LLM's for text-reuse: capturing legislative amendments success in Spain

Andreu Rodilla

BSC-CNS
andreu.rodilla@bsc.es

Andreu Casas

Royal Holloway University of London
andreu.casas@rhul.ac.uk

Joan Giner

BSC-CNS
joan.giner@bsc.es

August 18, 2025

Abstract

Comparing legislative texts has long been a central task for political scientists and legal scholars addressing diverse research questions. Since the late 20th century, the rise of computational social sciences has introduced automated approaches to this process, enabling researchers to scale their analyses and explore broader patterns. One such advancement is the systematic analysis of legislative amendments. Using rule-based methods, previous studies have employed text-reuse techniques to determine whether legislative amendments proposed for a bill are incorporated into the final law. While these methods have facilitated large-scale analyses of amendments, they face notable limitations, such as accurately identifying relevant text across documents (amendment, bill, and law) and managing cross-amended sections. Recent advancements in large language models (LLMs), however, provide new opportunities to overcome these challenges by 1) improving the text-reuse workflow and 2) increasing measurement accuracy. This study develops an LLM-based approach, including subsequent fine-tuning, to analyse all legislative amendments in Spain between 1996 and 2019 (more than 93.000 amendments). Validation is conducted using a hand-coded dataset of over 5,000 amendments.

1 Introduction

Measuring the success (or failure) of legislative amendments is crucial for a comprehensive and accurate understanding of the policymaking process. Amendments are reflective of the ideological positions of individual legislators and parliamentary groups, of negotiations between parties and institutions (e.g. intra-chamber negotiation), and of political compromises – all key aspects of policymaking (Krehbiel, 1998; Martin and Vanberg, 2005; Ryan, 2018; König et al., 2023). The study of the proposal and adoption of legislative amendments provides relevant insights into political priorities, insitutional dynamics, and policy shifts.

Until recently, most studies of amendment success focused on a small set of amendments. As Palau et al. (2023) pointed out, this is in part because many see the role of parliaments in most parliamentary systems as residual, with the executive having the dominant legislative power (Olson, 1995). But the absence of large-N analyses is also a function of data unavailability, and more importantly, of a lack of automatic methods for scaling up the analysis. In the last decade, advances in automatic text analysis (Wilkerson and Casas, 2017), and text-reuse methods in particular (Wilkerson et al., 2015; Casas et al., 2020), have facilitated the automatic study of legislative edits at scale.

However, existing text-reuse approaches often fail at capturing granular amendment success for three main reasons. First, existing research aims to capture amendment success at a more aggregate level (e.g. during the committee stage; or when the other chamber amends the text) by comparing different version of the same bill as it progresses (Gava et al., 2021; Casas et al., 2020). For example, by looking at the version of the bill that passed one chamber, compared to the version of the bill amendment by the other chamber. However, this level of granularity makes it very difficult to trace influence back to the individual legislator, or parliamentary group, responsible for each new edit – making it not suitable for the study of individual amendments. Second, those that do look at individual amendment success, use traditional text-reuse methods that simply compare the similarity between two documents (e.g. a proposed amendment *v.* the enacted legislation). Yet, amendments often describe the edit to be made (e.g. *remove last sentence of Art. 3.1*) rather than propose a full alternative text (e.g. a full replacement text for Art. 3.1) – making conventional text-reuse methods unsuitable. And third, to account for the latter, existing research takes a rule-based approach to distinctly deal with each different type of amendment. For instance, in the previous example the rules would be to first parse the text of the original bill into (sub)articles, pull the text of Art. 3.1, pull the text of the last sentence, and then use traditional text-reuse methods to see if the sentence made it to the final text (or the next version) of the bill. This is an incredibly arduous approach because of the many inconsistencies in how legislators and parliamentary groups draft amendments.

To tackle these challenges, this study explores the use of Large Language Models (LLMs) for measuring individual amendment success. To do so, we rely on an original dataset of 5,000 annotated amendments proposed to X enacted laws in Spain between XX and XXX – for which we have manually coded in each case whether the amendment was successfully incorporated in the final text of the law. We demonstrate that LLMs offer greater flexibility and linguistic competence than rule-based systems, and so have great potential for improving the study of amendments success at scale. However, they also come with their own limitations.

2 The Legislative Amendment Success Task

Before presenting our methodology, we clarify the structure of the amendment classification task. In this context, an amendment is a proposed change to a bill, which may fall into one of the following categories:

- **Addition:** Introducing new legal content not present in the original bill.
- **Deletion:** Removing content from the bill in the final law.
- **Modification:** Altering existing content in the bill (a mix of additions and deletions).

