When Better Codebooks Are Not Enough: Predictive Performance and Behavioral Reliability in LLM Political Event Coding

Anonymous ACL submission

Abstract

High accuracy does not necessarily make an LLM a faithful coder. This issue matters because many social-science studies rely on expert-written codebooks to turn text into structured data. We study this problem in political event coding, a challenging source–target relation classification task beyond ordinary sentence-level classification, where models must determine what one actor did to another using detailed coding rules.

We test whether expert codebooks become more effective when operationalized into LLM-friendly forms with clearer definitions, examples, retrieved context, and rules for difficult cases. We then evaluate behavioral reliability under controlled changes to label names, codebook order, and label–definition mappings.

Clearer codebooks substantially improve classification performance, especially for fine-grained event classification. However, stronger predictive performance does not necessarily imply behavioral reliability. Models may produce valid labels and recover definitions while still failing behavioral reliability tests under controlled codebook changes.

These findings reveal a gap between predictive performance and behavioral reliability, suggesting that codebook-guided LLM systems should be evaluated not only by accuracy, but also by whether they preserve the coding logic that makes coded outputs meaningful for social-science research. [1]

1 Introduction

Many social-science studies turn text into analyzable data, such as whether a news report describes cooperation, protest, threat, or violence. This conversion is a form of measurement: it turns qualitative evidence into structured categories that can be counted, compared, and used in downstream analysis. Expert-written codebooks make this process explicit by defining category inclusion, exclusion, and procedures for ambiguous cases (Krippendorff, 2019; Neuendorf, 2018). However, codebooks are costly to build and apply: they require domain expertise, annotator training, and reliability checks. They are therefore essential measurement resources, but also bottlenecks for scaling text analysis.

This bottleneck motivates the use of LLMs to apply codebooks directly. Recent work shows that definitions, examples, and coding instructions can improve zero-shot classification, especially when labels are specialized or difficult to infer from names alone (Hu et al., 2024; Ruckdeschel, 2025). Yet strong predictive performance does not necessarily mean that models are truly applying the codebook definitions and rules. Models may instead rely on familiar label semantics, prompt order, or pretrained associations rather than the operational logic defined by the codebook (Halterman and Keith, 2025).

Political event coding provides a demanding setting for codebook-grounded LLMs. Full event coding involves identifying actors, actions, and directed relations from text; this paper focuses on the relation-classification component. Given a text and a specified source–target actor pair, the model predicts what political action the source directs toward the target. Event coding builds on earlier event-data codebooks such as WEIS (McClelland, 1999). Ontologies such as CAMEO and PLOVER use hierarchical labels and event-mode rules that can assign different labels to similar wording depending on whether an action occurred, was promised, refused, or negated (Gerner et al., 2002; Hu et al., 2024). These distinctions make political event coding a demanding test of whether LLMs follow codebook rules rather than matching familiar label semantics.

We study two questions:

[1] Code available for review at https://drive.google.com/drive/folders/1Gt6fDxH-jm9X0KbdBqtaniHmH46gq8iN?usp=sharing

