Extracting Article-Level Legal Dependencies from Swiss Federal Law using LLMs

Steven Cho[1] **Anna Kiepura**[2] **Jessica Lam**[2] **Richard Hahnloser**[2]
[1]ETH Zurich
[2]Institute of Neuroinformatics, University of Zurich and ETH Zurich, Switzerland
stecho@student.ethz.ch
{akiepura, jlam, richardh}@ini.ethz.ch

Abstract

Understanding dependencies between legal provisions is essential for analyzing statutory corpora; yet, such relationships are rarely available in machine-readable form. We present a hybrid pipeline for extracting article-level dependencies from Swiss federal legislation on Fedlex, combining deterministic XML preprocessing with large language model (LLM)–based semantic resolution. Additionally, we release three complementary data splits—document-level JSON, structured citation candidates, and LLM-based article assignments—to support downstream legal NLP research. We evaluate our approach on 2,103 SR documents, yielding over 63,000 citation instances. While LLMs are effective at resolving semantically complex references, we observe substantial limitations in structured output reliability: approximately 21% of generated items violate the expected schema, with most errors being unrecoverable. Our findings highlight a key challenge in applying LLMs to structured legal information extraction and provide a new resource for tasks such as legal knowledge graph construction, citation analysis, and benchmarking structured prediction in the legal domain.

1 Introduction

Understanding relationships between legal provisions is essential for maintaining a coherent body of law. Legal texts frequently reference other statutes and articles, forming a dense dependency network across the legal corpus (Kjær, 2000). Extracting these dependencies enables applications such as legal retrieval, impact analysis, and knowledge graph construction. However, this task is challenging because citations appear in diverse linguistic forms and are rarely explicitly encoded.

In Switzerland, the official platform for federal legislation, Fedlex[1], provides access to the *Systematische Rechtssammlung (SR)*. Although documents are available in XML, article-level cross-references are not machine-readable and must be reconstructed from fragmented signals in text, footnotes, and hyperlinks.

To address this, we release three complementary data splits: (i) document-level JSON representations of SR documents, (ii) structured citation candidates for LLM processing, and (iii) LLM-based article assignments linking citing and cited provisions.

We evaluate our pipeline on 2,103 SR documents, yielding over 63,000 citation items. While LLMs can resolve semantically complex references, structured output reliability remains a key bottleneck: about 21% of outputs violate the expected schema.

Overall, we make the following contributions:

- We introduce and release three JSON data splits for Swiss federal legislation, covering article-level document content, LLM-ready citation candidates, and LLM-based article assignments.
- We propose a hybrid deterministic–LLM pipeline for extracting article-level legal dependencies from Fedlex.
- We present a large-scale empirical analysis showing that, while LLMs can resolve many semantically ambiguous legal references, structured output reliability remains a key bottleneck.

2 Related Work

Swiss legal corpora and resources. Recent work has introduced datasets for Swiss legal NLP, including multilingual corpora of legislation (Felici, 2025) and judicial decisions (Rolshoven et al., 2025). In particular, CHEU-lex provides a parallel corpus of Swiss and EU legislation, focusing on multilingual alignment and linguistic analysis.

[1]https://www.fedlex.admin.ch/

However, these resources do not model article-level dependencies or provide structured representations of cross-references within Fedlex. Earlier work on Swiss legal corpora (Höfler and Piotrowski, 2011) predates Fedlex and relies on outdated collections, limiting its applicability to modern legal data.

Structured information extraction with LLMs. Recent work highlights both the potential and limitations of LLMs for structured output generation. SLOT (Wang et al., 2025) shows that high schema fidelity can be achieved through large-scale synthetic training and fine-tuning, while Structured RAG (Lin et al., 2025) demonstrates that zero-shot prompting over complex inputs leads to unstable and often invalid structured outputs. These findings are particularly relevant for legal documents, which exhibit deep nesting and heterogeneous structures.

Positioning of this work. We extract article-level dependencies from Swiss federal legislation and release a structured dataset derived from Fedlex XML. Unlike prior work, our approach combines deterministic preprocessing with LLM-based resolution without relying on fine-tuning, enabling machine-readable legal dependency structures not available in existing resources.

