:  
 220

- 221 • **Retrieval-Augmented Generation (RAG):** Additional fields for retrieval\_query, retrieved\_documents  
 222 (with hashes), retrieval\_model, and chunk\_strategy enable tracing which external context influenced  
 223 the output.
- 224 • **Tool use and function calling:** Fields for tools\_available, tool\_calls (with arguments and results),  
 225 and tool\_call\_hashes capture the full tool-use chain.
- 226 • **Multi-turn dialogues:** A conversation\_history\_hash field and turn\_index enable linking each turn  
 227 to the full conversation state.
- 228 • **Chain-of-thought / agent workflows:** A parent\_run\_id field supports hierarchical provenance graphs  
 229 for multi-step reasoning chains.

230 These extensions are not evaluated in our current experiments but are specified in the reference implementa-  
 231 tion’s schema to support future adoption in production LLM pipelines.  
 232

Having defined the protocol’s components, we now evaluate it empirically along two dimensions: the repro-  
 233 ductibility characteristics it reveals across different models and conditions, and the overhead it imposes on the  
 234 experimental workflow.  
 235

## 236 4 Experimental Setup

237 We designed a controlled experiment to simultaneously evaluate (a) the reproducibility characteristics of LLM  
 238 outputs under varying conditions and (b) the overhead imposed by our logging protocol.  
 239

### 240 4.1 Models and Infrastructure

241 We evaluate two models representing fundamentally different deployment paradigms:

242 **LLaMA 3 8B** (Grattafiori et al. 2024): A locally deployed open-weight model served through Ollama (Ollama  
 243 2024) on an Apple M4 system with 24 GB unified memory running macOS 14.6. Local deployment provides  
 244 complete control over the execution environment, eliminating confounding factors such as network latency,  
 245 server-side batching, and silent model updates. The software stack comprised Ollama v0.5.4, Python 3.12.8, the  
 246 ollama Python SDK v0.4.7, and the LLaMA 3 8B Q4\_0 quantization (SHA-256 recorded per run).  
 247

248 **GPT-4** (Achiam et al. 2023): A cloud-based proprietary model accessed via the OpenAI API (openai Python SDK  
 249 v1.59.9) with controlled seed parameters. Although we requested `model="gpt-4"`, the API returned `gpt-4-0613`  
 250 as the resolved model version in all runs, which we recorded in the `model_id_returned` field of each run record.  
 251 This represents the typical deployment scenario where researchers have limited control over the inference  
 252 environment. The API introduces additional sources of variability: load balancing, server-side batching, potential  
 253 model-version updates, and floating-point non-determinism across different hardware.  
 254

### 255 4.2 Tasks

256 We evaluate two tasks that represent complementary points on the output-structure spectrum:

257 **Task 1: Scientific Summarization.** Given a scientific abstract, produce a concise summary in exactly three  
 258 sentences covering the main contribution, methodology, and key quantitative result. This is an open-ended  
 259 generation task in which the model has considerable freedom in word choice and phrasing.

260 **Task 2: Structured Extraction.** Given a scientific abstract, extract five fields (`objective`, `method`, `key_result`,  
 261 `model_or_system`, `benchmark`) into a JSON object. This is a constrained generation task in which the output  
 262 format is fixed and the model must select, rather than generate, content.

### 263 4.3 Input Data

264 We use 30 widely-cited scientific abstracts from landmark AI/ML papers, including Vaswani et al. (2017) (Trans-  
 265 former), Devlin et al. (2019) (BERT), Brown et al. (2020) (GPT-3), Raffel et al. (2020) (T5), Wei et al. (2022)  
 266 (Chain-of-Thought), as well as seminal works on GANs, ResNets, VAEs, LSTMs, CLIP, DALL-E 2, Stable Diffusion,  
 267 LLaMA, InstructGPT, PaLM, and others. These abstracts vary in length (74–227 words), technical complexity, and  
 268 the number of quantitative results reported, thereby providing substantial diversity in the generation challenge.  
 269

### 270 4.4 Experimental Conditions

271 We define five conditions (Table 2) that systematically vary the factors hypothesized to affect reproducibility:

272 **C1 (Fixed seed, greedy decoding):** Temperature = 0, seed = 42 for all 5 repetitions. This represents the  
 273 maximum-control condition and should yield deterministic outputs.

274 **C2 (Variable seeds, greedy decoding):** Temperature = 0, seeds = {42, 123, 456, 789, 1024}. This condition tests  
 275 whether seed variation affects outputs when greedy decoding is used.

276 **C3 (Temperature sweep):** Three sub-conditions at  $t \in \{0.0, 0.3, 0.7\}$  with 3 repetitions each, using different  
 277 seeds per repetition. This condition characterizes how temperature affects output variability.

278 For LLaMA 3, each task × abstract combination is evaluated under conditions C1 (5 runs), C2 (5 runs), and C3 (9  
 279 runs = 3 temperatures × 3 reps), yielding 19 runs per pair, or  $19 \times 30 \times 2 = 1,140$  runs. For GPT-4, conditions C2 and  
 280 C3 are included: C2 (5 runs) and C3 (9 runs) per pair; due to API quota exhaustion, 724 valid runs were collected  
 281

282

Table 2. Experimental design: conditions, parameters, and expected outcomes.

Cond.	Description	Temp.	Seed	Reps	Expected Outcome
C1	Fixed seed, greedy	0.0	42 (fixed)	5	Deterministic output
C2	Variable seeds, greedy	0.0	5 different	5	Near-deterministic
$C3_{t=0.0}$	Temp. baseline	0.0	per-rep	3	Deterministic
$C3_{t=0.3}$	Low temperature	0.3	per-rep	3	Low variability
$C3_{t=0.7}$	High temperature	0.7	per-rep	3	High variability

Note: Each condition is applied to 30 abstracts  $\times$  2 tasks = 60 groups per condition. Total: 1,864 logged runs (1,140 LLaMA 3 + 724 GPT-4). For GPT-4, C2 uses the same fixed seed (42) as C1 across all repetitions; C2 therefore subsumes C1 as the definitive test of API determinism under greedy decoding. GPT-4 C1 (8/300 runs collected before quota exhaustion) is excluded from all analyses.

(C2: 300/300 complete; C3: 416/450). Note that for GPT-4, C2 uses the same fixed seed (= 42) and temperature (= 0) as C1, because the API’s seed parameter is advisory and does not guarantee determinism—the distinction between “fixed seed” and “variable seeds” is meaningful only for locally controlled models like LLaMA 3. Consequently, GPT-4 C2 serves as the definitive test of API determinism under greedy decoding, and the incomplete C1 data (8/300 runs) are excluded from all analyses. **Total: 1,864 valid runs.**

## 4.5 Metrics

We adopt an operational definition of reproducibility at three levels, each mapped to a specific metric:

- **Exact reproducibility** (string-level): Two outputs are identical character-by-character. Measured by *Exact Match Rate (EMR)*.
- **Near reproducibility** (edit-level): Two outputs differ only in minor surface variations (punctuation, whitespace, synonym substitution). Measured by *Normalized Edit Distance (NED)*.
- **Semantic reproducibility** (meaning-level): Two outputs convey the same information despite different phrasing. Measured by *ROUGE-L F1* and *BERTScore F1*.

This three-level framework allows us to distinguish between outputs that are bitwise identical ( $EMR = 1$ ), textually close ( $NED < 0.05$ ), and semantically equivalent ( $ROUGE-L > 0.90$ ). All variability metrics are computed over all  $\binom{n}{2}$  unique output pairs within each experimental group (defined by model, task, condition, and abstract):

**Exact Match Rate (EMR):** The fraction of output pairs that are character-for-character identical.  $EMR = 1.0$  indicates perfect reproducibility;  $EMR = 0.0$  indicates that no two outputs match exactly.

**Normalized Edit Distance (NED):** The Levenshtein edit distance (Levenshtein 1966) between each pair, normalized by the length of the longer string.  $NED = 0.0$  indicates identical outputs; higher values indicate greater textual divergence.

**ROUGE-L F1:** The F1 score based on the longest common subsequence at the word level (Lin 2004). This captures semantic similarity even when surface forms differ.  $ROUGE-L = 1.0$  indicates identical word sequences.

Our primary metrics (EMR, NED, ROUGE-L) focus on exact and near reproducibility, which are the most direct measures for our research question. To complement these surface-level metrics, we also compute **BERTScore F1** (T. Zhang et al. 2020)—an embedding-based semantic similarity metric—for all conditions. BERTScore captures meaning-level equivalence that surface metrics may miss (e.g., paraphrases), providing a fourth perspective on reproducibility. For the structured extraction task, we additionally report **JSON validity rate**, **schema compliance rate**, and **field-level accuracy**, which measure whether outputs are syntactically valid JSON, contain all expected fields, and agree on individual field values across runs, respectively (see Appendix D for detailed results).