We define an amendment as *successful* if its proposed change is reflected in the final version of the law. Figure 1 illustrates a sample amendment alongside the corresponding sections of the bill

and the enacted law. This example represents the basic unit of analysis used to evaluate amendment success.

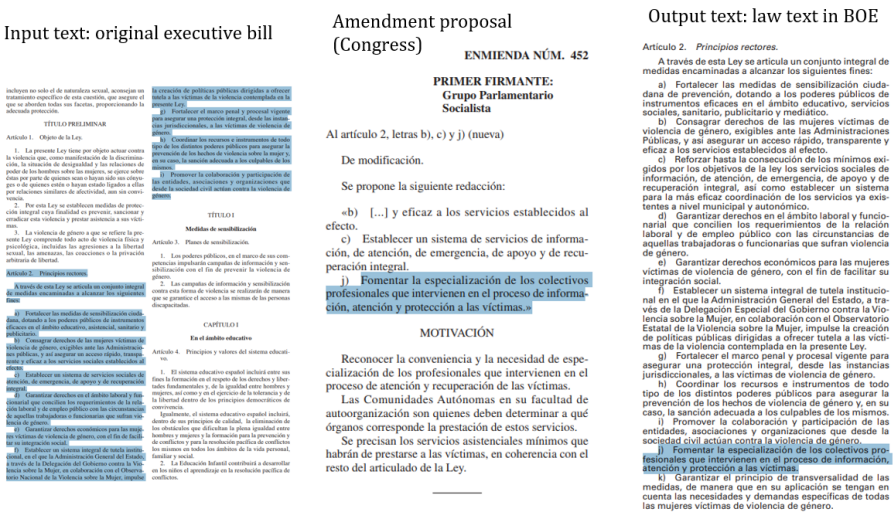


Figure 1: Example of a legislative amendment, with corresponding bill text and final law text.

3 Rule-Based Amendment Tracking Workflow

Figure 2 illustrates the rule-based approach that we first used to measure amendment success. To do so, for a given piece of legislation we compare the text of proposed amendments, to the text of the originally bill, and to the text of the final enacted law. Our dataset of Spanish amendments already contains information about whether each of them is an Addition, Deletion, or Modification type amendment (we collected this information from the parliamentary records during data collection). A first rule hence is to treat Addition and Modification amendments, compared to Deletion (Supression) ones, differently.

The former types (Addition and Modification) propose new text, and are often complete replacements, proposing full alternative text rather than describing the edits to be made. However, the amendment text often includes text that was part of the original bill. For example, when an amendments proposes an alternative text for Art 3.1 but what is actually new is simply the last sentence. To only assess the success of the newly proposed text, a rule in our approach is, for these cases, to go back to the original text of the bill and use text-reuse methods (e.g. full-sentence matching) to first disregard text that had already been proposed in the original bill. Finally, we use the same text-ruse method (e.g. full-sentence matching) to measure amendment success by looking at whether the newly proposed text *is* in the final text of the law.

In our dataset, Deletion amendments sometimes include the full text to be deleted from the bill. For these cases we can go ahead and use the chosen text-reuse method (e.g. full-sentence matching) to assess amendment success by looking at whether the text is *not* in the final text of the law. However, one important caveat is that rather than stating the full text to be deleted, Deletion amendments in our dataset instead often describe the edits to be made: e.g. *delete part (j) of Art. 3.1.* For these cases, we use a different set of rules. First, we recursively parse the text of the bill down to the Subsubarticle level (Title > Chapter > Article > Subarticle > Subsubarticle). Then we pull the target (deletion) text for the particular amendment (Art. 3.1.j), and finally use the rules described below for dealing with regular Deletion amendments.