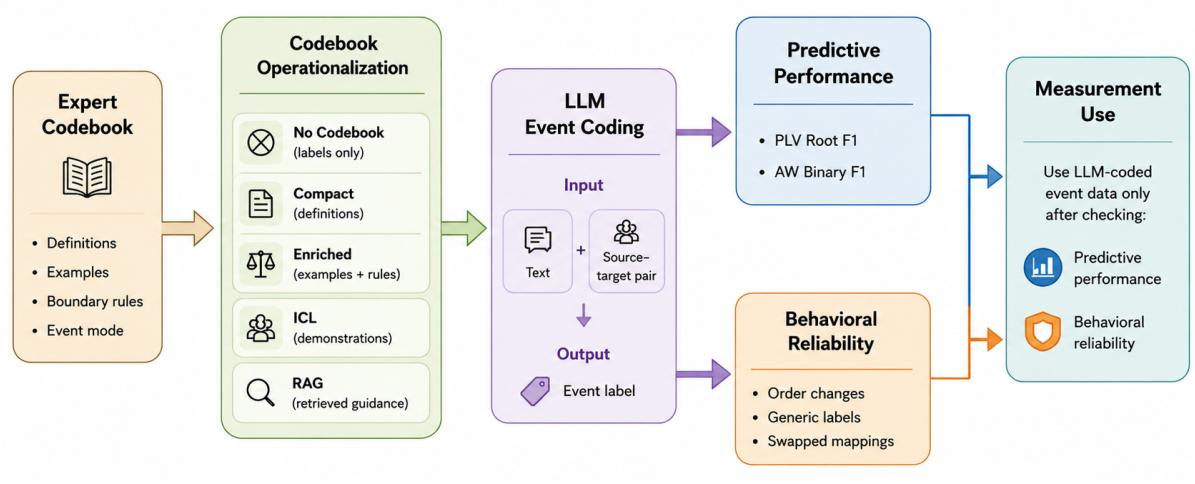

Figure 1: Predictive performance and behavioral reliability in codebook-grounded LLM event coding. Expert codebooks are converted into LLM-usable prompts or retrieval inputs and evaluated through predictive performance and behavioral reliability under controlled codebook perturbations.

1. Can richer codebook operationalization improve predictive performance?

2. Does stronger predictive performance necessarily imply reliable codebook-following behavior?

Our results reveal a gap between scoring well and following the codebook. Richer codebook operationalization improves fine-grained event classification, but these gains do not guarantee behavioral reliability under controlled codebook changes. We therefore argue that event-coding LLMs should be evaluated not only by predictive performance, but also by whether they preserve the coding logic defined by the codebook. A model that predicts the correct label is not necessarily a model that follows the codebook.

Our contributions are threefold: (1) we study codebook-guided LLM evaluation in a demanding political event-coding setting with hierarchical labels and event-mode distinctions; (2) we show that richer codebook operationalization substantially improves fine-grained event classification across open-source LLMs; and (3) we reveal a gap between predictive performance and reliable codebook-following behavior through controlled perturbation tests.

2 Preliminaries

2.1 Codebooks as Measurement Systems

Text-based social-science measurement often requires turning documents into structured variables. Codebooks make this process explicit by defining category boundaries, exclusion rules, and procedures for ambiguous cases (Krippendorff, 2019; Neuendorf, 2018). They are therefore more than label inventories: they are measurement instruments that translate qualitative evidence into analyzable observations.

The same structure that makes codebooks useful also makes them costly to build and apply. Codebook construction requires domain expertise and iterative refinement, while consistent use depends on annotator training and reliability assessment (Krippendorff, 2019; Neuendorf, 2018). These costs motivate the use of large language models as potential codebook users, without training a separate supervised model for each coding scheme. Political event coding provides a useful testbed because its labels depend on fine-grained coding rules rather than keyword matching alone.

2.2 Political Event Coding as a Challenging Testbed

Political event coding converts news text into structured records of political interactions. Full event coding may involve identifying actors, actions, and relations; this paper focuses on the relation-classification component. Given a text with a specified source and target actor, the model assigns the political relation directed from the source to the target.

CAMEO is a widely used ontology for political event coding, and PLOVER simplifies its label space while preserving the hierarchy needed for source–target relation classification (Gerner et al.,

Event-mode distinction in protest-related coding

Source: Protesters
Target: Government

Future protest intention
Text: "Protesters said they would stage demonstrations against the government."
Binary: Conflict
Quad: Verbal Conflict
Root: THREATEN

Completed protest action
Text: "Protesters staged demonstrations against the government."
Binary: Conflict
Quad: Material Conflict
Root: PROTEST

Ended protest activity
Text: "Protesters ended demonstrations after negotiations with the government."
Binary: Cooperation
Quad: Verbal Cooperation
Root: YIELD

Figure 2: Illustration of event-mode distinctions in PLOVER. Similar protest-related content can map to different labels depending on whether the action is threatened, carried out, or halted.

2002; Open Event Data Alliance, 2018; Hu et al., 2024). PLOVER labels are evaluated at three levels: **binary** cooperation versus conflict, **quad-level** verbal or material cooperation and conflict, and **root-level** categories such as agreement, aid, rejection, threat, protest, coercion, and assault.

This hierarchy makes event coding a demanding test of codebook-grounded LLMs. As Figure 2 shows, similar protest-related content can map to different labels depending on whether the action is threatened, carried out, or halted. Such cases require applying the codebook rules rather than matching keywords to familiar labels.

