Precision-Oriented Japanese Document-Level Relation Extraction with Two-Stage Structured Verification and Type Constraints

Anonymous authors
Paper under double-blind review

Abstract

Japanese document-level relation extraction (DocRE) is a practical route to building knowledge graphs from long-form text, but low-cost large language models (LLMs) used in a single one-shot extraction step often produce many false positives, harming precision and downstream usability. This setting is difficult because documents induce a quadratic space of entity pairs, relations may require cross-sentence evidence, and generative LLM outputs are prone to format errors and hallucinated links. We propose a low-cost Two-Stage pipeline that separates recall-oriented candidate generation from precision-oriented verification using JSON Schema–constrained structured outputs, followed by deterministic filtering with relation-specific domain/range constraints learned from training data. We evaluate on 10 JacRED development documents sampled by character-length stratification and compare against a one-shot baseline across Gemini Flash family configurations (2.0/2.5/3 preview; with and without "thinking"). Logged micro-averaged results show consistent false-positive reductions and precision gains for the proposed pipeline, improving F1 in several configurations (best 0.27), while revealing that recall remains the primary bottleneck. These findings suggest that structured verification and data-driven type constraints can make LLM-based Japanese DocRE more reliable even under tight cost constraints.

1 Introduction

Document-level relation extraction (DocRE) aims to identify directed semantic relations between entity pairs using evidence that may span multiple sentences within a document **??**. DocRE is a foundational capability for knowledge graph construction from text, where extracted triples of the form (head entity, relation, tail entity) can populate and update structured databases used in search, analytics, and question answering. While supervised neural DocRE has progressed rapidly, it typically requires training and domain-matched labeled corpora, and it can be computationally demanding due to the large candidate space induced by all within-document entity pairs **?**.

Japanese DocRE presents additional challenges. Recent work introduced JacRED, a Japanese Wikipedia-based DocRE benchmark with evidence sentences, motivated by the limited transferability of English DocRE resources to Japanese and by Japanese-specific discourse phenomena **?**. In parallel, practitioners increasingly attempt to use large language models (LLMs) for information extraction in low-resource or rapid-deployment settings, often via prompting or in-context learning **?**. However, for DocRE in particular, one-shot generative extraction can be brittle: LLMs may hallucinate relations, confuse argument direction, or over-generate plausible but unsupported links, leading to low precision.

This paper studies an explicitly cost-constrained setting: using low-cost LLM configurations to extract relation triples from Japanese documents. Our working hypothesis is that one-shot extraction with such models yields a large number of false positives, producing low precision and limiting knowledge graph usability. This is hard for three intertwined reasons. First, DocRE requires global reading: many relations are expressed across sentences and depend on discourse and coreference **??**. Second, the search space is large: if a document contains E entities, there are $E(E-1)$ ordered pairs to consider, and multi-label relations may apply to a pair. Third, generative models must be controlled not only for semantic correctness but also for structural correctness of outputs; unconstrained decoding can yield malformed JSON or ambiguous text lists that complicate evaluation and post-processing. Recent studies show that constrained decoding to JSON schemas can improve compliance and sometimes downstream accuracy, but coverage and behavior vary by engine and setup **?**.

We propose a Two-Stage extraction pipeline designed to reduce false positives while staying compatible with low-cost LLM operation. Stage 1 is recall-oriented candidate generation: the LLM proposes entity and relation candidates, emitted as schema-valid JSON via structured outputs. Stage 2 is precision-oriented verification: the LLM is asked to verify each proposed candidate triple, again using schema-constrained decoding, and verification is performed in batches to control cost. Finally, we apply deterministic post-processing using relation-specific domain/range constraints (type-pair constraints) derived empirically from the training data. Intuitively, this third step serves as a lightweight consistency check that removes triples whose entity type combinations were never observed for that relation in training.

We validate the approach on JacRED dev data **?**. To ensure we only report results that were actually run, we restrict evaluation to 10 development documents selected via character-length–based stratified sampling. We compare a Baseline one-shot extraction against the Proposed Two-Stage pipeline across several Gemini Flash family configurations: Gemini 2.0 Flash, Gemini 2.5 Flash, and Gemini 3 Flash Preview, with and without the model's "thinking" mode. Evaluation uses micro-averaged Precision, Recall, and F1 over the 10 documents.