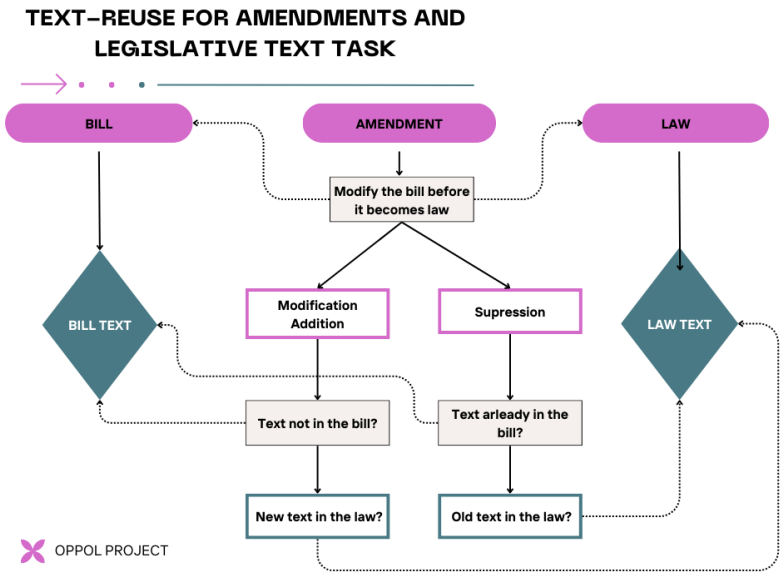


Figure 2: Text-reuse workflow for rule-based amendment tracking. The system categorizes changes as additions, modifications, or suppressions by comparing the bill and law texts.

We implemented this rule-based approach to predict amendment success in our annotated dataset of 5,000 amendments. We used exact sentence-matching as the backbone text-reuse method. We achieved an accuracy of around 75% for Addition and Modification amendments. However, it performed poorly for Deletion amendments. In particular, the problem was with Deletion amendments that described the edits to be made instead of directly stating the text to be deleted. This is because of many complexities associated with the additional rules/steps for dealing with these types of Deletion amendments.

4 Challenges in Rule-Based Approaches to Legislative Text Reuse

The limitations of rule-based systems fall into two main categories: (1) challenges in comparing legal texts—which is necessary to assess whether an amendment passed by comparing the initial (bill) and final (law) texts—and (2) difficulties in determining whether an amendment was successfully enacted—even when the relevant sections are identified, what is the correct mechanism for comparison? Did the amendment add text? How much? Did it modify text? Which parts? Did it delete text? Was the deleted text removed due to this amendment specifically?

4.1 Challenges in Text Comparison

In order to determine whether an amendment is incorporated into the final law, the proposed modification must be compared both to the original version of the bill and to the final version enacted into law. Accordingly, a central component of this method involves:

1. Identifying the type of modification proposed (e.g., addition, deletion, or alteration),
2. Locating the exact section of the bill the amendment seeks to modify,
3. Extracting the specific language introduced or changed by the amendment, and
4. Identifying the corresponding text in the final law for comparison.

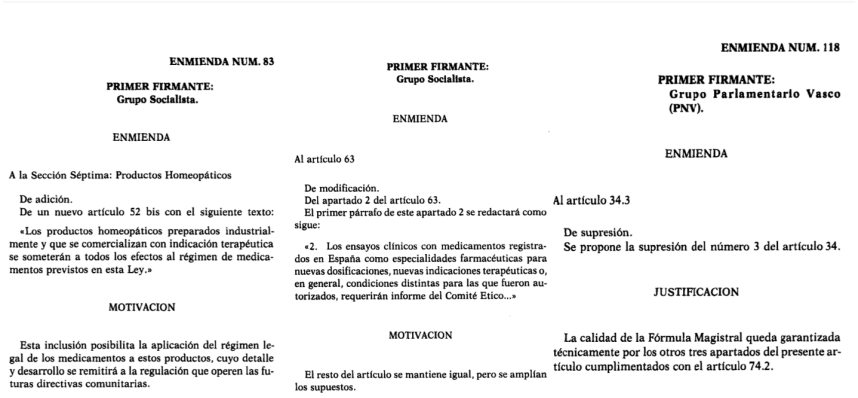
This is where most of the challenges arise when attempting to automatically assess whether an amendment has passed using rule-based approaches grounded in text reuse techniques. In the following subsections, we outline several key obstacles encountered when implementing such methods.

We then explain how Large Language Models (LLMs) can help overcome some of these limitations and present our initial LLM-based workflow.

4.1.1 Parsing and Digitization Issues

Many legislative documents are not born-digital and must be processed using Optical Character Recognition (OCR). OCR introduces noise due to inconsistent formatting, specialized fonts, or poor scan quality, all of which degrade the accuracy of parsing and text matching. Figure 3 illustrates how an amendment appears in the Spanish parliament. While the format is relatively stable—enabling this type of analysis—the quality of digitization varies depending on the legislature and the technologies available at the time.

In some parliaments, digitization improves over time, but in others it remains poor. Additionally, changes in parliamentary group leadership (who are responsible for submitting amendments in the Spanish parliament) can slightly alter amendment structure in our case study. The bill and law texts themselves are often poorly digitized, especially before 1996. Spanish legislative drafting is also less structured compared to countries like Germany (Engst, 2021), which imposes limitations on this study in two ways: (1) it makes pre-1996 analysis more difficult, and (2) it introduces considerable noise in text comparison, posing a particular problem for rule-based systems—which lack contextual awareness and interpretive ability.



