

Supplementary Material for

Span-Reference and Grounding Reliability in Generative Spanish Clinical Named Entity Recognition: A Validation Study

Eduardo Grande, Rafael Muñoz, Yoan Gutiérrez, and Estela Saquete

Contents

1	Overview	2
2	Exact Prompt Specifications	2
2.1	Primary prompt	2
2.2	Zero- and three-shot construction	3
2.3	Prompt-wording comparison	3
2.3.1	Short formulation	3
2.3.2	Expanded formulation	4
2.4	Position-reference instructions	4
3	Synthetic Output Examples	5
3.1	One entity	5
3.2	Repeated surface forms	5
3.3	Examples rejected during grounding	5
4	Executed Study Settings	6
4.1	Experiment matrices	6
4.2	Models and fine-tuning	6
4.3	Inference and evaluation	8
5	Evaluation Definitions	8
6	Supplementary Analysis Summaries	9
6.1	Pre-test annotation topology	9
6.2	Prompt wording	9
6.3	Zero- and three-shot prompting	9
6.4	Grounding repeated mention strings	10
6.5	Concurrent and serial inference	10
6.6	Completion length	10
7	Guide to Supplementary Data	10

1 Overview

This document provides the supporting information that is most useful to read alongside the article: the exact model-facing prompts, synthetic output examples, executed study settings, operational evaluation definitions, and summaries of the robustness analyses. The accompanying **Supplementary_Data.zip** contains a curated set of source-free CSV and JSON files underlying the manuscript tables, figures, and supplementary analyses. Where the same analysis was available in several export formats, the archive retains one canonical machine-readable source.

Training, inference, parsing, grounding, evaluation, statistical analysis, and table/figure regeneration code are maintained in the versioned public repository cited in the article. The supplementary files contain neither source clinical documents nor raw model generations, because some generated representations reproduce the input text.

The three annotation layers use the following task labels.

Table 1: Annotation tasks and model-facing entity labels.

Task	Label in the task release	English description
DisTEMIST	ENFERMEDAD	Disease
MedProcNER	PROCEDIMIENTO	Procedure
SympTEMIST	SINTOMA	Symptom, sign, or clinical finding

2 Exact Prompt Specifications

2.1 Primary prompt

The primary prompt uses the following shared skeleton. {LABEL} is replaced with the task label, {TEXT} with the clinical document, and {BLOCK} with one of the four output instructions below. The same model-specific chat template wraps the rendered prompt during fine-tuning and inference.

```
Eres un sistema de extracción de entidades clínicas. Extrae todas las menciones
de tipo {LABEL} del siguiente texto clínico.

### TEXTO
{TEXT}

### FORMATO DE SALIDA
{BLOCK}

### RESPUESTA
```

Inline XML (iXML).

```
Devuelve el texto original sin modificar. Inserta etiquetas XML de frontera para cada mención de
↪ tipo {LABEL}: una etiqueta de inicio antes de la mención y una etiqueta de fin después de la
↪ mención, compartiendo el mismo atributo data-span.
```

Mention-list JSON (mJSON).

```
Devuelve JSON válido exactamente con la forma {"entities":[{"text":"...", "label":"..."}]}. No
↪ incluyas texto fuera del JSON.
```

Tab-separated mention list (TSV).

Devuelve una línea por mención con dos columnas separadas por tabulador: texto<TAB>etiqueta. No
↪ incluyas cabecera.

Direct-offset JSON (oJSON).

Devuelve JSON válido exactamente con la forma
↪ {"entities":[{"start":0,"end":10,"text":"...", "label":"..."}]}. Los offsets son índices de
↪ caracteres.

2.2 Zero- and three-shot construction

The unadapted checkpoints used the same model-facing primary query, including the task-specific label, source document, and format block. In zero-shot inference this query was the only user message. In three-shot inference, three preceding user–assistant exchanges were added: each user message was a fully rendered primary query for a training document and each assistant message was that document’s format-specific gold target. The final user query was byte-identical to its zero-shot counterpart. No system message, output repair, retry, or prompt optimisation was added.

Three panels per task were selected with seeds 13, 17, and 23. For each seed, eligible positive training documents were sorted by identifier, shuffled deterministically, and the first three were retained in that order. The same task-specific panel documents and order were used for both models and all four formats; only the assistant target linearisation changed with format. Panel identifiers, hashes, target-fidelity checks, and context-fit records are included in the Supplementary Data. These panel seeds describe variation in demonstrations and are distinct from the fine-tuning seeds.