2.3 Codebook Operationalization and Reliability

Because event coding depends on codebook-specific boundary rules, recent work has explored codebook-based zero-shot approaches. In political event coding, Hu et al. (2024) convert PLOVER codebook knowledge into natural language inference (NLI) hypotheses, drawing on entailment-style formulations for zero-shot classification (Obamuyide and Vlachos, 2018; Yin et al., 2019). Related zero-shot event extraction work also studies schema transfer without task-specific supervision (Huang et al., 2018). Other approaches place definitions, examples, and coding instructions directly into prompts or promptbook-style reformulations (Ruckdeschel, 2025; Stuhler et al., 2025). These methods reduce the need for retraining when schemas or label definitions change (Hu et al., 2022; Parolin et al., 2021, 2022).

We use *codebook operationalization* to mean converting expert codebook content into LLM-usable forms, including definitions, examples, boundary rules, and retrieved context. Such representations may improve prediction by making coding rules easier to access. However, performance under a single prompt does not establish behavioral reliability. We therefore test whether predictions follow the intended codebook logic under controlled codebook changes that should not alter the underlying source–target relation (Halterman and Keith, 2025).

3 Approach

This section defines the source–target classification task, describes how codebook knowledge is represented for LLMs, and introduces behavioral reliability probes.

3.1 Task Setup

We study source–target political relation classification: given a text and a specified source–target actor pair, the model predicts the event label for the directed relation. We evaluate this task under the PLOVER hierarchy introduced above, with root-level coding serving as the strongest test of codebook boundary-rule application.

3.2 Codebook Operationalization Strategies

We compare several ways of presenting codebook knowledge to the model. The no-codebook baseline provides only the text, the marked source–target pair, and the valid label set. It tests what can be inferred from label names and pretrained knowledge alone. **Compact** codebook prompting adds short natural-language definitions and higher-level quad-level groupings for each label. **Enriched** codebook prompting keeps the same label inventory as Compact but adds worked examples, event-mode guidance, boundary notes, and disambiguation rules for commonly confused categories. These additions clarify how the original codebook should be applied without changing the underlying label space.

As shown in Table 1, the Enriched setting adds concrete event-mode guidance beyond Compact definitions. Because Enriched combines multiple

Table 1: Concrete example of Enriched codebook guidance over Compact guidance. Enriched guidance adds event-mode distinctions without changing the label space.

	Codebook content
Compact	Definitions only: THREATEN = verbal indication of conflict PROTEST = collective conflict action YIELD = de-escalation or concession
Enriched	Same label space, plus event-mode distinctions: future protest intention → THREATEN completed protest activity → PROTEST halted protest activity → YIELD

forms of guidance, we interpret Compact–Enriched differences as the effect of the enriched codebook package rather than as an ablation of individual components.

We also include general prompting and retrieval baselines. In-context learning (**ICL**) provides labeled demonstrations without full codebook definitions (Brown et al., 2020). **CoT** uses the Enriched codebook with an added step-by-step reasoning instruction before the final label prediction. Retrieval-augmented generation (**RAG**) retrieves input-relevant definitions, examples, or rules instead of presenting the full codebook (Lewis et al., 2020). We report the main retrieval variant in the main text; additional implementation details and variants are provided in Appendix B.

3.3 Behavioral Reliability Probes

We evaluate behavioral reliability through controlled codebook perturbations adapted to source–target event coding. Each probe holds the input text and source–target actor pair fixed while changing the codebook presentation. The goal is to test whether predictions remain tied to the intended coding rules rather than relying on label names, entry order, or pretrained semantic associations (Halterman and Keith, 2025).

The first set of probes tests whether the model recognizes the valid codebook structure. Legal-label compliance measures the share of parseable valid-label outputs, while definition recovery tests whether codebook definitions are matched to the intended political relation categories.

The second set tests behavioral reliability under codebook changes. Order probes reverse or shuffle codebook entries; generic-label probes replace meaningful labels with neutral placeholders while preserving definitions; and swapped-mapping probes alter label–definition associations. These probes are not intended to simulate realistic ontology revisions directly. Instead, they isolate whether predictions follow operational definitions or rely on surface-level prompt cues.

3.4 Behavioral Reliability Metrics

Predictive performance is measured with macro-F1 on held-out test examples. For PLV, macro-F1 is computed over the 15 root-level PLOVER labels. For AW, macro-F1 is computed over the binary Cooperation/Conflict labels. Outputs that cannot be parsed into a valid label are counted as incorrect rather than dropped.

Each behavioral probe is scored according to the behavior it tests. Original-condition accuracy measures the share of predictions matching the gold label under the unmodified codebook. Legal-label compliance measures the share of outputs that can be parsed into the valid label set, while definition recovery evaluates whether codebook definitions are mapped back to their intended labels. For order perturbations, we report agreement and prediction changes across original, reversed, and shuffled codebook orders. Generic-label and swapped-mapping probes are evaluated with weighted F1 against the expected perturbed labels.