3 Fedlex Dataset

3.1 Source Data: Swiss Federal Law on Fedlex

Fedlex is the official platform of the Swiss Confederation for publishing federal legislation. It provides access to legally binding documents, including laws, ordinances, and treaties, in multiple formats (XML, HTML, PDF) and languages. In this work, we focus on *Systematische Rechtssammlung (SR)*, which organizes federal law into a structured, topic-based collection.

Within SR, legal documents are organized hierarchically by subject domain and document identifier (e.g., SR 141.0). Each document represents a complete legal act and is internally structured into articles, which form the smallest unit of legal normativity (Kelsen, 1967). Because articles contain the operative legal content, recovering relationships at the article level is essential for a meaningful analysis of legal dependencies.

We restrict our study to the *Landesrecht* section of SR in German. We collected 2,103 SR XML documents by issuing a SPARQL query through the Fedlex interface and subsequently using a custom XML scraper to bulk download the corresponding files.

3.2 Dataset Construction and Data Splits

To enable both citation extraction and downstream NLP applications, we construct and release three complementary JSON data splits derived from the Fedlex corpus:

- **Document-level JSON (SR content)**. Each SR XML document is converted into a structured JSON file containing article-level content. This representation preserves document hierarchy, article boundaries, and metadata, making the data directly usable for NLP tasks.
- **LLM input (citation candidates)**. We extract citation-relevant context from the XML and represent it as structured JSON items. Each item contains a citing article, a target SR reference, and a local text snippet capturing the citation context. These inputs are designed to provide sufficient semantic and structural information for LLM-based citation disambiguation.
- **LLM output (article assignments)**. For each input item, an LLM assigns the corresponding cited article(s) within the target SR document or identifies document-level references when no article is specified. The outputs form a machine-readable representation of article-level legal dependencies.

3.3 Challenges to Article-Level Dependency Extraction

Although Fedlex provides structured XML, article-level dependencies are not explicitly encoded. Relevant information is instead distributed across article text, hyperlinks, and footnotes, requiring a combination of structural parsing and semantic interpretation.

Legal citation patterns are diverse and irregular. References may appear in implicit or non-standard forms, including acronym-based citations, preamble-defined references, intra-document mentions, missing article numbers, or cases where article numbers and SR identifiers are separated. Multiple reference types may also co-occur, creating ambiguity that cannot be resolved with simple heuristics.

[1]We release all code and data to facilitate future research: https://github.com/stevencho24/Extracting-Article-Level-Legal-Dependencies-from-Swiss-Federal-Law-using-LLMs

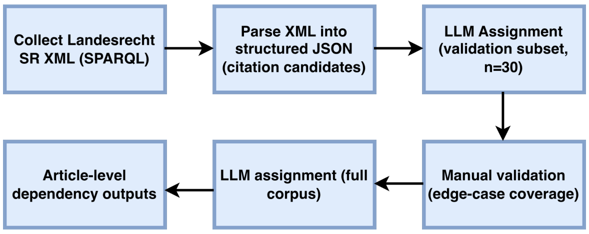

Figure 1: Pipeline for extracting article-level dependencies from Fedlex SR XML. A subset of documents is first processed and manually validated to ensure coverage of citation edge cases, after which the pipeline is applied to the full corpus.

We identify recurring citation edge cases that must be addressed to reconstruct article-level dependencies; a detailed taxonomy is provided in Appendix A.

Given this dataset, the task is to map each *home article* to the correct *cited article(s)* or document. This requires resolving ambiguous citation structures and distinguishing intra- from inter-document references, making the task non-trivial even for modern LLMs.

4 Methodology

We propose a hybrid pipeline that combines deterministic pre-processing of Fedlex XML documents with LLM-based semantic resolution. The pipeline consists of three stages: (i) data collection from Fedlex, (ii) structured extraction of citation candidates, and (iii) LLM-based article assignment.

