Structured Drafting and Evidence-Grounded Refinement for the LT4CPR 2026 Shared Task

Wataru Sakata
Stockmark
wataru.sakata@stockmark.co.jp

Wataru Hirota
Stockmark
wataru.hirota@stockmark.co.jp

Abstract

Generating crisis situation reports from numerous short, redundant tweets requires both information organization and support from cited evidence. We present a system for the LT4CPR Shared Task that combines structured draft generation with bidirectional evidence-grounded refinement. Draft generation separates claim extraction, section-wise aggregation, report assembly, and summary generation. Refinement collects report-driven correction candidates and tweet-driven additions, verifies them against source tweets, and integrates accepted edits. Across all 23 cells covering two public development crises, our structured-generation configuration achieved a higher Main Score than direct generation (0.5308 vs. 0.4198). Adding bidirectional refinement slightly reduces Main Score to 0.5216 but increases the automatic NLI-based source-support metric for body bullets from 0.7925 to 0.8566. Our submission, combining both components, ranked first among the nine participating teams in the official test evaluation with a primary score of 0.682. A model-based audit of 50 randomly sampled modified body bullets judged 35 edits beneficial, eight equivalent, and seven detrimental, suggesting that bidirectional refinement improves statement quality in many edits. However, the audit and NLI disagree on individual improvements, suggesting that increases in the NLI metric should not be interpreted directly as equivalent gains in factual source support.

1 Introduction

During a crisis, situational awareness requires reports that organize information about damage, evacuations, and response activities from numerous short social media posts. The LT4CPR Shared Task (Xia et al., 2026) takes collections of tweets as input and requires structured situation reports with citations to supporting tweets.

Generating such reports requires consolidating duplicate accounts and numerical updates while retaining important information and avoiding assertions that go beyond the sources. Attaching citations alone does not ensure that the cited sources support the full content of a statement.

LLMs may underutilize information in long inputs when summarizing (Ravaut et al., 2024). Approaches to addressing this challenge include generating summaries through staged processing of the input (Zhang et al., 2022) and revising generated outputs against source texts or retrieved evidence (Zhu et al., 2021; Gao et al., 2023; Kirstein et al., 2025). We combine these two approaches in a system comprising *Structured Draft Generation* and *Bidirectional Evidence-Grounded Refinement* to address this task (Figure 1). The former progressively structures information by separating claim extraction from tweets, section-wise aggregation, report assembly, and generation of an overview and timeline. The latter collects edit candidates in two directions: report-to-tweet correction, which starts from draft statements to identify errors, and tweet-to-report recovery, which starts from tweets to identify missing information. Candidates are verified and integrated using the source tweets before being applied to the draft.

In our experiments, we compare direct generation, structured draft generation, and bidirectional refinement applied to each draft across all 23 cells covering two crises in the public development set. Our structured-generation configuration achieved higher agreement with reference reports than direct generation. Our submitted system, combining both components, ranked first among the nine participating teams in the official test evaluation (Section 6.2). For structured drafts, bidirectional refinement slightly reduces Main Score while increasing the automatic NLI-based source-support metric. In addition, through these comparisons, ablations, and a model-based audit of randomly sampled modifications, we analyze the effects of both components on reference agreement and source support, exam-

Proceedings of the First Workshop on Language Technologies for Crisis Preparedness and Response, pages 141–155
November 9, 2026 ©2026 Association for Computational Linguistics

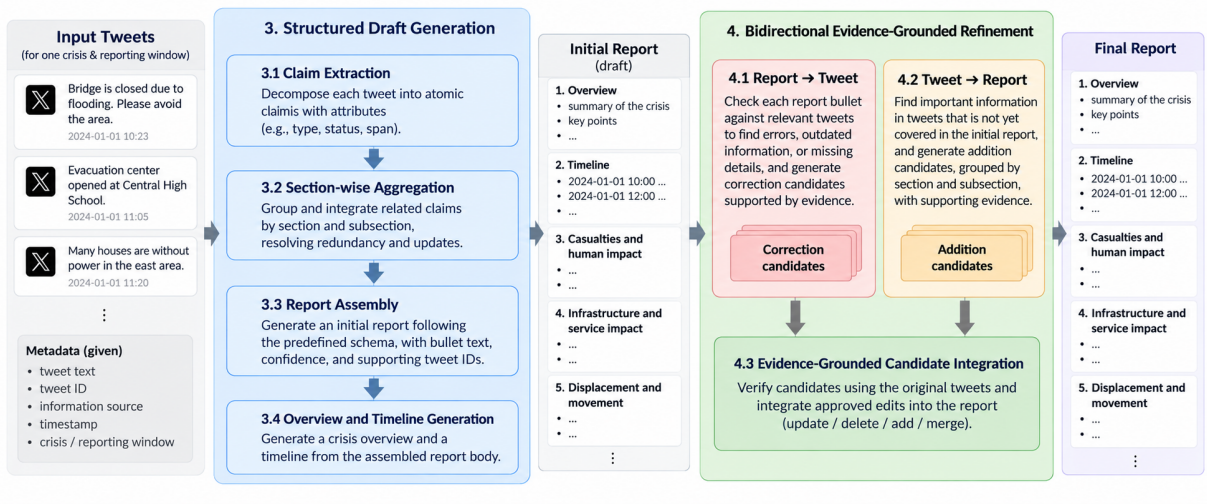


Figure 1: System overview: structured draft generation (Section 3) produces an initial report from the input tweets, and bidirectional evidence-grounded refinement (Section 4) collects correction and addition candidates from both report-to-tweet and tweet-to-report directions and integrates them into the final report.

ining disagreements with the automatic NLI metric and remaining challenges in information integration and unsupported relations.

2 Task Overview

The LT4CPR Shared Task requires concise, structured situation reports that can be verified against the input crisis-related social media messages. We follow the official task guidelines for the input and output formats and report schema.[1]

The data cover multiple crises, each divided into reporting windows. We refer to an independent input and report-generation unit corresponding to a crisis, a reporting window, and a replicate as a *cell*. Each cell provides the tweets within that period and crisis/window metadata. Reports are generated independently of other cells.

The output report consists of sections and subsections defined by the official schema. We refer to report Sections 3–11, which contain the principal factual information, as the *report body*. Each subsection contains one or more bullets, each with statement text, a confidence label, and supporting tweet IDs.