The logged results show that the Proposed pipeline consistently reduces false positives across all tested configurations, yielding clear precision gains. The best observed F1 is 0.27, achieved by Gemini 2.5 Flash with thinking budget 2048 and by Gemini 3 Flash Preview without thinking (also 0.27), while the baseline typically ranges from 0.17 to 0.26. Despite these gains, recall remains low (0.12–0.22) for both baseline and proposed methods, indicating that candidate generation and/or entity alignment is the dominant bottleneck.

Contributions: - We formulate a cost-oriented Two-Stage LLM pipeline for Japanese DocRE that separates candidate generation from verification using JSON Schema–constrained structured outputs, improving output reliability and enabling deterministic post-processing. - We introduce a simple, data-driven domain/range (head_type, tail_type) constraint filter learned from training data and apply it as a final precision guardrail. - We provide an empirical comparison on 10 stratified JacRED dev documents across multiple Gemini Flash family configurations, reporting micro-averaged Precision/Recall/F1 with full TP/FP/FN counts from logs. - We analyze failure patterns and highlight recall as the key limitation, motivating future work on entity alignment robustness and capturing implicit or multi-hop relations.

Future work should focus on improving recall without giving up the precision gains. Promising directions include better prompts or decompositions for candidate generation, more robust entity canonicalization and directionality handling, and verification strategies that avoid over-regularization when "thinking" is enabled in the verifier. More broadly, integrating explicit evidence selection, a recurring theme in DocRE research, may help connect extracted triples to supporting sentences for higher faithfulness **??**.

2 Related Work

Our work sits at the intersection of document-level relation extraction, LLM-based generative information extraction, and constrained decoding for structured outputs.

DocRE benchmarks and modeling. DocRED established DocRE as a distinct task by showing that many Wikipedia-derived facts require multi-sentence evidence and by providing entity/coreference annotations and evidence sentences, along with evaluation practices such as micro-F1 and overlap-robust "Ign" variants **?**. Subsequent work summarized in the DocRE survey highlights recurring challenges that also manifest in our setting: quadratic entity-pair scaling, multi-hop reasoning, and pipeline sensitivity to upstream entity processing **?**. Compared with supervised DocRE architectures (graph-based reasoning, transformer variants with localized pooling, joint evidence extraction), our approach does not train a task model and instead targets low-cost LLM prompting; therefore it addresses a different point in the accuracy–cost–engineering trade space. In particular, many surveyed methods assume availability of a trainable encoder and task-specific optimization, whereas our method relies only on inference-time decomposition and deterministic filtering.

Japanese DocRE. JacRED is the first public Japanese Wikipedia-based DocRE benchmark with evidence sentences and a relation schema derived from Wikipedia and Wikidata, created via a semi-automatic workflow **?**. The JacRED paper reports that LLM in-context learning performs poorly relative to supervised models, motivating reliability-focused pipelines when using LLMs. Our work differs by targeting low-cost LLM configurations and by focusing on precision failures of one-shot extraction, proposing verification and type constraints as mitigation. While JacRED's construction uses model recommendations to reduce annotation labor, our pipeline instead uses a verifier model call to reduce false positives at inference.

Generative IE with LLMs. Surveys of LLM-based generative IE emphasize that prompt-based extraction can be sensitive and that structured or "universal" extraction formats often help in strict settings such as relation extraction with rigid output requirements **?**. A key practical issue is that unconstrained generative decoding can produce malformed structures. Our pipeline adopts schema-constrained structured outputs for both extraction and verification, aligning with the survey's emphasis on constrained decoding as an enabling technique for reliable IE.

Structured and constrained decoding. Structured-output systems that constrain generation to JSON schemas or grammars have been benchmarked for compliance and coverage, with evidence that constrained decoding can improve accuracy and sometimes efficiency, though implementation details matter **?**. Our use of JSON Schema–constrained decoding is motivated primarily by robustness and determinism in downstream processing. Unlike general benchmarking work that studies a wide variety of schemas and engines, we focus on a concrete DocRE pipeline and use structured decoding as a building block to separate candidate generation from verification.

Joint extraction as an alternative. End-to-end joint NER and RE models using pre-trained transformers reduce pipeline error propagation by training a single model that predicts entities and relations jointly **?**. However, that line of work is primarily sentence-level and requires supervised training; it is not directly applicable to our cost-constrained, inference-only LLM setting on document-level relations. Nevertheless, it provides a useful contrast: where joint models improve precision/recall via learned coupling, we instead impose coupling via an explicit verification stage and type constraints.