2.3 Prompt-wording comparison

Two alternative Spanish formulations were evaluated on the DisTEMIST development partition for one training seed. The entity definition, target representations, parsers, and grounding rules were unchanged. These formulations were used only to test whether the main pattern depended on one wording of the instruction.

2.3.1 Short formulation

Extrae las menciones de tipo {LABEL} del texto clínico.

Texto:

{TEXT}

Formato:

{BLOCK}

Respuesta:

Table 2: Output blocks used with the short formulation.

Format	Exact output block
iXML	Copia el texto sin cambios e inserta etiquetas XML de frontera (inicio y fin, con el mismo data-span) alrededor de cada mención.
mJSON	Devuelve solo JSON: {"entities":[{"text":"...", "label":"..."}]}.
TSV	Una mención por línea: texto y etiqueta separados por un tabulador. Sin cabecera.

Format	Exact output block
oJSON	Devuelve solo JSON: {"entities":[{"start":0,"end":10,"text":"...", "label":"..."}]}. start/end son índices de carácter.

2.3.2 Expanded formulation

```
Actúas como un sistema experto de extracción de entidades clínicas en español. Identifica y
↪ extrae todas las menciones de tipo {LABEL} en el siguiente texto clínico, sin omitir
↪ ninguna.

### TEXTO CLÍNICO
{TEXT}

### INSTRUCCIONES DE FORMATO
{BLOCK}

### RESPUESTA
```

Table 3: Output blocks used with the expanded formulation.

Format	Exact output block
iXML	Reproduce el texto original carácter por carácter, sin añadir ni quitar nada. Alrededor de cada mención inserta una etiqueta XML de inicio justo antes y una de fin justo después, ambas con el mismo atributo data-span para emparejarlas.
mJSON	Devuelve únicamente un objeto JSON válido con la forma exacta {"entities":[{"text":"...", "label":"..."}]}. No incluyas ningún texto fuera del JSON.
TSV	Devuelve TSV con una mención por línea y dos columnas: el texto de la mención y su etiqueta, separados por un carácter de tabulación. Sin línea de cabecera.
oJSON	Devuelve únicamente un objeto JSON válido con la forma exacta {"entities":[{"start":0,"end":10,"text":"...", "label":"..."}]}. start y end son los índices de carácter (base cero) de inicio y fin de la mención.

2.4 Position-reference instructions

The diagnostic study retained the primary skeleton and compared the original oJSON instruction with the following two reference mechanisms.

Fully specified direct offsets.

```
Devuelve únicamente JSON válido con la forma
↪ {"entities":[{"start":0,"end":5,"text":"dolor", "label":"{LABEL}"}]}. Los índices cuentan
↪ puntos de código Unicode desde cero y se refieren al bloque de texto exacto y sin modificar
↪ mostrado bajo ### TEXTO. start es inclusivo y end es exclusivo. El texto debe coincidir
↪ carácter por carácter con TEXTO[start:end], incluidos espacios, saltos de línea, puntuación
↪ y acentos. Ejemplo sintético: para el texto «dolor leve», la mención «dolor» se representa
↪ como {"start":0,"end":5,"text":"dolor", "label":"{LABEL}"}. No incluyas texto fuera del JSON.
```

Occurrence-index JSON (occJSON).

Devuelve únicamente JSON válido con la forma

↪ `{"entities":[{"text":"dolor","label":"{LABEL}","occurrence":0}]}`. occurrence es el índice desde cero de la aparición exacta de text en el bloque de texto sin modificar, contando de izquierda a derecha. Si «dolor» aparece dos veces, sus índices son 0 y 1. text debe copiarse carácter por carácter, incluidos espacios, saltos de línea, puntuación y acentos. No incluyas texto fuera del JSON.

3 Synthetic Output Examples

All examples in this section are invented and contain no corpus text.

3.1 One entity

For the source *dolor óseo*, the entity *dolor óseo* occupies Unicode code-point positions 0–10 and has label SINTOMA.

Table 4: Valid targets for one synthetic entity.

