

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 3,604 controlled experiments. These experiments employ five models—three locally deployed (LLaMA 3 8B, Mistral 7B, Gemma 2 9B) and two API-served (GPT-4, Claude Sonnet 4.5)—on four NLP tasks. All five models are evaluated on single-turn extraction and summarization under five conditions varying seed, temperature, and decoding strategy (10–30 abstracts per model; GPT-4 primarily under variable-seed greedy decoding due to quota constraints). Multi-turn refinement and RAG extraction are evaluated for the three local models and Claude Sonnet 4.5 under greedy decoding (10 abstracts each). 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 ($t=0$), local models achieve near-perfect reproducibility: Gemma 2 9B reaches EMR = 1.000 across all tasks, LLaMA 3 attains EMR = 0.987 for extraction, and Mistral 7B achieves EMR = 0.960. By contrast, API-served models exhibit substantial hidden non-determinism: GPT-4 achieves only EMR = 0.443 for extraction, while Claude Sonnet 4.5 achieves EMR = 0.190 for extraction and EMR = 0.020 for summarization—the lowest observed in our study. This local-vs-API reproducibility gap (average single-turn EMR: 0.956 vs. 0.221, a more than 4-fold difference) is confirmed across two independent API providers. The gap extends to complex interaction regimes: under multi-turn refinement and RAG extraction, local models maintain high reproducibility ($\text{EMR} \geq 0.880$), while Claude Sonnet 4.5—the only API model tested on these tasks—achieves EMR = 0.040 for multi-turn and EMR = 0.000 for RAG. The protocol adds less than 1% overhead across all five models.

Conclusions: Our results provide evidence that (1) API-served models exhibit substantial non-determinism under greedy decoding that is not attributable to user-controllable parameters, a pattern observed independently for both GPT-4 and Claude; (2) locally deployed models achieve near-perfect to perfect bitwise reproducibility under greedy decoding; (3) the local-vs-API gap extends to multi-turn refinement and RAG extraction, where

*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:

1 Claude Sonnet 4.5 (the only API model tested on these tasks) achieves near-zero EMR while local models maintain
 2 EMR ≥ 0.880 ; (4) temperature is the dominant user-controllable factor affecting variability; and (5) comprehensive
 3 provenance logging adds negligible overhead (<1%). The protocol, reference implementation, and all experimental
 4 data are publicly available.

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

7 Additional Key Words and Phrases: reproducibility, large language models, non-determinism, provenance, generative
 8 AI, experiment tracking, W3C PROV, scientific methodology
 9

10 **JAIR Associate Editor:**

11 **JAIR Reference Format:**

12 Lucas Rover and Yara de Souza Tadano. 2026. Hidden Non-Determinism in Large Language Model APIs: A
 13 Lightweight Provenance Protocol for Reproducible Generative AI Research. *Journal of Artificial Intelligence
 14 Research* (2026), 37 pages.

15 1 Introduction

16 When a researcher queries a cloud-hosted LLM with the same prompt and temperature zero, one would
 17 reasonably expect identical outputs. Our experiments show otherwise: across five controlled seeds under
 18 greedy decoding, GPT-4 produces the same extraction result only 44% of the time, and Claude Sonnet
 19 4.5 achieves only 19%. Meanwhile, locally deployed models such as Gemma 2 9B produce *perfectly*
 20 *identical* outputs every time. This hidden, provider-dependent non-determinism exemplifies a fundamental
 21 challenge introduced by the rapid adoption of large language models (LLMs) in scientific research: how to
 22 ensure that studies relying on generative AI outputs are reproducible, auditable, and scientifically rigorous.
 23 Unlike traditional computational experiments, in which deterministic algorithms produce identical results
 24 given identical inputs, LLMs exhibit inherent variability in their outputs due to stochastic sampling,
 25 floating-point non-determinism, and opaque model-versioning practices (Y. Chen et al. 2023; Zhu et al.
 26 2023).

27 Importantly, “non-reproducible” does not necessarily mean “unreliable”: our experiments also show
 28 that semantic similarity (measured by BERTScore F1) remains above 0.94 across all conditions, even
 29 when exact textual match drops to zero. In other words, API outputs typically convey the same *meaning*
 30 despite differing in *phrasing*—but this distinction is invisible without systematic measurement, and many
 31 downstream analyses (meta-analyses, comparative studies, regulatory audits) require exact reproducibility.

32 A related subtlety concerns the `seed` parameter offered by some APIs. For API-served models, the seed
 33 parameter is advisory, not a guarantee of determinism: OpenAI explicitly documents that “determinism
 34 is not guaranteed” even when a seed is specified (OpenAI 2024), and Anthropic’s Claude API does not
 35 support a seed parameter at all. Our experimental design accounts for this by treating seed variation as a
 36 control condition and measuring actual output reproducibility directly, rather than relying on API-side
 37 determinism guarantees.

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

46

48 Existing experiment-tracking tools such as MLflow (Zaharia et al. 2018), Weights & Biases (Biewald
 49 2020), and DVC (Kuprieiev et al. 2024) were designed primarily for training pipelines and numerical
 50 metrics. Although valuable for their intended purposes, these tools lack features critical for generative AI
 51 studies: structured prompt versioning, cryptographic output hashing for tamper detection, provenance
 52 graphs linking outputs to their full generation context, and environment fingerprinting specific to
 53 inference-time conditions.

54 In this paper, we make three contributions, with the protocol design as the primary and most durable
 55 contribution:

- 56 (1) **A lightweight, standards-based protocol** for logging, versioning, and provenance tracking of
 57 generative AI experiments. The protocol introduces *Prompt Cards* and *Run Cards* as structured
 58 documentation artifacts, and adopts the W3C PROV data model (Moreau and Missier 2013) for
 59 machine-readable provenance graphs. It operationalizes—and extends to generative AI workflows—
 60 the reproducibility checklist and badge mechanisms recently adopted by JAIR (Gundersen,
 61 Helmert, et al. 2024), providing machine-readable infrastructure that automates what those
 62 mechanisms require researchers to document manually.
- 63 (2) **A large-scale empirical case study** demonstrating both the protocol’s effectiveness and the
 64 reproducibility characteristics of LLM outputs in the models and snapshots evaluated. Through
 65 3,604 controlled experiments with five models—three locally deployed (LLaMA 3 8B, Mistral 7B,
 66 Gemma 2 9B) and two API-served (GPT-4, Claude Sonnet 4.5)—across four tasks (extraction,
 67 summarization, multi-turn refinement, RAG extraction), 30 abstracts, and five conditions, we
 68 quantify output variability using four complementary metrics and measure the protocol’s overhead.
 69 Our results document a striking reproducibility gap between local and API-based inference in the
 70 evaluated models that is invisible without systematic logging.
- 71 (3) **A reference implementation** in Python that demonstrates the protocol’s practical applicability,
 72 together with all experimental data, to facilitate adoption and independent verification.

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

78 2 Related Work

79 2.1 Reproducibility in AI Research

80 The reproducibility crisis in AI has been documented extensively. Gundersen and Kjensmo (2018) surveyed
 81 400 AI papers and found that only 6% provided sufficient information for full reproducibility. Pineau
 82 et al. (2021) reported on the NeurIPS 2019 Reproducibility Program, which introduced reproducibility
 83 checklists and found significant gaps between reported and actual reproducibility. More recently, Gundersen,
 84 Helmert, et al. (2024) described four institutional mechanisms adopted by JAIR—reproducibility checklists,
 85 structured abstracts, badges, and reproducibility reports—establishing a community standard for what
 86 should be documented in AI research. Gundersen, Gil, et al. (2018) identified three levels of reproducibility
 87 in AI—method, data, and experiment—and argued that all three are necessary for scientific progress.
 88 Belz et al. (2021) conducted a systematic review of 601 NLP papers and confirmed pervasive under-
 89 reporting of experimental details, while Dodge et al. (2019) proposed improved reporting standards for
 90 ML experiments, including confidence intervals and significance tests across multiple runs. More broadly,
 91 Kapoor and A. Narayanan (2023) identified data leakage as a widespread driver of irreproducible results
 92 across 17 scientific fields that use ML-based methods.

95 For generative AI specifically, Y. Chen et al. (2023) demonstrated that ChatGPT’s outputs on NLP
 96 benchmarks exhibit non-trivial variability across identical queries, even with temperature set to zero.
 97 Zhu et al. (2023) showed that reproducibility degrades further when tasks involve subjective judgment,
 98 such as social computing annotations. Most recently, Atil et al. (2024) systematically measured the
 99 non-determinism of five LLMs under supposedly deterministic settings across eight tasks, finding accuracy
 100 variations up to 15% across runs and introducing the Total Agreement Rate (TAR) metric. Ouyang et al.
 101 (2024) confirmed that temperature zero does not guarantee determinism in ChatGPT code generation.
 102 Most recently, Yuan et al. (2025) traced such non-determinism to numerical precision issues in GPU
 103 kernels and proposed LayerCast as a mitigation strategy—a hardware-level fix that reduces but does
 104 not eliminate non-determinism, and that is not available to researchers using closed API services. Our
 105 Exact Match Rate (EMR) metric is closely related to Atil et al.’s Total Agreement Rate (TAR), which
 106 measures the fraction of runs producing the modal output; EMR instead measures the fraction of *all*
 107 *output pairs* that match exactly, providing a more sensitive measure when agreement is low and no clear
 108 modal output exists. Our work complements these studies in four specific ways. First, whereas prior
 109 studies (including Atil et al.’s five-model, eight-task study) measure variability post hoc, we provide
 110 a structured provenance protocol that enables *prospective* documentation and audit—answering not
 111 only “how much variability?” but also “why did these outputs differ?” through cryptographic hashing
 112 and W3C PROV graphs. Second, we directly compare local and API-based inference on identical tasks
 113 with identical prompts across *five* models and *two* independent API providers (OpenAI and Anthropic),
 114 isolating the deployment paradigm as a variable and suggesting that API non-determinism is a consistent
 115 pattern across providers rather than a provider-specific artifact. Third, we extend beyond single-turn
 116 evaluation to include multi-turn refinement and retrieval-augmented generation, demonstrating that
 117 reproducibility characteristics generalize across interaction regimes. Fourth, we quantify the overhead of
 118 systematic logging, demonstrating that the “cost of knowing” is negligible.

120 2.2 Experiment Tracking Tools

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

123 **MLflow** (Zaharia et al. 2018) provides experiment tracking, model packaging, and deployment. It logs
 124 parameters, metrics, and artifacts, but focuses on training pipelines and numerical outcomes rather than
 125 text-generation provenance.

126 **Weights & Biases** (Biewald 2020) offers experiment tracking with visualization dashboards. It
 127 supports prompt logging but lacks structured prompt versioning, cryptographic output hashing, and
 128 provenance graph generation.

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

131 **OpenAI Eval** (OpenAI 2023) is a framework for evaluating LLM outputs against benchmarks. It
 132 provides structured evaluation but is tightly coupled to OpenAI’s ecosystem and does not generate
 133 interoperable provenance records.

134 **LangSmith** (LangChain 2023) offers tracing and evaluation for LLM applications. It captures detailed
 135 execution traces but uses a proprietary format and requires cloud connectivity.

136 More broadly, Bommasani et al. (2022) identified reproducibility as a key risk for foundation models, and
 137 Liang et al. (2023) proposed the HELM benchmark for holistic evaluation of language models, including
 138 robustness and fairness dimensions that complement our reproducibility focus. In the provenance space,
 139 Padovani et al. (2025) recently introduced yProv4ML, a framework that captures ML provenance in
 140

142 Table 1. Comparison of our protocol with existing reproducibility tools and frameworks for GenAI experiments.
 143 Checkmarks (✓) indicate full support; tildes (~) indicate partial support; dashes (–) indicate no support.

144

145 Feature	Ours	MLflow	W&B	DVC	OpenAI Eval	LangSmith
146 Prompt versioning (Prompt Card)	✓	–	~	–	~	~
147 Run-level provenance (W3C PROV)	✓	–	–	–	–	–
148 Cryptographic output hashing	✓	–	–	✓	–	–
149 Seed & param logging	✓	✓	✓	–	✓	✓
150 Environment fingerprinting	✓	~	~	~	–	–
151 Model weights hashing	✓	–	~	✓	–	–
152 Overhead <1% of inference	✓	~	~	N/A	N/A	~
153 Designed for GenAI text output	✓	–	–	–	✓	✓
154 Open standard (PROV-JSON)	✓	–	–	–	–	–
155 Local-first (no cloud dependency)	✓	✓	–	✓	–	–

156

157

158

159

160 PROV-JSON format with minimal code modifications; our protocol shares the commitment to W3C
 161 PROV and SHA-256 hashing but differs in three key respects: (i) we target inference-time stochastic text
 162 generation rather than training pipelines; (ii) our Run Cards capture prompt-level metadata (prompt
 163 hash, seed status, interaction regime) not present in training-oriented schemas; and (iii) we provide
 164 empirical evidence quantifying why such logging is necessary for API-served models.

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

174

175 2.3 Provenance in Scientific Computing

176

177 Data provenance—the lineage of data through transformations—has a rich history in database systems and
 178 scientific workflows (Herschel et al. 2017). The W3C PROV family of specifications (Moreau and Missier
 179 2013) provides a standardized data model for representing provenance as directed acyclic graphs of *entities*,
 180 *activities*, and *agents*. Samuel and König-Ries (2022) applied provenance tracking to computational biology
 181 workflows, demonstrating its value for reproducibility. However, to our knowledge, no prior work has
 182 applied W3C PROV specifically to generative AI experiment workflows, in which the challenge involves
 183 not only tracking data lineage but also capturing the stochastic generation context that determines output
 184 variability.

