

Open-Schema Metadata Extraction from GitHub READMEs: A Cross-Model Agreement Study

Genet-Asefa Gesese^{1,2,*,†}, Zongxiong Chen^{3,*,†}, Sonja Schimmler^{3,4} and Harald Sack^{1,2}

¹FIZ Karlsruhe – Leibniz Institute for Information Infrastructure, Germany

²Karlsruhe Institute of Technology, Institute AIFB, Germany

³Fraunhofer FOKUS, Germany

⁴TU Berlin, Germany

Abstract

GitHub README files contain information needed to discover and reuse scientific artifacts, yet much of it is absent from conventional scholarly metadata. Large language models (LLMs) can convert this heterogeneous content into structured metadata, but open-schema extraction is difficult to evaluate without document-level gold annotations and consistent field names. To address this challenge, we applied an agreement-based evaluation methodology for assessing open-schema metadata extraction without requiring document-level gold metadata. The methodology is applied to metadata independently generated by six representative LLMs from 200 README files in the NERdME corpus using a zero-shot, evidence-grounded prompt. Output well-formedness, extraction cardinality, field-name diversity, and agreement over fields, values, and field–value pairs are evaluated under exact and embedding-based semantic matching. JSON validity ranges from 0.990 to 1.000 and record completeness is at least 0.998, although extraction cardinality varies substantially across models. Semantic matching raises multi-model field–value Jaccard similarity from 0.0000 to 0.1053 and Krippendorff’s alpha from 0.1784 to 0.5237. Agreement differences between open- and closed-model groups are limited, whereas within-family agreement consistently exceeds cross-family agreement. These results show that models often extract semantically similar metadata while organizing it differently, demonstrating that inter-model agreement is a useful diagnostic for evaluating open-schema metadata extraction, although it is not a substitute for correctness validation.

Keywords

Open-schema metadata extraction, Large language model, Scholarly information infrastructure, Agreement-based evaluation

1. Introduction

Scientific knowledge infrastructures increasingly represent research outputs beyond publications. Software, datasets, models, workflows, benchmarks, and other implementation artifacts are essential for understanding, reproducing, and extending scientific results. Open scholarly infrastructures, therefore, seek to connect publications with these complementary outputs [1], which requires rich, structured metadata about each artifact. Depending on the artifact, this metadata may describe its purpose, dependencies, datasets, evaluation procedures, license, usage instructions, or associated publications. In practice, however, such metadata is often incomplete, inconsistent, or unavailable in structured form.

GitHub repositories are an important source of artifact-level information. README files in these repositories commonly describe the artifact’s purpose, installation procedure, dependencies, supported environments, datasets, evaluation metrics, licenses, publications, and usage examples. Unlike bibliographic metadata, however, README files follow no universal schema. Relevant information may appear in headings, badges, prose, lists, tables, links, or code blocks. This heterogeneity makes conventional rule-based extraction and fixed-schema information extraction difficult.

6th International Workshop on Scientific Knowledge: Representation, Discovery, and Assessment, Oct 2026, Bari, Italy

*Corresponding author.

†These authors contributed equally.

✉ genet-asefa.gesese@fiz-karlsruhe.de (G. Gesese); zongxiong.chen@fokus.fraunhofer.de (Z. Chen);

sonja.schimmler@fokus.fraunhofer.de (S. Schimmler); harald.sack@fiz-karlsruhe.de (H. Sack)

ORCID 0000-0003-3807-7145 (G. Gesese); 0000-0003-2452-0572 (Z. Chen); 0000-0002-8786-7250 (S. Schimmler);

0000-0001-7069-9804 (H. Sack)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

Large language models (LLMs) have recently been applied to scientific information extraction and metadata generation [2, 3]. By following natural language instructions, they can extract structured metadata from heterogeneous documents without requiring task-specific extraction rules. More importantly, they can perform open-schema extraction, identifying metadata that is relevant for a particular repository rather than restricting extraction to a predefined set of fields. This flexibility is particularly valuable for GitHub README files, where the information exposed by a software library, dataset, benchmark, pretrained model, or experimental repository may differ substantially. This work does not argue that open-schema extraction is generally preferable to closed-schema extraction. Rather, it considers an open-schema setting because the metadata associated with research code repositories is heterogeneous and cannot always be anticipated by a fixed vocabulary.

However, evaluating LLM-based open-schema extraction for README metadata extraction presents two important challenges. First, useful metadata is inherently document-dependent. A software library, dataset repository, benchmark, pretrained model, or experimental artifact may expose substantially different metadata. Restricting extraction to a fixed schema may therefore omit information that is important for understanding or reusing a particular artifact. Although open-schema extraction allows models to identify metadata that is appropriate for each repository, different models may organize the same underlying information using different field names or metadata structures.

Second, existing README annotation resources are usually designed for mention-level named-entity recognition rather than document-level metadata evaluation. NERdME [4], for example, provides manually annotated entity mentions in research-related README files. It does not define complete document-level metadata records for each artifact. Consequently, conventional evaluation metrics such as precision, recall, and F1 cannot be directly applied to open-schema extraction outputs.

These challenges motivate our study. Rather than asking which model is most accurate, we investigate the reliability and compatibility of independently generated metadata in the absence of document-level gold annotations. We further examine whether apparent disagreement reflects genuinely different metadata selections or surface-form variation in how models express the same concepts. We address the following research questions (RQs): **RQ1: *How well-formed are LLM-generated open-schema metadata outputs, and how do extraction cardinality and field-name diversity vary across models?*** **RQ2: *To what extent do LLM-generated open-schema metadata outputs agree?*** **RQ3: *How does metadata-extraction agreement differ across model categories and model families?***

To answer these questions, this study evaluates contemporary LLMs on GitHub README files from the NERdME dataset, including six representative models used in the main cross-model analysis. Each model receives the same zero-shot prompt and produces evidence-grounded JSON metadata. We assess output validity, extraction cardinality, and field-name diversity, then compare model outputs using pairwise and multi-model agreement metrics under exact and semantic matching based on sentence-transformer embeddings at three levels of representation: metadata fields, metadata values, and field-value pairs. We additionally examine agreement within open- and closed-model groups and within and across model families. The main contributions of this study are as follows:

- **Open-schema metadata extraction.** Metadata extraction from GitHub README files is formulated as an evidence-grounded, open-schema information extraction task.
- **Agreement without gold metadata.** An agreement-based methodology is introduced for evaluating open-schema metadata extraction from README files without requiring document-level gold metadata.
- **Cross-model agreement analysis.** A systematic comparison across models, model groups, and model families characterizes where open-schema metadata outputs converge and where they remain inconsistent under exact and semantic matching.