Format	Target
iXML	<code><SINTOMA data-span="s0000" data-boundary="start"/>dolor óseo<SINTOMA data-span="s0000" data-boundary="end"/></code>
mJSON	<code>{"entities":[{"text":"dolor óseo","label":"SINTOMA"}]}</code>
TSV	<code>dolor óseo<TAB>SINTOMA</code>
oJSON	<code>{"entities":[{"start":0,"end":10,"text":"dolor óseo","label":"SINTOMA"}]}</code>
occJSON	<code>{"entities":[{"text":"dolor óseo","label":"SINTOMA","occurrence":0}]}</code>

3.2 Repeated surface forms

For the source *dolor y dolor*, both occurrences are entities. mJSON and TSV repeat the surface form; count-matched grounding assigns the first emitted row to the first exact occurrence and the second row to the second. occJSON identifies the two locations directly with occurrence numbers 0 and 1.

```
mJSON: {"entities":[{"text":"dolor","label":"SINTOMA"}, {"text":"dolor","label":"SINTOMA"}]}

TSV:   dolor<TAB>SINTOMA
       dolor<TAB>SINTOMA

oJSON: {"entities":[{"start":0,"end":5,"text":"dolor","label":"SINTOMA"}, {"start":8,"end":13,"text":"dolor","label":"SINTOMA"}]}

occJSON: {"entities":[{"text":"dolor","label":"SINTOMA","occurrence":0}, {"text":"dolor","label":"SINTOMA","occurrence":1}]}
```

3.3 Examples rejected during grounding

Both outputs below are syntactically valid JSON and satisfy the expected field types, but neither produces a scored span. In the first, the emitted text does not equal the specified source slice. In the second, the requested end position exceeds the length of the synthetic source *dolor*.

```

{"entities":[{"start":0,"end":5,"text":"fiebre","label":"SINTOMA"}]}

{"entities":[{"start":99,"end":104,"text":"dolor","label":"SINTOMA"}]}

```

4 Executed Study Settings

4.1 Experiment matrices

Table 5: Executed experiment matrices. The diagnostic confirmation partition was not used to train the diagnostic adapters and the official test set was not used in that study component. Prompted baselines used unadapted checkpoints and therefore required no additional adapters.

Component	Conditions	Tasks	Model/seed combinations	Adapters
Primary comparison	iXML, mJSON, TSV, oJSON	3	2 models $\times$ 3 seeds	72
Prompted baselines	Zero-shot and three-shot iXML, mJSON, TSV, oJSON	3	2 models $\times$ 1 or 3 panels	0
Initial position-reference comparison	Original and fully specified offsets on 3 tasks; occurrence index on DisTEMIST	3	2 models $\times$ 3 seeds	42
Descriptive cross-layer occurrence comparison	Occurrence index on MedProcNER and SympTEMIST	2	2 models $\times$ 3 seeds	12

The primary adapters use 675 training documents and the 75-document development partition used during study preparation. They are evaluated on the 250-document official test partition. The prompted baselines use the same test documents, final queries, output limits, parsers, grounding rules, and scorers: 24 zero-shot cells were run once and 24 three-shot task–model–format conditions were run with each of three fixed panels, giving 72 three-shot cells. For the position-reference comparison, the same 675-document pre-test pool is divided into 600 training documents and two fixed 75-document partitions: the original development partition and an internal confirmation partition.

4.2 Models and fine-tuning

Table 6: Model and fine-tuning settings.

Setting	Executed value
Base models	Qwen/Qwen2.5-3B-Instruct, revision aa8e72537993ba99e69dfaafa59ed015b17504d1; meta-llama/Llama-3.2-3B-Instruct, revision 0cb88a4f764b7a12671c53f0838cd831a0843b95
Training precision and quantisation	bfloat16 computation; 4-bit quantisation
Maximum training sequence length	8,192 tokens
Optimisation	Five epochs; batch size 1; gradient accumulation 8; learning rate 2×10^{-4} ; warm-up ratio 0.03; no weight decay
Loss and packing	Completion-only loss; packing disabled
LoRA	Rank 64; alpha 128; dropout 0.05; target modules <code>q_proj</code> , <code>k_proj</code> , <code>v_proj</code> , <code>o_proj</code> , <code>gate_proj</code> , <code>up_proj</code> , and <code>down_proj</code>

Setting	Executed value
Training seeds	13, 17, and 19
Chat templates	Tokenizer-provided model-specific templates; identical template and tokenizer used for fine-tuning and inference

4.3 Inference and evaluation

Table 7: Inference and primary evaluation settings.