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

188

189 **3 Protocol Design**

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

193 **3.1 Scope and Design Principles**

194 The protocol is designed around three principles:

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

204 **3.2 Prompt Cards**

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

- 208 • `prompt_id`: Unique identifier
- 209 • `prompt_hash`: SHA-256 hash of the prompt text, enabling tamper detection
- 210 • `version`: Semantic version number
- 211 • `task_category`: Classification of the task (e.g., summarization, extraction)
- 212 • `objective`: Natural-language description of what the prompt is designed to achieve
- 213 • `assumptions`: Explicit assumptions about inputs and expected behavior
- 214 • `limitations`: Known limitations or failure modes
- 215 • `target_models`: Models for which the prompt was designed and tested
- 216 • `expected_output_format`: Description of the expected output structure
- 217 • `interaction_regime`: Single-turn, multi-turn, or chain-of-thought
- 218 • `change_log`: History of modifications

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

225 **3.3 Run Cards**

226 A *Run Card* captures the complete execution context of a single generative AI run. Each Run Card
227 records 24 core fields organized into five groups (the complete JSON schema in Appendix B includes
228 these fields plus additional metadata such as `researcher_id`, `affiliation`, `system_logs`, and `errors`):

- 230 (1) **Identification:** `run_id`, `task_id`, `task_category`, `prompt_hash`, `prompt_text`
- 231 (2) **Model context:** `model_name`, `model_version`, `weights_hash`, `model_source`
- 232 (3) **Parameters:** `inference_params` (temperature, top_p, top_k, max_tokens, seed, decoding_strategy),
233 `params_hash`
- 234 (4) **Input/Output:** `input_text`, `input_hash`, `output_text`, `output_hash`, `output_metrics`

Run Card Schema (24 core + extension fields)	
236	1. Identification
237	<code>run_id · task_id · task_category · prompt_hash · prompt_text</code>
238	2. Model Context
239	<code>model_name · model_version · weights_hash · model_source</code>
240	3. Parameters
241	<code>inference_params {temp, top_p, top_k, max_tokens, seed, strategy} · params_hash</code>
242	4. Input/Output
243	<code>input_text · input_hash · output_text · output_hash · output_metrics</code>
244	5. Execution Metadata
245	<code>environment · environment_hash · code_commit · timestamps · duration_ms · overhead_ms · storage_kb</code>
246	API Extensions (optional)
247	<code>api_request_id · api_region · seed_status ∈ {sent, logged-only, not-supported}</code>
248	Workflow Extensions (optional)
249	<code>conversation_history_hash · turn_index · retrieval_context_hash · parent_run_id</code>
250	

251 Fig. 1. Run Card minimal schema. All SHA-256 hashes (5 total) enable tamper detection and automated comparison.
 252 API and workflow extension fields are optional.

- 253
 254
 255 (5) **Execution metadata:** `environment` (OS, architecture, Python version, hostname), `environment_hash`,
 256 `code_commit`, `timestamps` (start/end), `execution_duration_ms`, `logging_overhead_ms`, `storage_kb`

257 For API-served models, optional extension fields capture provider-specific metadata that may help diag-
 258 nose non-determinism: `api_request_id`, `api_response_headers`, `api_model_version_returned`, `api_region`,
 259 and a `seed_status` field that distinguishes between seeds that were “sent” to the API, “logged-only”
 260 (recorded for protocol parity but not sent, as with Claude), or “not-supported” by the provider. This
 261 formalization ensures that the advisory or absent nature of API seed parameters is captured as structured
 262 metadata rather than left as an undocumented assumption.

263 Figure 1 illustrates the Run Card schema as a minimal structured record.

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

266 3.3.1 *Normative Field Requirements*. To support adoption as a citable specification, we classify Run Card
 267 fields using normative language following RFC 2119 (Bradner 1997):

- 268 • **MUST** (required for audit completeness): `run_id`, `prompt_text`, `prompt_hash`, `model_name`,
 269 `model_version`, `inference_params` (including temperature, seed, decoding strategy), `output_text`,
 270 `output_hash`, `timestamp_start`.
- 271 • **SHOULD** (strongly recommended): `input_hash`, `params_hash`, `environment_hash`, `weights_hash`
 272 (local models), `code_commit`, `execution_duration_ms`, `logging_overhead_ms`, `seed_status` (API
 273 models).
- 274 • **MAY** (optional, context-dependent): `api_request_id`, `api_response_headers`, `api_region`, `conversation_history`,
 275 `turn_index`, `retrieval_context_hash`, `parent_run_id`, `researcher_id`, `affiliation`.

276 A conforming implementation MUST populate all MUST fields and SHOULD populate all SHOULD
 277 fields. The MUST set is minimal: removing any MUST field renders at least one audit question from
 278 Section 6.8 unanswerable.

283 **3.4 W3C PROV Integration**

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

- 287 • **Entities:** Prompt, InputText, ModelVersion, InferenceParameters, Output, ExecutionMetadata
 288 • **Activities:** RunGeneration (the inference execution)
 289 • **Agents:** Researcher, SystemExecutor (the execution environment)

291 PROV relations capture the causal structure:

- 293 • **used:** RunGeneration used Prompt, InputText, ModelVersion, InferenceParameters
 294 • **wasGeneratedBy:** Output wasGeneratedBy RunGeneration
 295 • **wasAssociatedWith:** RunGeneration wasAssociatedWith Researcher, SystemExecutor
 296 • **wasAttributedTo:** Output wasAttributedTo Researcher
 297 • **wasDerivedFrom:** Output wasDerivedFrom InputText

298 This standardized representation enables automated reasoning about experiment provenance, including
 299 detecting when two runs share identical configurations and identifying the specific factors that differ
 300 between non-identical outputs. The choice of W3C PROV over plain JSON logs is deliberate: PROV’s
 301 formal semantics allow automated tools to traverse the provenance graph and answer queries such as
 302 “what changed between these two runs?” without custom parsing logic. An abbreviated example document
 303 is given in Appendix C; to illustrate the structure concisely, the core provenance chain is:
 304

305 Prompt →_{used} RunGeneration →_{generated} Output
 306 InputText →_{used} RunGeneration →_{assoc.} Researcher
 307 ModelVersion →_{used} RunGeneration; Output →_{derived} InputText

309 **3.5 Reproducibility Checklist**

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

314 **3.6 Extensions for Advanced Workflows**

316 The protocol’s field schema accommodates complex workflows through optional extension fields. Our em-
 317 pirical evaluation exercises two of these extensions—multi-turn dialogues and RAG—while the remaining
 318 extensions are specified in the reference implementation’s schema:

- 320 • **Multi-turn dialogues:** A `conversation_history_hash` field and `turn_index` enable linking
 321 each turn to the full conversation state. *Evaluated in Task 3 (multi-turn refinement) using Ollama’s
 322 /api/chat endpoint.*
 323 • **RAG:** Fields for retrieval context (with hashes) trace which external information influenced the
 324 output. *Evaluated in Task 4 (RAG extraction) with prepended context passages.*
 325 • **Tool use and function calling:** Fields for available tools, tool calls (with arguments, results,
 326 and hashes) capture the full tool-use chain.
 327 • **Chain-of-thought / agent workflows:** A `parent_run_id` field supports hierarchical provenance
 328 graphs for multi-step reasoning chains.

330 **3.7 Formal Definition and Audit Completeness**

331 We define the protocol as a tuple $\mathcal{P} = (PC, RC, G, CL)$, where PC is a Prompt Card, RC is a Run
 332 Card, G is a W3C PROV graph, and CL is the reproducibility checklist. Each Run Card RC_i is itself a
 333 tuple of field groups: $RC_i = (Id, Mod, Par, IO, Env, H)$, where H denotes the set of five SHA-256 hashes
 334 (prompt, input, parameters, environment, output).

335 We define an *audit question* as a predicate Q over one or more Run Cards. The protocol satisfies
 336 the following *audit completeness* property: for a set of 10 audit questions $\{Q_1, \dots, Q_{10}\}$ (defined in
 337 Section 6.8), every Q_j is answerable if and only if all field groups are populated. Formally:

$$339 \quad \forall Q_j \in \{Q_1, \dots, Q_{10}\} : \text{answerable}(Q_j, RC_i) \Leftrightarrow \bigwedge_{g \in \text{required}(Q_j)} g \subseteq RC_i \quad (1)$$

340 where $\text{required}(Q_j)$ maps each question to its minimal set of required field groups. The ablation analysis in
 341 Section 6.8 demonstrates that every field group is in the required set of at least one question, establishing
 342 protocol *minimality*: removing any field group violates Equation 1 for at least one Q_j .

343 The *differential diagnosis* property follows from the hash fields: given two Run Cards RC_a, RC_b with
 344 $H_{\text{output}}^a \neq H_{\text{output}}^b$, the protocol enables automatic identification of the divergence source by comparing
 345 the remaining hashes. If $H_{\text{prompt}}^a = H_{\text{prompt}}^b$, $H_{\text{input}}^a = H_{\text{input}}^b$, $H_{\text{params}}^a = H_{\text{params}}^b$, and $H_{\text{env}}^a = H_{\text{env}}^b$, then
 346 the output difference is attributable to non-determinism in the generation process itself—precisely the
 347 phenomenon we measure empirically in Section 5.

348 Having defined the protocol’s components and formal properties, we now evaluate it empirically along
 349 two dimensions: the reproducibility characteristics it reveals across different models and conditions, and
 350 the overhead it imposes on the experimental workflow.

351 **4 Experimental Setup**

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

354 **4.1 Models and Infrastructure**

355 We evaluate five models representing two fundamentally different deployment paradigms: three locally
 356 deployed open-weight models and two cloud API-served proprietary models. All local models were served
 357 through Ollama v0.15.5 (Ollama 2024) on an Apple M4 system with 24 GB unified memory running
 358 macOS 14.6 with Python 3.14.3.

359 **4.1.1 Local Models. LLaMA 3 8B** (Grattafiori et al. 2024): An open-weight model in Q4.0 quantization.
 360 Local deployment provides complete control over the execution environment, eliminating confounding
 361 factors such as network latency, server-side batching, and silent model updates. The model’s SHA-256
 362 weights hash was recorded per run via the Ollama API.

363 **Mistral 7B** (Jiang et al. 2023): An open-weight model (Q4.0 quantization) with a sliding-window
 364 attention mechanism, providing a second data point for local inference reproducibility at a similar
 365 parameter scale.

366 **Gemma 2 9B** (Gemma Team et al. 2024): Google’s open-weight model (Q4.0 quantization), represent-
 367 ing a third local model from an independent model family. Gemma 2 proved to be the most deterministic
 368 model in our study, though its inference time is substantially higher than the other local models (~ 180 s
 369 per run vs. 8–14 s for LLaMA and Mistral), likely due to its larger context window and architectural
 370 differences at Q4.0 quantization on the M4 chip.

377 **4.1.2 API-Served Models.** **GPT-4** (Achiam et al. 2023): Accessed via the OpenAI API (`openai` Python
 378 SDK v1.59.9) with controlled seed parameters. The API returned `gpt-4-0613` as the resolved model
 379 version in all runs. The API introduces additional sources of variability: load balancing, server-side
 380 batching, potential model-version updates, and floating-point non-determinism across different hardware.

381 **Claude Sonnet 4.5** (Anthropic 2024): Accessed via the Anthropic API using a lightweight `urllib`-
 382 based runner (no SDK dependency). Claude’s API does not support a `seed` parameter; we set `temperature=0`
 383 for greedy decoding and logged a seed value for protocol parity (marked as `logged-only-not-sent-to-api`).
 384 This provides an independent replication of the API non-determinism phenomenon on a second cloud
 385 provider.

386

387 4.2 Tasks

388 We evaluate four tasks that span the output-structure spectrum and interaction complexity:

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

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

395 **Task 3: Multi-turn Refinement.** A three-turn dialogue in which the model first extracts structured
 396 information, then receives feedback requesting more detail, and finally produces a refined extraction. This
 397 tests reproducibility under conversational state accumulation, using Ollama’s `/api/chat` endpoint for
 398 local models.

399 **Task 4: RAG Extraction.** The same structured extraction task as Task 2, but with an additional
 400 retrieved context passage prepended to the input. This tests whether augmenting the prompt with
 401 external context affects reproducibility.

402

403 4.3 Input Data

404 We use 30 widely-cited scientific abstracts from landmark AI/ML papers, including Vaswani et al. (2017)
 405 (Transformer), Devlin et al. (2019) (BERT), Brown et al. (2020) (GPT-3), Raffel et al. (2020) (T5), Wei
 406 et al. (2022) (Chain-of-Thought), as well as seminal works on GANs, ResNets, VAEs, LSTMs, CLIP,
 407 DALL-E 2, Stable Diffusion, LLaMA, InstructGPT, PaLM, and others. These abstracts vary in length
 408 (74–227 words), technical complexity, and the number of quantitative results reported, thereby providing
 409 substantial diversity in the generation challenge.

410

411 4.4 Experimental Conditions

412 We define five conditions (Table 2) that systematically vary the factors hypothesized to affect repro-
 413 ducibility:

414 **Design principle for API models.** For cloud-hosted APIs whose `seed` parameter is advisory rather
 415 than deterministic (as documented by OpenAI for GPT-4 (OpenAI 2024)) or entirely absent (as with
 416 Claude), the fixed-vs.-variable seed distinction has no guaranteed effect server-side. We therefore treat C2
 417 as the primary test of determinism under greedy decoding for such models.

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

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

422

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: Tasks 1–2 are evaluated under all five conditions (C1, C2, C3). Tasks 3–4 (multi-turn, RAG) are evaluated under C1 only for the three local models and Claude Sonnet 4.5. Total: 3,604 logged runs across 5 models. For API-served models, C2 uses the same fixed seed as C1; the seed parameter is advisory and does not guarantee determinism.

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

Run counts. For Tasks 1–2 (extraction and summarization), each model is evaluated under C1 (5 runs), C2 (5 runs), and C3 (9 runs = 3 temperatures \times 3 reps) per abstract. LLaMA 3 uses 30 abstracts (1,140 runs); the newer models (Mistral 7B, Gemma 2 9B, Claude Sonnet 4.5) use 10 abstracts (380 runs each). For GPT-4, quota exhaustion limited collection to 724 runs (C2: 300/300; C3: 416/450; C1: 8/300 excluded). For Tasks 3–4 (multi-turn and RAG), the three local models and Claude Sonnet 4.5 are evaluated under C1 with 10 abstracts \times 5 repetitions = 50 runs each (400 runs total). **Grand total: 3,604 valid runs.**

Table 3 summarizes the per-model run distribution.

Table 3. Run distribution across models and tasks.

Model	Tasks 1–2	Tasks 3–4	Total
LLaMA 3 8B	1,140	100	1,240
Mistral 7B	380	100	480
Gemma 2 9B	380	100	480
GPT-4	724	—	724
Claude Sonnet 4.5	380	100	480
Chat-format control [†]	200	—	200
Total	3,204	400	3,604¹

[†]LLaMA 3 8B via /api/chat endpoint (Appendix H).

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)*.

¹One Claude run (0.03%) returned an empty output due to API timeout and is excluded from variability metrics.