The remainder of the paper is organized as follows. Section 2 reviews related work and identifies the research gap, followed by Section 3, which formalizes the open-schema metadata extraction task and presents the proposed agreement-based evaluation methodology. Section 4 then describes the experimental setup, evaluation protocol, and empirical results organized around the three research questions. Finally, Sections 5 and 6 discuss the study limitations and conclude the paper, respectively.

2. Related Work

2.1. Metadata Extraction from Scientific Documents

Metadata extraction has traditionally focused on scholarly publications. Systems such as GROBID [5] and CERMINE [6] transform scientific PDFs into structured bibliographic records, extracting titles, authors, affiliations, references, and document structure. These systems operate primarily with predefined metadata schemas appropriate for publications. Beyond individual systems, shared tasks such as MESD [7] have benchmarked metadata extraction from scholarly PDFs across a range of extraction paradigms. This progression spans rule-based reference parsing [8], CRF-based sequence labeling applied to research papers [9], neural sequence models such as bidirectional LSTM-CRF architectures [10], and computer-vision-based document layout analysis [11], underscoring the continued difficulty of the task for documents with non-standard layouts.

More recent work has explored LLM-based extraction. Dagdelen et al. [2] investigated structured information extraction from scientific text using LLMs. MOLE [3] introduced an LLM-based framework for extracting and validating metadata from scientific papers. Beyond scientific articles, LLMs have also been applied to extract descriptive metadata from literary texts, for example, generating catalog fields such as theme, keywords, and character information from digital-humanities corpora, with quality assessed through expert review rather than an automatic gold standard [12]. Across these studies, LLMs are used to generate structured metadata from heterogeneous documents, but they assume a predefined output schema, whereas the present study considers open-schema metadata extraction.

Our work differs in both source documents and output schema: we process repository README files rather than scientific articles. README metadata includes operational and reuse-oriented information such as installation commands, software dependencies, checkpoints, evaluation datasets, hardware requirements, and licenses. Moreover, rather than assuming a fixed target schema, we permit models to identify fields that are useful for the specific artifact.

2.2. Information Extraction from Code Repositories

Research repositories contain information that may be absent from associated publications. NERdME was introduced to support named-entity recognition over GitHub README files associated with research artifacts [4]. It represents entities such as datasets, software, publications, metrics, programming languages, and scientific events. NERdME nevertheless provides mention-level annotations. A README may contain repeated mentions of the same software, variations in capitalization, or multiple entities belonging to the same metadata category. Turning these mentions into document-level metadata requires canonicalization, aggregation, cardinality handling, and potentially contextual role classification.

Separately, community efforts such as CodeMeta and the Citation File Format (CFF) define *prescriptive* vocabularies that repository maintainers are encouraged to populate for software citation and discovery [13]. In practice, most repositories do not adopt these schemas, and the relevant information remains embedded in free-text README prose rather than in a machine-readable metadata file. This gap motivates extracting metadata directly from README text rather than assuming that structured metadata is already available.

Automatic extraction of software metadata from repositories has also been investigated. SOMEF [14] extracts predefined software metadata from GitHub repositories by combining supervised classifiers, heuristics, repository metadata, and documentation. The extracted information is mapped to established software metadata vocabularies, enabling repository discovery and knowledge graph construction. Like CodeMeta and CFF, SOMEF assumes a predefined metadata schema and evaluates extraction against known metadata categories.

Our study uses the README documents from NERdME as an evaluation corpus, but does not treat its mention annotations as document-level gold metadata. Rather than extracting predefined metadata categories, we examine whether LLMs agree when independently constructing open-schema metadata records from the same README documents.

2.3. Open-Schema Information Extraction

Our extraction task is open-schema: rather than filling values into a fixed, predefined set of metadata fields, a model must also select which fields are relevant for a given artifact. A useful conceptual analogue is Open Information Extraction (OpenIE) [15], which similarly removed a fixed target vocabulary from a different extraction task, i.e., relation extraction. Where classical relation extraction required target relations to be specified in advance, OpenIE removed this requirement, extracting relational tuples directly from text without a pre-specified relation vocabulary. Subsequent systems refined the OpenIE paradigm along different axes: ReVerb introduced syntactic and lexical constraints on verb-mediated relations to reduce uninformative extractions [16], while OLLIE relaxed the requirement that relations be verb-mediated and incorporated contextual information such as attribution and enabling conditions [17]. Across these systems, extractions are evaluated against a fixed tuple structure (subject, relation, object), and the open element is a short relation phrase drawn from a single sentence.

Our task differs from OpenIE in both unit and scope: the extracted item is a field–value pair describing a research artifact rather than a subject–relation–object triple, and the source is a README file rather than a single sentence. Our task shares OpenIE’s principle of avoiding a predefined output vocabulary, but applies it to document-level metadata extraction rather than relation extraction.

2.4. Agreement as an Evaluation Methodology

Our evaluation strategy builds on a body of work that treats agreement, rather than comparison against a gold standard, as the primary indicator of annotation reliability. Measures such as Krippendorff’s alpha were developed to quantify the reliability of human coding schemes by measuring agreement while accounting for chance [18]. Artstein and Poesio established Krippendorff’s alpha and related chance-corrected coefficients as standard measures of annotation reliability in computational linguistics, particularly for tasks where an objective reference annotation is unavailable or difficult to define [19].

More recently, agreement-based evaluation has been extended from human annotators to LLMs. Gilardi et al. showed that ChatGPT achieved intercoder agreement comparable to, and in some cases exceeding, that of crowd-workers and trained human annotators on text annotation tasks [20]. Ahmed et al. further proposed model–model agreement, rather than agreement with a human-authored gold standard, as an indicator of whether annotation tasks are suitable for LLM automation, demonstrating that model–model agreement correlates with human–model agreement across ten software engineering annotation tasks [21].

Our study extends agreement-based evaluation to open-schema metadata extraction. Rather than discrete annotation labels, the units of analysis are open-schema field–value pairs extracted from GitHub README files. Furthermore, rather than evaluating predictions against document-level reference metadata, the study examines agreement among independently prompted LLMs in the complete absence of document-level gold annotations.

3. Study Design

This section formalizes the open-schema metadata extraction task and introduces the agreement-based evaluation methodology used throughout the paper.

3.1. Task Definition

Let $\mathcal{D} = \{d_1, \dots, d_N\}$ denote a corpus of N GitHub README documents. For each document, the task is to extract metadata as a set of field–value pairs without assuming a predefined schema. The extracted information is represented using metadata fields generated from the document content rather than selected from a fixed vocabulary. Consequently, the resulting metadata schema is document-dependent and may differ across repositories. We refer to this setting as open-schema metadata extraction.