For protocol overhead, we measure:

330 Table 3. Output variability across experimental conditions for LLaMA 3 8B (local) and GPT-4 (API). Values are means over  
 331 30 abstracts; per-abstract standard deviations are available in the project repository. EMR = Exact Match Rate ( $\uparrow$ ), NED =  
 332 Normalized Edit Distance ( $\downarrow$ ), ROUGE-L = word-level LCS F1 ( $\uparrow$ ), BS-F1 = BERTScore F1 ( $\uparrow$ ). For GPT-4, C2 uses the same  
 333 parameters as C1 (seed = 42,  $t=0$ ), effectively subsuming C1 as the definitive test of API determinism (see Section 4). GPT-4  
 334 extraction C3 conditions are based on 14–17 abstracts (vs. 30 for all other conditions) due to API quota exhaustion.

Model	Task	Condition	EMR $\uparrow$	NED $\downarrow$	ROUGE-L $\uparrow$	BS-F1 $\uparrow$
LLaMA 3 8B	Summarization	C1 (fixed seed, $t=0$ )	0.947	0.0050	0.9945	0.9990
		C2 (var. seeds, $t=0$ )	0.947	0.0050	0.9945	0.9990
		C3 ( $t=0.0$ )	0.911	0.0083	0.9909	0.9984
		C3 ( $t=0.3$ )	0.000	0.2790	0.7441	0.9669
		C3 ( $t=0.7$ )	0.000	0.4438	0.5589	0.9432
	Extraction	C1 (fixed seed, $t=0$ )	0.987	0.0031	0.9966	0.9997
		C2 (var. seeds, $t=0$ )	0.987	0.0031	0.9966	0.9997
		C3 ( $t=0.0$ )	0.978	0.0052	0.9943	0.9996
		C3 ( $t=0.3$ )	0.211	0.1224	0.8838	0.9851
		C3 ( $t=0.7$ )	0.000	0.2530	0.7719	0.9693
GPT-4 (API)	Summarization	C2 (greedy, $t=0$ )	0.230	0.1365	0.8695	0.9839
		C3 ( $t=0.0$ )	0.144	0.1623	0.8479	0.9804
		C3 ( $t=0.3$ )	0.000	0.2832	0.7238	0.9662
		C3 ( $t=0.7$ )	0.000	0.4366	0.5554	0.9477
	Extraction	C2 (greedy, $t=0$ )	0.443	0.0724	0.9384	0.9904
		C3 ( $t=0.0$ )	0.381	0.0721	0.9356	0.9900
		C3 ( $t=0.3$ )	0.143	0.1477	0.8669	0.9799
		C3 ( $t=0.7$ )	0.000	0.2247	0.7890	0.9708

- **Logging time:** Wall-clock time spent on hashing, metadata collection, and file I/O, measured separately from inference time.
- **Storage:** Size of each run record (JSON) and total storage for all protocol artifacts.
- **Overhead ratio:** Logging time as a percentage of total execution time.

## 5 Results

### 5.1 Output Variability

Table 3 presents the main variability results for both models, aggregated across all 30 abstracts.

5.1.1 *LLaMA 3 8B (Local Inference).* **Finding 1: Structured extraction achieves near-perfect reproducibility under greedy decoding.** With  $t = 0$ , extraction produces EMR = 0.987 and NED = 0.0031 across conditions C1 and C2, meaning virtually every output is character-for-character identical. Summarization achieves an EMR of 0.947 with NED = 0.0050, indicating near-perfect but not complete reproducibility.

**Finding 2: Seed variation has no effect under greedy decoding.** Conditions C1 and C2 produce identical results despite using different seeds. With  $t = 0$ , the model always selects the highest-probability token, making the seed irrelevant. This finding confirms that greedy decoding provides reliably deterministic inference with locally deployed models.

5.1.2 *GPT-4 (API Inference).* **Finding 3: API-based inference is substantially less reproducible than local inference, even under greedy decoding.** This is the most striking result of our study. Under greedy

377 Table 4. Reproducibility comparison: LLaMA 3 8B (local) vs. GPT-4 (API) under greedy decoding, Condition C2 (variable  
 378 seeds,  $t=0$ ). GPT-4 shows significantly lower reproducibility, consistent with server-side non-determinism.

Task	Metric	LLaMA 3 8B	GPT-4
Summarization	EMR↑	0.947	0.230
	NED↓	0.0050	0.1365
	ROUGE-L↑	0.9945	0.8695
Extraction	EMR↑	0.987	0.443
	NED↓	0.0031	0.0724
	ROUGE-L↑	0.9966	0.9384

389 decoding ( $t = 0$ ) with controlled seeds, GPT-4 achieves only EMR = 0.230 for summarization and EMR = 0.443 for  
 390 extraction—compared to LLaMA’s 0.947 and 0.987, respectively, under the same C2 condition.

391 Table 4 highlights this reproducibility gap directly.

392 This gap is not due to user-side parameter differences: both models use  $t = 0$  with the same seed. The observed  
 393 variability is consistent with deployment-side factors that are invisible to the researcher, including hardware-level  
 394 floating-point non-determinism across different GPU types in the serving cluster, request-batching and scheduling  
 395 effects, prompt-format differences (completion vs. chat interface), and potential silent model updates during  
 396 the experimental window. To isolate the prompt-format contribution, we conducted a supplementary control  
 397 experiment running LLaMA 3 via Ollama’s /api/chat endpoint (chat template, matching GPT-4’s message  
 398 structure); the results (Appendix E) confirm that the format difference does not explain the reproducibility gap.  
 399 While our experimental design controls for user-side parameters, we note that a definitive decomposition of  
 400 API-side variability sources would require access to the serving infrastructure. *Without systematic logging, this*  
 401 *non-determinism would be entirely invisible.*

402 **5.1.3 Temperature Effects Across Models. Finding 4: Temperature is the dominant user-controllable factor  
 403 affecting variability.** Figure 1 shows the relationship between temperature and output variability for both  
 404 models.

405 Within the C3 temperature sweep, increasing temperature from 0.0 to 0.7 reduces ROUGE-L from 0.991 to  
 406 0.559 (LLaMA summarization) and from 0.994 to 0.772 (LLaMA extraction). For GPT-4, the same sweep reduces  
 407 ROUGE-L from 0.848 to 0.555 (summarization) and from 0.936 to 0.789 (extraction). The *relative* rate of degradation  
 408 is comparable, but GPT-4 starts from a lower baseline owing to its inherent server-side non-determinism.

409 Notably, BERTScore F1 remains above 0.94 across all conditions, even when EMR drops to zero at  $t = 0.7$  (Table 3).  
 410 This indicates that while textual outputs diverge substantially at higher temperatures, their semantic content  
 411 remains highly similar. The gap between surface-level metrics (EMR, NED) and semantic metrics (BERTScore)  
 412 underscores that non-determinism in LLM outputs is primarily a *phrasing* phenomenon rather than a *meaning*  
 413 phenomenon—a distinction with important practical implications for downstream applications that tolerate  
 414 paraphrase variation.

## 416 5.2 Cross-Model Comparison

417 Figure 2 provides a direct visual comparison of the two models under greedy decoding.

418 Figure 3 presents a comprehensive heatmap of EMR across all model-task-condition combinations.

419 To quantify the reproducibility gap between local and API-based inference, we performed paired  $t$ -tests on per-  
 420 abstract EMR values under condition C2 (greedy decoding,  $t = 0$ ) across all 30 abstracts. For summarization, the  
 421 difference is highly significant:  $t(29) = 17.250$ ,  $p < 0.0001$ , Cohen’s  $d = 3.149$  (LLaMA 3 mean EMR = 0.947, 95%