Given an SR XML document, the goal is to transform raw legal text into structured citation instances and assign each instance to the correct cited article(s) or document. Figure 1 provides an overview of the pipeline.

4.1 Data Collection

We collect SR documents from the *Landesrecht* of Fedlex using its SPARQL interface. The resulting dataset consists of 2,103 XML documents, each representing a complete legal act. These documents serve as the input to the preprocessing pipeline.

4.2 Structured Citation Extraction

The pre-processing stage converts each XML document into structured JSON items representing potential legal references. We first parse the XML and build a parent map to identify whether a reference occurs in an article or in the preamble and to recover its local context.

After thorough and exhaustive manual tracking of how Fedlex heuristically cites SR relations, we concluded citation candidates can be extracted from these four sources. First, we process **authorial notes**, which often contain hyperlink-based SR references. For each note with an **eli/cc** link, we extract the target SR, target URL, enclosing article identifier, normalized note content, and a paragraph-level snippet containing the current note while excluding unrelated neighboring notes.

Second, we detect **preamble acronyms** by linking parenthesized acronyms outside authorial notes to the nearest SR-bearing authorial note in the preamble and then propagating these mappings to later article occurrences.

Third, we capture **local acronyms** inside articles: if an all-caps acronym appears immediately before an authorial note, we collect matching non-footnote lines from the same article as additional citation context.

Finally, we extract **explicit article mentions** outside authorial notes using patterns such as “Art.” and “Artikel” to capture intra-document references not covered by hyperlink-based extraction. The result is a de-duplicated set of JSON items encoding SR-link detections, preamble-derived acronym references, local-acronym-enhanced references, and explicit article-reference snippets.

4.3 LLM-based Article Assignment

In the second stage, we use an LLM to assign cited articles to the structured citation items. Processing is performed at the document level: for each SR document, the model receives a task-specific prompt (details included in Appendix B), a project README as auxiliary context, and the corresponding citation-candidate JSON.

We query the gpt-5.1 model with deterministic decoding (temperature = 0.0) and require strictly valid JSON output. For each item, the model determines whether the reference corresponds to (i) specific cited articles in a target SR document, (ii) a document-level reference, or (iii) an intra-document reference. The output preserves the original item identifiers and augments them with assigned article(s), a confidence label, and a short justification.

[1]“eli/cc” refers to URL patterns used in the Swiss Fedlex system, where ELI (European Legislation Identifier) provides a standardized, machine-readable identifier for legal documents, and “cc” denotes consolidated versions of federal acts or ordinances. Together, such links point to specific, up-to-date legal provisions within the Swiss legal corpus.

This setup enables the resolution of implicit, acronym-based, and long-distance references that are difficult to capture with deterministic rules.

5 Results and Discussion

5.1 Validation on Edge Cases

Before applying the pipeline to the full corpus, we perform targeted manual validation to ensure the correct handling of diverse citation patterns. We select a representative subset of 30 Landesrecht SR documents so that all identified citation edge-case categories are covered at least once.

For each document, all extracted article-level dependencies are manually verified against the source text. The pipeline achieves full correctness on this subset after iterative refinement, indicating that the preprocessing and LLM-based assignment components are able to handle the known structural and semantic variations of legal references. In Appendix C, we provide a more detailed analysis of five SR documents.

This validation step provides confidence that errors observed at scale are not primarily due to unhandled edge cases, but rather arise from other factors such as output reliability.

5.2 Structural Consistency and Coverage

We analyze the outputs of the pipeline over the full dataset, comprising 2,103 SR documents and a total of 63,821 citation items. Each item is expected to contain at least the fields `ITEM`, `ARTICLE_EID`, `TARGET_SR`, and `assigned_articles`, which together define a valid article-level dependency. Here, `ITEM` denotes a unique identifier for the extracted citation instance, `ARTICLE_EID` identifies the citing (home) article within the source SR document, `TARGET_SR` specifies the referenced SR document, and `assigned_articles` contains the article(s) within the target document that are predicted to be cited.