Formally, given a GitHub README document d and a model m , the extracted metadata is represented as:

$$O_{d,m} = \{(f_i, v_i, e_i, l_i, c_i, r_i)\}_{i=1}^{K_{d,m}}, \quad (1)$$

where $K_{d,m} = |O_{d,m}|$ is the number of records extracted from document d by model m . Here f_i is a model-generated metadata field, v_i is the extracted value, e_i is the verbatim evidence span, l_i is the evidence location, c_i is the model-assigned confidence level, and r_i is a short rationale for why the item is useful. Although the agreement experiments consider only fields and values, the remaining attributes are retained to support inspection and future analyses of grounding and extraction quality.

The task is also evidence-grounded: every metadata item must be supported by a verbatim span from the README. The prompt explicitly prohibits unsupported inference and the use of external knowledge. The extracted metadata may include, for example, artifact identity, repository and paper links, datasets, models, dependencies, commands, configurations, hardware requirements, evaluation metrics, scientific context, licenses, citation information, and attribution.

3.2. Output Validity, Extraction Cardinality, and Field-Name Diversity

Before comparing metadata extracted by different models, we verify that outputs conform to the required format. These checks distinguish disagreement arising from extraction behavior from disagreement caused by malformed outputs. A model may produce well-formed structured metadata while differing substantially from other models in the amount and variety of metadata it extracts.

For each document-model pair, let $J_{d,m} = 1$ if the generated output parses successfully as JSON, and $J_{d,m} = 0$ otherwise. JSON validity is the proportion of the N README outputs from model m that parse successfully:

$$\text{JSONValid}(m) = \frac{1}{N} \sum_{d \in \mathcal{D}} J_{d,m}. \quad (2)$$

The successfully parsed outputs are therefore $\mathcal{D}_m^{\text{valid}} = \{d \in \mathcal{D} : J_{d,m} = 1\}$. We then measure record completeness at the item level. Let $C(o) = 1$ when all required attributes of record o are non-empty, and $C(o) = 0$ otherwise. For model m , record completeness is

$$\text{RecordComplete}(m) = \frac{\sum_{d \in \mathcal{D}_m^{\text{valid}}} \sum_{o \in O_{d,m}} C(o)}{\sum_{d \in \mathcal{D}_m^{\text{valid}}} |O_{d,m}|}. \quad (3)$$

The required attributes $(f_i, v_i, e_i, l_i, c_i, r_i)$ are defined in Formula 1. In addition, extraction cardinality is the number of metadata items produced for a README, while field-name diversity is the number of unique field names in that output. We summarize both distributions per model and report corpus-level exact and semantically normalized field-name counts. Together, these measures establish whether subsequent agreement differences may be influenced by malformed output, extraction cardinality, or field-name diversity.

3.3. Agreement Representations

Agreement between models may differ because they organize metadata using different fields, select different values, or do both. To distinguish these sources, we derive three set representations from each output.

The field representation contains the model-selected metadata fields:

$$F_{d,m} = \{f_i \mid (f_i, v_i, \dots) \in O_{d,m}\}. \quad (4)$$

The value representation contains extracted metadata values:

$$V_{d,m} = \{v_i \mid (f_i, v_i, \dots) \in O_{d,m}\}. \quad (5)$$

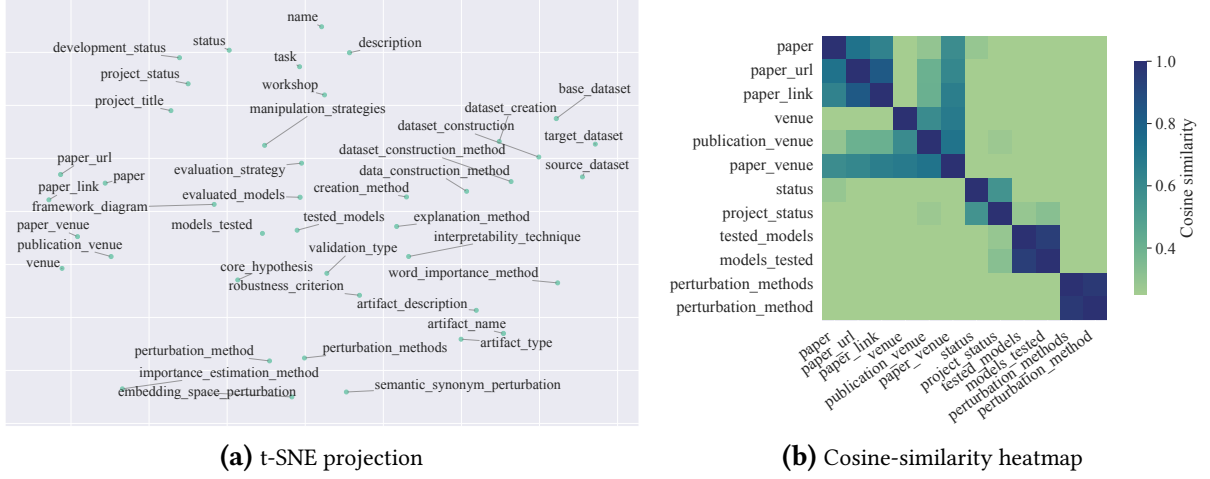


Figure 1: Semantic structure of field names extracted by six LLMs from one README using sentence-transformers/all-MiniLM-L6-v2. The t-SNE panel shows all 42 labels; the heatmap reports cosine similarities for 12 selected labels that illustrate surface-form variation.

The field-value representation contains complete metadata items as field-value pairs:

$$P_{d,m} = \{(f_i, v_i) \mid (f_i, v_i, \dots) \in O_{d,m}\}. \quad (6)$$

Field agreement evaluates whether models use similar metadata fields. Value agreement compares extracted content regardless of field naming. Field-value agreement requires agreement on both the content and its associated field.

3.4. Exact and Semantic Matching

We evaluate agreement using exact and semantic matching. Exact matching requires identical normalized strings; thus, labels such as `name` and `artifact_name` do not match even when they denote a similar concept. Semantic matching clusters related surface forms before scoring. We encode strings with a sentence-transformer [22], L2-normalize the embeddings, connect strings whose cosine similarity is at least τ , and treat each connected component as one semantic unit. Cluster identifiers replace the original strings when computing agreement. Section 4 specifies the encoder and thresholds. Figure 1 illustrates this distinction using 42 field names from one README. Exact matching separates variants such as `perturbation_methods/perturbation_method` and `tested_models/models_tested`, despite cosine similarities of 0.966 and 0.945. Semantic matching captures such naming variation but does not establish extraction accuracy. Connected components can also merge indirectly related strings through transitive links, so we retain exact matching as a baseline and evaluate threshold sensitivity.

3.5. Agreement Metrics and Statistical Testing

Agreement is evaluated using two complementary measures: Jaccard similarity, which quantifies the overlap between extracted metadata units, and Krippendorff’s alpha, which provides a chance-corrected measure of agreement. Both measures are computed separately for each representation (field, value, and field-value) under both exact and semantic matching.