Setting	Executed value
Decoding	Greedy (temperature 0; top-p 1); decoding seed 1729
Serving conditions	Up to 32 concurrent sequences for the primary analysis; one sequence for the robustness analysis
Generation limits	Task-model-format-specific; development-target 99th percentile plus 20%
Prompted checkpoints	The same unadapted Qwen and Llama model revisions; no fine-tuned adapter loaded
Prompted replication	One deterministic zero-shot run per task-model-format cell; three fixed training-derived demonstration panels per three-shot cell
Primary metric	Strict-span micro-precision, recall, and F_1 from the corresponding task-specific scorer
String grounding	Count-matched exact search for mJSON and TSV
Output handling	Deterministic parsing and grounding; no fuzzy matching, output repair, generation retry, or model-based post-processing
Uncertainty	2,000 paired document-bootstrap resamples for confidence intervals; an additional analysis resamples documents and training seeds
Hypothesis tests	10,000 paired randomisations for 12 specified tests; Holm adjustment across the 12-test family
Practical threshold	Absolute F_1 difference of 0.02, used as an engineering threshold rather than a clinically validated value

5 Evaluation Definitions

Table 8: Operational definitions and denominators.

Term	Operational definition	Denominator
Syntax-valid output	Balanced/tag-form output, well-formed JSON, or two fields on every non-empty TSV line, as appropriate.	All evaluated documents
Schema-accepted output	Surface syntax and the expected outer structure, field types, and label vocabulary are accepted.	All evaluated documents
Fully parsed output	The deterministic parser accepts the complete output.	All evaluated documents
Partially accepted output	The complete output is rejected but at least one candidate is retained.	All evaluated documents
Parsed candidate	One entity item accepted by the format-specific parser.	Candidates extracted from generated outputs
Grounded candidate	A parsed candidate aligned to one exact source span using tag position, validated offsets, occurrence reference, or the declared exact-string rule.	Parsed candidates
Strict true positive	A grounded candidate whose start, end, and label match a gold annotation under the task-specific scorer.	Scored predictions and gold annotations
Boundary error	A one-to-one same-label prediction overlapping an unmatched gold span without matching both boundaries.	Parser-accepted candidates
Wrong occurrence	The emitted surface is present but is grounded to a different occurrence than the gold span.	Parser-accepted candidates

Term	Operational definition	Denominator
Omission	A gold entity left unmatched after strict and boundary matching.	Gold entities

6 Supplementary Analysis Summaries

6.1 Pre-test annotation topology

Table 9: Annotation topology in the training and development partitions. The count-matched ceiling is the maximum exact-span recovery obtained when repeated mention strings are assigned to source occurrences from left to right.

Task	Split	Docs	Entities	Repeated (%)	Nested	Crossing	Count-matched ceiling (%)
DisTEMIST	Train	675	7,193	19.0	157	14	94.9
DisTEMIST	Development	75	872	20.4	12	2	96.3
DisTEMIST	Combined	750	8,065	19.1	169	16	95.1
SympTEMIST	Train	675	8,119	11.8	8	20	96.7
SympTEMIST	Development	75	973	12.4	0	2	96.6
SympTEMIST	Combined	750	9,092	11.8	8	22	96.7
MedProcNER	Train	675	9,875	17.2	160	37	96.2
MedProcNER	Development	75	1,190	17.6	20	4	96.1
MedProcNER	Combined	750	11,065	17.2	180	41	96.1

6.2 Prompt wording

Table 10: Range of DisTEMIST development F_1 across the primary, short, and expanded prompt formulations. Each range covers one seed and both models.

Format	Minimum F_1	Maximum F_1
iXML	0.707	0.753
mJSON	0.698	0.741
TSV	0.708	0.746
oJSON	0.002	0.007

The ordering of iXML, mJSON, and TSV varied with wording and model, but direct offsets remained near zero under all three formulations. The complete 24-cell results are provided in the Supplementary Data file `prompt_wording_sensitivity_summary.csv`.

6.3 Zero- and three-shot prompting

Across the 24 task-model-format conditions, zero-shot strict F_1 ranged from 0 to 0.131. Three-shot prompting raised mention-list performance: mJSON achieved 0.165–0.348 and TSV 0.203–0.386 when averaged across the three panels. In contrast, iXML remained at 0–0.028 and oJSON at zero. Every descriptive paired bootstrap interval for prompted minus fine-tuned F_1 was below zero. These comparisons use the same official test documents, but they estimate different sources of variation: fine-tuned values vary over three training seeds, whereas three-shot values vary over three demonstration panels.

