Argumentative Analysis of Legal Rulings: A Structured Framework Using Bobbitt's Typology

Carlotta Giacchetta[1], Raffaella Bernardi[2], Jacopo Staiano[1], Serena Tomasi[1] Barbara Montini[3]

[1]University of Trento, [2]Free University of Bozen-Bolzano, [3]University of Brescia

carlotta.giacchetta@studenti.unitn.it, raffaella.bernardi@unibz.it, jacopo.staiano@unitn.it, serena.tomasi_1@unitn.it, bmontini001@studenti.unibs.it

Abstract

Legal reasoning remains one of the most complex and nuanced domains for AI, with current tools often lacking transparency and domain adaptability. While recent advances in large language models (LLMs) offer new opportunities for legal analysis, their ability to structure and interpret judicial argumentation remains unexplored. We address this gap by proposing a structured framework for AI-assisted legal reasoning, centered on argumentative analysis. In this work, we use GPT-4o for discourse-level and semantic analysis to identify argumentative units and classify them according to Philippe Bobbitt's (Bobbitt, 1984) six constitutional modalities of legal reasoning. We apply this framework to legal rulings from the Italian Court of Cassation. Our experimental findings indicate that LLM-based tools can effectively augment and streamline legal practice, by e.g. preprocessing the legal texts under scrutiny; still, the limited performance of the state-of-the-art generative model tested indicates significant room for progress in human-AI collaboration in the legal domain.

1 Introduction

In this work, our aim is to develop a digital tool based on Argument Mining and Artificial Intelligence to support legal professionals (judges, lawyers, prosecutors, notaries, and legal trainees) in critically analysing and understanding judicial decisions in their full argumentative complexity. The tool is not intended to replace the legal expert, but rather to assist in navigating the often intricate and cognitively demanding task of interpreting judicial texts. A key theoretical premise of this work is that judicial decisions are fundamentally argumentative products, structured through layers of reasoning that go beyond the mere operative part of the ruling or its legal maxim. Traditional approaches, such as relying solely on summaries or syllogistic reduction, risk obscuring the deeper argumentative processes and implicit assumptions embedded in the decision-making. To address this, we draw from argumentation theory and computational linguistics to extract and classify the internal logic of judgments. The proposed system has been developed to perform two primary functions: *i)* to segment the judgment into discrete argumentative units, each representing an independent statement with argumentative value; and *ii)* to semantically label these units by identifying the type of legal reasoning they instantiate, as illustrated in Figure 1.

To this end, we design a classification framework based on Philippe Bobbitt's typology of constitutional argumentation (Bobbitt, 1984): originally developed in the context of U.S. Supreme Court decisions.

Our main contributions are twofold: first, we release the first corpus of Italian judicial decisions (civil and criminal rulings from the Court of Cassation) annotated with argumentative labels based on Bobbitt's constitutional modalities; second, we propose and evaluate a pipeline that integrates large language models with expert annotation to classify argumentative units in legal texts.

We find that GPT-4o, when guided with carefully designed prompts, can capture a significant portion of the argumentative structure, providing a useful framework for assisted legal analysis. However, human input remains essential in identifying subtle distinctions between modalities, especially in complex or ambiguous reasoning contexts. Notably, even junior expert annotators often struggle to reach full agreement, highlighting the intrinsic complexity and subjectivity of the task of argumentative classification in judicial texts.

2 Related Works

Over the past decade, Argument Mining (AM) has become an increasingly prominent area within the