For document d , representation $r \in \{\text{field, value, field-value}\}$, and matching mode $\mu \in \{\text{exact, semantic}\}$, let $S_{d,m}^{(r,\mu)}$ denote the set of extracted units produced by model m . For a set of models $\mathcal{M}$, Jaccard similarity is

$$J_d^{(r,\mu)}(\mathcal{M}) = \frac{|\bigcap_{m \in \mathcal{M}} S_{d,m}^{(r,\mu)}|}{|\bigcup_{m \in \mathcal{M}} S_{d,m}^{(r,\mu)}|}. \quad (7)$$

We aggregate document-level agreement using the median since it is robust to README files with unusually large or small metadata outputs. For the evaluated corpus $\mathcal{D}$, corpus-level Jaccard similarity is given:

$$J^{(r,\mu)}(\mathcal{M}) = \text{median}_{d \in \mathcal{D}} J_d^{(r,\mu)}(\mathcal{M}). \quad (8)$$

The overall analysis uses all six representative models in $\mathcal{M}$, while the Open and Closed group analyses use the three models in each group. For two models, the same definition reduces to pairwise Jaccard similarity, which we use for the heatmaps and family comparison. In every case, we compute one score per README and report the median across README files. We also report Krippendorff’s alpha [18] as a chance-corrected, joint multi-rater measure. Krippendorff’s alpha is computed over a global metadata universe induced by the exact or semantic units. For each document, each model either selects or does not select each possible unit. This common universe approximates evaluation under a shared schema, but its value can be influenced by sparse prevalence and shared absences; we therefore interpret alpha together with Jaccard similarity.

To assess observed differences, we perform two-sided paired randomization tests on README-level scores [23]. For each document, we compute the difference between two conditions, randomly invert the signs of nonzero differences, and estimate the probability of a mean difference at least as extreme as the observed value under the null hypothesis. We apply Holm correction [24] to the six outcomes in each comparison family (two metrics by three representations). Document-level pairing controls for README content; the resulting inference concerns the evaluated models across these documents, not a broader population of models.

4. Experimental Evaluation

4.1. Experimental Setup

We evaluate open-schema metadata extraction on 200 research-related GitHub README files from the NERdME corpus [4]. Each README is processed independently by every evaluated model using the same zero-shot, evidence-grounded prompt, and every model is instructed to return structured JSON without relying on external knowledge. To reduce sampling variability and support reproducibility, all requests are submitted through OpenRouter [25] with temperature set to 0. Reasoning was enabled using each model’s highest supported reasoning configuration. The exact prompt (Appendix A), model identifiers, request code, raw responses, and normalized JSON outputs are retained with the evaluation artifacts ¹.

The primary agreement analysis uses Qwen 3.7 Flash [26], DeepSeek V4 Flash [27], Kimi K3 [28], Claude Sonnet 5 [29], GPT-5.6 Luna Pro [30, 31], and Gemini 3.6 Flash [32]. The GPT citation documents the underlying Luna model, while the OpenRouter citation documents the Pro reasoning route used in our experiments. We compare the Open group (Qwen, DeepSeek, and Kimi) with the Closed group (Claude, GPT, and Gemini). The family analysis additionally includes Qwen 3.5 Flash, Qwen 3.6 Flash, DeepSeek V3.2, and DeepSeek Chat V3.1.

4.2. Evaluation Protocol

The experiments follow the methodology described in Section 3. Semantic matching uses the sentence-transformers/all-MiniLM-L6-v2 [33] model with a primary similarity threshold of $\tau = 0.85$. Threshold sensitivity is evaluated for $\tau \in \{0.70, 0.80, 0.85, 0.90\}$. README-level agreement scores are summarized by their median across documents. Group and family comparisons use the aggregation procedures described in Section 3.5, and statistical significance is assessed using two-sided paired randomization tests with Holm correction.

¹<https://github.com/chenzongxiong/metadata-extraction>

Table 1

Output validity and field-name diversity across representative models. JSON validity is the fraction of README outputs that parse successfully. Record completeness is the fraction of extracted metadata records containing all required attributes. Exact unique fields count distinct raw field names, while semantic unique fields count field-name clusters after semantic normalization.

Model	JSON validity	Record completeness	Unique fields	
			Exact	Semantic
Qwen	0.990	0.999	1,037	1,002
DeepSeek	0.995	0.999	2,007	1,938
Kimi	1.000	1.000	1,884	1,800
Claude	0.995	0.998	1,906	1,861
GPT	1.000	1.000	3,834	3,724
Gemini	0.995	1.000	1,044	1,006

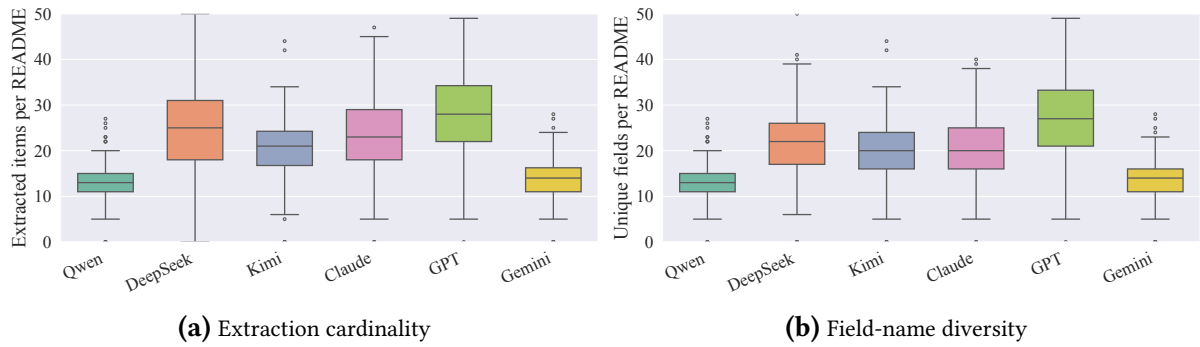


Figure 2: (a) Distribution of the number of metadata items extracted per README. (b) Distribution of the number of unique metadata field names extracted per README. Together, the two panels show how model verbosity and field-name diversity vary across representative LLMs.

4.3. Results and Analysis

This section presents the results according to the three research questions. We first assess output well-formedness, extraction cardinality, and field-name diversity (RQ1); we then quantify cross-model agreement under exact and semantic matching (RQ2); finally, we examine whether agreement varies across model groups and families (RQ3).

4.3.1. RQ1: How Well-Formed Are LLM-generated Open-Schema Metadata Outputs, and How Do Extraction Cardinality and Field-name Diversity Vary Across Models?

Agreement analysis first requires well-formed structured outputs. Differences in extraction cardinality and field-name diversity can also depress overlap independently of output well-formedness. We therefore establish whether the models produce parseable, structurally complete outputs and how much they extract before interpreting cross-model agreement.