The demonstrations substantially improved structural compliance for mJSON and TSV, especially for Llama, but did not establish reliable source copying or numerical span references. Generation stopped at the configured length limit in 956 of 6,000 zero-shot outputs (15.9%) and 2,096 of 18,000 three-shot outputs (11.6%). The complete source-free run metrics, aggregates, descriptive contrasts, panel records, and context checks are supplied in the three `prompted_baseline_*.json` files.

6.4 Grounding repeated mention strings

Relative to count-matched grounding, always assigning each emitted surface to its first source occurrence reduced strict F_1 by 0.023–0.034 across the primary task–model cells. Assigning an emitted surface to every source occurrence increased F_1 by 0.001–0.007. These policies change the false-positive/false-negative trade-off and are reported as sensitivity analyses rather than alternative primary scores. Complete results are provided in the Supplementary Data file `grounding_sensitivity_summary.csv`.

6.5 Concurrent and serial inference

Across the 24 task–model–format conditions, the absolute difference between seed-averaged concurrent and serial strict F_1 was at most 0.0025 (mean 0.0008). Across 18,000 paired document outputs, 71.5% of raw generations, 92.8% of grounded span sets, and 93.8% of document-level TP/FP/FN counts were identical. These results support stability of the aggregate conclusions under the two tested serving settings; they do not imply byte-identical generation. Detailed results are provided in the Supplementary Data files `concurrency_robustness.csv` and `concurrency_identity.csv`.

6.6 Completion length

Generation stopped at the configured length limit in 94 of 18,000 primary concurrent outputs (0.52%): 7 iXML, 22 mJSON, 24 TSV, and 41 oJSON outputs. No completion reached 90% of either model’s context window. Completion-length stopping therefore cannot account for the near-zero direct-offset result. The audit cannot distinguish every possible end-of-generation condition recorded under a generic normal stop.

7 Guide to Supplementary Data

`Supplementary_Data.zip` contains CSV and JSON supplementary files that support the manuscript. A README inside the package explains its contents and groups the files by analysis. The package does not include clinical documents or raw model generations.

The principal machine-readable files are grouped as follows:

- **Primary strict scores and uncertainty:** `run_scores.csv`, `bootstrap_cis.csv`, `delta_f1_forest.csv`, `primary_seed_ranges.csv`, `primary_two_stage_cis.csv`, and `primary_multiplicity.csv`.
- **Reliability diagnostics:** `primary_diagnostic_taxonomy_formats.csv`, `primary_diagnostic_taxonomy_runs.csv`, `primary_diagnostic_taxonomy_stratified.csv`, `primary_stratified_results.csv`, `table5_diagnostics.csv`, `offset_decomposition.csv`, and `offset_taxonomy_confirmation.csv`.
- **Position-reference study:** `position_reference_initial_development_runs.csv`, `position_reference_initial_confirmation_runs.csv`, `occurrence_index_extension_development_runs.csv`, `occurrence_index_extension_confirmation_runs.csv`, and `position_reference_comparison.json`, which contains the matched contrasts and confidence intervals.
- **Prompted baselines:** `prompted_baseline_results.json`, `prompted_baseline_demonstrations.json`, and `prompted_baseline_context_audit.json`.
- **External validation:** `external_validation_results.json`, `external_validation_data_profile.json`, and `external_validation_context_audit.json`.
- **Robustness analyses:** `prompt_wording_sensitivity_summary.csv`, `grounding_sensitivity_summary.csv`, `concurrency_robustness.csv`, and `concurrency_identity.csv`.
- **Data and computation summaries:** `dataset_profiles.json`, `annotation_topology.json`, `shared_document_comparison.json`, `target_reconstruction_check.json`, `primary_compute_exposure.csv`, and `final_inference_lengths.json`.
- **Table and figure source data:** `table4_strict_f1.csv`, `table5_diagnostics.csv`, `prompted_baseline_results.json`, `external_validation_results.json`, `delta_f1_forest.csv`, and the position-reference and taxonomy files listed above.

The GitHub repository cited in the manuscript contains the code, configuration files, and scripts needed to interpret the Supplementary Data and reproduce the analyses.