Proceedings of the 12th Argument Mining Workshop, pages 107–115

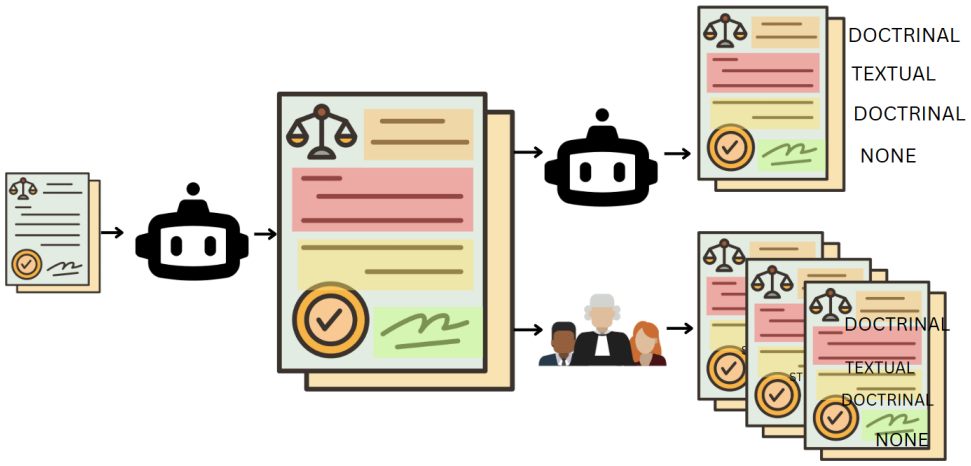

Figure 1: Starting from a legal document, a language model segments the text into paragraphs; then, these are classified by an LLM and domain experts according to Bobbitt's argumentative categories.

intersection of artificial intelligence, computational linguics, and legal informatics.

Our approach builds on a rich body of work by integrating discourse segmentation with semantic classification of legal arguments. Unlike prior studies that focused on broad categories such as premise/conclusion or rhetorical roles(Palau and Moens, 2009), (Santin et al., 2023), (Grundler et al., 2022) we adopt Bobbitt's (Bobbitt, 1984) constitutional modalities as a semantic framework, allowing for a more refined classification of legal reasoning. Further, while much of the previous literature has concentrated on English-language (Chalkidis et al., 2020) decisions from supranational courts (Chlapanis et al., 2024), our work expands the scope by applying the methodology to Italian Court of Cassation rulings. This contributes to the growing interest in multilingual and civil law systems within the AM community, as evidenced by recent shared tasks and datasets, such as the AMELIA (Grundler et al., 2024) challenge for Italian legal texts.

3 Data Collection and Categories

To implement and evaluate our framework, we compiled a custom corpus of Italian judicial decisions and adopted a classification scheme grounded in constitutional legal theory. The process involved selecting representative rulings, curating the textual material, and mapping argumentative content to a set of predefined categories. In what follows, we describe the composition of the dataset and the typology of argumentation used for annotation.

3.1 Corpus Description

The corpus used for this study consists of 20 judgments from the Italian Supreme Court of Cassation, which is the highest judicial authority responsible for ensuring uniform interpretation of the law (a function known as nomofilachy). The selected rulings are among the most significant ones highlighted by the Court's official website[1] and include 10 civil and 10 criminal cases.

All rulings were written in Italian and sourced from the *De Jure* legal database.[2] Civil cases span the years 2018 to 2025, while criminal cases are drawn from 2023 and 2024, reflecting the most recent developments in judicial language and practice.

The rulings vary significantly in length and complexity, ranging from concise decisions of 4 pages to more elaborate ones extending up to 26 pages. This diversity reflects the variability of legal practice and provides a realistic testbed for evaluating both human and model-based annotation of argumentative content.

Prior to annotation, all decisions were preprocessed to extract the full text, removing non-argumentative sections such as headers, metadata, or procedural summaries.[3]

[1] https://www.italgiure.giustizia.it/sncass/

[2] Available at: https://dejure.it/

[3] Preliminary experiments were conducted using the *Demosthenes dataset* (CJEU decisions on fiscal state aid), as it already featured a well-defined argumentative structure. This allowed us to initially focus on the categorization task. However, we later decided to shift our main focus to Italian judicial decisions.

3.2 Categories

To classify the argumentative content of each decision, we adopted Bobbitt's typology of constitutional reasoning. This framework identifies six primary categories (Historical, Textual, Structural, Prudential, Doctrinal, Ethical), each corresponding to a different mode of legal justification.[4]

In addition to these six, we introduced a residual category, `None`, to capture instances where no clear argumentative function could be assigned, either due to lack of information or the presence of purely descriptive or procedural content.