LLMs almost always produce parseable and structurally complete metadata. Table 1 reports JSON validity from 0.990 to 1.000 and record completeness of at least 0.998. Thus, serialization failures are unlikely to be the primary explanation for the agreement patterns below.

Extraction cardinality varies substantially across models despite their similarly high JSON validity and record completeness. Figure 2 shows that Qwen and Gemini are the most concise, with median extraction cardinalities of 13 and 14 items per README, respectively. GPT is the most verbose, with a median of 28 items per README, followed by DeepSeek with 25, Claude with 23, and Kimi with 21. The same pattern appears for unique fields per README: Qwen and Gemini use a median of 13 and 14 unique fields, while GPT uses 27.

Table 2

Multi-model agreement among six representative LLMs under exact and semantic matching. Agreement is computed across Qwen 3.7 Flash, DeepSeek V4 Flash, Kimi K3, Claude Sonnet 5, GPT-5.6 Luna Pro, and Gemini 3.6 Flash.

Representation	Jaccard Similarity		Krippendorff’s Alpha	
	Exact	Semantic	Exact	Semantic
Fields	0.0133	0.0385	0.2789	0.3788
Values	0.0547	0.1266	0.4243	0.5650
Field–Value	0.0000	0.1053	0.1784	0.5237

Table 3

Significance summary from two-sided paired randomization tests over README-level scores. Cells report Holm-adjusted significance levels; model-group and family comparisons use semantic matching.

Comparison	Jaccard Similarity			Krippendorff Alpha		
	Field	Value	Field–Value	Field	Value	Field–Value
Semantic vs. Exact	***	***	***	***	***	***
Open-group vs. Closed-group	*	n.s.	n.s.	***	n.s.	n.s.
Within-family vs. Cross-family	***	***	***	***	***	***

Note: n.s. denotes $p \geq 0.05$; * denotes $p < 0.05$; ** denotes $p < 0.01$; *** denotes $p < 0.001$.

Corpus-level field-name diversity remains high after semantic normalization. GPT produces 3,834 exact unique field names, DeepSeek 2,007, Claude 1,906, Kimi 1,884, Gemini 1,044, and Qwen 1,037. Semantic clustering reduces these counts only modestly under the selected encoder, suggesting that simple surface variation alone does not explain the observed field-name diversity.

RQ1 Takeaway: LLMs consistently produce well-formed structured outputs, but extraction cardinality and field-name diversity vary substantially across models. High JSON validity and record completeness, therefore, do not imply consistency in the metadata selected or how it is organized.

4.3.2. RQ2: To What Extent Do LLM-generated Open-Schema Metadata Outputs Agree?

Open-schema extraction allows models to express the same concept using different field names or value formulations, making exact overlap potentially too strict. We therefore quantify cross-model agreement under both exact and semantic matching to distinguish surface-form variation from deeper differences in metadata selection.

Exact agreement is limited and is weakest for complete field–value records. Table 2 reports Jaccard similarities of 0.0133 for fields, 0.0547 for values, and 0.0000 for field–value pairs. Chance-corrected alpha gives the same ordering: values are the most stable representation, while complete field–value records are the least stable. Figure 3 further shows that this disagreement is broad rather than caused by one outlier model: every model pair has limited exact field–value overlap.

Semantic matching reveals significantly more agreement over extracted content than exact matching exposes. Value Jaccard increases from 0.0547 to 0.1266 and value alpha from 0.4243 to 0.5650; field–value Jaccard increases from 0.0000 to 0.1053, while alpha increases from 0.1784 to 0.5237. Field-only gains are smaller, indicating greater compatibility in extracted values than in the fields used to organize them. Table 3 shows that every exact-versus-semantic improvement remains significant after Holm correction, and Figure 3 shows an increase for every model pair. Figure 3 further shows that every model pair benefits from semantic normalization, indicating that the improvement is not driven by only a small subset of models. Nevertheless, Jaccard similarity remains low when agreement requires support from all six models, so normalization does not eliminate differences in selection and granularity.

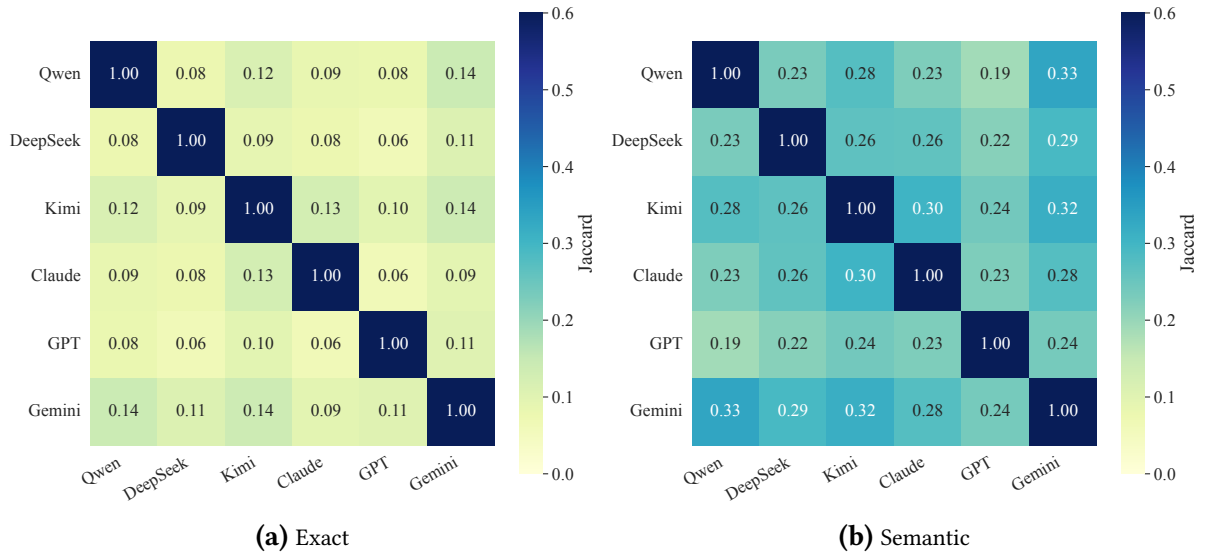


Figure 3: Pairwise agreement between representative LLMs. Heatmaps show median pairwise Jaccard similarity for extracted field-value metadata units across the six representative models. The exact-matching heatmap compares raw field-value strings, while the semantic-matching heatmap first clusters surface-form variants before computing Jaccard similarity. Darker cells indicate stronger agreement between model pairs.