ENMIENDA NUM. 83

PRIMER FIRMANTE:
Grupo Socialista.

ENMIENDA

A la Sección Séptima: Productos Homeopáticos

De adición.
De un nuevo artículo 52 bis con el siguiente texto:

«Los productos homeopáticos preparados industrialmente y que se comercializan con indicación terapéutica se someterán a todos los efectos al régimen de medicamentos previstos en esta Ley.»

MOTIVACION

Esta inclusión posibilita la aplicación del régimen legal de los medicamentos a estos productos, cuyo detalle y desarrollo se remitirá a la regulación que operen las futuras directivas comunitarias.

PRIMER FIRMANTE:
Grupo Socialista.

ENMIENDA

Al artículo 63

De modificación.
Del apartado 2 del artículo 63.
El primer párrafo de este apartado 2 se redactará como sigue:

«2. Los ensayos clínicos con medicamentos registrados en España como especialidades farmacéuticas para nuevas dosificaciones, nuevas indicaciones terapéuticas o, en general, condiciones distintas para las que fueron autorizados, requerirán informe del Comité Ético...»

MOTIVACION

El resto del artículo se mantiene igual, pero se amplían los supuestos.

ENMIENDA NUM. 118

PRIMER FIRMANTE:
Grupo Parlamentario Vasco (PNV).

ENMIENDA

Al artículo 34.3

De supresión.
Se propone la supresión del número 3 del artículo 34.

JUSTIFICACION

La calidad de la Fórmula Magistral queda garantizada técnicamente por los otros tres apartados del presente artículo cumplimentados con el artículo 74.2.

Figure 3: Example of a legislative amendment in Spain. This structure is used to track text reuse in rule-based workflows.

4.1.2 Complex Legal Language and Structure

Legal texts use formal and often highly nested language, with cross-references, specialized phrasing, and embedded clauses. Rule-based systems struggle to process and interpret such complexity, especially without semantic understanding or access to document structure. Amendments frequently propose changes across multiple sections of the law, which requires comparing a large number of text segments. This adds substantial complexity to the task and necessitates additional preprocessing steps to prepare the data and define comparison boundaries.

4.1.3 Identifying Amendment Targets

Amendments often refer to the bill text in indirect or abstract terms, such as "the second paragraph of article 3.3.a." Mapping these references to specific segments of text requires deep structural parsing and contextual interpretation—something difficult to accomplish with fixed rules. This was a key motivation for initiating this project with Large Language Models (LLMs): we rely on their ability to interpret the task flexibly and semantically.

4.2 Challenges in Measuring Amendment Success

Once the amendment and its target text have been identified, the next step is to determine whether it passed. Designing a mechanism to compute and represent success is one of the most critical parts of this workflow. Amendments often contain multiple proposals, some of which may be incorporated while others are rejected. This raises a key question: how do we measure partial success?

In our rule-based approach, we relied on calculating the percentage of phrases in the amendment that appeared in the final law. This method has the advantage of depending on exact matching, which aligns with the precision required in legal language. However, percentage-based metrics can be unintuitive and difficult to interpret consistently.

In this first round of evaluation, we opted for a binary classification scheme—passed or not passed—based on a hand-coded sample of 2,035 amendments. This allows for clearer research analysis, validation, and model training.

That said, developing richer representations of amendment success—beyond binary or percentage-based labels—has strong potential to advance both machine learning applications in text analysis and the study of legislative change itself. Most legislative studies rely heavily on proxies to track lawmaking impact, and improving how we measure success directly from text would be a meaningful contribution.

5 Why Use Large Language Models (LLMs)?

Traditional rule-based systems often struggle with the ambiguity, complexity, and variability inherent in legislative texts. These systems require extensive rule engineering and still fall short when confronted with vague references, non-standard phrasing, or structurally complex amendments. In contrast, Large Language Models (LLMs) offer a more flexible, context-aware alternative, particularly well-suited for interpreting natural language in legal and procedural contexts.

LLMs offer several significant advantages for the task of amendment classification. First, they can interpret amendments written in indirect, paraphrased, or fragmented language—something rule-based systems handle poorly. Second, LLMs are robust to structural complexity, including nested references, hierarchical phrasing, and legal cross-citations. These features are especially relevant in legislative texts, where clauses often refer to multiple parts of a bill or law. Third, LLMs reduce the need for handcrafted rules and labor-intensive preprocessing by enabling workflows to be implemented directly through prompt engineering. This not only streamlines the pipeline but also enhances scalability across jurisdictions and amendment formats.