- 471 • **Semantic reproducibility** (meaning-level): Two outputs convey the same information despite
 472 different phrasing. Measured by *ROUGE-L F1* and *BERTScore F1*.

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

477 **Exact Match Rate (EMR):** The fraction of output pairs that are character-for-character identical.
 478 $\text{EMR} = 1.0$ indicates perfect reproducibility; $\text{EMR} = 0.0$ indicates that no two outputs match exactly.
 479 With $n = 5$ repetitions per group ($\binom{5}{2} = 10$ pairs), per-abstract EMR values are discrete: $\{0.0, 0.1, \dots, 1.0\}$;
 480 with $n = 3$ (C3 conditions), EMR takes values in $\{0.0, 0.333, 0.667, 1.0\}$. This granularity should be
 481 considered when interpreting standard deviations and confidence intervals for small sample sizes.

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

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

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

496 For protocol overhead, we measure:

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

502 All EMR values in Tables 4 and 8 are accompanied by 95% bootstrap confidence intervals (10,000
 503 resamples over per-abstract EMR values, percentile method).

504 5 Results

506 5.1 Reproducibility Under Greedy Decoding

507 Table 4 presents the headline result: Exact Match Rates under greedy decoding for all five models. Table 5
 508 provides the full three-level reproducibility assessment.

510 **5.1.1 Local Models: Near-Perfect to Perfect Reproducibility.** **Finding 1: Gemma 2 9B achieves perfect**

511 bitwise reproducibility under greedy decoding. Across all tasks and conditions with $t=0$, Gemma 2
 512 9B produces $\text{EMR} = 1.000$ with $\text{NED} = 0.000$ —every single output is character-for-character identical
 513 across repetitions. This includes not only single-turn extraction and summarization but also multi-turn
 514 refinement and RAG extraction.

515 **Finding 2: All three local models achieve high reproducibility.** LLaMA 3 8B attains $\text{EMR} =$
 516 0.987 for extraction and 0.947 for summarization; Mistral 7B achieves 0.960 and 0.840, respectively. The

518 Table 4. Exact Match Rate (EMR) under greedy decoding ($t=0$) across five models and two single-turn tasks, with
 519 95% bootstrap confidence intervals ($n_{\text{boot}}=10,000$). For local models, values reflect condition C1 (fixed seed); for
 520 GPT-4, C2 (variable-seed greedy, as C1 has insufficient coverage); for Claude, C1 (Claude’s API does not support a
 521 seed parameter). Higher is more reproducible.

Model	Source	Extraction EMR	Summarization EMR
Gemma 2 9B	Local	1.000 [1.00, 1.00]	1.000 [1.00, 1.00]
	Local	0.987 [0.96, 1.00]	0.947 [0.89, 0.99]
	Local	0.960 [0.88, 1.00]	0.840 [0.72, 0.96]
GPT-4	API	0.443 [0.32, 0.57]	0.230 [0.16, 0.30]
Claude Sonnet 4.5	API	0.190 [0.05, 0.40]	0.020 [0.00, 0.05]

530 Table 5. Three-level reproducibility assessment under greedy decoding ($t=0$). L1: bitwise identity (EMR), L2: surface
 531 similarity (NED, ROUGE-L), L3: semantic equivalence (BERTScore F1). Values are means across abstracts.

Model	Task	L1: Bitwise		L2: Surface		L3: Semantic	
		EMR	σ	NED↓	ROUGE-L↑	BERTScore F1↑	
Gemma 2 9B	Extraction	1.000	0.000	0.000	1.000	1.0000	
	Summarization	1.000	0.000	0.000	1.000	1.0000	
Mistral 7B	Extraction	0.960	0.120	0.001	1.000	0.9999	
	Summarization	0.840	0.196	0.046	0.955	0.9935	
LLaMA 3 8B	Extraction	0.987	0.072	0.003	0.997	0.9997	
	Summarization	0.947	0.139	0.014	0.986	0.9979	
GPT-4	Extraction	0.443	0.335	0.072	0.938	0.9904	
	Summarization	0.230	0.193	0.137	0.870	0.9839	
Claude Sonnet 4.5	Extraction	0.190	0.291	0.101	0.904	0.9878	
	Summarization	0.020	0.040	0.242	0.764	0.9704	

548 small deviations from perfect reproducibility in LLaMA 3 and Mistral 7B appear to be associated with a
 549 warm-up effect on the first inference call after model loading, which affects 2–4 of the 10–30 abstracts per
 550 model; we hypothesize this reflects GPU cache initialization, though this was not formally tested. Seed
 551 variation (C1 vs. C2) has *no effect* under greedy decoding for any local model: the model always selects
 552 the highest-probability token, making the seed irrelevant.

555 **5.1.2 API-Served Models: Substantial Hidden Non-Determinism. Finding 3: Both API-served models**
 556 **exhibit substantial non-determinism under greedy decoding, observed independently across**
 557 **two providers.** Under $t=0$ with controlled seeds, GPT-4 achieves EMR = 0.443 for extraction and 0.230
 558 for summarization. Claude Sonnet 4.5 is even less deterministic: EMR = 0.190 for extraction and EMR
 559 = 0.020 for summarization—meaning that across 10 abstracts \times 5 repetitions, Claude produced the same
 560 summarization output only 2% of the time.

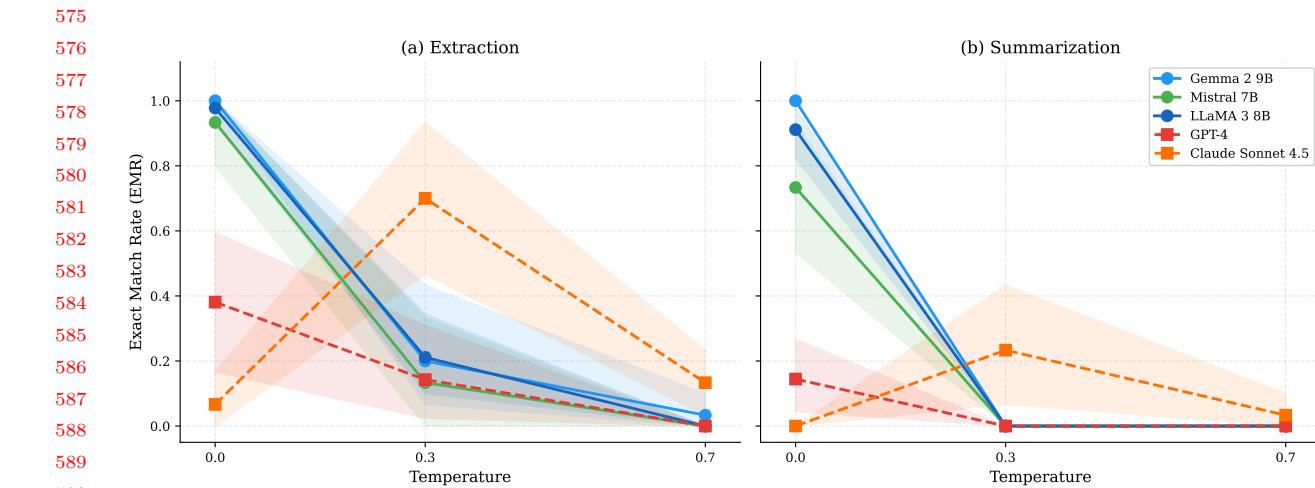
561 Table 6 summarizes the deployment-paradigm gap.

562 Under the representative greedy condition for each model (C1 for local models and Claude, C2 for GPT-
 563 4; see Table 4), the average single-turn EMR is **0.956 for local models vs. 0.221 for API models**—a
 564

565 Table 6. API-served vs. locally deployed models under greedy decoding (single-turn tasks only). Local averages: simple
 566 mean across 3 models \times 2 tasks (C1+C2 combined). API averages: simple mean across 2 models \times 2 tasks (GPT-4
 567 C2, Claude C1). Local models exhibit substantially higher bitwise reproducibility, consistent with deployment-side
 568 factors—rather than user-controllable parameters—as a major contributor to API output variability.
 569

Deployment	EMR↑	NED↓	ROUGE-L↑	BS-F1↑
Local (3 models)	0.956	0.011	0.990	0.9985
API (2 models)	0.221	0.138	0.869	0.9831

574 Effect of Sampling Temperature on Reproducibility



576 Fig. 2. Effect of temperature on Exact Match Rate across five models. (a) Extraction task. (b) Summarization task.
 577 Local models (solid lines) start from near-perfect or perfect reproducibility at $t=0$, while API models (dashed lines)
 578 start from a much lower baseline. All models converge toward $EMR = 0$ at $t=0.7$.
 579

580

581 more than 4-fold reproducibility gap. This gap is not due to user-side parameter differences: all models
 582 use $t=0$ with the same decoding strategy. The observed variability is consistent with deployment-side
 583 factors invisible to the researcher—such as hardware-level floating-point variability, request batching, and
 584 model routing. This pattern, observed independently across two API providers (OpenAI and Anthropic),
 585 is consistent with non-determinism arising from factors common to cloud-hosted LLM inference rather
 586 than being a provider-specific artifact. *Without systematic logging, this non-determinism would be entirely*
 587 *invisible.*
 588

589 **5.1.3 Temperature Effects Across Models. Finding 4:** Temperature is the dominant *user-controllable*
 590 factor affecting variability for local models; for API-served models, the relationship is more
 591 complex. Figure 2 shows the relationship between temperature and EMR for all five models. Table 7
 592 provides the full temperature sweep data.

593 Within the C3 temperature sweep, increasing temperature from 0.0 to 0.7 reduces EMR to zero for all
 594 models on summarization. For extraction, local models drop from $EMR > 0.93$ to near zero, while API
 595 models drop from their already-low baselines. Notably, BERTScore F1 remains above 0.94 in all conditions
 596 (minimum: 0.943 for LLaMA summarization at $t=0.7$) even when EMR drops to zero, indicating that
 597

598

612 Table 7. Effect of sampling temperature on Exact Match Rate (EMR) under condition C3. For local models, increasing
 613 temperature monotonically reduces EMR. For API models, the relationship is more complex: Claude Sonnet 4.5 exhibits
 614 *higher* EMR at $t=0.3$ than at $t=0.0$ (see text). At $t=0.7$, all models converge toward $\text{EMR} \approx 0$ for summarization.

Model	Task	$t=0.0$	$t=0.3$	$t=0.7$
Gemma 2 9B	Extraction	1.000	0.200	0.033
	Summarization	1.000	0.000	0.000
Mistral 7B	Extraction	0.933	0.133	0.000
	Summarization	0.733	0.000	0.000
LLaMA 3 8B	Extraction	0.978	0.211	0.000
	Summarization	0.911	0.000	0.000
GPT-4	Extraction	0.381	0.143	0.000
	Summarization	0.144	0.000	0.000
Claude Sonnet 4.5	Extraction	0.067	0.700	0.133
	Summarization	0.000	0.233	0.033

629
 630 non-determinism is primarily a *phrasing* phenomenon rather than a *meaning* phenomenon: even when
 631 outputs differ textually, they convey equivalent information. This distinction is practically important—
 632 researchers whose downstream analyses depend on semantic content rather than exact wording may find
 633 API outputs acceptable despite low EMR.

634 However, the temperature–reproducibility relationship is not uniformly monotonic across all models.
 635 Claude Sonnet 4.5 exhibits an anomalous pattern under the C3 sweep: extraction EMR *increases* from
 636 0.067 at $t=0.0$ to 0.700 at $t=0.3$ before declining to 0.133 at $t=0.7$; summarization shows a similar
 637 inversion (EMR = 0.000 at $t=0.0$, rising to 0.233 at $t=0.3$). This counterintuitive behavior—where a small
 638 positive temperature *improves* reproducibility relative to greedy decoding—may reflect how Anthropic’s
 639 infrastructure implements the $t=0$ decoding path: at exactly zero temperature, server-side stochastic
 640 processes (e.g., speculative decoding, hardware-level floating-point non-determinism across GPU types,
 641 or request batching effects) may dominate output variability, whereas a small positive temperature
 642 may activate a more stable sampling path that happens to converge on similar tokens. With $n=10$
 643 abstracts and 30 runs per temperature level (standard deviation $\sigma = 0.38$ for the 0.700 extraction EMR),
 644 this observation should be interpreted cautiously. Nevertheless, it underscores that the temperature–
 645 reproducibility relationship for API-served models depends on provider-specific implementation details
 646 that are opaque to researchers. Finding 4 therefore holds robustly for local models and for the overall $t=0$
 647 to $t=0.7$ trajectory, but the precise shape of the temperature–response curve for individual API providers
 648 merits further investigation with larger sample sizes.

650 5.2 Multi-Turn and RAG Reproducibility

651 **Finding 5:** The local-vs-API reproducibility gap extends to complex interaction regimes.
 652 Table 8 and Figure 3 present results for multi-turn refinement and RAG extraction across the three local
 653 models and Claude Sonnet 4.5.

654 Gemma 2 9B and Mistral 7B achieve perfect $\text{EMR} = 1.000$ for both multi-turn refinement and RAG
 655 extraction, demonstrating that conversational state accumulation and context augmentation do not
 656 degrade reproducibility when the underlying model is deterministic. LLaMA 3 8B shows $\text{EMR} = 0.880$

659 Table 8. Reproducibility under complex interaction regimes (C1 fixed seed, $t=0$), with 95% bootstrap confidence
 660 intervals on EMR. Multi-turn refinement involves three successive prompt-response exchanges. RAG extraction augments
 661 the prompt with a retrieved context passage. Claude Sonnet 4.5 is included as a representative API-served model;
 662 its near-zero EMR across all four scenarios confirms that the local-vs-API reproducibility gap extends to complex
 663 interaction regimes.