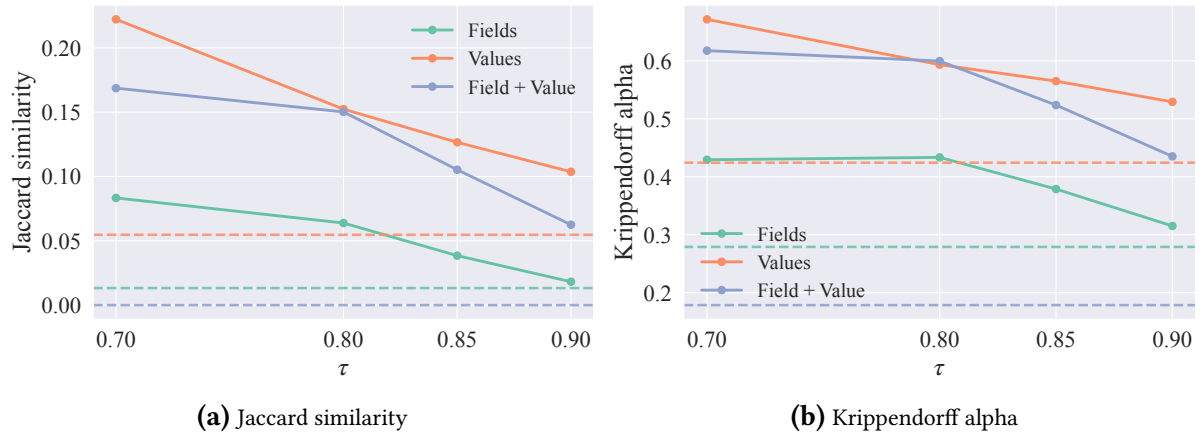


Figure 4: Sensitivity of semantic agreement to the clustering threshold τ . Jaccard similarity and Krippendorff's alpha are computed across the six representative models. Solid lines show semantic scores for fields, values, and field-value pairs; dashed lines show the corresponding exact baselines. Lower τ values permit more aggressive clustering and therefore increase agreement.

The semantic-matching benefit is robust, although its magnitude depends on the similarity threshold. Figure 4 shows that lower thresholds produce more aggressive clustering and higher absolute agreement, especially for values and field-value pairs, while field agreement is comparatively stable. Across all tested thresholds, semantic matching benefits values and field-value pairs more than fields. Semantic agreement should therefore be reported with its embedding model and threshold, but the central content-level finding is not tied to a single threshold choice.

RQ2 Takeaway: Exact matching substantially understates cross-model compatibility, especially for extracted values and complete field-value records. Sentence-embedding-based matching recovers meaningful conceptual overlap, but agreement remains limited since models also differ in metadata selection, granularity, and schema organization.

Table 4

Agreement within the open- and closed-model groups under exact and semantic matching. The open group comprises Qwen 3.7 Flash, DeepSeek V4 Flash, and Kimi K3; the closed group comprises GPT-5.6 Luna Pro, Claude Sonnet 5, and Gemini 3.6 Flash.

Group	Representation	Jaccard Similarity		Krippendorff’s Alpha	
		Exact	Semantic	Exact	Semantic
Open models	Fields	0.0732	0.1277	0.2933	0.4015
	Values	0.1270	0.2474	0.4002	0.5583
	Field–Value	0.0320	0.2121	0.1794	0.5123
Closed models	Fields	0.0566	0.1023	0.2412	0.3427
	Values	0.1578	0.2596	0.4273	0.5658
	Field–Value	0.0305	0.2209	0.1588	0.5204

Table 5

Agreement within and across the Qwen and DeepSeek families. Both metrics are computed for model pairs. Within-family entries give equal weight to the median Qwen–Qwen and DeepSeek–DeepSeek pair scores; cross-family entries are medians over the nine Qwen–DeepSeek pairs. Entries are medians over 197 README files. Semantic matching uses sentence-transformers/all-MiniLM-L6-v2 with $\tau = 0.85$.

Comparison	Representation	Jaccard Similarity		Krippendorff’s Alpha	
		Exact	Semantic	Exact	Semantic
Within-family	Fields	0.2222	0.2971	0.3572	0.4502
	Values	0.3167	0.4534	0.4690	0.6160
	Field–Value	0.1464	0.4148	0.2463	0.5784
Cross-family	Fields	0.1667	0.2381	0.2839	0.3802
	Values	0.2609	0.3889	0.4129	0.5591
	Field–Value	0.1000	0.3478	0.1805	0.5150

4.3.3. RQ3: How Does Agreement Differ Across Model Groups and Families?

Overall agreement can conceal systematic variation associated with model grouping or family. We therefore compare agreement within the Open and Closed groups and within and across model families to determine whether either grouping helps explain extraction behavior.

The Open–Closed grouping explains little of the observed agreement variation. Table 4 computes Jaccard similarity and Krippendorff’s alpha jointly across the three models in each group. Under semantic matching, the Open models show higher agreement for metadata fields (Jaccard: 0.1277 versus 0.1023; alpha: 0.4015 versus 0.3427), whereas the Closed models are slightly higher for values and field–value pairs. After Holm correction, only the field differences remain significant (Table 3); value and field–value differences are not significant.

Within-family agreement exceeds cross-family agreement across all representations. Table 5 uses the same pairwise aggregation for both conditions and balances the Qwen and DeepSeek within-family medians. Under semantic matching, within-family agreement is consistently higher across all three representations, i.e., fields (Jaccard: 0.2971 versus 0.2381; alpha: 0.4502 versus 0.3802), values (0.4534 versus 0.3889; 0.6160 versus 0.5591), and field–value pairs (0.4148 versus 0.3478; 0.5784 versus 0.5150).

The within-family advantage is statistically significant under semantic matching. Table 3 confirms Holm-adjusted significance for both metrics and all three representations. For the evaluated Qwen and DeepSeek variants, family membership therefore provides a clearer grouping than the Open–Closed distinction.

RQ3 Takeaway: The Open–Closed grouping yields significant differences only for fields, whereas Qwen and DeepSeek variants agree more strongly within their respective families than across families for every representation. These results apply to the evaluated models and should not be generalized to all model groups or families.

5. Limitations

Agreement is not accuracy. The reported agreement scores quantify the consistency of metadata extracted by independently prompted models, not its correctness. Models can agree on incorrect or incomplete metadata, and correct outputs can disagree because of differences in selection or granularity. Although the prompt requires verbatim evidence, this study does not independently evaluate extraction correctness or grounding. **The model and document scope is limited.** The main analysis covers six models and 200 research-related README files. The family comparison includes only Qwen and DeepSeek variants. The statistical tests assess consistency across the evaluated README files for the selected models; they do not guarantee generalization beyond this corpus or these models. **Semantic agreement depends on the normalization procedure.** We use one sentence-transformer and connected-component clustering. Threshold sensitivity reduces dependence on a single τ , but another encoder or clustering rule may yield different matches, and transitive links can merge indirectly related strings. **Global-universe alpha is sensitive to sparse units.** The common metadata universe supports a shared-schema interpretation, but many shared absences can affect chance correction. We therefore report alpha together with Jaccard similarity rather than treating either metric as definitive.