In addition to the body, the system generates an overview summarizing the report and a timeline organizing key events chronologically. We distinguish the generation and refinement of report Sections 3–11 from the generation of the overview and timeline.

3 Structured Draft Generation

To reduce omissions, duplication, and conflation when organizing information distributed across many tweets, we use a multistage pipeline that separates claim extraction, section-wise aggregation, report assembly, and summary generation. We describe each component below.

3.1 Claim Extraction

This stage extracts structured atomic claims from the input tweets, decomposing their factual content into units that can be independently integrated and verified in later stages (Table 1). Tweets within each reporting window are split into batches of at most 80 in input order and processed by an LLM. Each claim includes a source tweet ID, event type, section, subsection, event key, claim text, numerical figure, and status.

To assign claims consistently, we first ask the LLM to identify the type of event being reported and then output its corresponding section, rather than selecting a section directly. The extraction prompt includes a mapping between the 29 event types defined in the official taxonomy guide[2] and their sections. For example, a reported death is classified as KILL and assigned to report Section 3 (Casualties), while an evacuation is classified as EVACUATE and assigned to report Section 5 (Displacement).

[1] https://github.com/LT4CPR/2026_aacl_workshop/blob/c8bd925e1c21d82248b98c5faabb75146418966f/shared_task/data/DATA_FORMAT.md

[2] https://github.com/LT4CPR/2026_aacl_workshop/blob/c8bd925e1c21d82248b98c5faabb75146418966f/shared_task/data/TAXONOMY_GUIDE.md

We extend this mapping to subsections by identifying the placement of each event type in the training gold reports. In Section 6.4, we compare this configuration with one in which the mapping is omitted and the LLM selects sections and subsections directly.

When a tweet describes multiple distinct event types, each is extracted as a separate claim. The tweet in Table 1 yields two claims: identification of the chemical (HAZARDOUS_CONDITION, assigned to report Section 6, Hazard assessment) and designation of an evacuation zone (EVACUATE, assigned to report Section 5, Evacuations).

3.2 Section-wise Aggregation

Fine-grained claims extracted from individual tweets often include duplicate reports of the same event and numerical updates. This stage consolidates these claims and generates subsections (Table 2). Specifically, the claims from Section 3.1 are grouped by report section (3–11), and the LLM generates subsections and their report bullets using a fixed subsection vocabulary. The prompt includes the following instructions:

- **Duplicate event aggregation:** Merge duplicate accounts of the same event into a single bullet.
- **Numerical update handling:** Treat updates as separate bullets for each value rather than as duplicates. If a later report explicitly corrects an earlier value, describe it as a correction. Do not add the values together.
- **Length control:** Control report length using explicit target bullet counts in the prompt. These targets depend on the number of input tweets and are derived from training-report statistics.
- **Gold report style examples:** Provide gold reports from the training data as examples of the desired writing style.

3.3 Report Assembly

To ensure consistent application of the required output schema and confidence labels, we assemble the generated bullets into a report using deterministic rules. Each bullet retains its supporting tweet IDs and receives a confidence label based on the status of its content and the number of supporting tweets.

3.4 Overview and Timeline Generation

Finally, an LLM generates an overview and timeline from the completed body to provide high-level summaries of the detailed report. If output validation detects a structural issue in either component, the LLM regenerates that component while keeping the body fixed. Raw tweets are not supplied again at this stage.

4 Bidirectional Evidence-Grounded Refinement

Although structured draft generation produces drafts containing the main information, LLM-based extraction and aggregation can leave statements that misrepresent source tweets or omit important information. To detect errors and omissions in complementary ways, we collect edit candidates in both the report-to-tweet and tweet-to-report directions, verify them against source tweets, and integrate them into the body in a single pass.

4.1 Report-to-Tweet Correction Candidate Generation

This stage processes each draft bullet sentence by sentence, using the sentence to retrieve related tweet evidence and existing report bullets. These are used to check factual errors and within-report duplication and to propose corrections where needed. We retrieve two types of related information for each target bullet:

- **Related tweets:** Given the target bullet and the tweets it originally cites, the LLM generates three queries to search for supporting evidence, counterevidence, and related or updated information. Together with the bullet text itself, these form four BM25 (Robertson and Zaragoza, 2009) queries over tweets in the same cell, retrieving up to six tweets per query. Retrieved tweets serve as evidence for checking and correcting the target bullet.
- **Related report bullets:** Using the target bullet text as a BM25 query, we retrieve up to eight other bullets from the draft to check for duplication and whether one statement subsumes another.

The LLM receives the target bullet, its cited tweets, retrieved tweets, and related report bullets, and proposes one of three actions: *keep* the bullet unchanged, *replace* it using tweet evidence,

Input tweet [39]: *RT @northlinefreigh: WCEM just confirmed the leaking tank car is carrying anhydrous ammonia. Evacuation zone is 800 metres, please stay clear #ArdleyJunction #WexfordDerailment http://bit.ly/z6WHd6*

	Claim 1	**Claim 2**
event type	HAZARDOUS_CONDITION	EVACUATE
section / subsection	6 / Hazard assessment	5 / Evacuations
event key	chemical-identification	evacuation-radius
claim text	WCEM confirmed the leaking tank car was carrying anhydrous ammonia.	The evacuation zone was set at 800 metres for the Ardley Junction incident.
figure (type)	—	800 (radius_m)
status	reported	reported

Table 1: Example of claim extraction (dev cell `chemical.W1.k1`, tweet ID 39). The tweet reports two distinct events—the identification of the chemical and the setting of the evacuation zone—so each is extracted as a separate claim.

Input claims (excerpt)	**Generated bullets**
[18] WCEM ordered a precautionary evacuation within 800 metres of the Ardley Junction rail yard. [39] The evacuation zone was set at 800 metres for the Ardley Junction incident. (similar claims from 18 tweets in total: 18–78)	Wexford County Emergency Management ordered a precautionary evacuation within 800 metres of the Ardley Junction rail yard. *[confirmed]* cites [18, 26, 30, 31, 32, 39]
[80] The evacuation zone was extended to 1.6 kilometres for the Wexford derailment. (similar claims from 10 tweets in total: 79–97)	The evacuation zone was extended to 1.6 kilometres for the anhydrous ammonia release at Ardley Junction. *[confirmed]* cites [79, 80, 82, 83, 86, 87]

Table 2: Example of section-wise aggregation (dev cell `chemical.W1.k1`, section 5 Evacuations). Eighteen duplicate reports of the 800 m evacuation radius are merged into a single bullet, while the extension to 1.6 km becomes a separate bullet as a numerical update.

or *remove* it if it is unnecessary, for example because it duplicates another bullet or lacks supporting tweets.