Despite these strengths, LLMs are not without limitations. One major constraint is **prompt capacity**: ideally, the model should receive the full text of the bill, the amendment, the final law, and task instructions. If any of these elements are truncated, the model may lack sufficient context, leading to unreliable classifications. A second challenge is the risk of **hallucination**. While generally robust, LLMs can occasionally generate factually incorrect but plausible-sounding responses. In the legal domain, even minor errors—such as changing the plurality or gender of a word—can significantly alter the meaning of a clause. Additionally, the **opaque nature** of LLM outputs can hinder interpretability, making it difficult for researchers to audit or justify specific decisions.

To address these challenges, we use the Qwen2.5-14B-Instruct-1M model, which supports input sequences of up to one million tokens. This extended context window makes it feasible to include all relevant documents and instructions in a single prompt, minimizing information loss. To reduce hallucinations, we tested various prompts by modifying individual components and comparing outputs. To improve explainability, we also asked the model to provide justifications for its classifications. These justifications were then used to: (1) assess the model's output against a manually coded reference, and (2) compare the model's reasoning with the comments provided by human annotators.

Using this model, we developed a zero-shot classification workflow and tested it on a curated dataset of 5,000 manually coded amendments, focusing on a subset of 2,035 cases that

met input length constraints. This design enables consistent comparisons across amendment types—additions, modifications, and deletions—while preserving interpretability.

The integration of LLMs in this domain not only improves the accuracy of amendment tracking but also opens new avenues for rethinking how legislative change is measured. In the following section, we detail our classification workflow and discuss its performance. Later, we revisit the limitations of this approach and outline potential directions for refining prompts, enhancing transparency, and extending the methodology to broader legislative corpora.

6 First LLM-based Workflow

6.1 Workflow for Zero-Shot Classification Using LLMs

To evaluate the viability of Large Language Models (LLMs) for amendment classification, we developed a structured, modular workflow implemented in Python. This workflow operates on a curated dataset of 2,035 manually coded amendments, each matched to its corresponding legislative text via a shared identifier. The aim is to assess whether each amendment—categorized as an addition, modification, or deletion—was successfully incorporated into the final law.

The workflow proceeds through the following key stages:

- **Amendment Selection:** The pipeline sequentially processes each amendment in the dataset. For each record, the unique identifier is used to retrieve both the corresponding bill (proposed legislation) and the law (enacted text).
- **Preprocessing of Legal Texts:** Before prompt generation, both the bill and law texts undergo a cleaning routine. This includes the removal of HTML tags and formatting artifacts, as well as the standardization of whitespace and character encoding to ensure consistency in the input passed to the language model.
- **Amendment Typing and Prompt Routing:** Each amendment in the dataset includes a pre-parsed classification indicating its type—addition, modification, or deletion. Based on this classification, the workflow dynamically selects one of three tailored prompts specifically designed for the corresponding task type. This routing ensures that the prompt provides the model with both the contextual framing and the linguistic cues best suited to the kind of textual change being evaluated.
- **Prompt Execution:** The selected prompt is then submitted to the LLM (Qwen2.5-14B-Instruct-1M), which processes the task in a zero-shot setting. The model returns both a binary classification (accepted or rejected) and a natural language justification for its response.
- **Result Logging and Evaluation:** Outputs from the model—including predictions and justifications—are stored for each amendment. These are later evaluated against the hand-coded labels to assess model accuracy and understand areas of agreement and divergence. Justifications are also compared with the manual annotations provided by human coders, supporting an audit trail and interpretability analysis.

After multiple iterations of prompt tuning and validation across different amendment categories, we arrived at a final set of optimized prompts. These prompts reflect not only the structural complexity of legal language but also the specific semantic expectations of each amendment type. Notably, we opted not to include worked examples in the prompts. While examples are often helpful in instructional prompting, we found that in our case they introduced semantic noise and consumed valuable context space. Additionally, we adopted an open-ended format for justifications to maintain interpretability and facilitate qualitative review. Below, we present the final versions of the prompts used for each amendment type.