6. Conclusion and Future Work

This paper presented an agreement-based methodology for evaluating zero-shot, open-schema metadata extraction from GitHub README files without requiring document-level gold metadata across different representative LLMs. The results show that outputs are almost always well-formed and structurally complete, but differ substantially in extraction cardinality and field-name diversity. While exact agreement is particularly limited for complete field–value records, semantic matching reveals considerably greater conceptual overlap, indicating that exact string matching substantially understates compatibility among LLM-generated metadata.

Agreement differs only modestly between the Open and Closed model groups, whereas agreement is consistently higher within the Qwen and DeepSeek model families than across model families. These findings suggest that inter-model agreement provides a useful diagnostic for assessing the consistency of open-schema metadata extraction, although it should not be interpreted as a measure of extraction correctness. Future work should incorporate manually curated document-level metadata to assess extraction accuracy, verify evidence grounding, investigate alternative semantic normalization methods, and extend the evaluation to additional model families, prompting strategies, and inference configurations.

Acknowledgments

This work has been partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation), NFDI4DS (Grant number 460234259).

Declaration on Generative AI

During the preparation of this work, the author(s) used ChatGPT-5.6 in order to: Grammar and spelling check. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication’s content.

References

- [1] J. Priem, H. Piwowar, R. Orr, OpenAlex: A fully-open index of scholarly works, authors, venues, institutions, and concepts, 2022. URL: <https://arxiv.org/abs/2205.01833>. doi:10.48550/arXiv.2205.01833. arXiv:2205.01833.
- [2] J. Dagdelen, A. Dunn, S. Lee, N. Walker, A. S. Rosen, G. Ceder, K. A. Persson, A. Jain, Structured information extraction from scientific text with large language models, *Nature Communications* 15 (2024) 1418. URL: <https://doi.org/10.1038/s41467-024-45563-x>. doi:10.1038/s41467-024-45563-x.
- [3] Z. Alyafeai, M. S. Al-shaibani, B. Ghanem, MOLE: Metadata extraction and validation in scientific papers using LLMs, in: C. Christodoulopoulos, T. Chakraborty, C. Rose, V. Peng (Eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, Association for Computational Linguistics, Suzhou, China, 2025, pp. 12236–12264. URL: <https://aclanthology.org/2025.findings-emnlp.655/>. doi:10.18653/v1/2025.findings-emnlp.655.
- [4] G. A. Gesese, Z. Chen, S. Jiang, M. A. Tan, Z. Liu, S. Schimmler, H. Sack, NERdME: A named entity recognition dataset for indexing research artifacts in code repositories, in: *Proceedings of the ACM Web Conference 2026, WWW '26*, Association for Computing Machinery, New York, NY, USA, 2026, pp. 8661–8664. URL: <https://doi.org/10.1145/3774904.3792934>. doi:10.1145/3774904.3792934.
- [5] P. Lopez, GROBID: Combining automatic bibliographic data recognition and term extraction for scholarship publications, in: M. Agosti, J. L. Borbinha, S. Kapidakis, C. Papatheodorou, G. Tsakonas (Eds.), *Research and Advanced Technology for Digital Libraries*, volume 5714 of *Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, 2009, pp. 473–474. URL: https://doi.org/10.1007/978-3-642-04346-8_62. doi:10.1007/978-3-642-04346-8_62.
- [6] D. Tkaczyk, P. Szostek, M. Fedoryszak, P. J. Dendek, Ł. Bolikowski, CERMINE: Automatic extraction of structured metadata from scientific literature, *International Journal on Document Analysis and Recognition* 18 (2015) 317–335. URL: <https://doi.org/10.1007/s10032-015-0249-8>. doi:10.1007/s10032-015-0249-8.
- [7] Z. Boukhers, C. Yang, MESD: Metadata extraction from scholarly documents – a shared task overview, in: *Proceedings of the 2nd International Workshop on Natural Scientific Language Processing and Research Knowledge Graphs (NSLP 2025) co-located with the 22nd Extended Semantic Web Conference (ESWC 2025)*, volume 3977 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2025. URL: <https://ceur-ws.org/Vol-3977/NSLP-06.pdf>.
- [8] I. Councill, C. L. Giles, M.-Y. Kan, ParsCit: an open-source CRF reference string parsing package, in: N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, D. Tapias (Eds.), *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, European Language Resources Association (ELRA), Marrakech, Morocco, 2008. URL: <https://aclanthology.org/L08-1291/>.
- [9] F. Peng, A. McCallum, Information extraction from research papers using conditional random fields, *Information processing & management* 42 (2006) 963–979.
- [10] J. P. Chiu, E. Nichols, Named entity recognition with bidirectional lstm-cnns, *Transactions of the association for computational linguistics* 4 (2016) 357–370.
- [11] X. Zhong, J. Tang, A. J. Yepes, Publaynet: largest dataset ever for document layout analysis, in: *2019 International conference on document analysis and recognition (ICDAR)*, IEEE, 2019, pp. 1015–1022.
- [12] A. Morejón, B. Navarro-Colorado, C. García-Barceló, A. Berenguer, D. Tomás, J.-N. Mazón, Automatic metadata extraction leveraging large language models in digital humanities, *Electronics* 14 (2025) 4962.
- [13] M. B. Jones, C. Boettiger, A. Cabunoc Mayes, A. Smith, P. Slaughter, K. Niemeyer, Y. Gil, M. Fenner, K. Nowak, M. Hahnel, et al., Codemeta: an exchange schema for software metadata. version 2.0, *KNB data repository* (2017).
- [14] A. Mao, D. Garijo, S. Fakhraei, Somef: A framework for capturing scientific software metadata from its documentation, in: *2019 IEEE International Conference on Big Data (Big Data)*, IEEE,