Model	Scenario	EMR [95% CI]	NED \downarrow	ROUGE-L \uparrow	BS-F1 \uparrow
Gemma 2 9B	Single-turn Extraction	1.000 [1.00, 1.00]	0.000	1.000	1.0000
	Single-turn Summarization	1.000 [1.00, 1.00]	0.000	1.000	1.0000
	Multi-turn Refinement	1.000 [1.00, 1.00]	0.000	1.000	1.0000
	RAG Extraction	1.000 [1.00, 1.00]	0.000	1.000	1.0000
Mistral 7B	Single-turn Extraction	0.960 [0.88, 1.00]	0.001	1.000	0.9999
	Single-turn Summarization	0.840 [0.72, 0.96]	0.046	0.955	0.9935
	Multi-turn Refinement	1.000 [1.00, 1.00]	0.000	1.000	1.0000
	RAG Extraction	1.000 [1.00, 1.00]	0.000	1.000	1.0000
LLaMA 3 8B	Single-turn Extraction	0.987 [0.96, 1.00]	0.003	0.997	0.9997
	Single-turn Summarization	0.947 [0.89, 0.99]	0.014	0.986	0.9979
	Multi-turn Refinement	0.880 [0.76, 1.00]	0.012	0.988	0.9986
	RAG Extraction	0.960 [0.88, 1.00]	0.012	0.985	0.9987
Claude Sonnet 4.5	Single-turn Extraction	0.190 [0.05, 0.40]	0.101	0.904	0.9878
	Single-turn Summarization	0.020 [0.00, 0.05]	0.242	0.764	0.9704
	Multi-turn Refinement	0.040 [0.00, 0.08]	0.189	0.834	0.9780
	RAG Extraction	0.000 [0.00, 0.00]	0.256	0.748	0.9714

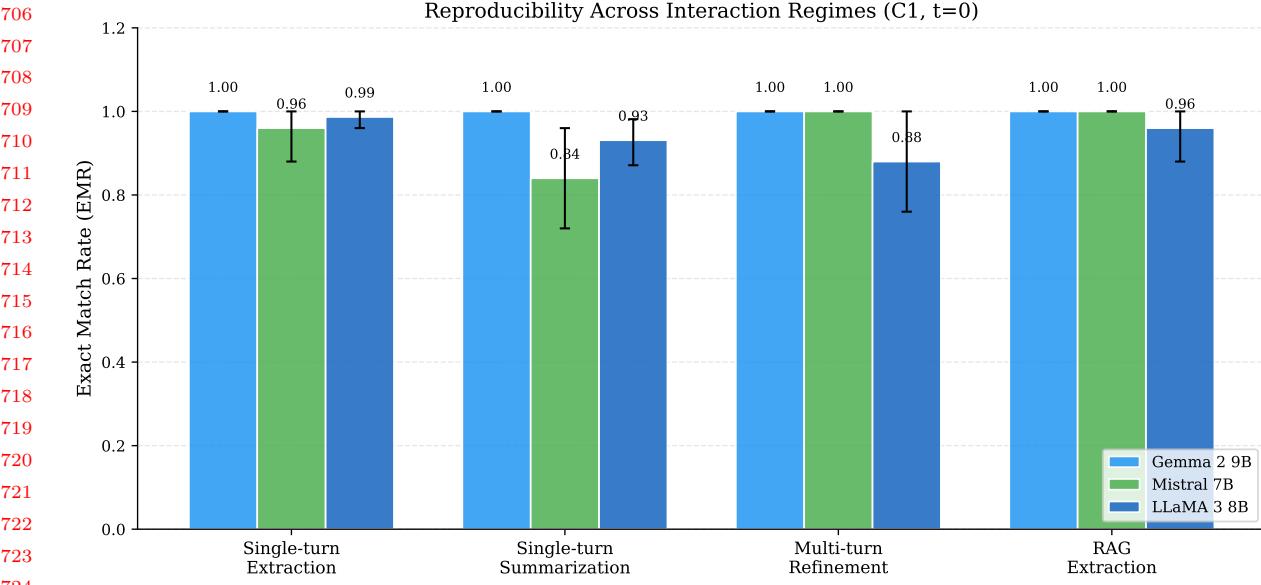
684 for multi-turn and 0.960 for RAG—slightly lower than its single-turn extraction performance (0.987),
 685 consistent with error accumulation across dialogue turns.

686 Claude Sonnet 4.5, the only API-served model evaluated on these tasks, achieves EMR = 0.040 for
 687 multi-turn refinement and EMR = 0.000 for RAG extraction—the lowest values observed in our study.
 688 The RAG result is particularly striking: across 50 runs (10 abstracts \times 5 repetitions), not a single pair of
 689 outputs was character-for-character identical (NED = 0.256). This confirms that API non-determinism is
 690 not limited to single-turn tasks but persists—and may even worsen—under complex interaction regimes
 691 where longer outputs and additional context amplify server-side variability.

693 5.3 Cross-Model Comparison

694 Figure 4 provides a comprehensive heatmap of EMR across all model-task combinations, and Figure 5
 695 shows the three-level reproducibility profile for each model.

696 The reproducibility gap between local and API-based inference is statistically significant. Because
 697 per-abstract EMR is a bounded, discrete metric (taking values in $\{0.0, 0.1, \dots, 1.0\}$ with $n=5$ repetitions
 698 per group), we report the non-parametric Wilcoxon signed-rank test as our primary analysis. Across
 699 the 30 paired LLaMA 3/GPT-4 abstracts under greedy decoding: for summarization, $W = 0$, $p < 0.001$;
 700 for extraction, $W = 3.5$, $p < 0.001$. Parametric paired t -tests yield consistent results: summarization
 701 $t(29) = 17.250$, $p < 0.0001$, Cohen's $d = 3.149$; extraction $t(29) = 8.996$, $p < 0.0001$, Cohen's $d = 1.642$.
 702 Both effect sizes are very large ($d > 1.6$), and all p -values survive Bonferroni correction for the four
 703 primary comparisons ($\alpha_{\text{adjusted}} = 0.0125$).



729
730 Table 9. Provenance logging overhead across five models under greedy decoding (C_1). The protocol adds negligible
731 overhead (<1%) to inference latency across all models and deployment modes.

732
733

Model	Source	Mean Inference (ms)	Mean Overhead (ms)	Overhead (%)
Gemma 2 9B	Local	181,579.3	30.6	0.234
Mistral 7B	Local	13,931.3	27.3	0.281
LLaMA 3 8B	Local	7,524.8	26.7	0.456
GPT-4	API	4,519.7	24.5	0.564
Claude Sonnet 4.5	API	4,359.3	26.5	0.727

734
735
736
737
738
739
740
741

742 Importantly, the effect is not driven by a few outlier abstracts: under greedy decoding, LLaMA 3
743 achieves $\text{EMR} \geq 0.8$ for 29 of 30 abstracts in extraction and 28 of 30 in summarization, while GPT-4
744 achieves $\text{EMR} \leq 0.6$ for 20 of 30 abstracts in extraction and 28 of 30 in summarization. The gap is
745 pervasive across the abstract set, not concentrated in a few difficult inputs. Power analysis (Cohen 1988)
746 confirms that with $n = 30$ paired abstracts and the observed effect sizes ($d > 1.6$), statistical power
747 exceeds 0.999 for all primary comparisons; with $n = 10$ abstracts (as used for the newer models), power
748 remains above 0.95 for effects of this magnitude.

749 5.4 Protocol Overhead

750 Table 9 presents the protocol's overhead metrics across all five models.
751

Bitwise Reproducibility Under Greedy Decoding

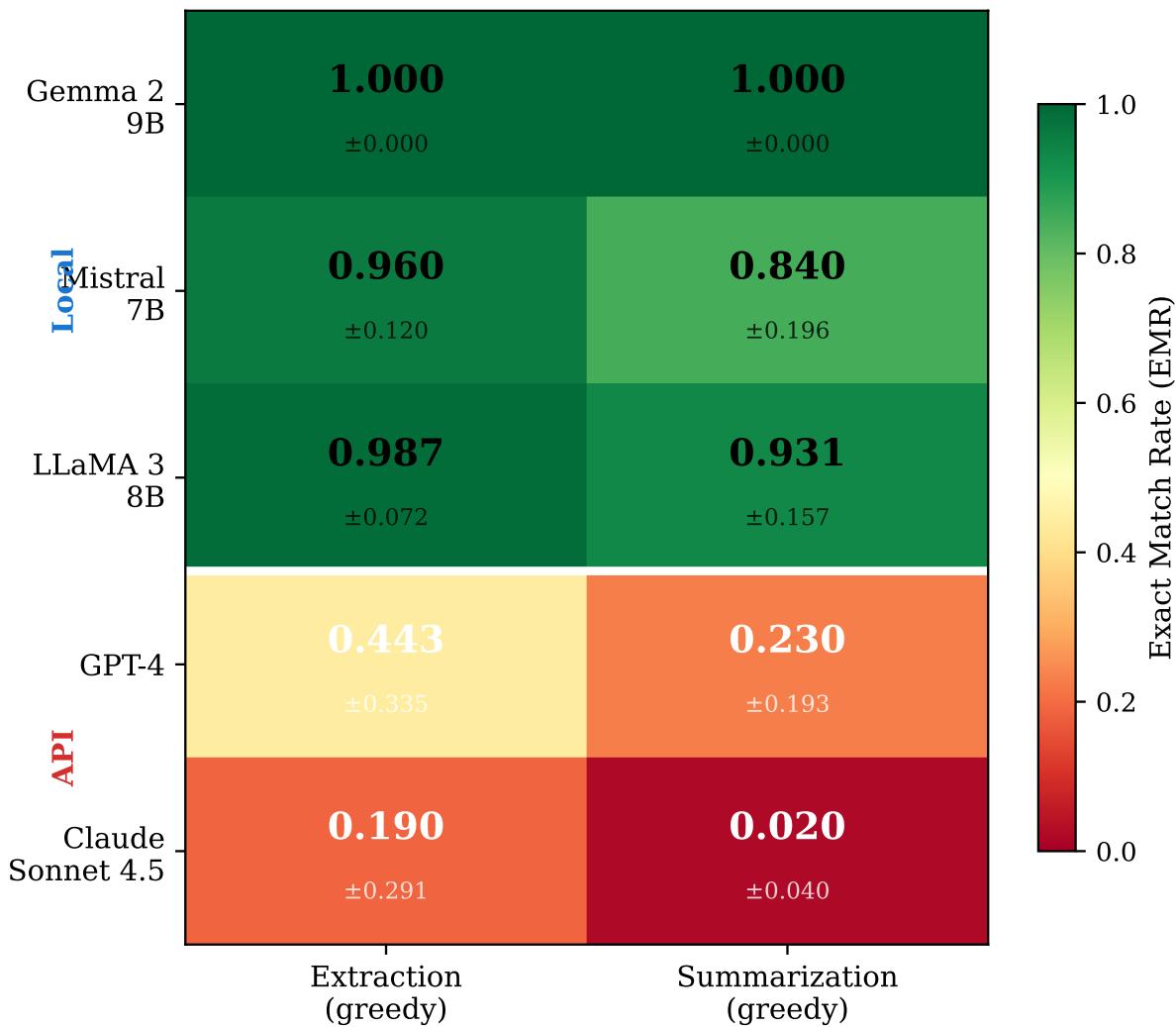


Fig. 4. Heatmap of Exact Match Rate under greedy decoding for five models. The horizontal white line separates local models (top three, green) from API-served models (bottom two, red). Gemma 2 9B achieves perfect 1.000 across all tasks.

The protocol adds less than 1% overhead for all five models, with mean logging time ranging from 21–30 ms depending on the model and task. Storage overhead remains modest at approximately 4 KB per run record. The overhead is consistent across local and API deployment modes, indicating that the protocol is deployment-agnostic.

Figure 6 provides an additional perspective on surface-level variability across models.

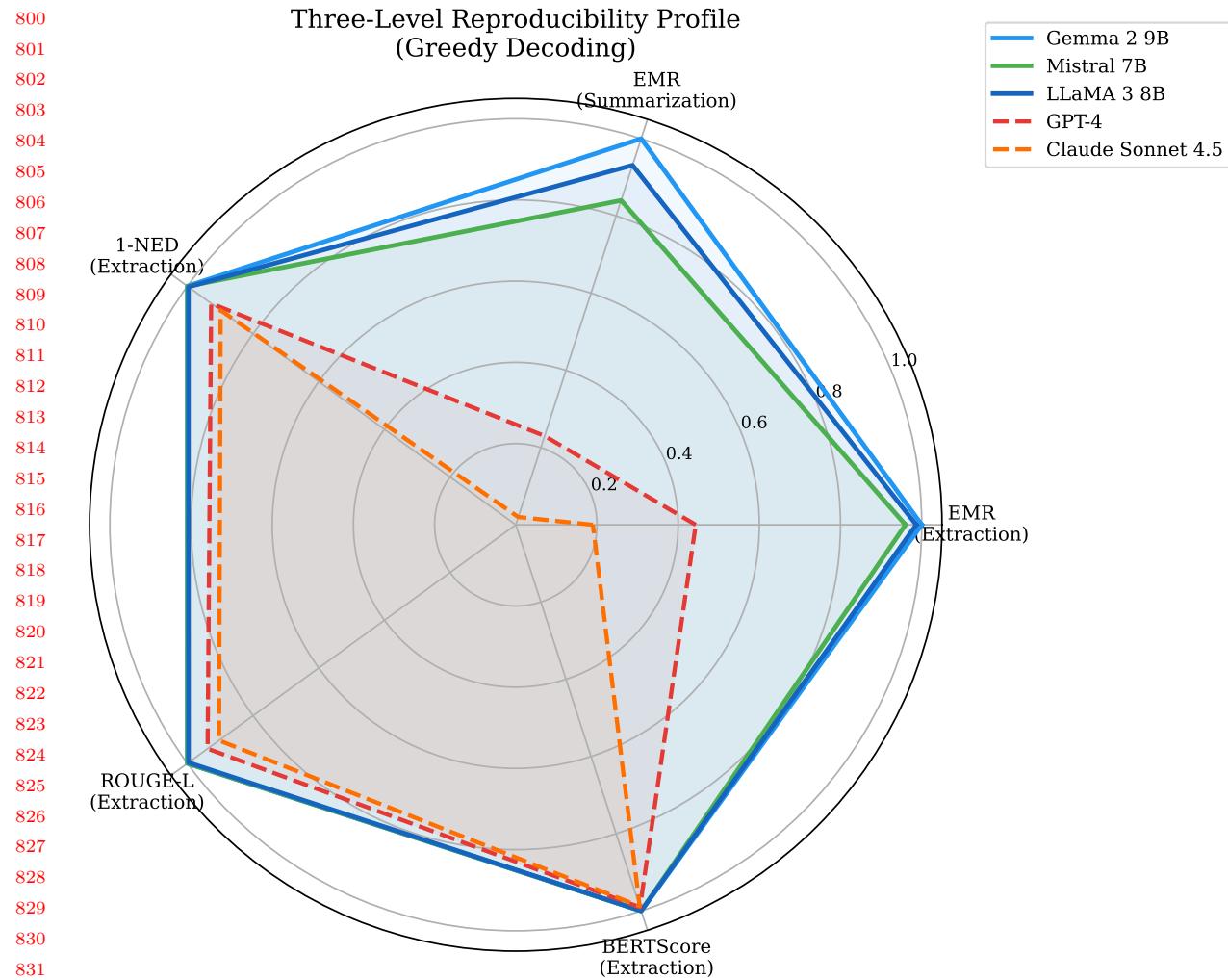


Fig. 5. Three-level reproducibility profiles under greedy decoding. Local models (solid lines) occupy the outer region across all five metrics, while API models (dashed lines) show pronounced deficits in EMR and NED while maintaining high BERTScore, indicating that API non-determinism is primarily lexical rather than semantic.