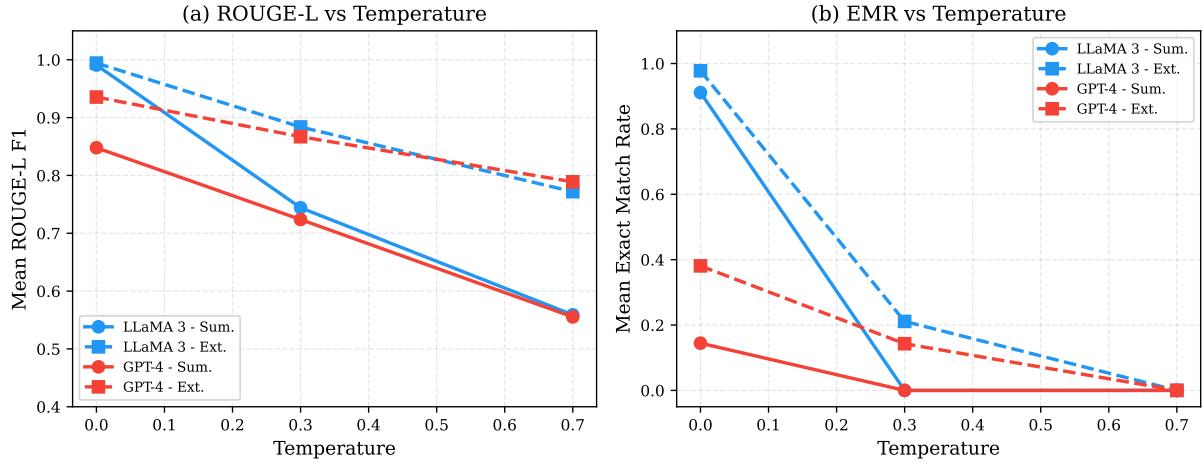


Fig. 1. Effect of temperature on output variability for both models. (a) ROUGE-L F1 decreases monotonically with temperature. (b) Exact Match Rate: LLaMA 3 starts from near-perfect reproducibility at  $t = 0$ , whereas GPT-4 starts from a lower baseline; however, both degrade at comparable rates with increasing temperature.

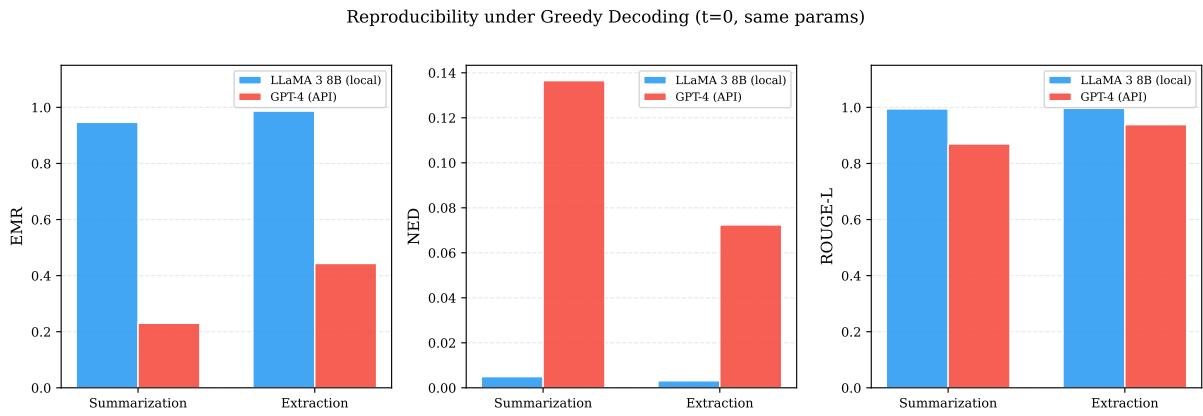


Fig. 2. Reproducibility under greedy decoding ( $t=0$ ), averaged across conditions C1, C2, and C3 at  $t=0$ : LLaMA 3 8B (local) vs. GPT-4 (API). LLaMA 3 achieves near-perfect to perfect reproducibility, while GPT-4 shows measurable variability across all metrics, particularly for summarization. Condition-specific values are given in Table 3.

CI [0.895, 0.998]; GPT-4 mean EMR = 0.230, 95% CI [0.157, 0.303]). For extraction, the gap is equally significant:  $t(29) = 8.996$ ,  $p < 0.0001$ , Cohen's  $d = 1.642$  (LLaMA 3 EMR = 0.987, 95% CI [0.959, 1.000]; GPT-4 EMR = 0.443, 95% CI [0.316, 0.571]). Both effect sizes are very large ( $d > 1.6$ ), confirming that the reproducibility difference is not only statistically significant but practically meaningful. All  $p$ -values survive Bonferroni correction at the per-family threshold  $\alpha = 0.05/6 \approx 0.008$  (six primary comparisons: EMR, NED, and ROUGE-L for each of the two tasks). A post hoc power analysis confirms that  $n = 30$  abstracts provides statistical power  $> 0.999$  for all primary comparisons (Cohen 1988).

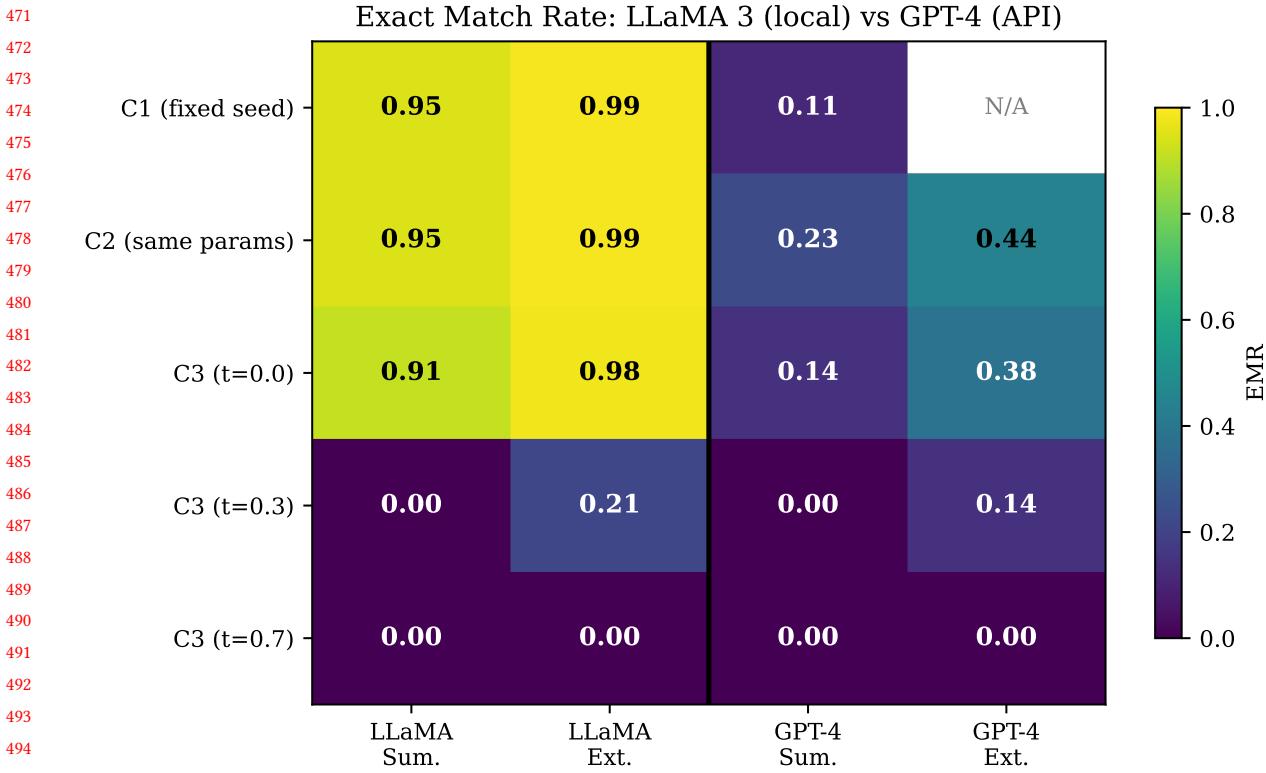


Fig. 3. Heatmap of Exact Match Rate across all experimental conditions. The left columns (LLaMA 3) show high EMR under greedy decoding, while the right columns (GPT-4) show lower EMR even at  $t = 0$ . The vertical black line separates the two models.

**Robustness check.** Since EMR values are bounded on  $[0, 1]$  and the paired differences are not normally distributed (Shapiro-Wilk:  $W = 0.894, p = 0.006$  for summarization;  $W = 0.885, p = 0.004$  for extraction), we additionally report non-parametric Wilcoxon signed-rank tests. All results remain highly significant: summarization  $W = 0.0, p < 0.001$ ; extraction  $W = 0.0, p < 0.001$ . The convergence of parametric and non-parametric tests confirms that the reproducibility gap is robust to distributional assumptions.

### 5.3 Protocol Overhead

Table 5 presents the protocol's overhead metrics across all 1,864 runs.

The protocol adds a mean overhead of **25.43 ms** per run, representing **0.545%** of the mean inference time. This is well within our target of <1%. The overhead is dominated by SHA-256 hashing and environment metadata collection; JSON serialization and file I/O contribute minimally.