An LLM judge then verifies the proposal, checking whether (1) the cited tweets support the entire revised statement, (2) useful information from the original bullet is preserved, and (3) the edit introduces unnecessary duplication with existing bullets. Only proposals that pass these checks are retained as correction candidates. The draft remains unchanged at this stage; edits are applied during candidate integration.

4.2 Tweet-to-Report Missing Information Recovery

The procedure in Section 4.1 generates correction candidates from existing report bullets and does not systematically search for information without a corresponding draft statement. This stage instead starts from tweets to identify information missing from the draft and propose additions.

Tweets in the same cell are processed in input order in batches of 20. The LLM classifies each tweet as *relevant*, *irrelevant*, or *uncertain* with respect to the crisis. For each relevant or uncertain tweet, it also extracts up to four claims, as in Section 3.1. To efficiently filter out information already present, extracted claims are compared with the full draft in batches of 16, and clearly covered claims are discarded. For each remaining claim, BM25 retrieves the top 12 related tweets and the top eight related report bullets.

The LLM receives the target claim, its originating tweet, retrieved tweets, tweets cited by the related report bullets, and the full draft. It classifies the claim as *covered* (adequately represented), *partial* (partly represented, with information still to add), *missing* (absent), or *not_relevant* (unnecessary for the report).

For partial or missing claims, the LLM proposes an additional bullet with a section, subsection, statement text, confidence label, and supporting tweet IDs. An LLM judge checks source support, duplication or contradiction with existing content, and confidence, among other criteria. Only proposals that pass verification are retained as addition candidates. As in Section 4.1, no changes are made to the draft at this stage.

4.3 Evidence-Grounded Candidate Integration

This stage combines the correction candidates from Section 4.1 and addition candidates from Section 4.2 and applies accepted edits to the draft. For each report section, the LLM receives all candidates for that section and their related source tweets, with the full report as context. It proposes at most six edits confined to the target section, accounting for duplication among candidates and their relationship to existing bullets.

Allowed edits include replacing or removing existing bullets, adding new bullets, and merging multiple bullets. Candidates are treated as editing hypotheses; source tweets provide the factual evidence. An LLM judge checks the proposed edits again. Only verified edits are applied, and existing bullets are retained when an edit is rejected. Table 3 illustrates candidate deduplication at this stage.

After all sections have been edited, the system validates bullet IDs, the schema, and citations. Finally, it regenerates the overview and timeline from the finalized body (report Sections 3–11) to produce the final report.

5 Experimental Setup

5.1 Data

We use the public training and development data from the LT4CPR Shared Task. We evaluate all configurations on the same 23 development cells, covering two crises: a chemical incident and an earthquake.

Training gold reports are used to construct the event-type-to-subsection mapping, set target bullet counts according to input tweet counts, and provide style examples during aggregation.

We limited the use of development data during system development where possible, but used some of it to improve the system. The results are therefore not fully independent of development and may overestimate performance on unseen data.

5.2 Evaluation Metrics

We use the official Main Score and Alignment F1 to evaluate agreement with reference reports. We additionally use NLI Support Rate as an unofficial auxiliary metric to assess whether cited tweets support the generated bullets.

Main Score is the arithmetic mean of BERTScore F1 (Zhang et al., 2020) and BLEURT (Sellam et al., 2020) between the generated and reference reports. Alignment F1 aligns their content to evaluate the balance between reference agreement, extraneous content, and omissions.

NLI Support Rate is the proportion of body bullets (report Sections 3–11) whose entire content is classified as `supported` by the cited tweets. We compute it for each cell using the diagnostic procedure in the shared-task development utilities.[3] Specifically, the diagnostic procedure decomposes each bullet into atomic claims, uses its cited tweets as the premise and each claim as the hypothesis for NLI, and counts a bullet as fully supported only when every claim is entailed. This is an auxiliary measure of automatically assessed source support, not a measure of factual accuracy established by human verification. Section 6.3 separately examines the relationship between this metric and source-support judgments through a model-based audit.

All metrics are aggregated using hierarchical averaging over replicates, then reporting windows, and finally crises.

5.3 Direct Generation Baseline

We use Direct Generation as a baseline. It receives all raw tweets in a cell, including tweet IDs, sources, and timestamps, together with crisis information and the reporting window, and generates the report in a single step following the common final output schema. It does not perform claim extraction or section-wise aggregation. Across the 23 evaluated cells, the maximum tweet count was 997 (earthquake.W4.k1), and the longest complete prompt contained 50,623 tokens (chemical.W4.k1), including instructions, schema, and the style example. Even with the 12,000-token output allowance, it fit within the 65,536-token context; no input tweets were truncated in any cell.

We provide one gold report from the training data, volcano.W2.k1, as a style example and instruct the model to write each bullet as an atomic sentence of approximately 10–20 words. Unlike Structured Generation, Direct Generation receives neither target bullet counts based on the number of input tweets nor section-specific length budgets. The LLM also generates confidence labels according to the criteria in the prompt.

[3] https://github.com/LT4CPR/2026_aacl_workshop/blob/c8bd925e1c21d82248b98c5faabb75146418966f/shared_task/tools/other_tools/audit_faithfulness.py

Existing content	Candidates	Integration result
Confirmed injuries near St. Marren ([3b.1]).	Four near-duplicate additions reporting injuries at the hall, all unconfirmed and citing [32, 34, 38, 47, 59].	Preserve [3b.1]; merge the four candidates into one new unconfirmed bullet with the same citations.

Table 3: Candidate deduplication during integration (`earthquake.W1.k1`, section 3 Injuries).

Although the initial output includes an overview and timeline, these components are subsequently regenerated and validated with the body held fixed, generally using the same procedure as Structured Generation.

Both methods use the same model and generation settings, detailed in Section 5.4. This comparison evaluates the draft-generation methods as a whole, including differences in length control and confidence assignment.