6 Discussion

The preceding results paint a clear and consistent picture: locally deployed models under greedy decoding achieve near-perfect to perfect bitwise reproducibility across all four tasks, while API-served models—from two independent providers—exhibit substantial hidden variability on single-turn tasks that researchers cannot control. Temperature is the dominant user-controllable factor for local models (though API models show a more complex temperature-reproducibility relationship; see Section 5), structured tasks are more reproducible than open-ended ones, and complex interaction regimes (multi-turn, RAG) do not degrade

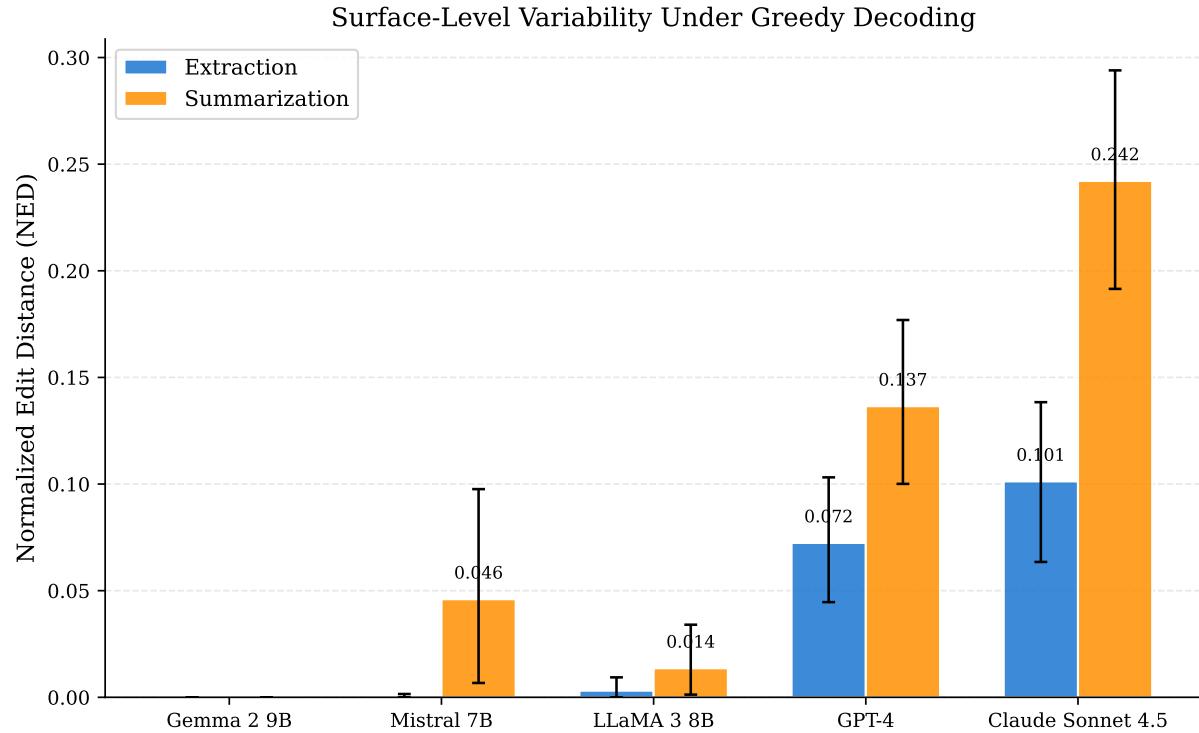


Fig. 6. Normalized Edit Distance (NED) under greedy decoding. Local models show near-zero NED (Gemma 2: 0.000, Mistral: 0.001), while API models exhibit NED 0.07–0.30, quantifying the surface-level divergence that accompanies the EMR gap.

local-model reproducibility. 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:

Use greedy decoding with local models for maximum reproducibility. Gemma 2 9B achieved *perfect* EMR = 1.000 across all tasks under greedy decoding. LLaMA 3 and Mistral 7B achieved EMR ≥ 0.840 . Local deployment with $t=0$ should be the default configuration for any study in which output consistency is critical.

API non-determinism is observed across providers. Our most consequential finding is that *both* GPT-4 (OpenAI) and Claude Sonnet 4.5 (Anthropic) exhibit substantial non-determinism under greedy decoding on single-turn tasks. Claude’s EMR of 0.020 for summarization means that effectively no two runs produce the same output. Researchers using *any* API-served model should never assume reproducibility without verification and should report multiple runs with variability metrics.

Prefer structured output formats when possible. The extraction task’s consistently higher reproducibility across all five models demonstrates that output-format constraints directly improve

894 reproducibility. This effect holds for both local models (EMR 0.960–1.000 for extraction vs. 0.840–1.000
 895 for summarization) and API models (EMR 0.190–0.443 for extraction vs. 0.020–0.230 for summarization).

896 **Include warm-up runs for local models.** The per-abstract analysis revealed that the first inference
 897 call after model loading may differ from subsequent calls due to cache initialization. This affects LLaMA 3
 898 and Mistral 7B on 2–4 of their abstracts, slightly reducing aggregate EMR.

899 **Log comprehensively; the cost is negligible.** At less than 1% overhead and approximately 4 KB
 900 per run across all five models, there is no practical reason not to apply comprehensive logging. The cost
 901 of not logging—namely, the inability to detect the kind of pervasive API non-determinism documented
 902 herein—far exceeds the protocol’s minimal requirements.

903

904 **6.2 Local vs. API Inference: A Persistent Reproducibility Gap**

905 The most significant finding of this study is the reproducibility gap between local and API-based inference,
 906 observed consistently across the two independent cloud providers evaluated. Under greedy decoding on
 907 single-turn tasks, local models average EMR = 0.956 while API models average EMR = 0.221—a more
 908 than 4-fold gap. The fact that Claude Sonnet 4.5 (Anthropic) exhibits *even lower* reproducibility than
 909 GPT-4 (OpenAI, snapshot gpt-4-0613) is inconsistent with provider-specific implementation as the
 910 sole explanation and suggests that non-determinism arises from factors common to distributed cloud
 911 inference infrastructure, such as hardware-level floating-point variability, request batching, and model
 912 routing. We emphasize that this gap is documented for the specific model versions and API snapshots
 913 evaluated; whether it generalizes to other providers, model families, or future API implementations is an
 914 open question that our protocol is designed to help answer systematically.

915 This gap has practical implications for the scientific use of the API-based LLMs evaluated. *Without*
 916 *systematic logging, a researcher using the models and configurations tested in our study would have*
 917 *no way of knowing that their “deterministic” experiment produces different outputs across runs.* Our
 918 protocol makes this hidden non-determinism visible, measurable, and documentable—and provides
 919 the infrastructure for researchers to assess whether the pattern holds for their specific models and
 920 configurations.

921

922 **6.3 Task-Dependent Reproducibility**

923 The difference between summarization and extraction reproducibility—observed consistently across all
 924 five models—is consistent with and extends our earlier two-model finding. The reproducibility hierarchy
 925 (extraction > summarization) holds for local models (EMR gap of 0.03–0.12) and is amplified for API
 926 models (EMR gap of 0.17–0.25). This finding suggests a spectrum ranging from highly constrained tasks
 927 (structured extraction) to open-ended tasks (summarization), with the degree of output-space constraint
 928 serving as a primary determinant.

929

930 **6.4 Multi-Turn and RAG: Reproducibility Under Complexity**

931 Our multi-turn and RAG results address a key limitation of prior work (including our own earlier
 932 two-model study): reproducibility under complex interaction regimes. The finding that Gemma 2 9B
 933 and Mistral 7B maintain perfect EMR = 1.000 for both multi-turn refinement and RAG extraction
 934 demonstrates that conversational state accumulation and context augmentation do not inherently degrade
 935 reproducibility for deterministic local models. LLaMA 3’s slight degradation (EMR = 0.880 for multi-turn)
 936 suggests model-specific sensitivity to dialogue-turn interactions, possibly related to the warm-up effect
 937 observed in single-turn experiments. Crucially, Claude Sonnet 4.5’s near-zero EMR for both multi-turn
 938 (0.040) and RAG (0.000) confirms that the local-vs-API reproducibility gap extends beyond single-turn
 939

940

941 tasks. The RAG result—zero exact matches across 50 runs—suggests that longer outputs and additional
 942 retrieval context may amplify server-side variability, though a single API model cannot establish this as a
 943 general principle.

944

945 6.5 The Role of Provenance

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

947

- 948 (1) **Automated comparison:** By comparing PROV graphs of two runs, one can automatically
 949 identify which factors differed (e.g., same prompt and model but different temperatures), enabling
 950 systematic diagnosis of non-reproducibility.
- 951 (2) **Lineage tracking:** When outputs are used as inputs to downstream processes (e.g., summarization
 952 outputs fed into a meta-analysis), the provenance chain can be extended to trace any final result
 953 back to its full generation context.
- 954 (3) **Compliance:** For regulated domains (healthcare, legal, finance), PROV documents provide the
 955 formal evidence trail required by audit standards ([National Institute of Standards and Technology
 956 2023](#)) and emerging regulations such as the EU AI Act ([European Parliament and Council of the
 957 European Union 2024](#)).

958

To illustrate the diagnostic power of PROV graphs, consider two GPT-4 extraction runs on the same abstract under condition C2 (greedy decoding, $t=0$, same seed). Although the PROV entities for Prompt, InputText, ModelVersion, and InferenceParameters are identical (verified via matching SHA-256 hashes), the Output entities differ: `output_hash` values diverge, and the `wasGeneratedBy` timestamps differ by several seconds. The PROV graph thus automatically pinpoints the source of non-reproducibility: the only varying factor is the RunGeneration activity itself, consistent with non-determinism arising from server-side factors.

959

To demonstrate that PROV-based reasoning goes beyond what plain JSON logs provide, we implemented three programmatic queries over our 3,604-run PROV dataset:

960

- 961 (1) **Divergence attribution:** “For all abstract-condition groups with non-identical outputs, identify
 962 which PROV entities diverge.” Result: across 348 GPT-4 and Claude groups with output divergence,
 963 100% share identical Prompt, InputText, ModelVersion, and InferenceParameters entities—the *only*
 964 varying component is the RunGeneration activity, providing systematic evidence for server-side
 965 non-determinism across the entire dataset rather than anecdotal examples.
- 966 (2) **Cross-provider comparison:** “Find all abstract-task pairs where both GPT-4 and Claude
 967 were given identical Prompt and InputText entities (verified by matching `genai:hash` attributes)
 968 but produced different Output entities.” Result: 20 such pairs exist (10 abstracts \times 2 tasks); in
 969 every case, *both* providers produced non-identical outputs across repetitions, confirming provider-
 970 independent non-determinism on shared inputs.
- 971 (3) **Provenance chain traversal:** “Starting from any Output entity, traverse `wasGeneratedBy` →
 972 `used` relations to reconstruct the full generation context, then verify integrity via hash comparison.”
 973 This query validates that every output in our dataset can be traced back to its complete generation
 974 context with no broken links—a guarantee that plain JSON logs cannot provide without custom
 975 graph-traversal code.

976

These queries exploit PROV’s formal graph structure (entity–activity–agent relations with typed edges) to answer questions that would require bespoke parsing logic on unstructured JSON logs. The queries and their results are included in the project repository.

977

988 **6.6 Pipeline Threat Model**

989 A natural objection is whether the observed output variability in API-served models could originate from
 990 our client-side pipeline rather than from server-side non-determinism. We address this systematically.

991 **No retries or parallelism.** Our API runners issue exactly one HTTP request per run, with no
 992 retry logic, exponential backoff, or concurrent requests. Each run is executed sequentially with a fixed
 993 delay between calls. Any request that fails (e.g., the single Claude timeout) is logged with the error and
 994 excluded from variability metrics rather than retried.

995 **Deterministic client-side processing.** All pre-processing (prompt construction, input hashing) and
 996 post-processing (output hashing, metadata collection) are deterministic operations verified by SHA-256
 997 hashes. The Run Card records the exact prompt text sent (`prompt_hash`), the exact input (`input_hash`),
 998 and the exact parameters (`params_hash`). For any pair of runs within a group, these three hashes are
 999 identical by construction.

1000 **No text normalization.** Outputs are stored and compared as received from the API, with no
 1001 whitespace normalization, encoding conversion, or post-processing. The `output_hash` is computed on the
 1002 raw response string.

1003 **PROV-based differential diagnosis.** Our PROV graphs provide formal evidence: across all 348
 1004 experimental groups with non-identical outputs for GPT-4 and Claude, 100% share identical Prompt,
 1005 InputText, ModelVersion, and InferenceParameters entities (verified via SHA-256 hash comparison). The
 1006 *only* varying component is the RunGeneration activity itself. This rules out client-side divergence as an
 1007 explanation and is consistent with server-side factors (hardware-level floating-point variability, request
 1008 routing, speculative decoding) as the source of non-determinism.

1009 **API metadata logging.** For API-served models, Run Cards capture `api_request_id`, `api_response_headers`,
 1010 and `api_model_version_returned`. In all GPT-4 runs, the returned model version was consistently
 1011 `gpt-4-0613`, ruling out silent model updates during the experiment window.

1013 **6.7 Limitations**

1014 We organize threats to validity following standard categories:

1015 **6.7.1 Internal Validity. Sample size.** LLaMA 3 uses 30 abstracts per condition, while the newer models
 1016 (Mistral, Gemma 2, Claude) use 10 abstracts. With $n = 30$, statistical power exceeds 0.999 for all primary
 1017 comparisons (Cohen 1988). With $n = 10$, the study is adequately powered for the large observed effect
 1018 sizes ($d > 1.6$) but may miss subtler effects. To verify that the unbalanced design does not inflate
 1019 the local-vs-API gap, we conducted a balanced subsample analysis restricting all models to the same
 1020 10 abstracts. Under this balanced comparison, local models average EMR = 0.953 while API models
 1021 average EMR = 0.190 (5.0× gap), confirming that the observed reproducibility gap is robust to sample-size
 1022 equalization and, if anything, slightly larger under balanced conditions.