Storage overhead is similarly modest: each run record occupies approximately 4.1 KB, and the complete set of 1,864 run logs, provenance documents, and Run Cards totals 19.52 MB. Note that provenance documents are generated per experimental group (i.e., per unique model–task–condition–abstract combination), yielding 331 PROV-JSON files that aggregate the individual runs within each group.

Figure 4 shows the overhead distribution broken down by model.

Table 5. Protocol overhead: logging time and storage costs for 1,864 runs (1,140 LLaMA 3 + 724 GPT-4).

Metric	Value	Unit
<i>Logging time overhead</i>		
Mean per run	25.43 ± 9.00	ms
Min / Max	11.05 / 51.88	ms
Total (1,864 runs)	47,393	ms
Mean overhead ratio	0.545%	of inference time
Max overhead ratio	1.621%	of inference time
<i>Storage overhead</i>		
Run logs (1,864 files)	7,729	KB
PROV documents (331 files)	1,736	KB
Run Cards (1,864 files)	2,610	KB
Prompt Cards + input data	7,445	KB
Total output	19.52	MB

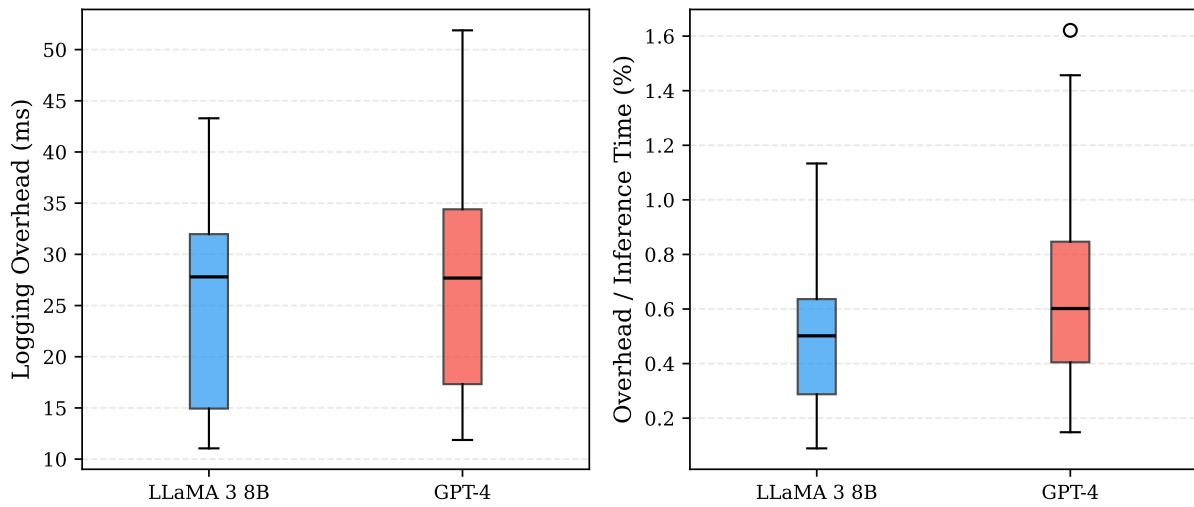


Fig. 4. Distribution of protocol overhead by model. Left: Absolute logging time (ms). Right: Overhead as a percentage of inference time. Overhead is comparable between local (LLaMA 3) and API (GPT-4) inference, consistently below 1.7%.

## 6 Discussion

The preceding results paint a nuanced picture: local inference under greedy decoding is near-perfectly reproducible, but API-based inference exhibits substantial hidden variability that researchers cannot control. Temperature is the dominant user-controllable factor, and structured tasks are inherently more reproducible than open-ended ones. We now consider what these findings mean for research practice, what the protocol enables that was previously invisible, and where the current study’s limitations lie.

### 6.1 Implications for Reproducibility Practice

Our results yield several actionable recommendations for researchers conducting generative AI experiments:

565    **Use greedy decoding with local models for maximum reproducibility.** Under  $t = 0$  with LLaMA 3 (local),  
 566 extraction achieved 98.7% EMR and summarization reached 94.7% EMR across 30 abstracts. This configuration  
 567 should be the default for any study in which output consistency is critical.

568    **Be aware of API non-determinism.** Our most consequential finding is that GPT-4, even with  $t = 0$  and  
 569 a fixed seed, produces substantially variable outputs (EMR = 0.230 for summarization, 0.443 for extraction).  
 570 Researchers using API-based models should *never assume reproducibility* without verification, and should report  
 571 multiple runs with variability metrics.

572    **Prefer structured output formats when possible.** The extraction task's consistently higher reproducibility  
 573 across both models demonstrates that output-format constraints directly improve reproducibility. Researchers  
 574 should consider whether their tasks can be reformulated as structured extraction rather than open-ended  
 575 generation.

576    **Include warm-up runs for local models.** The per-abstract analysis revealed that the first inference call  
 577 after model loading may differ from subsequent calls owing to cache initialization effects. Discarding the first run  
 578 is a straightforward practice that improves measured reproducibility. Consequently, the LLaMA 3 summarization  
 579 EMR of 0.947 may represent a conservative lower bound: with a warm-up run excluded, the effective EMR would  
 580 approach 1.000 for the remaining repetitions. Future studies should incorporate an explicit warm-up run as part  
 581 of their experimental protocol.

582    **Log comprehensively; the cost is negligible.** At 0.545% overhead and approximately 4 KB per run, there is  
 583 no practical reason not to apply comprehensive logging. The cost of not logging—namely, the inability to detect  
 584 the kind of API non-determinism documented herein—far exceeds the protocol's minimal requirements.

## 586    6.2 Local vs. API Inference: A Reproducibility Gap

587 The most significant finding of this study is the reproducibility gap between local and API-based inference. Under  
 588 nominally identical greedy decoding conditions, LLaMA 3 (local) achieves EMR = 0.987 for extraction while  
 589 GPT-4 (API) achieves only 0.443. For summarization, the gap is 0.947 vs. 0.230.

590 This gap has profound implications for the scientific use of API-based LLMs. *Without systematic logging, a*  
 591 *researcher using GPT-4 would have no way of knowing that their “deterministic” experiment produces different*  
 592 *outputs across runs.* Since our experimental design controls all user-side parameters (temperature, seed, prompt,  
 593 input), the observed variability is consistent with opaque server-side factors as the primary source. Our protocol  
 594 makes this hidden non-determinism visible, measurable, and documentable.

## 596    6.3 Task-Dependent Reproducibility

598 The difference between summarization and extraction reproducibility under identical conditions—observed  
 599 consistently across both models—is, to our knowledge, the first empirical quantification of how task structure  
 600 affects LLM output reproducibility. This finding suggests a spectrum ranging from highly constrained tasks (struc-  
 601 tured extraction, classification) to open-ended tasks (summarization, dialogue), with the degree of output-space  
 602 constraint serving as a primary determinant. Notably, even GPT-4’s extraction task (EMR = 0.443) substantially  
 603 outperforms its summarization task (EMR = 0.230), confirming that this effect is not specific to any single model.

## 604    6.4 The Role of Provenance

606 The W3C PROV graphs generated by our protocol serve multiple purposes beyond simple audit trails:

- 608    (1) **Automated comparison:** By comparing PROV graphs of two runs, one can automatically identify which  
 609 factors differed (e.g., same prompt and model but different temperatures), enabling systematic diagnosis  
 610 of non-reproducibility.

611

- 612 (2) **Lineage tracking:** When outputs are used as inputs to downstream processes (e.g., summarization  
 613 outputs fed into a meta-analysis), the provenance chain can be extended to trace any final result back to  
 614 its full generation context.  
 615 (3) **Compliance:** For regulated domains (healthcare, legal, finance), PROV documents provide the formal  
 616 evidence trail required by audit standards ([National Institute of Standards and Technology 2023](#)) and  
 617 emerging regulations such as the EU AI Act ([European Parliament and Council of the European Union  
 618 2024](#)).