In summary, our method is most directly comparable to prompt-based, generative extraction baselines in low-resource settings. Approaches that require supervised training or specialized document encoders are complementary but not directly substitutable under our constraints. Consequently, our experimental section focuses on comparing one-shot LLM extraction to our Two-Stage verification and filtering pipeline under identical model families and evaluation scripts.

3 Background

Document-level relation extraction (DocRE) can be described as follows. Given a document D consisting of a sequence of sentences, and a set of entity mentions grouped into canonical entities, the goal is to predict a set of directed labeled relations between entity pairs. Each predicted relation can be represented as a triple (h, r, t), where h is the head entity, t is the tail entity, and r is a relation label from a fixed schema. DocRE differs from sentence-level RE because the evidence supporting a relation may span multiple sentences, and coreference or discourse can be required to connect mentions to the correct entities **??**.

Problem setting and notation. Let $E(D)$ be the set of entities in a document D, and R be the set of relation labels. A system outputs a set of predicted triples $\hat{T}(D) \subseteq E(D) \times R \times E(D)$. Gold annotations provide $T(D)$. Evaluation in this work uses micro-averaged precision, recall, and F1 over the union of triples across a small set of documents. If TP is the number of correctly predicted triples, FP the number of predicted but incorrect triples, and FN the number of missing gold triples, then precision $= \text{TP}/(\text{TP} + \text{FP})$, recall $= \text{TP}/(\text{TP} + \text{FN})$, and F1 $= 2PR/(P + R)$. Because relations are directed, swapping head and tail constitutes a different triple and is counted as incorrect unless both directions are annotated.

Generative LLM extraction and structural control. In a prompt-based extraction approach, an LLM is asked to produce $\hat{T}(D)$ directly as text. This creates two practical issues. First, semantic reliability: models may output plausible but unsupported relations (false positives) or miss implicit relations (false negatives), especially when documents are long and the model must aggregate information. Second, structural reliability: outputs may violate expected formats, complicating parsing and automated evaluation. Constrained decoding and "structured outputs" address the latter by forcing the model to emit JSON conforming to a schema, improving compliance and enabling deterministic downstream steps **?**. In this paper, structured outputs are used not to change the underlying semantic reasoning of the model directly, but to make the pipeline modular and auditable.

Data-driven domain and range constraints. Many relation schemas implicitly constrain the types of entities that can appear as head and tail (domain/range). For example, a relation might only occur between a person and an organization. In our setting, entity types are those provided by JacRED (eight IREX-style types) **?**. We derive relation-specific allowable type pairs (head_type, tail_type) from the training data by collecting the set of observed type pairs for each relation. At inference, any predicted triple whose (head_type, tail_type) is not in the allowed set for its relation is deterministically filtered out. This strategy assumes that the training data provides a reasonably complete coverage of valid type combinations for frequent relations; it is intended as a precision-oriented guardrail and may reduce recall if the training set is incomplete.

Evaluation dataset context. JacRED provides Japanese Wikipedia documents annotated with relations and evidence sentences, designed to address limitations of cross-lingual transfer from English DocRE datasets and to support Japanese-native DocRE research **?**. While prior work reports strong supervised baselines on the full JacRED benchmark, our experiments intentionally use a small, stratified subset of the development set to reflect a lightweight evaluation regime suited to rapid iteration under API-based LLM usage constraints.

4 Method

We propose a Two-Stage pipeline for Japanese DocRE that decomposes extraction into candidate generation, verification, and deterministic constraint filtering. The key design goal is to reduce false positives from one-shot extraction while preserving compatibility with low-cost LLM inference.

Stage 1: recall-oriented candidate generation with structured outputs. Given a document D and its entity inventory (as provided by the dataset input to the pipeline), Stage 1 prompts an LLM to propose a set of candidate relation triples. The prompt is designed to be permissive to favor recall: the model is encouraged to list plausible triples rather than to be conservative. Crucially, the output is constrained to a predefined JSON schema (EXTRACTION_SCHEMA) using structured outputs so that each candidate is returned in a machine-parseable format. This eliminates ambiguity in downstream parsing and allows consistent handling across model versions. The output of Stage 1 is a multiset $C(D)$ of candidates; candidates may include duplicates or near-duplicates depending on the model behavior.

Stage 2: precision-oriented verification in batches. Stage 2 takes the candidate set $C(D)$ and verifies each candidate triple using a second LLM call that is explicitly calibrated toward precision. For each candidate $c = (h, r, t)$, the verifier is asked whether the relation is supported by the document. Verification is performed in batches of size 10 to control overhead and API cost. The verifier outputs a schema-conforming JSON object (VERIFICATION_SCHEMA) indicating, for each candidate, whether it should be accepted. The verified set is $V(D) = \{c \in C(D) : \text{verifier\_accept}(c) = \text{true}\}$.