4 Methodology

The core objective of our approach is to transform legal rulings into structured argumentative representations. This involves two main steps: (1) segmenting the ruling into coherent textual units (typically paragraphs), and (2) identifying the role of each segment within the broader argumentative structure of the document.

We operationalize this by first extracting the ruling text and dividing it into paragraphs, which are then labeled based on their argumentative function—either as premises, conclusions, or non-argumentative content. Each paragraph is uniquely identified and embedded in a hierarchical structure that reflects the flow of reasoning. Subsequently, we group semantically related segments and assign them to one of Bobbitt's constitutional categories of argumentation. This process results in a multi-layered representation of the ruling that supports both human interpretability and machine processing.

4.1 Text Segmentation into Paragraphs

The tool's primary function is to divide complex legal texts into coherent paragraphs. This is essential as it lays the groundwork for structuring the text, which will later be analyzed at the sentence level. Each paragraph is analyzed and classified into one of the following categories: *premise*, *conclusion* or *null*.

To maintain a structured representation of the argumentation, each paragraph is assigned a unique identifier built via *i)* a single character indicating the argument chain (e.g. A, B..) and *ii)* a progressive number denoting the order within the chain (e.g., A1, A2, B1)[5]. This structured XML representation ensures that the text remains both machine-readable and systematically organized, thereby facilitating downstream processing and analysis. Our pipeline design follows the structure adopted in the Demosthenes dataset proposed by Santin et al. (2023).[6]

4.2 LLM Annotations

The second phase of the pipeline is the annotation process, which is divided into two main steps.

The first step is **semantic grouping**, in which paragraphs are clustered based on their semantic similarity using a GPT-based model. Each group is assigned a unique `group_id` and can include up to eight paragraphs. Paragraphs that do not semantically align with others remain ungrouped and are labeled with `group_id: null` [7] The goal of this phase is to identify groups of argumentative units that address the same topic or rely on a shared line of reasoning. This semantic grouping serves a crucial functional role: it establishes the granularity at which constitutional argumentation categories (as described in Section 3.2) are assigned. Rather than classifying individual paragraphs—which may be too short or context-poor for accurate labeling—we classify entire semantic groups. Each group typically represents a coherent argumentative theme, making it a more suitable unit for the assignment of one of Bobbitt's six categories.

In the categorization step, each semantic group is passed to the LLM, which selects the most appropriate constitutional argument type from the predefined set, or assigns the label `None` if no category applies. The model is instructed to justify its choice with a brief explanation [8].

Both grouping and classification were conducted using a *zero-shot prompt* with structured instructions describing each category, and a temperature setting of 0.2. At the end, each paragraph in the XML file is annotated with three tags: an `<ID>` - corresponding to the paragraph identifier; a `<Group>` – corresponding to the `group_id` assigned to the paragraph and a `<Category>` – representing the classification label of the paragraph.

[4]Details on the description of the Categories are provided in Appendix A.

[5]For details on the prompt's syntactic structure, see Appendix D.

[6]Implemetation details, including file conversions and annotation formatting, are provided in Appendix B.

[7]For details on the prompt's semantic grouping, see Appendix E.

[8]For details on the prompt's categorization, see Appendix F.

4.3 Human Annotation

To validate the output of the automatic annotation pipeline, we collected manual annotations from a panel of five human experts with varying legal backgrounds. The annotators included: Junior experts (one law student, two legal trainees, one PhD candidate), and Senior expert (an university professor of constitutional law).

Each annotator was provided with a structured Excel file containing the paragraphs grouped and labeled with `group_id`, as generated by the model. For each paragraph, they were asked to assign one of Bobbitt's categories using a drop-down menu. To ensure comparability with the model's behavior, all annotators received the exact same prompts used by GPT during the automatic annotation phase. In cases where a semantic group appeared incoherent or internally inconsistent, annotators were instructed to assign the most appropriate category nonetheless—based on the dominant argumentative theme—and to flag the group as "incorrect." They could also provide suggestions for a more appropriate regrouping. This protocol allowed us to both preserve comparability with GPT outputs and collect qualitative feedback on grouping validity.