619 To illustrate the diagnostic power of PROV graphs, consider two GPT-4 extraction runs on the same abstract  
 620 under condition C2 (greedy decoding,  $t = 0$ , same seed). Although the PROV entities for Prompt, InputText,  
 621 ModelVersion, and InferenceParameters are identical (verified via matching SHA-256 hashes), the Output entities  
 622 differ: output\_hash values diverge, and the wasGeneratedBy timestamps differ by several seconds. The PROV  
 623 graph thus automatically pinpoints the source of non-reproducibility: the only varying factor is the RunGeneration  
 624 activity itself, consistent with the non-determinism originating server-side. This kind of automated differential  
 625 diagnosis is infeasible without structured provenance records.  
 626

## 627 6.5 Limitations

628 We organize threats to validity following standard categories:  
 629

630 *6.5.1 Internal Validity. Sample size and statistical power.* With  $n = 30$  abstracts per condition, our study has  
 631 adequate statistical power for the primary comparisons. A post hoc power analysis using the observed effect  
 632 sizes ( $d > 1.6$ ) and  $\alpha = 0.05$  yields power  $> 0.999$  for all primary comparisons ([Cohen 1988](#)). However, for one  
 633 secondary comparison (extraction EMR under  $C3_{t=0.3}$ ,  $d = 0.207$ ), power is low (0.084), meaning that subtler  
 634 effects may go undetected in some conditions.

635 **GPT-4 C3 incomplete coverage.** Due to API quota exhaustion, GPT-4 extraction under C3 conditions covers  
 636 14–17 of 30 abstracts (summarization C3 is complete). The missing abstracts are those that would have been  
 637 processed last in execution order; since abstracts were processed sequentially by ID, this missingness is effectively  
 638 independent of abstract content or difficulty. Importantly, our central claims rest on the C2 condition (300/300  
 639 runs complete for GPT-4), and the C3 temperature sweep serves as a secondary analysis. Restricting the C3  
 640 analysis to the 14 fully-complete abstracts does not change the direction or significance of any reported trend.  
 641

642 **Warm-up confound.** As noted above, the first LLaMA 3 inference after model loading may differ from  
 643 subsequent calls due to cache initialization. This affects a small number of abstracts (4 of 30 for summarization),  
 644 reducing the aggregate EMR from  $\sim 1.0$  to 0.947. It represents an uncontrolled confound in our experimental  
 645 design.

646 **Prompt format confound.** LLaMA 3 was queried via Ollama’s /api/generate endpoint (raw completion),  
 647 whereas GPT-4 was queried via the OpenAI Chat Completions API (structured messages with system/user roles).  
 648 This difference in prompt format is inherent to the deployment paradigms under study and mirrors real-world  
 649 usage. To assess whether this confound explains the reproducibility gap, we conducted a supplementary control  
 650 experiment running LLaMA 3 via Ollama’s /api/chat endpoint (chat template matching GPT-4’s message  
 651 structure) on 10 abstracts under conditions C1 and C2 (200 runs). The results (Appendix E) show that LLaMA 3  
 652 maintains near-identical reproducibility under the chat format, confirming that the prompt-format difference  
 653 does not account for the observed local-vs-API gap.  
 654

655 *6.5.2 External Validity. Two models.* Our evaluation covers LLaMA 3 8B (local) and GPT-4 (API), representing  
 656 two deployment paradigms but only one model per category. Other models—including Claude ([Anthropic  
 657 2024](#)), Gemini ([Gemini Team et al. 2024](#)), Mixtral, and larger or smaller LLaMA variants—may exhibit different  
 658 reproducibility characteristics. Our findings about the local-vs-API gap should therefore be interpreted as a case

study of this paradigm difference rather than a universal claim. The protocol itself is model-agnostic, and we note that gpt-4-0613 is now a legacy snapshot; the very fact that newer model versions may behave differently illustrates exactly the kind of silent evolution that our protocol is designed to detect and document.

**Two tasks.** Summarization and extraction represent distinct points on the output-structure spectrum but do not cover the full range of generative AI applications (e.g., dialogue, code generation, reasoning chains). A broader task suite would strengthen generalizability.

**English-only, single domain.** Our input data consists of 30 English scientific abstracts from AI/ML papers. While this is a substantial and diverse sample within one domain, reproducibility characteristics may differ for other languages, domains (e.g., biomedical, social science), or document types.

**No multi-turn evaluation.** All experiments use single-turn interactions. Multi-turn dialogues introduce additional variability through conversation history, which our protocol logs but our experiments do not evaluate.

**6.5.3 Construct Validity. Surface-level metrics.** Our metrics (EMR, NED, ROUGE-L) capture textual rather than semantic similarity. Two outputs that are semantically equivalent but syntactically different will register as non-matching under EMR and partially divergent under NED. This is by design—our focus is on *exact* reproducibility—but it means our results may overstate the practical impact of non-determinism for downstream applications where semantic equivalence suffices.

**6.5.4 Other Considerations. Privacy.** The protocol’s environment metadata includes the machine hostname, which may reveal institutional information. Deployments in privacy-sensitive settings should anonymize this field.

**Computational cost.** The total cost was modest: ~2 GPU-hours on a consumer laptop (Apple M4, 24 GB) for 1,140 LLaMA 3 runs, plus 724 API calls to GPT-4. The carbon footprint is negligible at this scale, and the logging overhead (25 ms per run) would not materially increase energy consumption even at thousands of runs.

## 6.6 Protocol Minimality: An Ablation Analysis

To substantiate our claim that the protocol captures a *minimal* set of metadata, we conducted an ablation analysis in which we systematically removed each field group from the protocol schema and assessed which audit questions became unanswerable. We defined 10 audit questions that a reproducibility-oriented researcher might ask (e.g., “Can we verify the exact prompt used?”, “Can we detect output tampering?”, “Can we trace full provenance?”) and mapped each to the protocol fields required to answer it. For this analysis, we decomposed the Run Card’s five sections into eight finer-grained field groups by separating cross-cutting concerns: Identification, Model Context, Parameters, Input Content, Output Content, Hashing (all SHA-256 digests), Environment, and Overhead (timing and storage metadata).

The results show that removing *any* of these eight field groups renders at least one audit question unanswerable, confirming that no group is redundant. The Hashing group (SHA-256 hashes for prompts, inputs, outputs, parameters, and environment) has the highest information density: its removal affects 6 of 10 questions despite contributing only 410 bytes per run. Conversely, the Overhead group (logging time metadata) is the least connected but remains necessary for overhead assessment. The complete ablation results are available in the project repository.

This analysis demonstrates that the protocol is *minimal* in the sense that every field group is necessary for at least one audit capability, while the total overhead remains at approximately 4,052 bytes per run.

## 6.7 Practical Costs and Adoption

One concern with any new protocol is whether the adoption burden is justified. We address this concretely:

- **Implementation effort:** Our reference implementation adds approximately 600 lines of Python (the protocol core) to an existing workflow. Integration requires 3–5 function calls per run.
- **Runtime cost:** 25 ms per run, negligible compared to inference times of seconds to minutes for typical LLM calls.
- **Storage cost:** 4 KB per run. Even at scale (10,000 runs), total storage is approximately 40 MB—less than a single model checkpoint.
- **Learning curve:** The protocol uses standard JSON and W3C PROV, requiring no specialized knowledge beyond basic Python.

Against these modest costs, the protocol provides complete audit trails, automated provenance graphs, tamper-detectable outputs via cryptographic hashing, and structured metadata that enable systematic reproducibility analysis.

## 7 Conclusion

We presented a lightweight protocol for logging, versioning, and provenance tracking of generative AI experiments, introducing Prompt Cards and Run Cards as novel documentation artifacts and adopting the W3C PROV data model for machine-readable provenance graphs. Through 1,864 controlled experiments with LLaMA 3 8B (local) and GPT-4 (API) across 30 scientific abstracts and two NLP tasks, we demonstrated four key findings:

- (1) **Local inference is substantially more reproducible than API-based inference.** Under identical greedy decoding settings, LLaMA 3 achieves EMR = 0.987 for extraction while GPT-4 achieves only 0.443, consistent with substantial server-side non-determinism that is invisible without systematic logging (paired  $t$ -test:  $p < 0.0001$ , Cohen's  $d > 1.6$ ).
- (2) **Task structure is a primary determinant of reproducibility.** Structured extraction consistently outperforms open-ended summarization across both models, with the JSON format constraint reducing the model's output space.
- (3) **Temperature is the dominant user-controllable factor.** Within the C3 temperature sweep, increasing from  $t = 0.0$  to  $t = 0.7$  reduces ROUGE-L from 0.991 to 0.559 (LLaMA summarization) and from 0.936 to 0.789 (GPT-4 extraction), while seed variation has no measurable effect under greedy decoding for local models.
- (4) **Comprehensive provenance logging adds negligible overhead:** 0.545% of inference time and approximately 4 KB per run, thereby removing any practical argument against systematic documentation.

These findings carry a broader implication: a substantial portion of published research that relies on API-based LLMs may contain non-reproducible results without the authors' knowledge. The cost of systematic provenance logging—half a percent of inference time and four kilobytes per run—is trivially small compared to the cost of publishing non-reproducible science.