An item is considered *structurally inconsistent* if any of these required fields are missing. Overall, 50,240 items (78.72%) are structurally consistent, while 13,581 items (21.28%) violate the expected schema. Thus, approximately one in five generated items is not directly usable for downstream processing.

From a pipeline perspective, this also corresponds to an end-to-end *coverage* of 78.72%, defined as the proportion of extracted citation candidates that result in structurally valid article-level assignments. In other words, roughly four out of five detected citation instances can be successfully transformed into usable structured dependencies.

One likely reason why, despite explicit prompting to the LLM, about 20 percent of citation items were not structurally consistent is due to task load + ambiguous precedence: long inputs, duplicated `SR_link_detection` guidance, and soft rules compete for attention with JSON format rules, which weakens strict JSON compliance in practice (Liu et al., 2024). Together, (i) missing hard - structural decoding constrained -schema constraints, (ii) pressure toward natural-language slots when unsure, and (iii) heavy, partly overlapping instructions over large lists may explain the structural misses.

5.3 Recoverability of Inconsistent Items

We further distinguish between *recoverable* and *unrecoverable* inconsistencies. An item is considered recoverable if missing fields can plausibly be reconstructed from auxiliary information (e.g., the reasoning field), and unrecoverable otherwise. In particular, items missing the citing article identifier (`ARTICLE_EID`) or the pair (`TARGET_SR`, `assigned_articles`) are treated as unrecoverable.

Among the 13,581 inconsistent items, only 1,624 (11.96%) are recoverable, corresponding to 2.54% of all items. The remaining 11,957 items (88.04% of inconsistent items, 18.74% overall) are unrecoverable. This indicates that most structural failures are not minor omissions but reflect fundamental breakdowns in output formation, limiting the effectiveness of post hoc correction.

5.4 Failure Modes

To better understand these failures, we analyze which fields are missing in unrecoverable items. The dominant failure mode is the absence of `ARTICLE_EID`, which occurs in 11,919 items (99.68% of unrecoverable cases). In contrast, only a negligible number of items fail due to missing target information alone.

This suggests that the primary bottleneck lies in reliably maintaining the association between extracted citation contexts and their originating articles, rather than in identifying target statutes or articles.

Finally, a key room for improvement is vetting the accuracy of the citation links that were given by this paper's methodology. While ideally one would manually review at least 100 randomly selected structurally consistent SR documents, due to time and financial constraints, we could not conduct such tests outside of the 30 mentioned in section 5.1.

6 Conclusion

We present a hybrid pipeline for extracting article-level dependencies from Swiss federal legislation, combining deterministic XML preprocessing with LLM-based semantic resolution. We also release three complementary data splits—document-level JSON, citation candidates, and LLM-based article assignments—providing a reusable resource for Swiss legal NLP. Our results show that, while LLMs can resolve complex legal references, structural reliability remains a key bottleneck, with over 20% of outputs failing to meet schema requirements. The resulting dataset enables downstream applications such as legal knowledge graph construction, citation analysis, and benchmarking of LLMs for structured information extraction.

7 Limitations

Our approach has several limitations. First, the pipeline relies on an LLM for article assignment, which introduces variability and structural unreliability: over 20% of outputs do not conform to the expected schema. Second, the method is evaluated only on German-language *Landesrecht* documents, and its generalizability to other languages or legal domains remains to be tested. Third, while manual validation confirms correctness on representative edge cases, large-scale semantic accuracy beyond structural consistency is not exhaustively evaluated.

Finally, the approach depends on heuristic preprocessing tailored to Fedlex XML, which may limit portability to other legal corpora with different structures. Addressing these limitations would require stronger guarantees on structured output, broader multilingual evaluation, and more comprehensive annotation for large-scale benchmarking.

References

Annarita Felici. 2025. Cheu-lex: a parallel multilingual corpus of swiss and eu legislation. *Applied Corpus Linguistics*, 5(3):100151.

Stefan Höfler and Michael Piotrowski. 2011. Building corpora for the philological study of swiss legal texts. *Journal for Language Technology and Computational Linguistics*, 26:77–89.

Hans Kelsen. 1967. *Pure Theory of Law*. University of California Press, Berkeley. Translated by Max Knight.