1023 **GPT-4 C3 incomplete coverage.** Due to API quota exhaustion, GPT-4 extraction under C3
 1024 conditions covers 14–17 of 30 abstracts (summarization C3 is complete at 30). Our central claims rest on
 1025 the C2 condition (300/300 runs complete), and the C3 temperature sweep serves as a secondary analysis.

1026 **Warm-up confound.** The first inference after model loading may differ from subsequent calls for
 1027 LLaMA 3 and Mistral 7B. This affects 2–4 abstracts per model, slightly reducing aggregate EMR.
 1028 Gemma 2 9B appears immune to this effect.

1029 **Prompt format confound.** Single-turn experiments use Ollama’s `/api/generate` endpoint for local
 1030 models, whereas API models use their respective chat APIs. A supplementary control experiment (200
 1031 additional runs using Ollama’s `/api/chat` endpoint; see Appendix H) shows that this format difference

1032

1035 does not explain the reproducibility gap: LLaMA 3 produces *identical* variability metrics (summarization
 1036 EMR = 0.929, extraction EMR = 1.000) under both completion and chat formats.

1037 **6.7.2 External Validity. Five models, two paradigms.** Our evaluation covers three local models and
 1038 two API-served models. However, other models—including Gemini (Gemini Team et al. 2024), larger
 1039 LLaMA variants, and open-weight models served via cloud APIs—may exhibit different characteristics.
 1040 Notably, our GPT-4 experiments used the gpt-4-0613 snapshot (June 2023); more recent models (GPT-4
 1041 Turbo, GPT-4o) may exhibit different reproducibility characteristics. Prior work by Ouyang et al. (2024)
 1042 suggests that non-determinism persists across ChatGPT model versions, but confirmation with current
 1043 models is warranted.

1044 **Four tasks.** Our task suite now includes single-turn extraction/summarization, multi-turn refinement,
 1045 and RAG extraction. However, it does not cover code generation, mathematical reasoning, or creative
 1046 writing, which may exhibit different reproducibility patterns.

1047 **English-only, single domain.** Our input data consists of 30 English scientific abstracts from AI/ML
 1048 papers. Reproducibility characteristics may differ for other languages, domains, or document types.

1049 **Multi-turn limited to one API model.** Multi-turn and RAG experiments include Claude Sonnet
 1050 4.5 as the sole API representative; GPT-4 was not evaluated on Tasks 3–4 due to quota exhaustion. While
 1051 Claude’s near-zero EMR is consistent with the single-turn API pattern, other API providers may exhibit
 1052 different multi-turn reproducibility characteristics.

1053 **6.7.3 Construct Validity. Surface-level metrics.** Our metrics (EMR, NED, ROUGE-L) capture textual
 1054 rather than semantic similarity. Two outputs that are semantically equivalent but syntactically different
 1055 will register as non-matching under EMR and partially divergent under NED. This is by design—our focus
 1056 is on *exact* reproducibility—but it means our results may overstate the practical impact of non-determinism
 1057 for downstream applications where semantic equivalence suffices.

1058 **6.7.4 Other Considerations. Privacy.** The protocol’s environment metadata includes the machine host-
 1059 name, which may reveal institutional information. Deployments in privacy-sensitive settings should
 1060 anonymize this field.

1061 **Computational cost.** The total cost was modest: approximately 8 GPU-hours on a consumer laptop
 1062 (Apple M4, 24 GB) for 2,000 local-model runs (including multi-turn and RAG experiments), plus 1,204
 1063 API calls to GPT-4 and Claude. The carbon footprint is negligible at this scale, and the logging overhead
 1064 (<30 ms per run) would not materially increase energy consumption even at thousands of runs.

1065 **6.8 Protocol Minimality: An Ablation Analysis**

1066 To substantiate our claim that the protocol captures a *minimal* set of metadata, we conducted an ablation
 1067 analysis in which we systematically removed each field group from the protocol schema and assessed
 1068 which audit questions became unanswerable. We defined 10 audit questions that a reproducibility-oriented
 1069 researcher might ask (e.g., “Can we verify the exact prompt used?”, “Can we detect output tampering?”,
 1070 “Can we trace full provenance?”) and mapped each to the protocol fields required to answer it. For this
 1071 analysis, we decomposed the Run Card’s five sections into eight finer-grained field groups by separating
 1072 cross-cutting concerns: Identification, Model Context, Parameters, Input Content, Output Content,
 1073 Hashing (all SHA-256 digests), Environment, and Overhead (timing and storage metadata).

1074 The results show that removing *any* of these eight field groups renders at least one audit question
 1075 unanswerable, demonstrating that no group is redundant. The Hashing group (SHA-256 hashes for
 1076 prompts, inputs, outputs, parameters, and environment) has the highest information density: its removal
 1077 affects 6 of 10 questions despite contributing only 410 bytes per run. Conversely, the Overhead group
 1078

1082 (logging time metadata) is the least connected but remains necessary for overhead assessment. The
 1083 complete ablation results are available in the project repository.

1084 This analysis demonstrates that the protocol is *minimal* in the sense that every field group is necessary
 1085 for at least one audit capability, while the total overhead remains at approximately 4,052 bytes per run.
 1086

1087 6.9 Practical Costs and Adoption

1088 One concern with any new protocol is whether the adoption burden is justified. We address this concretely:
 1089

- 1090 • **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.
- 1091 • **Runtime cost:** <30 ms per run across all five models, negligible compared to inference times of
 seconds to minutes for typical LLM calls.
- 1092 • **Storage cost:** ~4 KB per run. Our 3,604 runs total approximately 14 MB—less than a single
 model checkpoint.
- 1093 • **Learning curve:** The protocol uses standard JSON and W3C PROV, requiring no specialized
 knowledge beyond basic Python.

1098 Against these modest costs, the protocol provides complete audit trails, automated provenance graphs,
 1099 tamper-detectable outputs via cryptographic hashing, and structured metadata that enable systematic
 1100 reproducibility analysis.

1101 6.10 Minimum Reporting Checklist for Generative AI Studies

1103 Based on our findings and the protocol design, we recommend that researchers conducting generative
 1104 AI experiments report, at minimum, the following five items (the full 15-item checklist is provided in
 1105 Appendix A):

- 1106 (1) **Model identity and version:** Exact model name, version string, and—for local models—weights
 hash.
- 1107 (2) **Inference parameters:** Temperature, seed, top_p, top_k, max_tokens, and decoding strategy.
 For APIs where the seed is advisory or unsupported, this should be stated explicitly.
- 1108 (3) **Reproducibility metrics over multiple runs:** Report at least EMR (or an equivalent exact-
 match metric) and one semantic metric (e.g., BERTScore) over ≥ 3 repetitions per condition. A
 single run is insufficient to characterize output stability.
- 1109 (4) **Environment and deployment mode:** Whether inference was local or API-based, and the
 execution environment (hardware, OS, library versions).
- 1110 (5) **Output hashes:** SHA-256 or equivalent cryptographic hashes of outputs, enabling tamper
 detection and automated comparison across studies.

1113 Run Cards generated by our protocol automatically capture all five items, providing a machine-readable
 1114 record that satisfies this checklist with no additional effort from the researcher.

1115 7 Conclusion

1118 We presented a lightweight protocol for logging, versioning, and provenance tracking of generative AI
 1119 experiments, introducing Prompt Cards and Run Cards as novel documentation artifacts and adopting the
 1120 W3C PROV data model for machine-readable provenance graphs. Through 3,604 controlled experiments
 1121 with five models—three locally deployed (LLaMA 3 8B, Mistral 7B, Gemma 2 9B) and two API-served
 1122 (GPT-4, Claude Sonnet 4.5)—across four NLP tasks and 30 scientific abstracts, we demonstrated five key
 1123 findings:
 1124

- 1129 (1) **API non-determinism is consistent across the two providers evaluated.** Both GPT-4
 1130 (OpenAI, snapshot gpt-4-0613) and Claude Sonnet 4.5 (Anthropic) exhibit substantial non-
 1131 determinism under greedy decoding on single-turn tasks (average EMR = 0.221), while all three
 1132 local models achieve average EMR = 0.956. This more than 4-fold reproducibility gap, observed
 1133 independently for two cloud providers, is consistent with non-determinism arising from factors
 1134 common to cloud-hosted inference infrastructure rather than being a provider-specific artifact.
 1135 Whether this pattern generalizes to other API providers and more recent model snapshots remains
 1136 an empirical question that the protocol is designed to help answer.
- 1137 (2) **Local models can achieve perfect bitwise reproducibility.** Gemma 2 9B attains EMR
 1138 = 1.000 across all four tasks under greedy decoding—every output is character-for-character
 1139 identical across repetitions.
- 1140 (3) **The local-vs-API gap extends to complex interaction regimes.** Multi-turn refinement and
 1141 RAG extraction achieve EMR ≥ 0.880 for all local models (Gemma 2 9B and Mistral 7B: perfect
 1142 EMR = 1.000), while Claude Sonnet 4.5—the only API model tested on these tasks—achieves
 1143 EMR = 0.040 (multi-turn) and EMR = 0.000 (RAG).
- 1144 (4) **Temperature is the dominant user-controllable factor for local models.** Increasing from
 1145 $t=0.0$ to $t=0.7$ reduces EMR to zero for all five models on summarization, while seed variation
 1146 has no effect under greedy decoding for local models. For API-served models, the temperature–
 1147 reproducibility relationship is more complex and may be non-monotonic (see Section 5).
- 1148 (5) **Comprehensive provenance logging adds negligible overhead:** less than 1% of inference
 1149 time and approximately 4 KB per run across all five models, removing any practical argument
 1150 against systematic documentation.

1151
 1152 These findings carry a broader implication: if the pattern observed across the two API providers
 1153 and model snapshots in our study generalizes, a substantial portion of published research that relies
 1154 on API-based LLMs may contain non-reproducible results without the authors’ knowledge. Regardless
 1155 of whether API non-determinism proves universal or provider-specific, the protocol itself provides the
 1156 infrastructure to detect, measure, and document such variability—making hidden non-determinism visible
 1157 wherever it occurs. The cost of systematic provenance logging—less than one percent of inference time—is
 1158 trivially small compared to the cost of publishing non-reproducible science.

1159 Looking ahead, we plan to (i) extend the model suite to include Gemini (Gemini Team et al. 2024) and
 1160 open-weight models served via cloud APIs (e.g., Hugging Face Inference Endpoints) to further disentangle
 1161 model architecture from deployment infrastructure; (ii) extend the task coverage to code generation,
 1162 mathematical reasoning, and agentic workflows; and (iii) develop automated reproducibility scoring based
 1163 on provenance graph analysis. Ultimately, we envision a future in which every generative AI output
 1164 carries a provenance certificate, and reproducibility metrics are reported alongside accuracy as a standard
 1165 component of empirical evaluation.

1166 The reference implementation, all 3,604 run records, provenance documents, and analysis scripts are
 1167 publicly available to support adoption and independent verification.

1170 Acknowledgments

1171 This work was supported by UTFPR – Universidade Tecnológica Federal do Paraná. The experiments were
 1172 conducted using locally deployed open-weight models to ensure full reproducibility of the computational
 1173 environment.

1176 **Data Availability Statement**

1177 The reference implementation, all 3,604 run records (JSON), PROV-JSON provenance documents, Run
 1178 Cards, Prompt Cards, input data, analysis scripts, and generated figures are publicly available at:

1179 <https://github.com/Roverlucas/genai-reproducibility-protocol>

1180 The repository includes instructions for reproducing all experiments and regenerating all tables and
 1181 figures from the raw data.

1183 **Author Contributions**

1184 Following the CRediT (Contributor Roles Taxonomy) framework: **Lucas Rover**: Conceptualization,
 1185 Methodology, Software, Validation, Formal Analysis, Investigation, Data Curation, Writing – Original
 1186 Draft, Writing – Review & Editing, Visualization, Project Administration. **Yara de Souza Tadano**:
 1187 Supervision, Conceptualization, Methodology, Writing – Review & Editing, Project Administration.

1189 **Conflict of Interest**

1190 The authors declare no conflicts of interest. This research was conducted independently at UTFPR with
 1191 no external funding from commercial AI providers. The use of OpenAI's GPT-4 API was for research
 1192 evaluation purposes only and does not constitute an endorsement.

1194 **Use of AI-Assisted Tools**

1195 The authors used AI-assisted tools (Claude, Anthropic) during the preparation of this manuscript for
 1196 language editing, code development support, and data analysis scripting. All AI-generated content was
 1197 critically reviewed, validated, and revised by the authors, who take full responsibility for the accuracy
 1198 and integrity of the final manuscript. The scientific design, experimental execution, interpretation of
 1199 results, and intellectual contributions are entirely the authors' own work.

1201 **References**

- 1202 J. Achiam et al.. 2023. *GPT-4 Technical Report*. arXiv preprint. (2023). arXiv: [2303.08774 \(cs.CL\)](https://arxiv.org/abs/2303.08774).
- 1203 Anthropic. 2024. *The Claude Model Family*. (2024). <https://www.anthropic.com/clause>.
- 1204 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).
- 1205 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).
- 1206 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 Linguistics, 381–393. doi:[10.18653/v1/2021.eacl-main.29](https://doi.org/10.18653/v1/2021.eacl-main.29).
- 1207 L. Biewald. 2020. *Experiment Tracking with Weights and Biases*. (2020). <https://wandb.com/>.
- 1208 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).
- 1209 S. Bradner. 1997. *Key Words for Use in RFCs to Indicate Requirement Levels*. RFC 2119. Internet Engineering Task Force. doi:[10.17487/RFC2119](https://doi.org/10.17487/RFC2119).
- 1210 T. Brown et al.. 2020. “Language Models are Few-Shot Learners.” In: *Advances in Neural Information Processing Systems*. Vol. 33, 1877–1901. arXiv: [2005.14165 \(cs.CL\)](https://arxiv.org/abs/2005.14165).
- 1211 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).
- 1212 J. Cohen. 1988. *Statistical Power Analysis for the Behavioral Sciences*. (2nd ed.). Lawrence Erlbaum Associates. ISBN: 978-0-8058-0283-2.
- 1213 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 Computational Linguistics, 4171–4186. doi:[10.18653/v1/N19-1423](https://doi.org/10.18653/v1/N19-1423).