5.4 Implementation Details

We use Gemma-4-31B-it (Gemma Team, 2026) for all LLM operations in draft generation and bidirectional refinement. Candidate generation and LLM judging use the same model through separate requests; judging is therefore not evaluation by an independent model.

Appendix A provides implementation details, including generation-time model settings, retrieval settings, processing limits, retry behavior, and output validation for the overview and timeline. Appendix C provides an overview of the prompts for each LLM call.

6 Results and Analysis

6.1 Overall Results

Table 4 presents Direct Generation, Structured Generation, and Bidirectional Refinement applied to each. The Structured Generation configuration achieved higher scores than Direct Generation on all metrics: Main Score increases from 0.4198 to 0.5308, Alignment F1 from 0.1665 to 0.2496, and NLI Support Rate from 0.2938 to 0.7925.

The effects of Refinement differed by draft-generation method. Applied to Direct Generation, Refinement increased all three metrics: Main Score from 0.4198 to 0.4511, Alignment F1 from 0.1665 to 0.1775, and NLI Support Rate from 0.2938 to 0.6525. Applied to Structured Generation, it increased NLI Support Rate from 0.7925 to 0.8566 but decreased Main Score from 0.5308 to 0.5216 and Alignment F1 from 0.2496 to 0.2291.

Thus, Refinement consistently improved NLI-based source support, while its effects on agreement with reference reports depended on the draft-generation method. In particular, for Structured Generation, which already achieved high reference agreement, the source-support gains from additional edits did not translate into improvements in reference-based metrics. On the official test set, the submitted configuration ranked first among the nine participating teams (Section 6.2).

6.2 Submission Configuration

We submitted the combination of Structured Draft Generation and Bidirectional Evidence-Grounded Refinement as our primary system for the shared task.

Because the Structured Generation configuration achieved a higher Main Score than Direct Generation, we selected it as the draft-generation method.

Adding Refinement slightly reduced Main Score from 0.5308 to 0.5216, while increasing NLI Support Rate from 0.7925 to 0.8566.

Manual comparison with source tweets in a small number of examples also identified corrections of erroneous statements and recovery of omitted information. Based on these observations, we submitted the Full configuration with the intention of improving source support while preserving most of the reference agreement. This manual inspection was limited to a small number of examples.

The model-based audit in Section 6.3 and the ablations in Sections 6.4 and 6.5 were conducted after submission and were not used to select the submitted configuration. The official shared-task evaluation (Xia et al., 2026) ranked our submission first among the nine participating teams, with a primary score of 0.682 against 0.664 for the second-ranked team and 0.469 for the organizer baseline (Table 5). This is a system-level comparison; a matched draft-only test result was unavailable at submission time (see Section 6.7 for a post-hoc decomposition using the later-released references).

System	Main Score	Alignment F1	NLI Support Rate
Direct Generation	0.4198	0.1665	0.2938
Direct Generation + Refinement	0.4511	0.1775	0.6525
Structured Generation	**0.5308**	**0.2496**	0.7925
Structured Generation + Refinement	0.5216	0.2291	**0.8566**

Table 4: Effects of structured draft generation and bidirectional refinement: direct generation, structured generation, and each with refinement.

Metric	Organizer baseline	Our submission
Primary	.469	.682
BERTScore F1	.463	.727
BLEURT	.475	.636
Alignment F1	.208	.463
ROUGE-1	.290	.616
ROUGE-2	.172	.448
ROUGE-L	.251	.549

Table 5: Official test results. Source: workshop leaderboard, checked 14 September 2026.

6.3 Model-Based Audit of Refinement

In Section 6.1, adding refinement to Structured Generation increased NLI Support Rate but decreased Main Score and Alignment F1, motivating an additional evaluation of whether individual edits actually improve the report when checked against the sources. We randomly sampled 50 of the 466 body bullets whose content, confidence, citations, or placement was changed by Full refinement. Of these, 34 were modifications to existing bullets with a one-to-one correspondence between the before and after versions, while the remaining 16 had no matched pre-refinement bullet and were treated as additions.

We conducted a blind evaluation by presenting the before and after versions as A/B with their temporal order hidden, using a separate reasoning model, `gpt-6-astra`, with medium reasoning effort. The overall assessment considered source support, epistemic status, information preservation, placement, and citations: 35 edits (70%) were judged beneficial, eight (16%) equivalent, and seven (14%) detrimental. Among the 34 edits with a one-to-one correspondence between before and after statements, the number of fully source-supported statements increased from 16 to 25. However, of the 45 post-refinement statements labeled supported by automatic NLI, nine were judged in this blind evaluation to retain unsupported content. These results suggest that refinement often improves source-grounded quality, while increases in NLI Support Rate cannot be interpreted directly as the magnitude of improvement in factual support. Details of the evaluation protocol are provided in Appendix D.

6.4 Ablation Study of Structured Draft Generation

Table 6 presents ablation results for three design choices in structured draft generation. All three ablations decrease Main Score, with the largest decrease occurring when event type mapping is removed. Removing length control increases the number of body bullets to approximately 1.9 times the original count, while Alignment F1 decreases from 0.2496 to 0.1822 and NLI Support Rate increases from 0.7925 to 0.9128. These results indicate that length control helps adjust the amount of information to match reference reports, while reference agreement and NLI-based source support do not necessarily change in the same direction.

6.5 Ablation Study of Bidirectional Refinement

Table 7 compares Correction, Recovery, and Full refinement. NLI Support Rate is 0.8429 for Correction, 0.8096 for Recovery, and 0.8566 for Full, with Full achieving the highest value by combining both directions.

However, Full has lower Main Score and Alignment F1 than either unidirectional configuration. We therefore observe no improvement in the official metrics from combining both directions.

Full applied replacement operations to 359 of the 1,069 draft body bullets (33.6%), so the small change in Main Score is not due to a low volume of edits.

These quality effects come with steeply increasing computational cost. Moving from Direct Generation to Structured Draft Generation increases LLM calls from 8 to 138 and total input-plus-output tokens from 0.23M to 0.96M (measured on the same eight cells of chemical and earthquake, W1–W4). Refinement is costlier still: across all 23 development cells, Full refinement took 1,507 sec-

System	Main Score	Alignment F1	NLI Support Rate
Structured Generation	**0.5308**	**0.2496**	0.7925
Without event type mapping	0.5126	0.2116	0.8557
Without length control	0.5180	0.1822	**0.9128**
Without gold report style examples	0.5262	0.2395	0.8269

Table 6: Ablation study of structured draft generation.