Each Junior Expert annotated a subset of the rulings, while the Senior Expert annotated the entire corpus. This design allows us to compute both human–AI agreement and human–human agreement, with a focus on the differences between expertise levels and the model's alignment with legal reasoning across varying levels of legal training.[9] Humans took from 30 to 120 minutes per judgment.

5 Experimental Results

To assess the consistency of GPT annotations relative to human judgment, we compute agreement using two complementary strategies: intersection-based and union-based evaluation – Both approaches rely on Cohen's κ (Cohen, 1960). These two perspectives allow us to evaluate GPT both on highly reliable human annotations (intersection) and in more flexible, real-world scenarios (union).

Intersection-based agreement ($\cap$). In this setting, we first construct a subset of the dataset consisting only of those instances where both human annotators independently assigned the same label to a given paragraph. These agreed-upon labels are treated as the gold standard, and GPT's output is compared against them. This provides a high-precision evaluation, focusing only on cases where human consensus exists.

[9]Details on the annotation interface and inter-annotator agreement metrics are provided in Appendix C.

Table 1: **Cohen's Kappa** agreement scores between human annotators (senior and juniors) and GPT for the annotated documents in Criminal and Civil Law.

	Criminal Law	Civil Law
Senior vs Junior	0.17	0.27
Senior vs GPT	0.15	-0.03
Junior vs GPT	0.07	-0.09
Senior $\cap$ Junior vs GPT	0.15	-0.0936
Senior $\cup$ Junior vs GPT	0.46	0.1874

Table 2: Detailed classification report for GPT vs Human (union-based agreement) on annotated documents from Italian Criminal Law.

Categories	Precision	Recall	F1-score	N.
Doctrinal	0.41	0.80	0.54	30
None	0.95	0.87	0.91	212
Prudential	0.00	0.00	0.00	2
Structural	0.00	0.00	0.00	2
Textual	0.00	0.00	0.00	8

Union-based agreement ($\cup$). In the union-based setting, we take a more permissive approach: GPT is considered correct if its predicted label matches either of the two human annotators. This strategy accounts for cases where annotators diverge but GPT still aligns with one of them, thereby capturing partial alignment with human judgment. This formulation is particularly suited for analyzing noisy or ambiguous labels, and reflects the inherent subjectivity of legal interpretation.

Looking at the agreement scores in Table 1, we observe that the Senior Expert achieves consistently higher agreement with GPT than the Junior Expert does, particularly in the Criminal Law domain. This suggests that the model tends to align more closely with interpretations grounded in deeper legal reasoning and experience.

However, when comparing the individual category distributions (Tables 2 and 3), we note that GPT often selects different categories than the experts, especially in cases where legal argumentation is subtle or multi-layered. Moreover, we noticed that GPT captures only a limited subset of argumentative categories, missing the semantic nuances that legal experts can identify thanks to their domain

Table 3: Detailed classification report for GPT vs Human annotators (union-based agreement) on documents from Italian Civil Law.

Categories	Precision	Recall	F1-score	N.
Ethical	0.00	0.00	0.00	1
Doctrinal	0.39	0.76	0.52	66
None	0.77	0.45	0.57	130
Textual	0.50	0.18	0.27	11

knowledge. These findings highlight the complexity of modeling judicial argumentation, where even human annotators often disagree.

Overall, GPT's behavior appears more comparable to that of a Junior Expert: while it demonstrates basic familiarity with argumentative distinctions, it lacks the consistency and depth shown by the Senior Expert, particularly in capturing less frequent or more conceptually demanding categories like Textual, Ethical, and Prudential.