Anne Lise Kjær. 2000. *On the structure of legal knowledge : The importance of knowing legal rules for understanding legal texts*, pages 127–162. De Gruyter Mouton, Berlin, New York.

Teng Lin, Yizhang Zhu, Yuyu Luo, and Nan Tang. 2025. Srag: Structured retrieval-augmented generation for multi-entity question answering over wikipedia graph. *Preprint*, arXiv:2503.01346.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Preprint*, arXiv:2307.03172. Preprint; frequently cited as Liu et al. 2023 on arXiv.

Luca Rolshoven, Vishvaksenan Rasiah, Srinanda Brügger Bose, Sarah Hostettler, Lara Burkhalter, Matthias Stürmer, and Joel Niklaus. 2025. Unlocking legal knowledge: A multilingual dataset for judicial summarization in switzerland. *Preprint*, arXiv:2410.13456.

Darren Yow-Bang Wang, Zhengyuan Shen, Soumya Smruti Mishra, Zhichao Xu, Yifei Teng, and Haibo Ding. 2025. Slot: Structuring the output of large language models. *Preprint*, arXiv:2505.04016.

A Citation Edge Case Taxonomy

In order to develop a Fedlex processing pipeline that preserves relevant contextual information for the LLM to assign article assignments, we exhaustively identified the semantic edge cases that LLMs will struggle with without additional context.

A.1 Edge Case 1: Very Short SR Documents Without Articles or Footnotes

Some Landesrecht SR documents contain extremely short legislative texts that do not include any formal articles or footnotes. In these cases, the *home article* does not exist as a discrete structural article element; instead, the entire SR document functions as a minimal unarticulated law. When such a document is referenced from another SR document, the reference relationship cannot rely on article-to-article mapping, since the *cited article* is structurally absent. This requires semantic handling that identifies the entire cited SR document as the target rather than an article-level node. Automated extraction must therefore detect missing article tags and treat the law as a single undivided article. See SR 901.2[2] for an example.

[2]https://www.fedlex.admin.ch/eli/cc/2020/68/de

A.2 Edge Case 2: References Appearing in the Preamble and Then Reused by Later Articles

In some cases, the *home article* indirectly references a cited SR document via a citation that originally appears only in the preamble of the *home SR document* with an acronym. Later, another article within the same SR document makes a direct reference to the same SR link using the acronym declared in the preamble, creating a chain of implicit and explicit relationships. Semantically, this requires connecting preambular citations with downstream article-level acronym references, even though the preamble itself is not an article and therefore cannot be treated as a normal *home article*. Extraction systems must store and propagate preamble-level citations as valid contextual citations for subsequent articles by remembering the cited SR document, its associated acronym, and article information. See article 9 of SR 901.022.2[3] for an example.

A.3 Edge Case 3: Article Citing Another Article Within the Same SR Document

A further complication arises when a *home article* cites another *cited article* located within the same *home SR document*. In these cases, the reference does not point to a different *cited SR document* but instead refers internally to a separate article of the same law. These internal references often appear inline, embedded within the same sentence as other legislative content, and may cite ranges of articles (e.g., "Articles 4–7"). Because the citation is internal to the same SR document, the extraction system must avoid incorrectly interpreting it as an external reference. Additionally, this case can conflict with Edge Case 8, where subheaders or preamble-derived acronyms may give the illusion of an intra-document reference when the citation actually points outward. Therefore, the algorithm must restrict its evaluation strictly to the sentence-level context when determining whether an article-to-article reference is internal or external. Semantic disambiguation is required to ensure that same-law citations are correctly classified as intradocument article relations rather than external SR dependencies. See paragraph 1, article 9 of SR 901.0[4] for an example.

[3] https://www.fedlex.admin.ch/eli/cc/2016/350/de#art_9

[4] https://www.fedlex.admin.ch/eli/cc/2007/136/de#art_9

A.4 Edge Case 4: Missing Article Numbers for Cited SR Documents

Another challenge arises when the *home article* cites a *cited SR document* but does not specify in words or with article-granular hyperlink the *cited article* within that document. Automated systems must parse the sentence and surrounding sentences to infer which SR identifiers correspond to which article reference, while also allowing "article unspecified" relationships when appropriate. See article 18 of SR 725.11[5] for an example.