1221

1222

- 1223 J. Dodge, S. Gururangan, D. Card, R. Schwartz, and N. A. Smith. 2019. “Show Your Work: Improved Reporting of
 1224 Experimental Results.” In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.
 1225 Association for Computational Linguistics, 2185–2194. doi:[10.18653/v1/D19-1224](https://doi.org/10.18653/v1/D19-1224).
- 1226 European Parliament and Council of the European Union. 2024. *Regulation (EU) 2024/1689 Laying Down Harmonised
 1227 Rules on Artificial Intelligence (AI Act)*. (2024). <https://eur-lex.europa.eu/eli/reg/2024/1689/oj>.
- 1228 T. Gebru, J. Morgenstern, B. Vecchione, J. W. Vaughan, H. Wallach, H. Daumé III, and K. Crawford. 2021. “Datasheets
 1229 for Datasets.” *Communications of the ACM*, 64, 12, 86–92. doi:[10.1145/3458723](https://doi.org/10.1145/3458723).
- 1230 Gemini Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, et al.. 2024.
 1231 *Gemini: A Family of Highly Capable Multimodal Models*. arXiv preprint. (2024). arXiv: [2312.11805 \(cs.CL\)](https://arxiv.org/abs/2312.11805).
- 1232 Gemma Team et al.. 2024. *Gemma 2: Improving Open Language Models at a Practical Size*. arXiv preprint. (2024). arXiv:
 1233 [2408.00118 \(cs.CL\)](https://arxiv.org/abs/2408.00118).
- 1234 A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, et al.. 2024.
 1235 *The LLaMA 3 Herd of Models*. arXiv preprint. (2024). arXiv: [2407.21783 \(cs.AI\)](https://arxiv.org/abs/2407.21783).
- 1236 O. E. Gundersen, Y. Gil, and D. W. Aha. 2018. “On Reproducible AI: Towards Reproducible Research, Open Science, and
 1237 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).
- 1238 O. E. Gundersen, M. Helmert, and H. H. Hoos. 2024. “Improving Reproducibility in AI Research: Four Mechanisms Adopted
 1239 by JAIR.” *Journal of Artificial Intelligence Research*, 81, 1019–1041. doi:[10.1613/jair.1.16905](https://doi.org/10.1613/jair.1.16905).
- 1240 O. E. Gundersen and S. Kjensmo. 2018. “State of the Art: Reproducibility in Artificial Intelligence.” *Proceedings of the
 1241 AAAI Conference on Artificial Intelligence*, 32, 1, 1644–1651. doi:[10.1609/aaai.v32i1.11503](https://doi.org/10.1609/aaai.v32i1.11503).
- 1242 M. Herschel, R. Diestelkämper, and H. Ben Lahmar. 2017. “A Survey on Provenance: What for? What form? What from?”
 1243 *The VLDB Journal*, 26, 6, 881–906. doi:[10.1007/s00778-017-0486-1](https://doi.org/10.1007/s00778-017-0486-1).
- 1244 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).
- 1245 A. Q. Jiang et al.. 2023. *Mistral 7B*. arXiv preprint. (2023). arXiv: [2310.06825 \(cs.CL\)](https://arxiv.org/abs/2310.06825).
- 1246 S. Kapoor and A. Narayanan. 2023. “Leakage and the Reproducibility Crisis in Machine-Learning-Based Science.” *Patterns*,
 1247 4, 9, 100804. doi:[10.1016/j.patter.2023.100804](https://doi.org/10.1016/j.patter.2023.100804).
- 1248 R. Kuprieiev, D. Petrov, and Iterative. 2024. *DVC: Data Version Control*. (2024). <https://dvc.org/>.
- 1249 LangChain. 2023. *LangSmith: A Platform for Building Production-Grade LLM Applications*. (2023). <https://smith.langchain.com/>.
- 1250 V. I. Levenshtein. 1966. “Binary Codes Capable of Correcting Deletions, Insertions, and Reversals.” *Soviet Physics Doklady*,
 1251 10, 8, 707–710.
- 1252 P. Liang et al.. 2023. “Holistic Evaluation of Language Models.” *Transactions on Machine Learning Research*. <https://openreview.net/forum?id=iO4LZibEqW>.
- 1253 C.-Y. Lin. 2004. “ROUGE: A Package for Automatic Evaluation of Summaries.” In: *Proceedings of the ACL-04 Workshop
 1254 on Text Summarization Branches Out*. Association for Computational Linguistics, 74–81. <https://aclanthology.org/W04-1013>.
- 1255 M. Mitchell, S. Wu, A. Zaldivar, P. Barnes, L. Vasserman, B. Hutchinson, E. Spitzer, I. D. Raji, and T. Gebru. 2019.
 1256 “Model Cards for Model Reporting.” In: *Proceedings of the Conference on Fairness, Accountability, and Transparency*.
 1257 ACM, 220–229. doi:[10.1145/3287560.3287596](https://doi.org/10.1145/3287560.3287596).
- 1258 L. Moreau and P. Missier. 2013. *PROV-DM: The PROV Data Model*. W3C Recommendation. World Wide Web Consortium.
 1259 <https://www.w3.org/TR/prov-dm/>.
- 1260 National Institute of Standards and Technology. 2023. *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*.
 1261 Tech. rep. U.S. Department of Commerce. doi:[10.6028/NIST.AI.100-1](https://doi.org/10.6028/NIST.AI.100-1).
- 1262 Ollama. 2024. *Ollama: Run Large Language Models Locally*. (2024). <https://ollama.com/>.
- 1263 OpenAI. 2024. *API Reference: Create Chat Completion — Seed Parameter*. “If specified, our system will make a best
 1264 effort to sample deterministically [...] Determinism is not guaranteed.” (2024). [https://platform.openai.com/docs/api-reference/chat/create](https://platform.openai.com/docs/api-

 1265 reference/chat/create).
- 1266 OpenAI. 2023. *OpenAI Eval: A Framework for Evaluating LLMs*. (2023). <https://github.com/openai/evals>.
- 1267 S. Ouyang, J. M. Zhang, M. Harman, and M. Wang. 2024. “An Empirical Study of the Non-determinism of ChatGPT in
 1268 Code Generation.” *ACM Transactions on Software Engineering and Methodology*, 34, 2, 1–28. doi:[10.1145/3697010](https://doi.org/10.1145/3697010).
- 1269 G. Padovani, V. Ananthraj, and S. Fiore. 2025. “yProv4ML: Effortless Provenance Tracking for Machine Learning Systems.”
 1270 *SoftwareX*, 29, 102028. doi:[10.1016/j.softx.2025.102028](https://doi.org/10.1016/j.softx.2025.102028).
- 1271 J. Pineau, P. Vincent-Lamarre, K. Sinha, V. Larivière, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and H. Larochelle. 2021.
 1272 “Improving Reproducibility in Machine Learning Research: A Report from the NeurIPS 2019 Reproducibility Program.”
 1273 *Journal of Machine Learning Research*, 22, 164, 1–20. <https://jmlr.org/papers/v22/20-303.html>.

- 1270 C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. 2020. “Exploring the
 1271 Limits of Transfer Learning with a Unified Text-to-Text Transformer.” *Journal of Machine Learning Research*, 21, 140,
 1272 1–67. <https://jmlr.org/papers/v21/20-074.html>.
- 1273 S. Samuel and B. König-Ries. 2022. “A Provenance-based Semantic Approach to Support Understandability, Reproducibility,
 1274 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).
- 1275 V. Stodden, M. McNutt, D. H. Bailey, E. Deelman, Y. Gil, B. Hanson, M. A. Heroux, J. P. Ioannidis, and M. Taufer. 2016.
 1276 “Enhancing Reproducibility for Computational Methods.” *Science*, 354, 6317, 1240–1241. doi:[10.1126/science.aah6168](https://doi.org/10.1126/science.aah6168).
- 1277 A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 2017. “Attention
 1278 is All You Need.” In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc. arXiv:
 1279 [1706.03762 \(cs.CL\)](https://arxiv.org/abs/1706.03762).
- 1280 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
 1281 Prompting Elicits Reasoning in Large Language Models.” In: *Advances in Neural Information Processing Systems*.
 1282 Vol. 35, 24824–24837. arXiv: [2201.11903 \(cs.CL\)](https://arxiv.org/abs/2201.11903).
- 1283 M. D. Wilkinson et al.. 2016. “The FAIR Guiding Principles for Scientific Data Management and Stewardship.” *Scientific
 1284 Data*, 3, 160018. doi:[10.1038/sdata.2016.18](https://doi.org/10.1038/sdata.2016.18).
- 1285 J. Yuan et al.. 2025. “Understanding and Mitigating Numerical Sources of Nondeterminism in LLM Inference.” In: *Advances
 1286 in Neural Information Processing Systems*. Vol. 38. arXiv preprint. Curran Associates, Inc. arXiv: [2506.09501 \(cs.LG\)](https://arxiv.org/abs/2506.09501).
- 1287 M. Zaharia et al.. 2018. “Accelerating the Machine Learning Lifecycle with MLflow.” *IEEE Data Engineering Bulletin*, 41,
 1288 4, 39–45. <http://sites.computer.org/debull/A18dec/p39.pdf>.
- 1289 T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. 2020. “BERTScore: Evaluating Text Generation with
 1290 BERT.” In: *Proceedings of the 8th International Conference on Learning Representations*. <https://openreview.net/forum?id=SkeHuCVFDr>.
- 1291
- 1292
- 1293
- 1294
- 1295
- 1296
- 1297
- 1298
- 1299
- 1300
- 1301
- 1302
- 1303
- 1304
- 1305
- 1306
- 1307
- 1308
- 1309
- 1310
- 1311
- 1312
- 1313
- 1314
- 1315
- 1316

1317 **A Reproducibility Checklist**

1318 The following checklist is designed for self-assessment of reproducibility in generative AI studies. Each
 1319 item maps to a specific field or artifact in our protocol.

1320

1321 **Prompt Documentation**

- 1322 (1) Is the exact prompt text recorded and versioned? [Prompt Card: prompt_text, prompt_hash]
- 1323 (2) Are design assumptions and limitations documented? [Prompt Card: assumptions, limitations]
- 1324 (3) Is the expected output format specified? [Prompt Card: expected_output_format]
- 1325 (4) Is the interaction regime documented (single/multi-turn)? [Prompt Card: interaction_regime]

1327 **Model and Environment**

- 1328 (5) Is the model name and version recorded? [Run Card: model_name, model_version]
- 1329 (6) Are model weights hashed for identity verification? [Run Card: weights_hash]
- 1330 (7) Is the execution environment fingerprinted? [Run Card: environment, environment_hash]
- 1331 (8) Is the source code version recorded? [Run Card: code_commit]

1333 **Execution and Output**

- 1335 (9) Are all inference parameters logged? [Run Card: inference_params]
- 1336 (10) Is the random seed recorded? [Run Card: inference_params.seed]
- 1337 (11) Is the output cryptographically hashed? [Run Card: output_hash]
- 1338 (12) Are execution timestamps recorded? [Run Card: timestamp_start, timestamp_end]
- 1339 (13) Is logging overhead measured separately? [Run Card: logging_overhead_ms]

1340 **Provenance**

- 1342 (14) Is a provenance graph generated per group? [PROV-JSON document]
- 1343 (15) Are provenance documents in an interoperable format? [W3C PROV standard]

1344 **B Run Card Schema**

1346 The complete Run Card schema, with data types and descriptions:

1347

1348 Listing 1. Run Card JSON schema (simplified).

```

1349 {
1350   "run_id": "string (unique identifier)",
1351   "task_id": "string (task identifier)",
1352   "task_category": "string (e.g., summarization)",
1353   "prompt_hash": "string (SHA-256 of prompt)",
1354   "prompt_text": "string (full prompt text)",
1355   "input_text": "string (input to the model)",
1356   "input_hash": "string (SHA-256 of input)",
1357   "model_name": "string (e.g., llama3:8b)",
1358   "model_version": "string (e.g., 8.0B)",
1359   "weights_hash": "string (SHA-256 of weights)",
1360   "model_source": "string (e.g., ollama-local)",
1361   "inference_params": {
1362     "temperature": "float",
1363     "top_p": "float",

```

```

1364 16   "top_k": "integer",
1365 17   "max_tokens": "integer",
1366 18   "seed": "integer|null",
1367 19   "decoding_strategy": "string"
1368 20 },
1369 21   "params_hash": "string (SHA-256 of params)",
1370 22   "environment": {
1371 23     "os": "string",
1372 24     "os_version": "string",
1373 25     "architecture": "string",
1374 26     "python_version": "string",
1375 27     "hostname": "string",
1376 28     "timestamp": "ISO 8601 datetime"
1377 29 },
1378 30   "environment_hash": "string (SHA-256)",
1379 31   "code_commit": "string (git commit hash)",
1380 32   "researcher_id": "string",
1381 33   "affiliation": "string",
1382 34   "timestamp_start": "ISO 8601 datetime",
1383 35   "timestamp_end": "ISO 8601 datetime",
1384 36   "output_text": "string (model output)",
1385 37   "output_hash": "string (SHA-256 of output)",
1386 38   "output_metrics": "object (task-specific)",
1387 39   "execution_duration_ms": "float",
1388 40   "logging_overhead_ms": "float",
1389 41   "storage_kb": "float",
1390 42   "system_logs": "string (raw system info)",
1391 43   "errors": "array of strings",
1392 44
1393 45 // --- API-specific optional fields ---
1394 46   "api_request_id": "string|null (provider request ID)",
1395 47   "api_response_headers": "object|null (selected headers)",
1396 48   "api_model_version_returned": "string|null",
1397 49   "api_region": "string|null (if available)",
1398 50   "seed_status": "string (sent|logged-only|not-supported)",
1399 51
1400 52 // --- Multi-turn extension fields ---
1401 53   "conversation_history_hash": "string|null (SHA-256)",
1402 54   "turn_index": "integer|null",
1403 55   "parent_run_id": "string|null",
1404 56
1405 57 // --- RAG extension fields ---
1406 58   "retrieval_context": "string|null",
1407 59   "retrieval_context_hash": "string|null (SHA-256)"
1408 }
```