Interestingly, the underlying classification framework shows limitations: Bobbitt's categories, developed for U.S. constitutional contexts, are often too broad or rigid to account for the fact-based and procedural reasoning typical of Italian jurisprudence. This mismatch likely contributes to the observed difficulties in annotation and model prediction, and suggests the need for more refined and context-sensitive taxonomies tailored to the Italian legal system.

Note: In Tables 2 and 3, the last column ("N.") indicates the number of paragraphs in the test set that were assigned to each category according to human annotations. This provides context for interpreting class imbalance and the model's relative performance per category.

6 Conclusion

This project advances the development of epistemically responsible legal AI by addressing the practical and conceptual challenges of Argument Mining in judicial decisions. Through expert-guided annotation, we exposed the limitations of current models in handling complex legal reasoning, particularly in terms of time demands, semantic ambiguity, and segmentation. Yet, the process proved valuable for enhancing legal understanding, with potential applications in education, research, and decision support. By refining classification strategies and prompt design, our work contributes to more transparent and trustworthy AI systems in the legal domain.

Limitations

Our current pipeline relies solely on GPT-4o, a general-purpose model not specifically tuned for legal tasks. This may limit its precision, especially in domains requiring up-to-date legal knowledge or fine-grained distinctions in terminology. Moreover, no systematic comparison has been made with alternative models—either proprietary or open-source.

Future work will benchmark multiple LLMs and investigate domain-adapted models optimized for legal argument classification.

Acknowledgments

References

Philip Bobbitt. 1984. *Constitutional fate: Theory of the Constitution*. Oxford University Press.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. Legal-bert: The muppets straight out of law school. *arXiv preprint arXiv:2010.02559*.

Odysseas S Chlapanis, Dimitrios Galanis, and Ion Androutsopoulos. 2024. Lar-echr: A new legal argument reasoning task and dataset for cases of the european court of human rights. *arXiv preprint arXiv:2410.13352*.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Giulia Grundler, Andrea Galassi, Piera Santin, Alessia Fidelangeli, Federico Galli, Elena Palmieri, Francesca Lagioia, Giovanni Sartor, and Paolo Torroni. 2024. Amelia - argument mining evaluation on legal documents in italian: A calamita challenge. In *CLICIT*.

Giulia Grundler, Piera Santin, Andrea Galassi, Federico Galli, Francesco Godano, Francesca Lagioia, Elena Palmieri, Federico Ruggeri, Giovanni Sartor, and Paolo Torroni. 2022. Detecting arguments in CJEU decisions on fiscal state aid. In *Proceedings of the 9th Workshop on Argument Mining*, pages 143–157, Online and in Gyeongju, Republic of Korea. International Conference on Computational Linguistics.

Raquel Mochales Palau and Marie-Francine Moens. 2009. Argumentation mining: the detection, classification and structure of arguments in text. In *Proceedings of the 12th international conference on artificial intelligence and law*, pages 98–107.

Piera Santin, Giulia Grundler, Andrea Galassi, Federico Galli, Francesca Lagioia, Elena Palmieri, Federico Ruggeri, Giovanni Sartor, and Paolo Torroni. 2023. Argumentation structure prediction in cjeu decisions on fiscal state aid. *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law*.

A Categories

- `Historical arguments`: refer to reasoning based on the original intentions of lawmakers, often invoking the legislative history or founding principles behind a norm.
- `Textual arguments`: rely on the literal or grammatical meaning of the legal text itself, emphasizing the surface structure of statutory language.
- `Structural arguments`: are concerned with the internal logic and architecture of the legal system, drawing connections between different institutional functions or constitutional provisions.
- `Prudential arguments`: take into account the practical consequences of a legal interpretation, including its potential benefits, risks, or social implications.
- `Doctrinal arguments`: are grounded in legal precedents and established jurisprudential interpretations, aiming to ensure consistency and stability in the application of the law.
- `Ethical arguments`: appeal to moral values or societal ideals, often drawing on broader cultural or philosophical principles.