A.5 Edge Case 5: Hyperlink Footers Containing Multiple Sources and Article Information

Many *home articles* contain footnotes or hyperlink footers listing multiple legislative sources in compact footnotes. These footers may include several cited AS documents and, in some cases, specific *cited articles* of cited SR law. Because the formatting compresses multiple references into a shared footnote, an extraction system must identify and isolate references corresponding only to the cited SR documents. Then the semantic task is to analyze the surrounding sentences of the sentence that had the footnote and disaggregate mixed citation lists into discrete article-to-article relationships. See article 14 of SR 725.11[6] for an example.

A.6 Edge Case 6: Article Number Provided but Appearing in a Different Location Than the Cited SR Link

In this case, a *home article* cites an SR document via a hyperlink but provides the *cited article number* at a different location in the same paragraph. This breaks the typical assumption that the SR hyperlink and article number appear adjacently. Extraction must use semantic association—linking the textual location of the SR number to the nearest or logically corresponding SR hyperlink—to distinguish this case from mixed-source footers (Edge Case 5). The mapping requires reading the sentence semantics of surrounding sentences within the paragraph rather than purely structural XML patterns. See article 27d of SR 725.11[7] for an example.

[5] https://www.fedlex.admin.ch/eli/cc/1960/525_569_555/de#art_18

[6] https://www.fedlex.admin.ch/eli/cc/1960/525_569_555/de#art_14

[7] https://www.fedlex.admin.ch/eli/cc/1960/525_569_555/de#art_27_d

A.7 Edge Case 7: Combined Intra-Document Article Citations and External SR References Within the Same Sentence

This edge case represents a hybrid scenario involving the complexities of both Edge Case 3 and Edge Case 4. The *home article* contains intra-document references to other *cited articles* within the same *home SR document*—often expressed as inline ranges such as "Articles 4–7"—while simultaneously referencing an external *cited SR document* within the exact same sentence. The external SR reference may even appear without a corresponding *cited article* number, further increasing ambiguity. Because both reference types appear in close proximity and share similar surface patterns, a naive heuristic extraction system may incorrectly merge them, misinterpret internal article ranges as belonging to the external SR document, or otherwise conflate distinct citation relationships.

Resolving this edge case requires sentence-level semantic decomposition. The algorithm must determine whether each article number or article range is an intra-document citation (i.e., referencing another article in the same SR document) or whether the reference is directed toward a different *cited SR document*. External SR references lacking article numbers must be preserved as valid cross-document links, while article ranges such as "4–7" must be interpreted exclusively within the context of the home SR document. Only through semantic differentiation can the system avoid misclassifying internal article citations as external legal dependencies, or vice versa. See article 114 of SR 711[8] for an example. The first SR hyperlink 113 is edge case 3 of not mentioning an article and referencing the whole document. Succeeding that line is a self-reference article line. This cannot easily be heuristically filtered as two separate article references.

A.8 Edge Case 8: Cited SR Link Acronym Mentioned in the Subheader of the Home Article

Some SR documents include acronyms or abbreviated SR references in the subheader of a *home article*, often derived from the cited SR document's preamble or title. These references do not follow standard hyperlink formatting and therefore require semantic recognition to determine whether the acronym genuinely refers to a *cited SR document*. The example below shows that the cited article reference and the related cited article are written above the home article content but below the home article title. Extraction must account for these references that appear outside the article body, treating them as valid citations even when no explicit hyperlink is present. See article 1 of SR 611.01[9] for an example.