Deterministic post-processing with type constraints. Even after verification, some candidates may be semantically plausible but structurally inconsistent with the relation schema or dataset conventions. We therefore apply a deterministic filter based on relation-specific domain/range constraints. For each relation r, we precompute an allowed set $A(r)$ of (head_type, tail_type) pairs observed in the training data. Each candidate $c = (h, r, t)$ is associated with entity types $\mathrm{type}(h)$ and $\mathrm{type}(t)$. The final output is:

$$\hat{T}(D) = \{(h, r, t) \in V(D) : (\mathrm{type}(h), \mathrm{type}(t)) \in A(r)\}.$$

This step is intentionally simple and transparent: it does not require model calls and can be audited and adjusted per relation.

Discussion of expected effects. The pipeline targets precision improvement via two mechanisms. The verifier reduces hallucinated or weakly supported relations by requiring explicit support in the document. The type constraint filter removes a class of systematic errors where the model assigns a relation between incompatible entity types. However, both mechanisms can reduce recall: the verifier may be overly conservative (especially when “thinking” is enabled, as observed in our analysis), and the type constraints may reject valid but unseen type combinations due to training set sparsity.

Implementation notes. The method is implemented with the Google GenAI SDK and Gemini API, leveraging structured outputs for both stages. Verification is performed with batch size 10. No model fine-tuning is performed; all improvements come from decomposition, constrained decoding, and deterministic filtering.

5 Experimental Setup

Dataset and sampling. We use the JacRED dataset development set **?**. To keep the evaluation lightweight and to ensure results reflect actual runs, we select 10 documents from the JacRED dev set using character-length–based stratified sampling. The goal is to cover a range of document lengths while limiting API calls.

Task and evaluation. The task is document-level relation extraction: for each document, predict a set of directed relation triples between entities. We evaluate using micro-averaged precision, recall, and F1 over the 10-document set. We also report TP, FP, and FN counts aggregated across the documents, where TP is the number of predicted triples matching the gold set exactly, FP are predicted triples not in gold, and FN are gold triples not predicted.

Methods compared. (1) Baseline: One-shot extraction. A single LLM call extracts entities and relations simultaneously in one step, followed by simple filtering. This baseline represents the common prompting-based approach where the model outputs the entire set of triples directly. (2) Proposed: Two-Stage KG Extraction. Stage 1 generates candidates with recall orientation and structured outputs (EXTRACTION_SCHEMA). Stage 2 verifies candidates with precision orientation and structured outputs (VERIFICATION_SCHEMA), performing batch verification with batch size 10. The final output is filtered with relation-specific (head_type, tail_type) constraints derived from the training data.

Models and configurations. We evaluate across multiple Gemini Flash family models: Gemini 2.0 Flash, Gemini 2.5 Flash, and Gemini 3 Flash Preview. For Gemini 2.5 Flash and Gemini 3 Flash Preview, we test configurations with “thinking” off and with thinking budget 2048, and for Gemini 2.0 Flash we test thinking set to none. These configurations reflect the cost–performance variants available within the family.

Implementation details. The extraction and verification stages are implemented using a Gemini API–based pipeline with the Google GenAI SDK. Structured outputs are enforced via JSON Schema–constrained decoding for both extraction and verification. Verification requests are batched (size 10). After verification, deterministic filtering applies empirically observed type-pair constraints per relation, derived from the training data. No hardware assumptions are required beyond a Python runtime capable of making Gemini API calls with an API key.

Fairness considerations. For each model configuration, we compare Baseline and Proposed under the same document subset and evaluation script. The primary difference is the pipeline decomposition and filtering steps. Because the Proposed method includes additional LLM calls (verification), its cost is higher than one-shot extraction; our study focuses on accuracy outcomes under low-cost model choices rather than wall-clock or dollar cost measurements.

6 Results

We report micro-averaged precision, recall, and F1 over the 10 sampled JacRED dev documents for each model configuration, comparing Baseline (one-shot) against Proposed (Two-Stage with verification and type constraints). We only present results that were logged.

Overall trend: precision gains via false-positive reduction. Across all configurations, the Proposed pipeline reduces FP substantially relative to Baseline, which directly increases precision. This is consistent with our hypothesis that one-shot extraction with low-cost LLMs produces many false positives. However, recall remains low for both methods, indicating that missing gold relations (FN) dominate and that better candidate generation and entity alignment are needed.