System	Main Score	Alignment F1	NLI Support Rate
Structured Generation	**0.5308**	**0.2496**	0.7925
Correction	0.5264	0.2435	0.8429
Recovery	0.5256	0.2354	0.8096
Full	0.5216	0.2291	**0.8566**

Table 7: Comparison of the Correction, Recovery, and Full refinement variants.

onds per cell on average (range 395–3,296), 9.6 hours in total on a single H200. Refinement improves NLI Support Rate and audit quality, but the computational cost rises sharply at each stage, so its application should be decided by weighing quality requirements against the computational budget.

6.6 Error Analysis

Table 8 illustrates errors that remain in Full. The first example combines reports of *multiple derailed cars* and *14 cars involved in the incident* into a statement that 14 cars derailed. Even when individual facts appear in the sources, the association between a quantity and a state may be unsupported. Our method fails to remove this over-inference.

In the second example, refinement links dangerous conditions and a closure reported together in the source, adding a causal explanation for the closure. This unsupported relation was absent from the draft but introduced by refinement and was not detected by automatic NLI.

These examples highlight the need to verify not only individual facts in the sources but also the associations between quantities and states and the causal relations asserted in generated statements.

6.7 Post-hoc Analysis with Released Test References

After the submission deadline, the organizers released the reference reports for the test set and the system outputs of all participating teams. We used these to conduct a post-hoc analysis of our submitted system on the test set. Our local evaluation exactly reproduces the official primary score of our submission (0.6816).

Before submission, we adopted refinement based on its improvement in NLI Support Rate and manual inspection of a small number of edits (Section 6.2); the post-submission audit (Section 6.3) provided additional evidence of edit quality. On the test set, the draft-only outputs score 0.6791, and the submitted version, which applies refinement to the same drafts, scores 0.6816 ($+0.0025$); the slight decrease observed on the development set (0.5308 to 0.5216) did not recur. These results are consistent with the decision to retain refinement, although the gain is small.

Compared with the second-ranked team crisis by crisis, our submission scores higher on *damsafety* ($+0.055$), *indfire* ($+0.048$), and *landslide* ($+0.039$), and lower only on *collapse* (-0.030) among the six crises.

The loss on collapse occurs in all six replicates of window W2, for two reasons. First, while the reference sentences are short and template-like, our output tends to append location qualifiers, which lowers BERTScore precision and BLEURT. We attribute this tendency to our editing policy, which prioritizes retaining source-supported place and scope qualifications over brevity: the instructions prohibit merging places or temporal scopes merely to shorten the text. Second, in `collapse.W2.k1`, information that belongs in the Injuries subsection was missing from the generation stage onward.

By window, our primary-score advantage over the second-ranked team is $+0.036$ at W1 and shrinks monotonically to approximately zero at W4. The advantage consists almost entirely of BERTScore precision ($+0.105$ at W1 to $+0.025$ at W4), while recall is slightly below the second-ranked team in every window. The evaluator dilutes precision by scoring system subsections that have no corresponding reference subsection as zero, and our outputs consistently contain fewer such sub-

System output (after refinement)	Source tweets (excerpt)
Over-inference across sources (`chemical.W1.k1`, section 4 Damage)	
Fourteen freight cars derailed near the Ardley yard, causing infrastructure damage at the junction. *[confirmed]* cites [2, 3, 5, 6, 11, 12]	[6] Just saw multiple freight cars derailed near the Ardley yard [12] RT @WCEM: CONFIRMED: WCEM has deployed to #ArdleyJunction #WexfordDerailment site. 14 cars involved in the incident #ammonia
Unsupported causality introduced by refinement (`earthquake.W2.k4`, section 4 Service disruptions)	
Officials stated closures were in place around Caldera, and local reports indicated the coastal area was closed **due to dangerous conditions**. *[confirmed]* cites [71, 82, 83, 89, 195]	[71] officials: closures are in place around Caldera #earthquake. Avoid the area if possible #CalderaQuake [195] Dangerous conditions reported at the coast, with the area closed per local reports as well...

Table 8: Two errors in the refined report body. Top: the report binds the count of cars involved in the incident to the derailment itself, a correspondence no source tweet states. Bottom: refinement adds a causal explanation for a closure, although the source reports dangerous conditions and closure together without establishing causality.

sections (on average 0.2 vs. 0.6 at W1 and 0.7 vs. 1.4 at W2, with comparable levels from W3 onward). Because the reference reports grow from 38 tokens on average at W1 to 254 at W4, suppressing content absent from the reference—through the bullet-count budgets in aggregation and the worth-reporting check in refinement—matters most for precision in the early windows with small references, whereas in later windows extra content increasingly finds a reference counterpart and the gap disappears. The observed gradient is consistent with this account. A comparison with a smaller draft-generation model, Gemma-4-12B-it, on the test and development sets is given in Appendix E.

7 Conclusion

Our structured-generation configuration achieved higher reference agreement than direct generation, while bidirectional refinement increased NLI Support Rate for both draft-generation methods but affected reference agreement differently depending on the method. The submitted combination of both components ranked first among the nine participating teams in the official test evaluation.

Limitations

The blind audit covers a random sample of 50 of the 466 modified body bullets and relies on a single LLM evaluator. Hiding system names and before/after labels does not eliminate model-specific biases, and the audit does not constitute human validation of factuality. It also excludes pure deletions, unchanged statements, and unrecovered information, so it does not assess overall report quality.

Adding bidirectional refinement to Structured Generation does not improve the primary reference-based evaluation metric. These experiments therefore do not establish the general effectiveness of refinement; its benefits may be limited or depend on the evaluation conditions.

The evaluation cells are not independent, as their windows and replicates overlap. The results cover only two crises and one seed and do not establish generalization to unseen disasters or statistical significance.

Ethical Considerations

We use the shared-task data solely for situation report generation. We do not link individuals in the data to external information to identify, contact, or profile them.