C Example PROV-JSON Document

1408 An abbreviated example of a PROV-JSON document generated for a single summarization run:

1409

1410

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

```

1411
1412 {
1413   "prefix": {
1414     "genai": "https://genai-prov.org/ns#",
1415     "prov": "http://www.w3.org/ns/prov#"
1416   },
1417   "entity": {
1418     "genai:prompt_c9644358": {
1419       "prov:type": "genai:Prompt",
1420       "genai:hash": "c9644358805b...",
1421       "genai:task_category": "summarization"
1422     },
1423     "genai:model_llama3_8b": {
1424       "prov:type": "genai:ModelVersion",
1425       "genai:name": "llama3:8b",
1426       "genai:source": "ollama-local"
1427     },
1428     "genai:output_590d0835": {
1429       "prov:type": "genai:Output",
1430       "genai:hash": "590d08359e7d..."
1431     }
1432   },
1433   "activity": {
1434     "genai:run_llama3_8b_sum_001_C1_rep0": {
1435       "prov:type": "genai:RunGeneration",
1436       "prov:startTime": "2026-02-07T21:54:34Z",
1437       "prov:endTime": "2026-02-07T21:54:40Z"
1438     }
1439   },
1440   "wasGeneratedBy": {
1441     "_:wGB1": {
1442       "prov:entity": "genai:output_590d0835",
1443       "prov:activity": "genai:run_llama3_8b..."
1444     }
1445   },
1446   "used": {
1447     "_:u1": {
1448       "prov:activity": "genai:run_llama3_...",
1449       "prov:entity": "genai:prompt_c9644358"
1450     }
1451   },
1452   "agent": {
1453     "genai:researcher_lucas_rover": {
1454       "prov:type": "prov:Person",
1455       "genai:affiliation": "UTFPR"
1456     }
1457   },
1458   "wasAssociatedWith": {
1459     "_:wAW1": {
1460

```

1458 Table 10. JSON extraction quality metrics by model and condition. *Raw Valid* = output parses directly as JSON;
 1459 *Extracted Valid* = JSON extracted via regex from outputs containing preamble text; *Schema* = all five expected fields
 1460 present; *Field EMR* = within-abstract pairwise exact match across runs for each extracted field, averaged over abstracts
 1461 (see Section D for interpretation). LLaMA 3 always prepends introductory text (e.g., “Here is the extracted information
 1462 in JSON format.”), yielding 0% raw validity but near-perfect extracted validity at $t=0$.
 1463

1464	Model	Cond.	Raw	Extr.	Schema	Within-Abstract Field EMR					Overall
						1465 Valid	Valid	Compl.	obj	meth	key_r
1466	LLaMA 3	C1 ($t=0$)	0%	100%	100%	0.987	0.987	0.987	1.000	0.987	0.989
		C2 ($t=0$)	0%	100%	100%	0.987	0.987	0.987	1.000	0.987	0.989
		C3 ($t=0.0$)	0%	100%	100%	0.978	0.978	0.978	1.000	0.978	0.982
		C3 ($t=0.3$)	0%	97.8%	97.8%	0.747	0.460	0.552	0.862	0.805	0.685
		C3 ($t=0.7$)	0%	92.2%	92.2%	0.522	0.167	0.267	0.611	0.711	0.456
1471	GPT-4	C2 ($t=0$)	100%	100%	100%	0.773	0.667	0.637	0.893	0.863	0.767
		C3 ($t=0.0$)	100%	100%	100%	0.833	0.571	0.667	0.905	0.810	0.757
		C3 ($t=0.3$)	100%	100%	100%	0.405	0.262	0.452	0.762	0.690	0.514
		C3 ($t=0.7$)	100%	100%	100%	0.137	0.157	0.255	0.667	0.725	0.388

```

49      "prov:activity": "genai:run_llama3_...",
50      "prov:agent": "genai:researcher_..."
51    }
52  }
53}
```

D JSON Extraction Quality

Table 10 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.

Second, within-abstract field-level exact match rates—computed by comparing only runs of the *same* abstract under the same condition, then averaging across abstracts—confirm the overall reproducibility hierarchy. Under greedy decoding, LLaMA 3 achieves near-perfect field EMR (0.982–0.989 overall), with all five fields at or above 0.978, consistent with the overall extraction EMR of 0.987 reported in Table 4. GPT-4 under greedy shows lower field EMR (0.757–0.767 overall), with open-ended fields (`method`: 0.667, `key_result`: 0.637) lagging behind structured fields (`model_or_system`: 0.893, `benchmark`: 0.863). As temperature increases, this gap widens: at $t=0.7$, `method` drops to 0.167 (LLaMA) and 0.157 (GPT-4), while `benchmark` retains 0.711 and 0.725 respectively—a 4–5× difference. This within-abstract formulation isolates true reproducibility (same input, same conditions, different runs) from between-abstract content variation, providing a methodologically clean measure of field-level consistency.

1505 E Prompt Card Example

1506 The following is a complete, filled-in Prompt Card for the summarization task as used in our experiments:

1507
1508 Listing 3. Prompt Card for the scientific summarization task.

```

1509 1 {
1510 2   "prompt_id": "summarization_v1",
1511 3   "prompt_hash": "c9644358805b4a7e...",
1512 4   "version": "1.0.0",
1513 5   "task_category": "summarization",
1514 6   "objective": "Produce a 3-sentence summary of a
1515 7     scientific abstract covering: (1) main
1516 8       contribution, (2) methodology, (3) key result.",
1517 9   "assumptions": [
1518 10    "Input is a single English scientific abstract",
1519 11    "Abstract contains identifiable methodology
1520 12      and quantitative results",
1521 13    "Model can produce coherent 3-sentence output"
1522 14  ],
1523 15   "limitations": [
1524 16    "Open-ended phrasing allows high output variance",
1525 17    "No explicit output-format constraint (unlike
1526 18      extraction task)"
1527 19  ],
1528 20   "target_models": [
1529 21    "llama3:8b", "mistral:7b", "gemma2:9b",
1530 22    "gpt-4", "claude-sonnet-4-5"
1531 23  ],
1532 24   "expected_output_format": "Three sentences of
1533 25     plain text, no JSON or structured markup",
1534 26   "interaction_regime": "single-turn",
1535 27   "change_log": [
1536 28     {"date": "2026-02-06", "change": "Initial version"}
1537 29  ]
1538 30 }
```

1538 F Representative Prompt Templates

1539 The following are the exact prompt templates used for each of the four experimental tasks. In all templates,
1540 {abstract} is replaced with the scientific abstract text at runtime.

1541 Task 1: Scientific Summarization

```
1542   Summarize the following scientific abstract in exactly 3 sentences. Cover: (1) the
1543   main contribution, (2) the methodology used, and (3) the key quantitative result.\n\nAbstract:
1544   {abstract}\n\nSummary:
```

1545 Task 2: Structured Extraction

```
1546   Extract the following fields from the scientific abstract below. Return JSON only, no
1547   explanation.\n\nFields: objective, method, key_result, model_or_system, benchmark\n\nAbstract:
1548   {abstract}\n\nJSON:
```

1549 Task 3: Multi-Turn Refinement (3 turns)

1552 Turn 1: [Same as Task 1 prompt]\n Turn 2: Now revise the summary to be more specific
 1553 about the quantitative results mentioned.\n Turn 3: Finally, add one sentence about
 1554 the limitations or future work mentioned in the abstract.

1555 Task 4: RAG Extraction

1556 Using the context passage below and the scientific abstract, extract the following
 1557 fields. Return JSON only.\n\nContext: {retrieved_passage}\nAbstract: {abstract}\n\nFields:
 1558 objective, method, key_result, model_or_system, benchmark\n\nJSON:

1559 G Experimental Coverage Matrix

1560 Table 11 provides a complete coverage matrix showing the number of abstracts and runs per model–task–
 1561 condition combination. This matrix enables readers to verify the sample sizes underlying all reported
 1562 metrics.

1563 Table 11. Experimental coverage: number of abstracts (runs) per model–task–condition. Dash (–) indicates the
 1564 combination was not evaluated. C1: fixed seed; C2: variable seed (C2_same_params for GPT-4); C3: temperature sweep
 1565 at $t \in \{0.0, 0.3, 0.7\}$.

Model	Task	C1	C2	C3 ($t=0.0$)	C3 ($t=0.3$)	C3 ($t=0.7$)
LLaMA 3 8B	Extraction	30 (150)	30 (150)	30 (90)	30 (90)	30 (90)
	Summarization	30 (150)	30 (150)	30 (90)	30 (90)	30 (90)
	Multi-turn	10 (50)	–	–	–	–
	RAG	10 (50)	–	–	–	–
Mistral 7B	Extraction	10 (50)	10 (50)	10 (30)	10 (30)	10 (30)
	Summarization	10 (50)	10 (50)	10 (30)	10 (30)	10 (30)
	Multi-turn	10 (50)	–	–	–	–
	RAG	10 (50)	–	–	–	–
Gemma 2 9B	Extraction	10 (50)	10 (50)	10 (30)	10 (30)	10 (30)
	Summarization	10 (50)	10 (50)	10 (30)	10 (30)	10 (30)
	Multi-turn	10 (50)	–	–	–	–
	RAG	10 (50)	–	–	–	–
GPT-4	Extraction	–	30 (150)	17 (51)	17 (51)	14 (42)
	Summarization	3 (8) [†]	30 (150)	30 (90)	30 (90)	30 (90)
	Multi-turn	–	–	–	–	–
	RAG	–	–	–	–	–
Claude Sonnet 4.5	Extraction	10 (49) [‡]	10 (50)	10 (30)	10 (30)	10 (30)
	Summarization	10 (50)	10 (50)	10 (30)	10 (30)	10 (30)
	Multi-turn	10 (50)	–	–	–	–
	RAG	10 (50)	–	–	–	–

1590 [†]GPT-4 C1 summarization: only 3 abstracts completed before quota exhaustion; excluded from primary analysis (C2 used
 1591 instead).

1592 [‡]Claude C1 extraction: 49 runs (1 empty output due to API timeout).

1593

1594 H Chat-Format Control Experiment

1595 To assess whether the prompt-format difference between LLaMA 3 (completion-style via /api/generate)
 1596 and GPT-4 (chat-style via Chat Completions) contributes to the observed reproducibility gap, we
 1597

1599 conducted a supplementary control experiment running LLaMA 3 8B through Ollama's `/api/chat`
 1600 endpoint, which applies the model's chat template (including special tokens for system/user/assistant
 1601 roles) in the same message structure used by GPT-4.

1602 **Design:** 10 abstracts \times 2 tasks \times 2 conditions (C1, C2) \times 5 repetitions = 200 runs, all under greedy
 1603 decoding ($t=0$).

1604 **Results:** Table 12 compares the chat-format control with the original completion-format results for
 1605 the same 10 abstracts. The two prompt formats produce *identical* variability metrics across all conditions:
 1606 summarization EMR = 0.929, NED = 0.0066, and ROUGE-L = 0.9922 in both modes; extraction achieves
 1607 perfect reproducibility (EMR = 1.000) regardless of interface. The 0.929 summarization EMR reflects the
 1608 warm-up effect on 2 of 10 abstracts—the same pattern observed in the full 30-abstract experiment. These
 1609 results confirm that prompt format is not a source of variability, and the reproducibility gap between
 1610 LLaMA 3 and GPT-4 is consistent with deployment-side factors (server infrastructure, floating-point
 1611 non-determinism across GPU types, request batching) rather than prompt-format differences.

1612 Table 12. Prompt-format control: LLaMA 3 8B via completion (`/api/generate`) vs. chat (`/api/chat`) for 10 abstracts
 1613 under greedy decoding ($t=0$). EMR computed over conditions C1 and C2 combined.
 1614

	Task	Metric	Completion	Chat
Summarization		EMR↑	0.929	0.929
		NED↓	0.0066	0.0066
		ROUGE-L↑	0.9922	0.9922
Extraction		EMR↑	1.000	1.000
		NED↓	0.0000	0.0000
		ROUGE-L↑	1.0000	1.0000

1623 *Note:* Completion and chat formats yield identical metrics for all 10 abstracts under greedy decoding, indicating that
 1624 prompt format is not a source of variability.

I API Payload Documentation

1628 To address potential “apples-to-oranges” concerns, we document the exact payload structures sent to each
 1629 inference endpoint. All payloads were constructed deterministically and logged as part of the Run Card.

1630 **Local models (Ollama).** Single-turn tasks use `POST /api/generate`:

1631 **Ollama generate payload (Tasks 1–2):**

```
1632 {"model": "llama3:8b",
1633   "prompt": "<full prompt text>",
1634   "options": {"temperature": 0.0, "seed": 42, "num_predict": 1024},
1635   "stream": false}
```

1636 The `model` field is set to `llama3:8b`, `mistral:7b`, or `gemma2:9b` as appropriate. Multi-turn tasks (Task 3)
 1637 use `POST /api/chat` with accumulated `messages` array. No system prompt, stop sequences, or post-
 1638 processing are applied.

1639 **GPT-4 (OpenAI).** Accessed via the `openai` Python SDK v1.59.9:

```
1640 {"model": "gpt-4",
1641   "messages": [{"role": "user", "content": "<prompt>"}],
1642   "temperature": 0.0, "seed": 42, "max_tokens": 1024}
```

1643 No system message, stop sequences, `top_p`, `frequency_penalty`, or `presence_penalty` were set (all
 1644 defaults). The resolved model version (`gpt-4-0613`) was extracted from the response object and logged.

1646 **Claude Sonnet 4.5 (Anthropic).** Accessed via `urllib` (no SDK dependency):

```
1647     {"model": "claude-sonnet-4-5-20250929",
1648         "messages": [{"role": "user", "content": "<prompt>"}],
1649         "temperature": 0.0, "max_tokens": 1024}
```

1650 No `seed` parameter (not supported by the Anthropic API), no system message, no stop sequences. The
1651 seed value in the Run Card is marked `seed_status: "logged-only-not-sent-to-api"`.

1652 **Key symmetry points.** Across all five models: (1) identical prompt text (verified by `prompt_hash`);
1653 (2) identical temperature ($t=0.0$); (3) identical max token limit (1,024); (4) no system messages; (5) no
1654 stop sequences; (6) no post-processing or text normalization of outputs.

1655

1656 Received February 2026

1657

1658

1659

1660

1661

1662

1663

1664

1665

1666

1667

1668

1669

1670

1671

1672

1673

1674

1675

1676

1677

1678

1679

1680

1681

1682

1683

1684

1685

1686

1687

1688

1689

1690

1691

1692