Prompt for Deletion Amendments

```
Compara el texto del proyecto de ley con el texto de la ley final y determina si la
↪ siguiente **enmienda de supresión** ha sido aceptada. Para ello, verifica si **las
↪ palabras o fragmentos que la enmienda propone eliminar han sido efectivamente
↪ suprimidos en el texto final de la ley**.

Devuelve exclusivamente un JSON con esta estructura exacta:
{
"respuesta": 1 si la enmienda ha sido aceptada (el texto ha sido eliminado), o 0 si no
↪ ha sido aceptada (el texto se mantiene),
"parte": Explica brevemente por qué has tomado esa decisión.
}

Sigue estrictamente este formato. No añadas texto fuera del JSON ni modifiques la
↪ estructura.

** Enmienda **
Tipo de enmienda: {amend_type}
Parte del texto que se pretende modificar:
{part}

Texto completo de la enmienda:
{texto}
** Fin de la enmienda **

A continuación, se presentan los textos oficiales:

** Proyecto de ley (versión inicial) **
{bill_text}
** Fin del proyecto de ley **

** Ley final (versión aprobada) **
{law_text}
** Fin de la ley final **
```

Prompt for Modification Amendments

```
Compara el texto del proyecto de ley con el texto de la ley final y determina si la
↪ siguiente **enmienda de modificación** ha sido aceptada. Para ello, identifica **qué
↪ fragmentos del texto original la enmienda propone modificar** y comprueba si dicha
↪ modificación aparece en el texto final.

Devuelve exclusivamente un JSON con esta estructura exacta:
{
"respuesta": 1 si la enmienda ha sido aceptada (la modificación está en la ley final), o
↪ 0 si no ha sido aceptada,
"parte": Explica brevemente por qué has tomado esa decisión.
}

Sigue estrictamente este formato. No añadas texto fuera del JSON ni modifiques la
↪ estructura.

** Enmienda **
Tipo de enmienda: {amend_type}
Motivación:
{motivacion}

Texto completo de la enmienda:
{texto}
** Fin de la enmienda **

A continuación, se presentan los textos oficiales:

** Proyecto de ley (versión inicial) **
{bill_text}
```

```
** Fin del proyecto de ley **

** Ley final (versión aprobada) **
{law_text}
** Fin de la ley final **
```

Prompt for Addition Amendments

```
Compara el texto del proyecto de ley con el texto de la ley final y determina si la
↪  siguiente **enmienda de adición** ha sido aceptada. Para ello, identifica **el
↪  contenido nuevo propuesto en la enmienda** y verifica si ha sido incorporado de
↪  forma íntegra o sustancial al texto final de la ley.

Devuelve exclusivamente un JSON con esta estructura exacta:
{
"respuesta": 1 si la enmienda ha sido aceptada (el nuevo contenido se ha incorporado), o
↪  0 si no ha sido aceptada,
"parte": Explica brevemente por qué has tomado esa decisión.
}

Sigue estrictamente este formato. No añadas texto fuera del JSON ni modifiques la
↪  estructura.

** Enmienda **
Tipo de enmienda: {amend_type}
Motivación:
{motivacion}

Texto completo de la enmienda:
{texto}
** Fin de la enmienda **

A continuación, se presentan los textos oficiales:

** Proyecto de ley (versión inicial) **
{bill_text}
** Fin del proyecto de ley **

** Ley final (versión aprobada) **
{law_text}
** Fin de la ley final **
```

These prompt templates serve as the operational foundation for our classification workflow and are central to both model performance and output interpretability. In the next section, we present a quantitative evaluation of the workflow's classification accuracy, followed by a qualitative analysis of model justifications and disagreement cases.

6.2 Overall Classification Performance

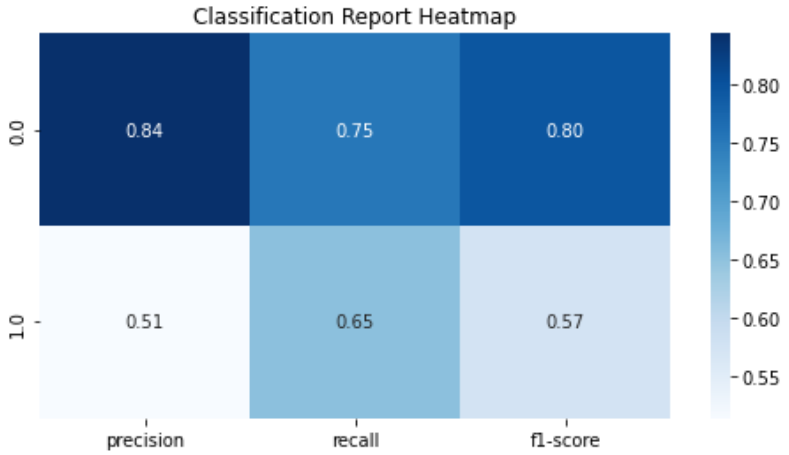


Figure 4: Overall classification metrics across all cases after cleaning.