Looking ahead, we plan to (i) expand the model suite to include Claude ([Anthropic 2024](#)), Gemini ([Gemini Team et al. 2024](#)), and open-weight models of varying sizes; (ii) extend the task coverage to dialogue, code generation, and multi-turn interactions; and (iii) develop automated reproducibility scoring based on provenance graph analysis. Ultimately, we envision a future in which every generative AI output carries a provenance certificate, and reproducibility metrics are reported alongside accuracy as a standard component of empirical evaluation.

The reference implementation, all 1,864 run records, provenance documents, and analysis scripts are publicly available to support adoption and independent verification.

## 753 Acknowledgments

754 This work was supported by UTFPR – Universidade Tecnológica Federal do Paraná. The experiments were  
 755 conducted using locally deployed open-weight models to ensure full reproducibility of the computational envi-  
 756 ronment.

## 757 Data Availability Statement

758 The reference implementation, all 1,864 run records (JSON), PROV-JSON provenance documents, Run Cards,  
 759 Prompt Cards, input data, analysis scripts, and generated figures are publicly available at:

760 <https://github.com/Roverlucas/genai-reproducibility-protocol>

761 The repository includes instructions for reproducing all experiments and regenerating all tables and figures from  
 762 the raw data.

## 763 Author Contributions

764 Following the CRediT (Contributor Roles Taxonomy) framework: **Lucas Rover**: Conceptualization, Methodol-  
 765 ogy, Software, Validation, Formal Analysis, Investigation, Data Curation, Writing – Original Draft, Writing –  
 766 Review & Editing, Visualization, Project Administration. **Yara de Souza Tadano**: Supervision, Conceptualization,  
 767 Methodology, Writing – Review & Editing, Project Administration.

## 768 Conflict of Interest

769 The authors declare no conflicts of interest. This research was conducted independently at UTFPR with no  
 770 external funding from commercial AI providers. The use of OpenAI’s GPT-4 API was for research evaluation  
 771 purposes only and does not constitute an endorsement.

## 772 Use of AI-Assisted Tools

773 The authors used AI-assisted tools (Claude, Anthropic) during the preparation of this manuscript for language  
 774 editing, code development support, and data analysis scripting. All AI-generated content was critically reviewed,  
 775 validated, and revised by the authors, who take full responsibility for the accuracy and integrity of the final  
 776 manuscript. The scientific design, experimental execution, interpretation of results, and intellectual contributions  
 777 are entirely the authors’ own work.

## 778 References

- 779 J. Achiam et al.. 2023. *GPT-4 Technical Report*. arXiv preprint. (2023). arXiv: [2303.08774 \(cs.CL\)](https://arxiv.org/abs/2303.08774).  
 780 Anthropic. 2024. *The Claude Model Family*. (2024). <https://www.anthropic.com/clause>.  
 781 B. Atil et al.. 2024. *Non-Determinism of “Deterministic” LLM Settings*. arXiv preprint. (2024). arXiv: [2408.04667 \(cs.CL\)](https://arxiv.org/abs/2408.04667).  
 782 M. Baker. 2016. “1,500 Scientists Lift the Lid on Reproducibility.” *Nature*, 533, 7604, 452–454. doi:[10.1038/533452a](https://doi.org/10.1038/533452a).  
 783 A. Belz, S. Agarwal, A. Shimorina, and E. Reiter. 2021. “A Systematic Review of Reproducibility Research in Natural Language Processing.” In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational  
 784 Linguistics, 381–393. doi:[10.18653/v1/2021.eacl-main.29](https://doi.org/10.18653/v1/2021.eacl-main.29).  
 785 L. Biewald. 2020. *Experiment Tracking with Weights and Biases*. (2020). <https://wandb.com/>.  
 786 R. Bommasani et al.. 2022. *On the Opportunities and Risks of Foundation Models*. arXiv preprint. (2022). arXiv: [2108.07258 \(cs.LG\)](https://arxiv.org/abs/2108.07258).  
 787 T. Brown et al.. 2020. “Language Models are Few-Shot Learners.” In: *Advances in Neural Information Processing Systems*. Vol. 33, 1877–1901.  
 788 Y. Chen, J. Li, X. Liu, and Y. Li. 2023. *On the Reproducibility of ChatGPT in NLP Tasks*. arXiv preprint. (2023). arXiv: [2304.02554 \(cs.CL\)](https://arxiv.org/abs/2304.02554).  
 789 J. Cohen. 1988. *Statistical Power Analysis for the Behavioral Sciences*. (2nd ed.). Lawrence Erlbaum Associates.  
 790 J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 2019. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*. Association for  
 791 Computational Linguistics, 4171–4186. doi:[10.18653/v1/N19-1423](https://doi.org/10.18653/v1/N19-1423).

- 800 J. Dodge, S. Gururangan, D. Card, R. Schwartz, and N. A. Smith. 2019. “Show Your Work: Improved Reporting of Experimental Results.”  
 801 In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics,  
 802 2185–2194. doi:[10.18653/v1/D19-1224](https://doi.org/10.18653/v1/D19-1224).
- 803 European Parliament and Council of the European Union. 2024. *Regulation (EU) 2024/1689 Laying Down Harmonised Rules on Artificial Intelligence (AI Act)*. (2024). <https://eur-lex.europa.eu/eli/reg/2024/1689/oj>.
- 804 T. Gebru, J. Morgenstern, B. Vecchione, J. W. Vaughan, H. Wallach, H. Daumé III, and K. Crawford. 2021. “Datasheets for Datasets.”  
 805 *Communications of the ACM*, 64, 12, 86–92. doi:[10.1145/3458723](https://doi.org/10.1145/3458723).
- 806 Gemini Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, et al. 2024. *Gemini: A Family of Highly Capable Multimodal Models*. arXiv preprint. (2024). arXiv: [2312.11805 \(cs.CL\)](https://arxiv.org/abs/2312.11805).
- 807 A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, et al. 2024. *The LLaMA 3 Herd of Models*. arXiv preprint. (2024). arXiv: [2407.21783 \(cs.AI\)](https://arxiv.org/abs/2407.21783).
- 808 O. E. Gundersen, Y. Gil, and D. W. Aha. 2018. “On Reproducible AI: Towards Reproducible Research, Open Science, and Digital Scholarship in AI Publications.” *AI Magazine*, 39, 3, 56–68. doi:[10.1609/aimag.v39i3.2816](https://doi.org/10.1609/aimag.v39i3.2816).
- 809 O. E. Gundersen, M. Helmert, and H. H. Hoos. 2024. “Improving Reproducibility in AI Research: Four Mechanisms Adopted by JAIR.” *Journal of Artificial Intelligence Research*, 81, 1019–1041. doi:[10.1613/jair.116905](https://doi.org/10.1613/jair.116905).
- 810 O. E. Gundersen and S. Kjensmo. 2018. “State of the Art: Reproducibility in Artificial Intelligence.” *Proceedings of the AAAI Conference on Artificial Intelligence*, 32, 1, 1644–1651. doi:[10.1609/aaai.v32i1.11503](https://doi.org/10.1609/aaai.v32i1.11503).
- 811 M. Herschel, R. Diestelkämper, and H. Ben Lahmar. 2017. “A Survey on Provenance: What for? What form? What from?” *The VLDB Journal*, 26, 6, 881–906. doi:[10.1007/s00778-017-0486-1](https://doi.org/10.1007/s00778-017-0486-1).
- 812 M. Hutson. 2018. “Artificial Intelligence Faces Reproducibility Crisis.” *Science*, 359, 6377, 725–726. doi:[10.1126/science.359.6377.725](https://doi.org/10.1126/science.359.6377.725).
- 813 S. Kapoor and A. Narayanan. 2023. “Leakage and the Reproducibility Crisis in Machine-Learning-Based Science.” *Patterns*, 4, 9, 100804. doi:[10.1016/j.patter.2023.100804](https://doi.org/10.1016/j.patter.2023.100804).
- 814 R. Kuprieiev, D. Petrov, and Iterative. 2024. *DVC: Data Version Control*. (2024). <https://dvc.org/>.
- 815 LangChain. 2023. *LangSmith: A Platform for Building Production-Grade LLM Applications*. (2023). <https://smith.langchain.com/>.
- 816 V. I. Levenshtein. 1966. “Binary Codes Capable of Correcting Deletions, Insertions, and Reversals.” *Soviet Physics Doklady*, 10, 8, 707–710.
- 817 P. Liang et al. 2023. “Holistic Evaluation of Language Models.” *Transactions on Machine Learning Research*. <https://openreview.net/forum?id=iO4LZibEqW>.
- 818 C.-Y. Lin. 2004. “ROUGE: A Package for Automatic Evaluation of Summaries.” In: *Proceedings of the ACL-04 Workshop on Text Summarization Branches Out*. Association for Computational Linguistics, 74–81.
- 819 M. Mitchell, S. Wu, A. Zaldivar, P. Barnes, L. Vasserman, B. Hutchinson, E. Spitzer, I. D. Raji, and T. Gebru. 2019. “Model Cards for Model Reporting.” In: *Proceedings of the Conference on Fairness, Accountability, and Transparency*. ACM, 220–229. doi:[10.1145/3287560.3287596](https://doi.org/10.1145/3287560.3287596).
- 820 L. Moreau and P. Missier. 2013. *PROV-DM: The PROV Data Model*. W3C Recommendation. World Wide Web Consortium. <https://www.w3.org/TR/prov-dm/>.
- 821 National Institute of Standards and Technology. 2023. *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*. Tech. rep. U.S. Department of Commerce. doi:[10.6028/NIST.AI.100-1](https://doi.org/10.6028/NIST.AI.100-1).
- 822 Ollama. 2024. *Ollama: Run Large Language Models Locally*. (2024). <https://ollama.com/>.
- 823 OpenAI. 2023. *OpenAI Evals: A Framework for Evaluating LLMs*. (2023). <https://github.com/openai/evals>.
- 824 S. Ouyang, J. M. Zhang, M. Harman, and M. Wang. 2024. “An Empirical Study of the Non-determinism of ChatGPT in Code Generation.” *ACM Transactions on Software Engineering and Methodology*, 34, 2, 1–28. doi:[10.1145/3697010](https://doi.org/10.1145/3697010).
- 825 G. Padovani, V. Ananthraj, and S. Fiore. 2025. “yProv4ML: Effortless Provenance Tracking for Machine Learning Systems.” *SoftwareX*, 29, 102028. doi:[10.1016/j.softx.2025.102028](https://doi.org/10.1016/j.softx.2025.102028).
- 826 J. Pineau, P. Vincent-Lamarre, K. Sinha, V. Larivière, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and H. Larochelle. 2021. “Improving Reproducibility in Machine Learning Research: A Report from the NeurIPS 2019 Reproducibility Program.” *Journal of Machine Learning Research*, 22, 164, 1–20. <https://jmlr.org/papers/v22/20-303.html>.
- 827 C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. 2020. “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer.” *Journal of Machine Learning Research*, 21, 140, 1–67. <https://jmlr.org/papers/v21/20-074.html>.
- 828 S. Samuel and B. König-Ries. 2022. “A Provenance-based Semantic Approach to Support Understandability, Reproducibility, and Reuse of Scientific Experiments.” *Journal of Biomedical Semantics*, 13, 1, 1–30. doi:[10.1186/s13326-022-00263-z](https://doi.org/10.1186/s13326-022-00263-z).
- 829 V. Stodden, M. McNutt, D. H. Bailey, E. Delelman, Y. Gil, B. Hanson, M. A. Heroux, J. P. Ioannidis, and M. Taufer. 2016. “Enhancing Reproducibility for Computational Methods.” *Science*, 354, 6317, 1240–1241. doi:[10.1126/science.aah6168](https://doi.org/10.1126/science.aah6168).
- 830 A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. 2017. “Attention is All You Need.” In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc.
- 831 J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. 2022. “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models.” In: *Advances in Neural Information Processing Systems*. Vol. 35, 24824–24837.