B Annotation Pipeline and File Formatting

The annotation pipeline consists of a multi-stage process aimed at converting legal rulings in PDF format into structured, machine-readable representations enriched with argumentative annotations. The process includes the following steps:

1. **PDF to XML conversion**: The raw PDF files are preprocessed to extract text, which is then segmented into paragraphs and stored in an XML structure. Each paragraph is enclosed within a `<par>` tag and assigned a unique identifier.
2. **Paragraph classification**: Using a language model, each paragraph is labeled as either a `Premise`, `Conclusion`, or `Null`, and wrapped in corresponding tags (`<prem>`, `<conc>`).
3. **Semantic grouping and categorization**: Related paragraphs are grouped semantically and assigned a `group_id` and a `Category`. A short explanation is generated for each group to justify both the grouping and the assigned category. These elements are stored in a structured JSON file and later used to augment the XML.
4. **XML augmentation**: The JSON-based annotations are reintegrated into the XML as new attributes: `Group` and `Category` are added to each paragraph node, while paragraph identifiers remain embedded as `<ID>` tags.
5. **Export to Excel**: For improved usability, the enriched XML is converted into an Excel spreadsheet in which each row represents a paragraph, and each column corresponds to one of the annotations (e.g., paragraph ID, group ID, argument role, Bobbitt category).

C Human Annotation Protocol

To support training and evaluation, we collect human-annotated data for a subset of legal rulings. The annotation process is carried out by legal experts, who were provided with a structured Excel file to guide and simplify the task. The annotation workflow follows these guidelines:

- **Pre-segmented input**: Annotators receive the ruling already segmented into paragraphs and grouped semantically. Each paragraph is associated with a pre-assigned `group_id`.
- **Category assignment**: For each paragraph, annotators select the most appropriate constitutional argument category from a drop-down menu. The available options correspond to Bobbitt's six constitutional categories: *Historical*, *Textual*, *Structural*, *Prudential*, *Doctrinal*, *Ethical*, or *None*.
- **Group-based validation**: Since all paragraphs belonging to the same group are visually adjacent in the spreadsheet, annotators can easily compare their content and ensure coherent category assignment across the group.

We provide annotators with clear definitions and examples for each label to ensure consistency. This setup reduces annotation ambiguity and improves efficiency. Inter-annotator agreement is evaluated using Cohen's kappa and F1 score. Results are reported in Section 5.