Figure 4 presents the classification performance of the model on 2,035 samples. The model achieves an overall accuracy of approximately 72%. For class 0.0 (non-passed amendments), it shows strong precision (0.84) and recall (0.75), yielding an F1-score of 0.80. Class 1.0 (passed amendments) shows lower precision (0.51) but higher recall (0.65), with an F1-score of 0.57. These results suggest the model is more precise in identifying failed amendments, while being better at recalling successful ones. The weighted average F1-score is 0.73, reflecting balanced performance despite class imbalance.

6.3 Performance by Amendment Type

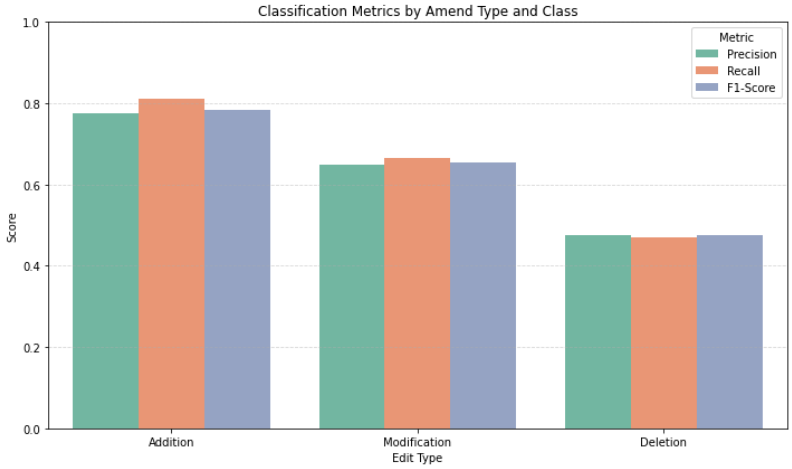


Figure 5: Classification metrics by amendment type: Addition (2.0), Modification (3.0), and Deletion (4.0).

Figure 5 shows model performance disaggregated by amendment type. Additions (type 2.0) show the strongest results with 84% accuracy, class 0.0 reaching an F1-score of 0.89 and class 1.0 achieving 0.68. Modifications (type 3.0) have an accuracy of 67%, with an F1-score of 0.74 for class 0.0 and 0.57 for class 1.0. Deletions (type 4.0) achieve around 70% accuracy overall, but the model struggles with class 1.0 (successful deletions), reflected in a low F1-score of 0.13. Class 0.0 for deletions performs reasonably well, with an F1-score of 0.82. These differences suggest the model performs best with additive amendments and faces challenges with deletion-type amendments, especially when identifying successful cases.

7 Discussion and Future Research

This study set out to evaluate the potential of Large Language Models (LLMs) for the classification of legislative amendments using a zero-shot prompting strategy. While the overall accuracy achieved is comparable to that of existing rule-based methods, the model demonstrates notable improvements in handling certain types of amendments—particularly modifications and additions. More significantly, this approach enables the analysis of deletion amendments, which are often beyond the reach of rule-based systems due to their structural complexity and the difficulty in identifying the target of removal. The ability to classify all three amendment types within a unified framework represents a substantial methodological advantage.

The results obtained in this first round of zero-shot classification are promising, especially given that they were achieved without any model fine-tuning. The labor saved through automation is considerable; manually reviewing thousands of amendments is time-consuming and resource-intensive, whereas the proposed workflow can produce predictions and justifications at scale with minimal human intervention. However, despite these strengths, several important limitations were encountered.

The most persistent technical constraint involves prompt length. In many cases, evaluating whether an amendment has been incorporated into the final law requires comparing long and hierarchically structured legislative texts. Some bills introduce new sections or shift the legal structure in ways that complicate the alignment between the amendment and the final law. When multiple articles are affected, or when the legal hierarchy is altered, the model needs access to a large context to make an informed judgment. Although the Qwen2.5-14B-Instruct-1M model supports a relatively large input size, this was still insufficient for nearly 3,000 amendments in our dataset, which could not be processed because the associated legal texts exceeded the model's input capacity.

Another source of error stems from hallucinations, though a detailed examination suggests that many of these are linked to upstream data quality rather than intrinsic model flaws. In particular, some amendments lack clear internal structure or use language that does not correspond to the type assigned during preprocessing. For example, certain submissions by Members of Parliament do not specify which article or paragraph is being modified, nor do they include the specific text to be changed. Instead, they offer broad descriptions of intent or policy goals. In such cases, the task posed to the model becomes fundamentally different from what the prompt assumes, leading to misclassifications or fabricated completions. This mismatch underscores the importance of well-structured inputs for reliable model behavior.