Aggregating crisis information with LLMs can nevertheless lead to overconfident statements about uncertain reports, inappropriate integration of distinct information, and misinterpretation of quantities or negation. Although our system seeks to reduce these risks through citations and source-grounded refinement, it does not guarantee the accuracy of generated content.

Generated reports should therefore be treated as research outputs. Any use in actual crisis response requires human verification against the original information.

References

Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, and Kelvin Guu. 2023. RARR: Researching and Revising What Language Models Say, Using Language Models. In *Proceedings of the 61st Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers), pages 16477–16508. Association for Computational Linguistics.

Gemma Team. 2026. Gemma 4 Technical Report. arXiv:2607.02770.

Frederic Kirstein, Terry Ruas, and Bela Gipp. 2025. What's Wrong? Refining Meeting Summaries with LLM Feedback. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 2100–2120. Association for Computational Linguistics.

Mathieu Ravaut, Aixin Sun, Nancy Chen, and Shafiq Joty. 2024. On Context Utilization in Summarization with Large Language Models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2764–2781. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning Robust Metrics for Text Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892. Association for Computational Linguistics.

Fei Xia, Haotian Zhu, Yoo Wun Lee, William D. Lewis, Antonis Anastasopoulos, and Belu Ticona. 2026. Overview of the LT4CPR 2026 Shared Task: Generating Situation Reports from Social Media Messages. In *Proceedings of the First Workshop on Language Technologies for Crisis Preparedness and Response (LT4CPR)*, in conjunction with AACL 2026, Hengqin, Zhuhai, China. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating Text Generation with BERT. In *International Conference on Learning Representations*.

Yusen Zhang, Ansong Ni, Ziming Mao, Chen Henry Wu, Chenguang Zhu, Budhaditya Deb, Ahmed Awadallah, Dragomir Radev, and Rui Zhang. 2022. Summ^N: A Multi-Stage Summarization Framework for Long Input Dialogues and Documents. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1592–1604. Association for Computational Linguistics.

Chenguang Zhu, William Hinthorn, Ruochen Xu, Qingkai Zeng, Michael Zeng, Xuedong Huang, and Meng Jiang. 2021. Enhancing Factual Consistency of Abstractive Summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 718–733. Association for Computational Linguistics.

A Implementation Details

Generation uses Gemma-4-31B-it with BF16 and a maximum context length of 65,536 tokens. We set temperature to 0.7, top-p to 0.95, top-k to 64, and seed to 42, disable thinking, and constrain outputs with JSON schemas. The generator and judge invoke the same model through separate requests. API/JSON retries for malformed outputs are limited to three attempts; semantically rejected edits are not repeatedly proposed until accepted.

BM25 uses $k_1 = 1.2, b = 0.75$. Correction retrieves up to six tweets per query and eight related report bullets. Recovery retrieves up to 12 tweets and eight report bullets. Recovery is capped at 120 relation assessments, 12 distinct addition candidates for acceptance, and 300 logical calls per cell. The integrator proposes at most six transactions per section and at most three replacement bullets per transaction.

The initial overview/timeline prompt specifies four overview sentences and four to ten timeline sentences. The subsequent finalizer lowers the timeline minimum to one sentence, leaves the body unchanged, and generates an overview of four sentences and a timeline of one to ten sentences, requiring supporting body IDs. Each timeline sentence inherits the union of citation IDs from its supporting body bullets. We allow up to three judging rounds and provide an extractive fallback, recording its use. In three Direct Generation cells, generation of these auxiliary components failed citation-scope checks, so we retained the original Direct Generation output, which had passed format validation, without editing its body.

A.1 Length-Control Rules

For a cell containing n tweets, the total body-bullet budget is $T = nr(n)$, where $r(n) = 0.10$ for $n < 400$, 0.09 for $400 \le n < 700$, and 0.07 for $n \ge 700$. These piecewise rates were chosen based on training-report statistics. Let c_s be the number of extracted claims assigned to report section s, and $C = \sum_s c_s$. For each section with $c_s > 0$, the target is

$$t_s = \max\Big(1, \operatorname{round}\Big[T\left(0.6\frac{c_s}{C} + 0.4p_s\right)\Big]\Big).$$

For sections 3–11, the fixed prior shares p_s are, respectively, 0.166, 0.137, 0.108, 0.108, 0.157, 0.163, 0.049, 0.011, and 0.100. Sections with no claims are skipped; the remaining shares are not renor-

malized. Rounding follows Python's round (ties to even), so section targets need not sum to $\mathrm{round}(T)$.

The aggregation prompt gives t_s as the target and an acceptable range from $\max(1, \mathrm{round}(0.7t_s))$ to $\mathrm{round}(1.3t_s) + 1$, together with the total-report guide $\mathrm{round}(T)$. It instructs the model to merge repeated events and, if still over target, omit the least-supported minor details first. For $n < 150$, an additional instruction favors fewer well-supported, merged bullets without padding. These are prompt-level targets, not hard output-count constraints; no subsection-level counts are allocated.

B Reproducibility and Case Identification

For each configuration, we retain drafts, inputs, prompts and schemas, candidates, integration edits and judgments, finalizer requests and responses, and evaluation results. Input and output hashes are used to verify that configurations correspond to the same cells and drafts. Replaying saved artifacts does not guarantee bitwise-identical stochastic generation across execution environments.

The quantitative result tables in this paper are generated from saved evaluation JSON files for which all 23 cells and NLI evaluations were complete. Main Score, Alignment F1, and NLI Support Rate all use hierarchical averaging over replicates, windows, and crises, in that order. The selected qualitative examples in the tables in Section 6 are identified by the cell IDs chemical.W1.k1 and earthquake.W2.k4 and by within-cell tweet IDs. These examples are not treated as a representative sample of the error distribution across all 23 cells.

The second example in Table 8 is audit case C011 (original sample ID R30). Identifiers R01–R50 follow the saved sampling order, whereas C001–C050 identify the anonymized evaluation cases; Appendix D describes sampling, criteria, and comparison with NLI.

The replacement proportion in Section 6.5 counts distinct draft body bullet IDs targeted by accepted integration transactions that contain both old IDs and replacement bullets. Across 23 cells, 350 such transactions targeted 359 of 1,069 draft body bullets, including multi-bullet merges; replacements may change confidence, citations, or placement without changing the text. Pure additions and pure deletions are excluded from this numerator. This operation-based count differs from the post-refinement sampling population in Appendix D.