D Prompt syntactic structure

This prompt guides the model in dividing a legal text into coherent paragraphs, each labeled as a "premise", "conclusion", or null. Each paragraph is assigned a unique identifier based on the logic of argument chains. It is used to generate machine-readable XML structures, as described in the main section of the paper.

```
{
    "role": "system",
    "content": (
        "You are an assistant skilled in
        ↪  analyzing and structuring legal
        ↪  texts. "
        "Your task is to divide the given text
        ↪  into coherent paragraphs and
        ↪  annotate each paragraph as either a
        ↪  'premise' or a 'conclusion', "
        "as part of an argument chain. An
        ↪  argument chain is defined as an
        ↪  argument supporting the final
        ↪  conclusion concerning a specific
        ↪  ground of appeal, "
        "together with all counterarguments
        ↪  considered by the Court. Multiple
        ↪  argument chains may be present in a
        ↪  single decision. "
        "Each premise and conclusion is denoted
        ↪  through a unique identifier (ID),
        ↪  composed of a letter (indicating
        ↪  the argument chain, e.g., A or B) "
        "and a progressive number (indicating
        ↪  the specific premise or conclusion
        ↪  within that chain, e.g., A1, A2, B1,
        ↪  B2).\n\n"
        "**Output Guidelines:**\n"
        "1. **Structure:** Return the output as
        ↪  a JSON array. Each element in the
        ↪  array must have the following
        ↪  structure:\n\n"
        "{\n"
        "  \"ID\": \"A1\",  // A unique ID for
        ↪  the paragraph (e.g., A1, A2, B1,
        ↪  etc.)\n"
        "  \"type\": \"premise\" or
        ↪  \"conclusion\",  // Type of
        ↪  paragraph\n"
        "  \"content\": \"The actual text of the
        ↪  paragraph\"\n"
        "}\n\n"
        "2. **Paragraph Coherence:** Ensure each
        ↪  paragraph represents a single
        ↪  logical unit.\n"
        "3. **Annotation:** Annotate paragraphs
        ↪  accurately as 'premise' or
        ↪  'conclusion'. "
        "If a paragraph does not fit clearly as
        ↪  a 'premise' or 'conclusion', leave
        ↪  the 'type' field as null.\n"
        "4. **ID Assignment:** Assign IDs using
        ↪  the following pattern:\n"
        "   - Use a letter (e.g., A, B) to
        ↪  indicate the argument chain the
        ↪  paragraph belongs to.\n"
        "   - Use a progressive number (e.g., A1,
        ↪  A2, B1, B2) to denote the order
        ↪  within the chain.\n"
        "5. **Ensure Consistency:** IDs must not
        ↪  restart for each chunk of text.
        ↪  Maintain continuity across all
        ↪  chunks.\n\n"
        "**Example Output:**\n"
        "[\n"
        "  {\"id\": \"A1\", \"type\":
        ↪  \"premise\", \"content\": \"The
        ↪  court finds that...\"},\n"
        "  {\"id\": \"A2\", \"type\":
        ↪  \"conclusion\", \"content\":
        ↪  \"Therefore, the appeal is
        ↪  dismissed.\"},\n"
        "  {\"id\": \"B1\", \"type\":
        ↪  \"premise\", \"content\": \"A
        ↪  counterargument is
        ↪  presented...\"},\n"
        "  {\"id\": \"B2\", \"type\": null,
        ↪  \"content\": \"Background context
        ↪  about the case.\"}\n"
        "]\n\n"
        "Now process the following text
        ↪  according to these guidelines."
    ),
},
{
    "role": "user",
    "content": (
        "Here is the text to process:\n\n"
        f"{text}\n\n"
        "Please divide it into coherent
        ↪  paragraphs, tag them as 'premise',
        ↪  'conclusion', or null, assign
        ↪  unique IDs, and return the output in
        ↪  JSON format."
        f"PAY ATTENTION: {self.state_message}"
    ),
}
```

This message is dynamically generated and included in the prompt to ensure that the numbering of argument chain IDs (e.g., A1, A2, ...) remains continuous, even when the text is processed in multiple chunks.

```
self.state_message = (
          f"The current chain is '{chain}'.
          ↪  "
          f"Ensure continuity of the chain
          ↪  IDs {self.current_chain}{s⌋
          ↪  elf.current_progressive}."
      )
```

E Prompt semantic grouping

The following is the prompt used to guide the language model in grouping legal sentences based on semantic meaning:

```
{
    "role": "system",
    "content": (
        "You are an assistant skilled in the
        ↪  structural and semantic analysis of
        ↪  legal sentences. "
```

```
        "You will receive sentences annotated
        ↪  with an ID and various attributes. "
        "Your task is to group the sentences
        ↪  that share a common semantic logic
        ↪  or address the same topic.\n\n"
        "Follow these **strict guidelines** when
        ↪  grouping the sentences:\n"
        "1. **Do not exceed 7/8 sentences per
        ↪  group**: Under no circumstances
        ↪  should a group contain more than 8
        ↪  sentences.\n"
        "2. **Group by semantic meaning**:
        ↪  Ignore the IDs and order. Base the
        ↪  grouping purely on the meaning of
        ↪  each sentence.\n"
        "3. **Leave unrelated sentences
        ↪  ungrouped**: Assign them to
        ↪  `group_id: null` with an
        ↪  explanation.\n"
        "4. **Provide clear reasons for
        ↪  grouping**: Explain why sentences
        ↪  are grouped together, focusing on
        ↪  their shared logic or theme.\n\n"
       "Format the response strictly as JSON:\n"
        "{\n"
        "  \"groups\": [\n"
        "    {\n"
        "      \"group_id\": 1,\n"
        "      \"sentence_ids\": [\"ID1\",
        ↪  \"ID2\", \"ID3\"],\n"
        "      \"reason\": \"Explanation for why
        ↪  these sentences are grouped
        ↪  together.\"\n"
        "    },\n"
        "    {\n"
        "      \"group_id\": 2,\n"
        "      \"sentence_ids\": [\"ID4\",
        ↪  \"ID5\"],\n"
        "      \"reason\": \"Explanation for
        ↪  this grouping.\"\n"
        "    },\n"
        "    {\n"
        "      \"group_id\": null,\n"
        "      \"sentence_ids\": [\"ID8\",
        ↪  \"ID20\"],\n"
        "      \"reason\": \"Ungrouped sentences
        ↪  due to lack of thematic
        ↪  connection.\"\n"
        "    }\n"
        "  ]\n"
        "}\n")
},
{
    "role": "user",
    "content": (
        "Here are some legal sentences annotated
        ↪  with IDs:\n\n"
        f"{json.dumps(chunk, indent=2)}\n\n"
        "Please strictly adhere to the
        ↪  guidelines "
        "Group unrelated sentences under
        ↪  `group_id: null`. Provide clear
        ↪  reasons for each group."
        f"{mex}")
}
```