For method-level comparison, we report two diagnostic summaries. The Codebook Alignment (**CB-Align.**) measures basic codebook recognition: whether the model outputs valid labels and can match codebook definitions to the intended categories. The Rule-following Score (**Rule-S**) measures behavioral reliability under codebook changes by combining the order, generic-label, and swapped-mapping probes. These summaries are compact diagnostics, so we interpret them together with the individual probe results rather than as standalone measures.

4 Experiments

4.1 Dataset and Setup

Our main evaluation uses the PLOVER-based PLV benchmark for source–target political relation classification (Hu et al., 2024). It contains 1,033 annotated source–action–target examples, each with an input text, a marked source actor, a marked target actor, and a directed event label. We report macro-F1 at three granularities: binary Cooperation/Conflict, quad-level verbal/material cooperation or conflict, and root-level categories from

Table 2: Main predictive results on PLV and AW. PLV is evaluated with root-level macro-F1, and AW is evaluated with binary macro-F1. Mean averages the four open-source LLMs. Bold marks the best open-source LLM result within each dataset–model column. The main comparison is No Codebook → Compact → Enriched. ICL and CoT are prompting baselines rather than additional codebook levels. UP and ZSP Tree are prior baselines from Hu et al. (2024).

	PLV Root F1					AW Binary F1				
Method	Gemma	Qwen	Mistral	Llama	Mean	Gemma	Qwen	Mistral	Llama	Mean
Main codebook comparison										
No Codebook	0.520	0.504	0.409	0.395	0.457	0.754	0.744	0.786	0.743	0.757
Compact	0.636	0.579	0.555	0.526	0.574	**0.788**	0.781	0.764	0.793	0.782
Enriched	**0.718**	**0.610**	**0.617**	**0.587**	**0.633**	0.761	**0.789**	**0.817**	0.810	**0.794**
Prompting baselines, not codebook levels										
ICL	0.573	0.514	0.444	0.429	0.490	0.774	0.774	0.798	**0.814**	0.790
CoT	0.520	0.503	0.405	0.479	0.477	0.493	0.431	0.687	0.441	0.513
Prior baselines										
UP			0.463					0.672		
ZSP Tree			0.824					0.880		

AGREE to ASSAULT.

We also evaluate on AW (ACE/WikiEvents), a cross-domain binary Cooperation/Conflict task with 805 test examples derived from the ACE event extraction corpus (Doddington et al., 2004) and WikiEvents (Li et al., 2021) by mapping seven NLP event types to PLOVER categories (Hu et al., 2024). Because the mapping between NLP event types and PLOVER Rootcodes is approximate, only binary evaluation is reliable on this dataset.

Predictive performance is measured on held-out test sets with macro-F1. For PLV, root-level predictions are also mapped upward through the PLOVER hierarchy; unparseable outputs are counted as incorrect. Behavioral reliability is evaluated on the available evaluation examples for each task by rerunning the same inputs under controlled codebook perturbations: reordered entries, generic label names, and swapped label–definition mappings. PLV reliability is evaluated at the root level, and AW reliability at the binary level.

The main predictive comparison focuses on three LLM-facing conditions: No Codebook, Compact, and Enriched. We report results across four open-source instruction-tuned models: Gemma2:9B, Qwen2.5:7B, Mistral-7B-Instruct, and Llama-3.1-8B-Instruct. Two prompting baselines are also included: ICL, which provides labeled demonstrations without any codebook definitions, and CoT, which prepends the Enriched codebook and adds a step-by-step reasoning instruction. We also report two prior baselines from Hu et al. (2024): Universal PETRARCH (**UP**) (Lu and Roy, 2017) , a traditional dictionary- and pattern-based event coder, and **ZSP Tree**, a hierarchical zero-shot prompting approach designed for codebook-guided event coding.

RAG refers to the strongest retrieval-based configuration from Appendix C, which retrieves input-relevant definitions and examples from the Enriched codebook instead of presenting it in full; it is reported alongside the behavioral probes rather than in the main predictive table. Additional ZSP, JSON, RAG, and prompting details are reported in Appendix C.