A.9 Edge Case 9: A Cited SR Document Named Explicitly in Text After a Hyperlink

In this case, it is similar to edge case 6, but instead of the article number, the cited SR document is referenced again by name elsewhere in the same home article. The *home article* references an SR document via hyperlink (e.g., "SR 311.0") but subsequently names the cited SR document explicitly in plain text (e.g., "Strafgesetzbuch"), extending the reference across multiple textual segments. This creates a compound citation that must be semantically unified: the hyperlink and the textual name must be merged into a single relationship with the correct *cited SR document*. Automated systems must therefore remember and treat adjacent textual mentions/acronyms as belonging to the same citation event rather than two separate references. See article 16 of SR 170.32[10] for an example.

B LLM Prompt

Below, we provide the prompt used for LLM-based article assignment.

```
Read the README.md file for context.

Task: For each input JSON item, assign the
↪  correct cited article(s).
Return a JSON array where each output item
↪  corresponds to the input "ITEM".

General rules:
- "Artikel" and "Art." are equivalent.
- Preamble SR references apply globally to all
↪  articles.

Item types:

1. SR_link_detection:
   Use the "snippet" to determine whether article
   ↪  numbers belong to TARGET_SR.
   Note: Presence of an article number does not
   ↪  guarantee assignment.
```

[8] https://www.fedlex.admin.ch/eli/cc/47/689_701_723/de#art_114

[9] https://www.fedlex.admin.ch/eli/cc/2006/228/de#art_1

[10] https://www.fedlex.admin.ch/eli/cc/1958/1413_1483_1489/de#art_16

```
2. preamble_acronym:
   Use "snippet" and "snippet_acronym" to assign
   ↪  articles via preamble mappings.

3. artikel_reference:
   Detect intra-document references (no
   ↪  TARGET_SR).

Edge cases:
- Conjunctions: "Art. 30, 33, 35 und 36"
- Ranges: "Artikel 4–6"

-----------------------------------------------⌋
↪ -------------
CRITICAL RULE: CONTEXTUAL RELATIONSHIP ANALYSIS
-----------------------------------------------⌋
↪ -------------

PRIMARY INDICATORS:
- Direct marking: "Art. 32 FHG"
- Prepositional link: "nach Art. 20 der
↪  Verordnung (SR ...)"
- Parenthetical association

SECONDARY INDICATORS:
- Clause dependency
- Sentence-level proximity with connectors

SELF-REFERENCE (DO NOT assign):
- No SR nearby
- Separate syntactic structure
- Internal references

Procedure:
1. Identify syntactic boundaries
2. Locate SR references
3. Check grammatical link
4. If unclear → self-reference

Example:
"(Art. 32 und 58 FHG) ... nach Artikel 12 Absatz
↪  4 FHG ...
nach den Artikeln 18 und 21"

→ Assign: 32, 58, 12
→ Do NOT assign: 18, 21

-----------------------------------------------⌋
↪ -------------

Output:
- Valid JSON only
- Preserve ITEM identifiers
- Include assigned_articles, TARGET_SR,
↪  confidence, reasoning

DO NOT use automated scripts. Perform full
↪  semantic analysis.
```

C Manual verification result

Below, we provide additional details on the manual validation procedure described in Section 5.1. We report representative results for a subset of SR documents selected to ensure coverage of all identified citation edge cases. For each document, all extracted article-level dependencies were manually checked against the source text.

Table 1 summarizes the presence of selected edge-case categories (C5–C9) and the corresponding verification accuracy. Across all evaluated documents, the pipeline achieves full correctness, and while it does not guarantee the same for the whole Fedlex, it indicates that the proposed approach reliably handles the diverse citation patterns observed in Swiss federal legislation.

	C5	C6	C7	C8	C9	Acc.
SR 711	Yes	Yes	No	Yes	No	**100%**
SR 725.11	Yes	Yes	Yes	Yes	No	**100%**
SR 420.1	Yes	Yes	No	Yes	No	**100%**
SR 611.01	Yes	Yes	Yes	No	Yes	**100%**

Table 1: Edge-case coverage and manual verification accuracy for four example Landesrecht SR documents. Columns C5–C9 indicate the presence of specific citation edge-case types (Cases 5–9), as defined in Appendix A. Each document is selected to ensure coverage of different edge-case categories, and all extracted article-level dependencies are manually verified.