C Prompt Overview

All LLM calls in our system use JSON-schema-constrained outputs and follow a common evidence policy: report text, editing candidates, and diagnostic results are not treated as factual evidence; only source tweets serve as evidence. For space reasons, we do not reproduce the prompts in full. Table 9 summarizes the role, inputs, outputs, and core constraints of every call, and the following subsections describe the prompt instructions relevant to the claims in the main text.

C.1 Claim Extraction Prompt

The extraction prompt first instructs the model to treat each tweet as expressing event types from a fixed taxonomy, identify the types present in each tweet, produce one claim per type, and assign each claim to the section specified by the mapping table. Table 10 presents this mapping; the actual prompt also includes typical expressions for each type and routing rules for ambiguous cases. An event key is defined as a short identifier shared by claims referring to the same event; successive numerical updates share the same key. The status field takes one of seven values: reported, negative, planned, rumour, denial, correction, or superseded.

C.2 Aggregation Prompt

The aggregation prompt specifies the following main rules.

- Use one bullet per event. Merge paraphrases, retweets, and repeated reports of the same event, but do not merge distinct events.
- When different values are reported for the same quantity, retain a separate bullet for each value in chronological order; never sum them. Only an explicit correction supersedes an earlier value, and the wording must make the correction clear.
- Keep categories distinct: missing $\neq$ dead, suspected $\neq$ confirmed cases, injuries $\neq$ treatment $\neq$ admissions, and planned $\neq$ completed actions.
- Retain reports of absence (e.g., "no deaths") as actual bullets without broadening the scope of the evidence. Attribute denials to their speakers; a denial alone does not establish that an event did not occur.

- Write each bullet as a single past-tense sentence in an official sitrep style, approximately 10–20 words long. Up to four reference bullets per subsection from three training documents are provided as style examples.
- The user message specifies target bullet counts (Section 3.2). When these are exceeded, remove the least-supported details first.
- Assign four flags to each bullet: negative, planned, rumour_or_denied, and contradicted. Confidence is assigned deterministically rather than generated by the LLM: any true flag yields unconfirmed; otherwise, at least two distinct cited tweets yield confirmed.

C.3 Refinement Judge Criteria

The acceptance criteria for each judge in Section 4 are explicitly represented as fields in its output schema.

- Correction judge (Section 4.1): accept only if unsupported_new (new unsupported content), lost_supported (lost supported information), and new_duplicates (new duplicate content) are all empty, and every evidence check includes an exact quotation from the cited tweets.
- Addition judge (Section 4.2): accept only if accept, whole_text_supported, same_event_window, confidence_appropriate, nonduplicate, and no_new_conflict are all true, and the set of tweet IDs in the evidence quotations matches the candidate's citation set.
- Integration judge (Section 4.3): apply a transaction only if accept, whole_support, preserves_useful_old, classification_correct, event_compression_correct, worth_reporting, and all checks on old and new items are true. The proposer's rationale is not passed to the judge.

C.4 Direct Generation Baseline Prompt

The Direct Generation prompt (Section 5.3) instructs the model to report only content supported by the supplied tweets, without adding plausible details or background knowledge; express one atomic factual claim per bullet in approximately 10–20 words; and cite one to six tweets providing the most direct evidence. One reference report (train volcano.W2.k1) is supplied as a style example. The LLM assigns confidence according to the prompt criteria: single reports, contradictions, unexecuted plans, and rumours are unconfirmed, whereas corroboration by multiple independent reports yields confirmed.

D Blind Evaluation Protocol

D.1 Population and Sampling

The post-submission evaluation in Section 6.3 compares the bodies (report Sections 3–11) of Structured Generation and Full refinement across 23 development cells. Eligible Full bullets differ in text, confidence, the cited tweet ID list, or section/subsection; changes only to bullet numbering are excluded. The unit is a post-refinement bullet, excluding pure deletions, unchanged statements, overviews, and timelines.

We first matched bullets identical in these attributes. For the remaining bullets, we applied Hungarian matching with a score combining citation-set Jaccard similarity (weight 0.6) and SequenceMatcher similarity of lowercased text (weight 0.4) for same-section pairs, assigning zero to cross-section pairs and discarding matches below 0.3. Unmatched Full bullets are treated as additions, so the distinction between additions and modifications depends on this procedure.

We sorted the resulting 466 items by cell ID and post-refinement bullet ID and sampled 50 without replacement using Python's `random.Random(20260916).sample`. The sample spans two crises and 19 cells, with 34 edits paired with draft statements and 16 additions without corresponding draft statements.

D.2 Evaluation Materials and A/B Assignment

Each case includes both versions' text, confidence, section/subsection, and citation IDs, together with the complete source tweets in the union of their citations, including source categories and timestamps. To assess duplication, information preservation, and placement, we supplied shared context containing other draft statements in the relevant section and the three most similar draft statements, excluding the corresponding target draft statement. The evaluator was instructed to treat this context as context rather than factual evidence and assess each version's source support using only its own citations.

Using seed 20260917, we assigned the post-refinement version to A/B in 17 paired cases each and eight addition cases each. The other version of an addition is `null`, denoting absence. We shuffled case order, source order, and shared context, assigning case IDs C001–C050 and source IDs beginning at T001 within each case. The evaluator did not receive the mapping key, system names, before/after labels, development history, reference reports, prior ratings, or NLI judgments. For additions, however, the absence of one version remains visible.

D.3 Evaluator and Execution

We used OpenAI `gpt-6-astra` with medium reasoning effort, separate from the Gemma model used for generation and refinement. Each case was evaluated in a fresh ephemeral Codex session from an isolated workspace containing only that case and the fixed task, rubric, and output schema. Sessions did not inherit development conversations or other case judgments; logged file reads were confined to the supplied materials. No dated model snapshot ID was recorded.

The evaluator returned JSON containing support labels, unsupported fragments, evidence quotations, confidence and placement assessments, an overall rating, and a rationale. Transient execution failures such as capacity errors could be retried; completed judgments were not overwritten or retried based on their content.