4.2 Codebook Guidance Improves Prediction

Table 2 reports the main predictive results on PLV and AW. On PLV, root-level macro-F1 improves from No Codebook to Compact and is highest under Enriched for all four models. This pattern is strongest at the root level, where models must distinguish neighboring event categories rather than only broad conflict/cooperation polarity. AW shows the same broad direction, but the gains are smaller and less uniform because the task is binary. Thus, the main comparison should be read as a controlled increase in codebook guidance, from no codebook to compact and then enriched guidance. The results show that explicit codebook information improves prediction, with the clearest gains in the harder PLV root-level setting.

The prior baselines provide context for the LLM results. UP, a classic dictionary-based event coder, performs well below the codebook-guided LLM settings, while ZSP Tree remains the strongest baseline. This suggests that simple codebook prompting improves over static dictionary matching, but still

falls short of specialized NLI-based event-mode disambiguation.

The PLV–AW contrast further suggests that codebook guidance matters most when labels require boundary-rule application. In AW, many cases can be resolved from broad Cooperation/Conflict polarity. In PLV root coding, however, the model must choose among neighboring categories that share surface-level wording but differ in polarity or mode, such as AGREE versus REQUEST, or AID versus SANCTION. These distinctions depend more directly on event-mode and boundary rules, which helps explain why Enriched yields its clearest gains at the root level.

Figure 2 illustrates why root-level coding is difficult: similar protest-related content can map to different labels depending on whether the action is threatened, carried out, or halted.

Enriched improves over Compact by making the same ontology easier to apply in difficult cases. As summarized in Table 1, Enriched adds examples, event-mode guidance, boundary notes, and disambiguation rules without changing the label space.

Additional JSON and RAG variants are reported in Appendix C. These results suggest that useful codebook representations must preserve contrastive context for boundary decisions, rather than merely shorten or restructure the prompt.

CoT is a cautionary case. Under strict held-out macro-F1, it does not consistently improve performance and remains weaker than the codebook-based conditions. This suggests that plausible reasoning does not guarantee correct codebook application: a reasoning prompt may help the model discuss the source, target, and action type, but the final label still depends on codebook-specific boundaries such as promised versus completed assistance, verbal threat versus material protest, or refused versus halted cooperation.

Full binary and quad-level cross-model results are reported in Appendix C.2.

To add uncertainty estimates for the main PLV codebook comparisons, we compute nonparametric bootstrap confidence intervals for root-level macro-F1 differences. Table 3 reports method differences rather than absolute scores, since the goal is to quantify uncertainty around the main comparisons. The intervals support the broad benefit of adding compact codebook guidance: Compact improves over the no-codebook baseline for all four models. In contrast, the additional Enriched-over-Compact gain is more model-dependent: it is positive for Gemma2 and Qwen2.5, while the intervals for Mistral and Llama3.1 overlap zero.

Table 3: Bootstrap uncertainty check for PLV root-level macro-F1 differences. The table reports method differences rather than absolute scores.

Model	Comparison	Δ F1	95% CI
Gemma2	Compact – No Codebook	+0.099	[0.076, 0.123]
	Enriched – Compact	+0.028	[0.013, 0.044]
Qwen2.5	Compact – No Codebook	+0.028	[0.003, 0.052]
	Enriched – Compact	+0.077	[0.057, 0.097]
Mistral	Compact – No Codebook	+0.093	[0.068, 0.119]
	Enriched – Compact	+0.014	[-0.007, 0.032]
Llama3.1	Compact – No Codebook	+0.097	[0.068, 0.124]
	Enriched – Compact	-0.009	[-0.033, 0.013]

Table 4: Behavioral reliability diagnostics on PLV and AW, averaged across four open-source models. Orig. Acc. is original-condition probe accuracy. CB-Align. summarizes valid-label compliance and definition recovery; Rule-S summarizes order, generic-label, and swapped-mapping probes. ICL uses no codebook definitions. CoT and RAG both use the Enriched codebook.

Setting	Method	Orig. Acc.	CB-Align.	Rule-S
PLV	Compact	0.612	0.967	0.471
	Enriched	0.619	0.990	0.461
	ICL	0.454	0.955	0.294
	CoT	0.642	0.996	0.507
	RAG	0.633	0.994	0.572
AW	Compact	0.869	0.995	0.585
	Enriched	0.841	1.000	0.570
	ICL	0.807	0.990	0.343
	CoT	0.834	1.000	0.580
	RAG	0.837	0.928	0.382

4.3 Behavioral Stress Tests Reveal Reliability Gaps

Having established that codebook guidance improves macro-F1, while the additional Enriched-over-Compact gain is more model-dependent, we next test whether these predictive gains reflect behavioral reliability. In other words, we test whether models are merely getting the right answer or actually following the intended coding rules. Table 4 reports behavioral diagnostics on PLV and AW, averaged across four instruction-tuned models. The table separates direct Compact and Enriched prompting from prompting and retrieval variants, so the Compact–Enriched reliability comparison is visible rather than averaged away.