- 847 M. D. Wilkinson et al. 2016. "The FAIR Guiding Principles for Scientific Data Management and Stewardship." *Scientific Data*, 3, 160018.  
848 doi:[10.1038/sdata.2016.18](https://doi.org/10.1038/sdata.2016.18).
- 849 J. Yuan et al. 2025. "Understanding and Mitigating Numerical Sources of Nondeterminism in LLM Inference." In: *Advances in Neural Information  
850 Processing Systems*. Vol. 38. Curran Associates, Inc. arXiv: [2506.09501 \(cs.LG\)](https://arxiv.org/abs/2506.09501).
- 850 M. Zaharia et al.. 2018. "Accelerating the Machine Learning Lifecycle with MLflow." *IEEE Data Engineering Bulletin*, 41, 4, 39–45.
- 851 T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. 2020. "BERTScore: Evaluating Text Generation with BERT." In: *Proceedings of the  
852 8th International Conference on Learning Representations*. <https://openreview.net/forum?id=SkeHuCVFDr>.
- 853 Y. Zhu, P. Zhang, E. Haq, P. Hui, and G. Buchanan. 2023. *Can ChatGPT Reproduce Human-Generated Labels? A Study of Social Computing  
854 Tasks*. arXiv preprint. (2023). arXiv: [2304.10145 \(cs.CL\)](https://arxiv.org/abs/2304.10145).
- 855
- 856
- 857
- 858
- 859
- 860
- 861
- 862
- 863
- 864
- 865
- 866
- 867
- 868
- 869
- 870
- 871
- 872
- 873
- 874
- 875
- 876
- 877
- 878
- 879
- 880
- 881
- 882
- 883
- 884
- 885
- 886
- 887
- 888
- 889
- 890
- 891
- 892
- 893

## 894 A Reproducibility Checklist

895 The following checklist is designed for self-assessment of reproducibility in generative AI studies. Each item  
 896 maps to a specific field or artifact in our protocol.

897

### 898 Prompt Documentation

- 899 (1) Is the exact prompt text recorded and versioned? [Prompt Card: prompt\_text, prompt\_hash]
- 900 (2) Are design assumptions and limitations documented? [Prompt Card: assumptions, limitations]
- 901 (3) Is the expected output format specified? [Prompt Card: expected\_output\_format]
- 902 (4) Is the interaction regime documented (single/multi-turn)? [Prompt Card: interaction\_regime]

### 904 Model and Environment

- 905 (5) Is the model name and version recorded? [Run Card: model\_name, model\_version]
- 906 (6) Are model weights hashed for identity verification? [Run Card: weights\_hash]
- 907 (7) Is the execution environment fingerprinted? [Run Card: environment, environment\_hash]
- 908 (8) Is the source code version recorded? [Run Card: code\_commit]

### 910 Execution and Output

- 912 (9) Are all inference parameters logged? [Run Card: inference\_params]
- 913 (10) Is the random seed recorded? [Run Card: inference\_params.seed]
- 914 (11) Is the output cryptographically hashed? [Run Card: output\_hash]
- 915 (12) Are execution timestamps recorded? [Run Card: timestamp\_start, timestamp\_end]
- 916 (13) Is logging overhead measured separately? [Run Card: logging\_overhead\_ms]

### 917 Provenance

- 919 (14) Is a provenance graph generated per group? [PROV-JSON document]
- 920 (15) Are provenance documents in an interoperable format? [W3C PROV standard]

## 922 B Run Card Schema

923 The complete Run Card schema, with data types and descriptions:

924

925 Listing 1. Run Card JSON schema (simplified).

```

926 {
927   "run_id": "string (unique identifier)",
928   "task_id": "string (task identifier)",
929   "task_category": "string (e.g., summarization)",
930   "prompt_hash": "string (SHA-256 of prompt)",
931   "prompt_text": "string (full prompt text)",
932   "input_text": "string (input to the model)",
933   "input_hash": "string (SHA-256 of input)",
934   "model_name": "string (e.g., llama3:8b)",
935   "model_version": "string (e.g., 8.0B)",
936   "weights_hash": "string (SHA-256 of weights)",
937   "model_source": "string (e.g., ollama-local)",
938   "inference_params": {
939     "temperature": "float",
940     "top_p": "float",
  
```

```

941   16   "top_k": "integer",
942   17   "max_tokens": "integer",
943   18   "seed": "integer|null",
944   19   "decoding_strategy": "string"
945 },
946 "params_hash": "string (SHA-256 of params)",
947 "environment": {
948   23   "os": "string",
949   24   "os_version": "string",
950   25   "architecture": "string",
951   26   "python_version": "string",
952   27   "hostname": "string",
953   28   "timestamp": "ISO 8601 datetime"
954 },
955 "environment_hash": "string (SHA-256)",
956 "code_commit": "string (git commit hash)",
957 "researcher_id": "string",
958 "affiliation": "string",
959 "timestamp_start": "ISO 8601 datetime",
960 "timestamp_end": "ISO 8601 datetime",
961 "output_text": "string (model output)",
962 "output_hash": "string (SHA-256 of output)",
963 "output_metrics": "object (task-specific)",
964 "execution_duration_ms": "float",
965 "logging_overhead_ms": "float",
966 "storage_kb": "float",
967 "system_logs": "string (raw system info)",
968 "errors": "array of strings"
969 }
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987