D.4 Rubric and Aggregation

Source support was labeled `supported` (the entire statement is supported), `partial` (some content is unsupported), `unsupported` (the citations do not support the statement), or `not_applicable` (the version is absent). Separately, overall quality was rated for *B relative to A*, considering source fidelity, epistemic status, information preservation, duplication, placement, and citations:

- 5: B repairs a clear factual error or important omission in A.
- 4: B makes a minor improvement in qualification, attribution, epistemic status, or information.
- 3: Approximately equivalent, or gains and losses balance.
- 2: B introduces a partial error, worsens placement, or loses useful information.
- 1: B introduces a severe error or reverses the meaning.

The evaluator was instructed not to reward citation count or polished wording alone, or automatically prefer a nonempty version.

After all 50 judgments were complete, we used the mapping key to orient ratings toward refinement: we retained rating r when B was the post-refinement version and used $6 - r$ when A was the post-refinement version. Ratings 4–5, 3, and 1–2 were grouped as beneficial, equivalent, and detrimental, yielding 35, eight, and seven edits, respectively. This is an overall edit-quality assessment, not a factual-support improvement rate.

D.5 Source Support and NLI Comparison

The before/after support comparison uses only the 34 paired edits, excluding the 16 additions. Fourteen pairs were fully supported before and after, 11 changed from partial to full support, seven remained partial, and two changed from full to partial support. Thus, fully supported statements increased from 16 to 25.

After judgments were fixed, we compared them with saved NLI results. Nine of the 45 post-refinement statements labeled `supported` by NLI were `partial` in the blind evaluation. Of four cases changing from non-supported to supported under NLI, only one also changed from partial to full support in the blind evaluation. These comparisons concern AI judgments on a sample of modified items; they neither estimate NLI accuracy against human ground truth nor re-estimate support rates across all 23 cells.

E Model-Size Comparison

Table 11 compares replacing the draft-generation model Gemma-4-31B-it with Gemma-4-12B-it. Prompts, generation settings, and the evaluator revision (`c8bd925`) are identical to the experiments in the main text. The 12B test run completed 116 of the 117 test cells, so the test column reports both models aggregated over the same 116 cells (the 31B figure over all 117 cells is 0.6791).

Model	dev (23 cells)	test (116 cells)
Gemma-4-12B-it	0.5030	0.6537
Gemma-4-31B-it	0.5308	0.6788

Table 11: Primary score by draft-generation model size.

Stage	Prompt	Key inputs	Output (JSON)	Core constraints
3.1	Claim extraction	80-tweet chunk; crisis/window metadata	claims: tweet ID, event type, section/subsection, event key, text, figure, status	identify event types first; section from the mapping table; one claim per distinct event in a tweet
3.2	Aggregation (per section 3–11)	section claims; style examples; target bullet counts	subsections with bullets: text, tweet IDs, 4 flags	merge duplicates; keep numerical updates separate; stay near targets; 10–20-word sitrep sentences
3.4	Initial overview / timeline	compiled body text (sections 3–11); crisis metadata	4 overview bullets; 4–10 timeline bullets with tweet IDs	no raw tweets; timeline citations copied from body bullets; no new facts
4.1	Audit	target bullet; its cited tweets	supported/unsupported fragments; support / challenge / related queries	cited tweets are the only evidence
4.1	Edit	target; retrieved tweets; related bullets; audit	keep / replace / remove; new bullets with quoted evidence	preserve supported relations; remove unsupported fragments; no style-only edits
4.1	Judge	proposal without the proposer's rationale	verdict; per-claim evidence checks with exact quotes	accept only with no unsupported content, no lost supported information, no new duplicates
4.2	Extraction	20-tweet batch; crisis/window metadata	per-tweet relevance; up to 4 relations (span, claim, evidence status)	labels are working hypotheses, not verified truth
4.2	Routing	16 relations; full draft	covered / review / uncertain / no-contribution	"covered" requires a literal witness string in the draft
4.2	Locate	claim; evidence tweets; full draft	covered / partial / missing / not-relevant; addition proposal (section, title, text, confidence, tweet IDs)	additions cite only reviewed tweets; canonical subsection titles
4.2	Judge	addition proposal; sources	six boolean checks; evidence quotes	all checks must pass; quotes must match the cited tweets
4.3	Propose (per section)	whole report as context; section candidates; source tweets	up to 6 transactions ($\leq$3 bullets each); reasons for unused candidates	edits restricted to the current section; candidates are hypotheses, tweets are evidence
4.3	Judge	one transaction; old/new items; sources	accept; per-item checks with exact quotes	whole support; preserve useful old content; correct classification; worth reporting
–	Finalization	final body (sections 3–11)	4 overview sentences; 1–10 timeline sentences	body fixed; at most 3 attempts, then extractive fallback
5.3	Direct baseline	all tweets; schema; one gold report example	full report JSON	report only what the tweets support; one atomic claim per bullet

Table 9: Inventory of LLM prompts: role, key inputs, outputs, and core constraints. All calls use JSON-schema-constrained outputs.

Section	Event type → subsection
3 Casualties	KILL→Fatalities; INJURE→Injuries (Cases in outbreaks); MISSING→Missing persons; TRAP→Trapped persons; RESCUE→Rescued persons; FOUND→Found persons; RECEIVE_HELP (medical)→Treatment and admissions
4 Infrastructure	DAMAGE→Damage; DESTROY→Destruction; DISRUPT_SERVICE→Service disruptions; RESTORE→Restoration progress
5 Displacement	EVACUATE→Evacuations; DISPLACE→Displacement; RETURN→Returns
6 Hazard	HAZARDOUS_CONDITION, SITUATION_STATE→Hazard assessment
7 Response	DEPLOY, COORDINATE→Operational response
8 Communication	WARN→Warnings and alerts; REQUEST→Requests for help; INFORM→Information updates
9 Aid	PROVIDE_HELP→Aid provided and distributed; RECEIVE_HELP (non-medical)→Aid received and needs coverage; TRANSFER, FUNDING (aid-directed)→Donations and financial transfers
10 Administrative	MEETING→Meetings and decisions; POLICY_CHANGE, TRANSFER (non-aid)→Administrative and economic measures
11 Social	SOCIAL_SUPPORT→Social support; ATTITUDE, PROTEST→Public sentiment

Table 10: Event-type-to-section/subsection mapping given to the extraction prompt. The mapping builds on the official taxonomy, with task-specific routing for medical help and aid-directed funding and subsection assignments added by this system.