Because the entire text is divided into chunks due to the maximum token length limitation of GPT, it is crucial to maintain the continuity of group assignments across different chunks. To achieve this, the IDs assigned in previous chunks are passed to subsequent chunks. This ensures that sentences that were already grouped together remain in the same group and that no sentence is reassigned to a different group incorrectly. To implement this, the following message (mex) is injected into the prompt, warning the model to preserve the group IDs from previous outputs:

```
mex =   (
            f"PAY ATTENTION:"
            f"The past groups are
            ↪  '{output_file}'. "
            f"Ensure continuity of the groups
            ↪  IDs and don't change groups of
            ↪  senteces that were yet
            ↪  asisigned."
        )
```

F Prompt semantic structure

To systematically categorize supporting arguments in legal texts, we use the following prompt. The goal is to classify a given argument into one of several predefined subcategories, ensuring a structured and interpretable classification process.

```
{
    "role": "system",
    "content": (
        "You are an expert assistant in
        ↪  analyzing legal texts. "
        "Your task is to classify a supporting
        ↪  argument into one of the following
        ↪  subcategories, "
       "or to indicate that none is appropriate:
        ↪  \n"
        "- **Historical Arguments**:
        ↪  Interpretation based on the
        ↪  original intentions of the framers
        ↪  and ratifiers.\n"
       "- **Textual Arguments**: Based solely on
        ↪  the literal meaning of the words.\n"
        "- **Structural Arguments**: Analysis of
        ↪  the overall constitutional system
        ↪  and interactions among its parts.\n"
        "- **Prudential Arguments**: Evaluation
        ↪  of practical pros and cons and
        ↪  social consequences.\n"
        "- **Doctrinal Arguments**: Use of legal
        ↪  precedents to resolve new cases.\n"
        "- **Ethical Arguments**: Based on moral
        ↪  principles and shared societal
        ↪  values.\n\n"
        "If none of the categories is suitable,
        ↪  you may indicate that the text does
        ↪  not fit into any of them.\n\n"
        "Please return the result in the
        ↪  following JSON format:\n\n"
        "{\n"
        f"   \"Group\": \"Group {group}\",\n"
        "   \"Category\": \"[Name of Category or
        ↪  'None']\",\n"
        "   \"Reason\": \"[Explanation for the
        ↪  classification]\"\n"
```

```
        "}\n"
    )
},
{
    "role": "user",
    "content": (
        f"The following text is a supporting
        ↪  argument: {text}. "
        f"Group: {group}"
        "Analyze the content and identify the
        ↪  most relevant subcategory from the
        ↪  provided options, "
        "or indicate if none of the
        ↪  subcategories is appropriate. "
        "Please ensure the response is formatted
        ↪  strictly as JSON, following the
        ↪  example provided."
    )
}
```