- 2019, pp. 3032–3037.
- [15] O. Etzioni, M. Banko, S. Soderland, D. S. Weld, Open information extraction from the web, *Communications of the ACM* 51 (2008) 68–74.
 - [16] A. Fader, S. Soderland, O. Etzioni, Identifying relations for open information extraction, in: *Proceedings of the 2011 conference on empirical methods in natural language processing*, 2011, pp. 1535–1545.
 - [17] M. Schmitz, S. Soderland, R. Bart, O. Etzioni, et al., Open language learning for information extraction, in: *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, 2012, pp. 523–534.
 - [18] K. Krippendorff, *Content Analysis: An Introduction to Its Methodology*, 4 ed., SAGE Publications, 2018.
 - [19] R. Artstein, M. Poesio, Survey article: Inter-coder agreement for computational linguistics, *Computational linguistics* 34 (2008) 555–596.
 - [20] F. Gilardi, M. Alizadeh, M. Kubli, Chatgpt outperforms crowd workers for text-annotation tasks, *Proceedings of the National Academy of Sciences* 120 (2023) e2305016120.
 - [21] T. Ahmed, P. Devanbu, C. Treude, M. Pradel, Can llms replace manual annotation of software engineering artifacts?, in: *2025 IEEE/ACM 22nd International Conference on Mining Software Repositories (MSR)*, IEEE, 2025, pp. 526–538.
 - [22] N. Reimers, I. Gurevych, Sentence-BERT: Sentence embeddings using siamese BERT-networks, in: K. Inui, J. Jiang, V. Ng, X. Wan (Eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Association for Computational Linguistics, Hong Kong, China, 2019, pp. 3982–3992. URL: <https://aclanthology.org/D19-1410/>. doi:10.18653/v1/D19-1410.
 - [23] R. Dror, G. Baumer, S. Shlomov, R. Reichart, The hitchhiker’s guide to testing statistical significance in natural language processing, in: I. Gurevych, Y. Miyao (Eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 1383–1392. URL: <https://aclanthology.org/P18-1128/>. doi:10.18653/v1/P18-1128.
 - [24] S. Holm, A simple sequentially rejective multiple test procedure, *Scandinavian Journal of Statistics* 6 (1979) 65–70. URL: <https://www.jstor.org/stable/4615733>.
 - [25] OpenRouter, Openrouter api documentation, 2026. URL: <https://openrouter.ai/docs/quickstart>.
 - [26] Qwen Team, Qwen3.7-Flash model documentation, QwenCloud, 2026. URL: <https://www.qwencloud.com/models/qwen3.7-flash>.
 - [27] DeepSeek-AI, DeepSeek-V4 preview release, 2026. URL: <https://api-docs.deepseek.com/news/news260424/>.
 - [28] Kimi Team, Kimi K3: Open frontier intelligence (2026). URL: <https://arxiv.org/abs/2607.24653>. doi:10.48550/arXiv.2607.24653. arXiv:2607.24653.
 - [29] Anthropic, Claude Sonnet 5 system card, 2026. URL: <https://www-cdn.anthropic.com/73ad94ca3c0502e75e46637cc62c8bd9532a7f2c/Claude%20Sonnet%205%20System%20Card.pdf>.
 - [30] OpenAI, GPT-5.6: Frontier intelligence that scales with your ambition, 2026. URL: <https://openai.com/index/gpt-5-6/>.
 - [31] OpenRouter, GPT-5.6 Luna Pro model route, 2026. URL: <https://openrouter.ai/openai/gpt-5.6-luna-pro>.
 - [32] Google DeepMind, Gemini 3.6 Flash model card, 2026. URL: <https://deepmind.google/models/model-cards/gemini-3-6-flash/>.
 - [33] Sentence Transformers, all-MiniLM-L6-v2 model card, Hugging Face, 2021. URL: <https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2>.

A. Metadata Extraction Prompt

The following prompt was used to instruct each model in the open-schema metadata extraction task.

A.1. Role and Objective

You are a metadata extraction system for artifacts documented on GitHub. Your task is to read the provided artifact document and extract useful metadata that would help someone understand, index, search, reproduce, or reuse the artifact.

A.2. Input and Evidence Grounding

The artifact document is a GitHub README file in Markdown format. The input is shown with line numbers for reference only. Use the line numbers to populate `evidence_location`, but do not include line number prefixes in the evidence field. Use the README structure (headers, code blocks, badges, lists) to help you locate and delimit evidence spans and populate `evidence_location`—for example, citing a Markdown header as the location, or treating a fenced code block as a single coherent span. This is a structural note only: it does not imply a fixed list of expected fields, and you should still identify whatever metadata is available in this specific document, not a generic README template.

Use only the information explicitly stated in the document. Do not use outside knowledge. Do not infer facts that are not directly supported by the text.

A.3. Open-Schema Field Selection

You may decide by yourself which metadata fields are relevant and how to name them, based entirely on what this specific document contains. Do not restrict yourself to any fixed list of field types, and do not assume the artifact belongs to any particular kind of project (e.g. software library, machine learning model, dataset, research paper, hardware project, documentation site, etc.) unless the document itself establishes that. Name each field using whatever term best describes what you found. Prefer short, snake_case field names (e.g. `programming_language`, not “the language used”).

Prioritize metadata that helps a user identify, understand, install, run, reproduce, evaluate, cite, or reuse the artifact.

High-priority metadata includes:

- **artifact identity:** name, purpose, description
- **access links:** repository, paper, project page, model, dataset, demo
- **reproducibility:** installation, dependencies, environment, hardware, checkpoints, commands
- **usage:** training, inference, evaluation, configuration
- **scientific context:** task, method, benchmark, metrics, results, citation
- **legal/reuse information:** license, terms, attribution

Avoid extracting low-value metadata such as generic section headings, vague benefits, repeated examples, marketing claims without concrete values, or ordinary prose that does not name a reusable fact. Extract commands only when they define installation, setup, training, evaluation, inference, reproduction, or deployment steps. Do not extract duplicate or near-duplicate commands.

A.4. Required Metadata Attributes

For each metadata item, return:

field: a short snake_case name for the metadata type you identified

value: the extracted value, as close to verbatim as possible

evidence: a text span copied VERBATIM (character-for-character) from the artifact text, including original Markdown syntax (e.g. keep “## Installation” as-is, keep code fence contents as-is). Do not include the input line number prefixes. Do not paraphrase, reflow, merge non-adjacent sentences, or fix typos in the evidence field. This field will be programmatically checked as a substring of the source document after JSON string decoding, so JSON escaping of newlines and quotes is acceptable.

evidence_location: line number or line range from the numbered input, optionally plus Markdown section/header title; use “unknown” only if not available

confidence: one of “high”, “medium”, “low”, using these criteria:

- **high:** the value is named explicitly and unambiguously, with no plausible alternative reading
- **medium:** the value is supported but requires minor interpretation (e.g. resolving an acronym used elsewhere in the doc, or a value split across adjacent sentences)
- **low:** the text is genuinely ambiguous between two or more readings, or the evidence is indirect/implicit

reason: a short explanation of why this value is useful metadata

A.5. Important Rules

1. Extract only metadata supported by evidence in the artifact text.
2. If a possible metadata value is ambiguous, include it with confidence “low” and explain the ambiguity in “reason”.
3. Do not output generic concepts unless the document names a concrete value.
4. Do not invent missing metadata.
5. If the same underlying fact could be tagged with more than one field name, choose the single field name you judge most precise, and do not emit duplicate items for the same evidence span under multiple field names.
6. If no useful metadata is found, return an empty list.
7. Output ONLY the JSON object below. No markdown code fences, no preamble, no explanation outside the JSON structure.

Return only valid JSON in the following format:

```
{
  "metadata": [
    {
      "field": "...",
      "value": "...",
      "evidence": "...",
      "evidence_location": "...",
      "confidence": "high | medium | low",
      "reason": "..."
    }
  ]
}
```

Input text: {input_text}