An additional pattern observed is the model's greater proficiency at identifying rejected amendments—those not reflected in the final law—than accepted ones. This may be partially due to ambiguity in the hand-coded labels or informal drafting practices that obscure the intended changes. Nevertheless, the model's capabilities are more evident when the input data is clean, the amendment is formally articulated, and the legal texts are properly aligned. Under such conditions, the model exhibits a strong ability to compare and reason about differences across legislative versions.

Despite its ability to process deletion amendments—a task that has traditionally required manual inspection or complex rules—the results in this category remain limited. Further improvements in text preprocessing, especially with regard to formatting consistency and section alignment, are likely necessary to improve outcomes.

Another unresolved issue concerns error analysis. While the current workflow logs predictions and justifications, it lacks systematic tools for identifying the sources of incorrect classifications. As a result, we are unable to distinguish clearly whether errors arise from prompt design, data inconsistencies, or limitations in model reasoning.

Looking ahead, several avenues for future research emerge. One natural progression is to fine-tune LLMs on domain-specific tasks using the hand-coded data, particularly to improve the model's ability to parse loosely structured amendments and distinguish between policy descriptions and concrete legal changes. Another promising direction involves the implementation of retrieval-augmented generation (RAG) techniques to dynamically locate and present only the most relevant portions of the bill and law within the prompt. This could reduce prompt size while preserving essential context.

The use of models with even larger context windows—such as those supporting 10,000 tokens or more—could allow for the inclusion of entire legislative documents without truncation, enabling end-to-end comparisons and reducing the need for preprocessing decisions that might omit important text. In parallel, agentic workflows that divide the task into smaller sub-tasks (e.g., amendment type detection, relevant text extraction, and final classification) could offer a more modular and interpretable pipeline, while also enhancing performance through stepwise reasoning.

In conclusion, while challenges remain in terms of input management, hallucination control, and explainability, the use of LLMs for amendment classification appears not only feasible but also highly scalable. With further refinements in prompt design, model selection, and preprocessing strategies, this approach has the potential to transform how legislative change is monitored and analyzed across diverse legal systems.

References

Casas, A., Denny, M. J. and Wilkerson, J. (2020), 'More effective than we thought: Accounting for legislative hitchhikers reveals a more inclusive and productive lawmaking process', *American Journal of Political Science* **64**(1), 5–18.
URL: *https://onlinelibrary.wiley.com/doi/abs/10.1111/ajps.12472*

Engst, B. G. (2021), *The Two Faces of Judicial Power*, Oxford Univ. Press.

Gava, R., Jaquet, J. M. and Sciarini, P. (2021), 'Legislating or rubber-stamping? assessing parliament's influence on law-making with text reuse', *European Journal of Political Research* **60**(1), 175–198.
URL: *https://doi.org/10.1111/1475-6765.12388*

Krehbiel, K. (1998), *Pivotal Politics: A Theory of U.S. Lawmaking*, University of Chicago Press, Chicago.

König, T., Lin, N. and Silva, T. N. (2023), 'Government dominance and the role of opposition in parliamentary democracies', *European Journal of Political Research* **62**(2), 594–611.
URL: *https://doi.org/10.1111/1475-6765.12510*

Martin, L. W. and Vanberg, G. (2005), 'Coalition policymaking and legislative review', *American Political Science Review* **99**(1), 93–106.

Olson, D. M. (1995), *Democratic Legislative Institutions: A Comparative View*, Routledge, London.

Palau, A. M., Casas, A. and Muñoz, L. (2023), 'To amend or not to amend: Under what circumstances do spanish legislators propose amendments to executive bills?', *West European Politics* . Advance online publication.
URL: *https://doi.org/10.1080/01402382.2023.2241253*

Ryan, J. M. (2018), *The Congressional Endgame: Interchamber Bargaining and Resolution*, University of Chicago Press, Chicago, IL.

Wilkerson, J. and Casas, A. (2017), 'Large-scale computerized text analysis in political science: Opportunities and challenges', *Annual Review of Political Science* **20**, 529–544.
URL: *https://doi.org/10.1146/annurev-polisci-052615-025542*

Wilkerson, J., Smith, D. and Stramp, N. (2015), 'Tracing the flow of policy ideas in legislatures: A text reuse approach', *American Journal of Political Science* **59**(4), 943–956.
URL: *https://doi.org/10.1111/ajps.12184*