Experiment 1: Gemini 2.0 Flash (thinking: none). Baseline achieves precision 0.20, recall 0.15, F1 0.17 with TP=22, FP=86, FN=126. Proposed improves precision to 0.35 and recall to 0.19, yielding F1 0.25 with TP=28, FP=51, FN=121. The primary change is a reduction of FP by 35 while also increasing TP by 6.

Experiment 2: Gemini 2.5 Flash (thinking: off). Baseline achieves precision 0.17, recall 0.16, F1 0.17 with TP=24, FP=115, FN=124. Proposed increases precision to 0.30 but recall drops to 0.12, leaving F1 unchanged at 0.17 with TP=18, FP=42, FN=130. Here verification and constraints strongly reduce FP (-73) but also reduce TP (-6), suggesting over-conservatism in the verification stage under this configuration.

Experiment 3: Gemini 2.5 Flash (thinking: 2048). Baseline achieves precision 0.18, recall 0.17, F1 0.17 with TP=25, FP=115, FN=123. Proposed achieves precision 0.36, recall 0.21, F1 0.27 with TP=31, FP=54, FN=117. Compared to Baseline, the Proposed pipeline both reduces FP (-61) and increases TP (+6), producing the best observed F1 in our runs.

Experiment 4: Gemini 3 Flash Preview (thinking: off). Baseline achieves precision 0.26, recall 0.16, F1 0.20 with TP=24, FP=70, FN=124. Proposed improves to precision 0.36, recall 0.22, F1 0.27 with TP=32, FP=56, FN=116. This configuration provides a favorable cost–performance trade-off in our analysis report because it reaches F1 0.27 without enabling thinking.

Experiment 5: Gemini 3 Flash Preview (thinking: 2048). Baseline achieves precision 0.31, recall 0.22, F1 0.26 with TP=33, FP=74, FN=115. Proposed achieves precision 0.37, recall 0.20, F1 0.26 with TP=30, FP=52, FN=118. Proposed reduces FP (-22) but also reduces TP (-3), leaving F1 unchanged. This aligns with the observation that enabling thinking improves Baseline performance but can over-regularize verification in the Proposed pipeline.

Limitations and failure analysis. First, recall is uniformly low (0.12–0.22), even when precision improves. This indicates that many gold relations are not being proposed in Stage 1 or are being lost due to entity alignment or directionality mismatches. Second, the interaction between thinking mode and verification appears nontrivial: while thinking improves Baseline in Gemini 3 (F1 0.26 vs 0.20), it does not improve Proposed (0.26 vs 0.27) and in some cases reduces TP. Third, our evaluation is small (10 documents), chosen for stratified coverage but not for statistical estimation; we do not report confidence intervals because no per-document variance logs are provided.

Figures. No figures were produced or provided in the experiment logs; therefore, we include no figures in this section.

All metrics reported above are from the saved logs and reflect micro-averaged evaluation over the sampled documents.

7 Conclusion

This paper examined a cost-constrained setting for Japanese document-level relation extraction on JacRED, where low-cost LLM one-shot extraction produces many false positives and thus low precision. We proposed a Two-Stage pipeline that (i) generates relation candidates with recall orientation using JSON Schema–constrained structured outputs, (ii) verifies candidates in a second, precision-oriented stage with batch verification, and (iii) applies deterministic domain/range filtering via relation-specific (head_type, tail_type) constraints derived from training data.

Across Gemini Flash family configurations, logged results on 10 stratified JacRED dev documents show that the proposed pipeline consistently reduces false positives and yields clear precision improvements. F1 improves notably in several settings, reaching 0.27 for Gemini 2.5 Flash with thinking budget 2048 and for Gemini 3 Flash Preview without thinking. At the same time, recall remains low overall, and in some configurations verification appears over-conservative, reducing true positives.

The main implication is that practical reliability improvements for LLM-based knowledge graph extraction can be achieved without model fine-tuning by combining structured decoding, explicit verification, and lightweight data-driven constraints. Future work should target recall improvements through better candidate generation, more robust entity alignment and relation directionality handling, and verification designs that preserve true positives even under different inference modes.

This work was generated by AIRAS (Tanaka et al., 2025).

References

Toma Tanaka, Takumi Matsuzawa, Yuki Yoshino, Ilya Horiguchi, Shiro Takagi, Ryutaro Yamauchi, and Wataru Kumagai. AIRAS, 2025. URL https://github.com/airas-org/airas.