```

## C Example PROV-JSON Document

An abbreviated example of a PROV-JSON document generated for a single summarization run:

Listing 2. Abbreviated PROV-JSON for a summarization run.

```

1 {
2   "prefix": {
3     "genai": "https://genai-prov.org/ns#",
4     "prov": "http://www.w3.org/ns/prov#"
5   },
6   "entity": {
7     "genai:prompt_c9644358": {
8       "prov:type": "genai:Prompt",
9       "genai:hash": "c9644358805b...",
10      "genai:task_category": "summarization"
11    },
12    "genai:model_llama3_8b": {
13      "prov:type": "genai:ModelVersion",
14      "genai:name": "llama3:8b",
15      "genai:source": "ollama-local"
16    }
17  }
18}
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
887
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
917
918
919
919
920
921
922
923
924
925
926
927
927
928
929
929
930
931
932
933
934
935
936
937
937
938
939
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987

```

```

988     16 },
989     17   "genai:output_590d0835": {
990     18     "prov:type": "genai:Output",
991     19     "genai:hash": "590d08359e7d..."
992     20   }
993     21 },
994     22   "activity": {
995     23     "genai:run_llama3_8b_sum_001_C1_rep0": {
996     24       "prov:type": "genai:RunGeneration",
997     25       "prov:startTime": "2026-02-07T21:54:34Z",
998     26       "prov:endTime": "2026-02-07T21:54:40Z"
999     27     }
1000    28   },
1001    29   "wasGeneratedBy": {
1002    30     "_:wGB1": {
1003    31       "prov:entity": "genai:output_590d0835",
1004    32       "prov:activity": "genai:run_llama3_8b..."
1005    33     }
1006    34   },
1007    35   "used": {
1008    36     "_:u1": {
1009    37       "prov:activity": "genai:run_llama3_...",
1010    38       "prov:entity": "genai:prompt_c9644358"
1011    39     }
1012    40   },
1013    41   "agent": {
1014    42     "genai:researcher_lucas_rover": {
1015    43       "prov:type": "prov:Person",
1016    44       "genai:affiliation": "UTFPR"
1017    45     }
1018    46   },
1019    47   "wasAssociatedWith": {
1020    48     "_:wAW1": {
1021    49       "prov:activity": "genai:run_llama3_...",
1022    50       "prov:agent": "genai:researcher_..."
1023    51     }
1024    52   }
1025    53 }

```

## D JSON Extraction Quality

Table 6 presents JSON-specific quality metrics for the structured extraction task. Two notable patterns emerge.

First, LLaMA 3 never produces raw-valid JSON: all 570 extraction outputs contain preamble text (e.g., “Here is the extracted information in JSON format:”) before the JSON object, despite the prompt explicitly requesting “JSON only, no explanation.” After extracting the embedded JSON via regex, validity rates reach 100% under greedy decoding, degrading slightly at higher temperatures (92.2% at  $t = 0.7$ ). GPT-4, by contrast, always produces raw-valid JSON with 100% schema compliance across all conditions. This instruction-following gap is consistent with the different prompt interfaces: the chat completion API’s structured message format may better signal the expected output format.

1035 Table 6. JSON extraction quality metrics by model and condition. *Raw Valid* = output parses directly as JSON; *Extracted*  
 1036 *Valid* = JSON extracted via regex from outputs containing preamble text; *Schema* = all five expected fields present; *Field EMR*  
 1037 = pairwise exact match across runs for each extracted field. LLaMA 3 always prepends introductory text (e.g., “Here is the  
 1038 extracted information in JSON format.”), yielding 0% raw validity but near-perfect extracted validity at  $t=0$ .

1040	Model	Cond.	Raw	Extr.	Schema	Field-Level EMR					Overall			
						1041 Valid	1041 Valid	1041 Compl.	1042 obj	1042 meth	1042 key_r	1042 mod/sys	1042 bench	1043 Field EMR
1045	LLaMA 3	C1 ( $t=0$ )	0%	100%	100%	0.027	0.027	0.040	0.090	0.174	0.071	0.071	0.071	0.071
		C2 ( $t=0$ )	0%	100%	100%	0.027	0.027	0.040	0.090	0.174	0.071	0.071	0.071	0.071
		C3 ( $t=0.0$ )	0%	100%	100%	0.022	0.022	0.036	0.085	0.170	0.067	0.067	0.067	0.067
		C3 ( $t=0.3$ )	0%	97.8%	97.8%	0.017	0.010	0.027	0.092	0.213	0.072	0.072	0.072	0.072
		C3 ( $t=0.7$ )	0%	92.2%	92.2%	0.012	0.004	0.022	0.068	0.295	0.080	0.080	0.080	0.080
1049	GPT-4	C2 ( $t=0$ )	100%	100%	100%	0.021	0.018	0.017	0.037	0.080	0.035	0.035	0.035	0.035
		C3 ( $t=0.0$ )	100%	100%	100%	0.027	0.018	0.020	0.036	0.121	0.045	0.045	0.045	0.045
		C3 ( $t=0.3$ )	100%	100%	100%	0.015	0.011	0.016	0.048	0.159	0.050	0.050	0.050	0.050
		C3 ( $t=0.7$ )	100%	100%	100%	0.003	0.003	0.005	0.028	0.102	0.028	0.028	0.028	0.028

1052 Second, field-level exact match rates reveal that structured fields (benchmark, model\_or\_system) are 2–6×  
 1053 more reproducible than open-ended fields (method, key\_result), consistent with the finding that structured tasks  
 1054 achieve higher overall EMR. Note that field-level EMR measures pairwise exact match on individual extracted  
 1055 field values (short strings sensitive to minor wording differences), whereas the overall extraction EMR in Table 3  
 1056 measures whole-output character identity; these are complementary but distinct metrics.

## 1058 E Chat-Format Control Experiment

1059 To assess whether the prompt-format difference between LLaMA 3 (completion-style via /api/generate) and  
 1060 GPT-4 (chat-style via Chat Completions) contributes to the observed reproducibility gap, we conducted a sup-  
 1061 plementary control experiment running LLaMA 3 8B through Ollama’s /api/chat endpoint, which applies the  
 1062 model’s chat template (including special tokens for system/user/assistant roles) in the same message structure  
 1063 used by GPT-4.

1064 **Design:** 10 abstracts × 2 tasks × 2 conditions (C1, C2) × 5 repetitions = 200 runs, all under greedy decoding  
 1065 ( $t = 0$ ).

1066 **Results:** Table 7 compares the chat-format control with the original completion-format results for the same 10  
 1067 abstracts. The two prompt formats produce *identical* variability metrics across all conditions: summarization  
 1068 EMR = 0.929, NED = 0.0066, and ROUGE-L = 0.9922 in both modes; extraction achieves perfect reproducibility  
 1069 (EMR = 1.000) regardless of interface. The 0.929 summarization EMR reflects the warm-up effect on 2 of 10  
 1070 abstracts—the same pattern observed in the full 30-abstract experiment. These results confirm that prompt  
 1071 format is not a source of variability, and the reproducibility gap between LLaMA 3 and GPT-4 is attributable  
 1072 to deployment-side factors (server infrastructure, floating-point non-determinism across GPU types, request  
 1073 batching).

1075 Received February 2026

1082 Table 7. Prompt-format control: LLaMA 3 8B via completion (/api/generate) vs. chat (/api/chat) for 10 abstracts under  
 1083 greedy decoding ( $t=0$ ). EMR computed over conditions C1 and C2 combined.

1084	Task	Metric	Completion	Chat
1085	Summarization	EMR $\uparrow$	0.929	0.929
1086		NED $\downarrow$	0.0066	0.0066
1087		ROUGE-L $\uparrow$	0.9922	0.9922
1088	Extraction	EMR $\uparrow$	1.000	1.000
1089		NED $\downarrow$	0.0000	0.0000
1090		ROUGE-L $\uparrow$	1.0000	1.0000

1091 Note: Completion and chat formats yield identical metrics for all 10 abstracts under greedy decoding, confirming prompt format is not a  
 1092 source of variability.

1093  
 1094  
 1095  
 1096  
 1097  
 1098  
 1099  
 1100  
 1101  
 1102  
 1103  
 1104  
 1105  
 1106  
 1107  
 1108  
 1109  
 1110  
 1111  
 1112  
 1113  
 1114  
 1115  
 1116  
 1117  
 1118  
 1119  
 1120  
 1121  
 1122  
 1123  
 1124  
 1125  
 1126  
 1127  
 1128