CB-Align. scores are consistently high across settings, indicating that most models can recognize valid labels and basic definition-category as-

Table 5: Enriched guidance does not consistently improve PLV Rule-S. Values compare Compact and Enriched versions of each setting; Δ is Enriched minus Compact.

Setting	Compact	Enriched	Δ
Direct prompt	0.471	0.461	-0.010
CoT reasoning	0.470	0.507	+0.037
RAG retrieval	0.527	0.572	+0.045

Table 6: Probe-level PLV behavioral reliability breakdown. Order κ measures agreement across original, reversed, and shuffled codebook orders; Generic F1 and Swap F1 report performance under generic-label and swapped-mapping probes. Values are diagnostic and should be read by probe type rather than as a single leaderboard. ICL uses no codebook definitions; CoT and RAG use the Enriched codebook.

Method	Order κ	Generic F1	Swap F1
Compact	0.743	0.566	0.105
Enriched	0.782	0.519	0.081
ICL	0.668	0.152	0.064
CoT	0.745	0.625	0.152
RAG	1.000	0.522	0.193

sociations. By contrast, Rule-S is more diagnostic because it measures whether predictions continue to follow the supplied codebook when label names, codebook order, or label–definition mappings change. Across both datasets, surface-level codebook alignment is much stronger than behavioral reliability, suggesting that recognizing codebook form is easier than reliably following codebook logic.

The Compact–Enriched comparison reinforces the decoupling between predictive performance and behavioral reliability. In Table 4, Enriched slightly improves PLV original-condition accuracy and Codebook Alignment over Compact, but does not improve Rule-S; on AW, it reaches perfect Codebook Alignment but has slightly lower original-condition accuracy and Rule-S. Table 5 summarizes this pattern across direct codebook prompting, CoT, and RAG: the Rule-S changes are small and method-dependent, with a slight decrease for direct codebook prompting and modest increases for CoT and RAG. We therefore treat these differences as directional diagnostics rather than statistically resolved effects.

Table 6 shows where the PLV behavioral reliability gap is concentrated. Order perturbations are less damaging than meaning-level perturbations: several methods maintain moderate to high agreement under reordered codebooks, but all methods perform poorly when labels are anonymized or label–definition mappings are swapped. The especially low Swap F1 scores suggest that models often preserve familiar label semantics even when the supplied codebook changes.

The Compact–Enriched contrast clarifies the main pattern. Enriched improves order agreement relative to Compact, but it does not improve Generic F1 or Swap F1. This supports the interpretation that enrichment helps with the original ontology and some presentation changes, while deeper meaning-level perturbations remain difficult. The perfect order agreement for RAG should be interpreted cautiously because retrieval selects local chunks rather than presenting the full ordered codebook.

Order perturbations preserve label semantics, whereas generic-label and swapped-mapping probes weaken or break the connection between label names and their usual meanings. The resulting drop in Generic F1 and Swap F1 suggests that models often rely on familiar semantic associations rather than fully following the supplied operational definitions.

4.4 Analysis

Taken together, the results show that richer codebook guidance improves hard event coding, but high predictive accuracy does not necessarily imply faithful codebook-following behavior. Figure 3 illustrates this decoupling on PLV: original-condition accuracy and Rule-S rank methods differently, with RAG having the highest Rule-S among the evaluated variants. By contrast, Enriched is strongest in the main predictive comparison, yet its gains on the behavioral diagnostics are much smaller than its gains in predictive macro-F1.

A model may reach the correct prediction while relying on shortcuts unrelated to the supplied coding rules. Models can perform well under the original ontology while remaining vulnerable to meaning-level perturbations. The swapped-mapping probe is the clearest case: models often continue to output the familiar semantic label rather than the label specified by the modified codebook.

One swapped-mapping case illustrates this failure mode. In a PLV example, Merkel met Erdogan for talks expected to focus on a migration deal. Under the original codebook, the gold label is CONSULT and the model predicts CONSULT. Under

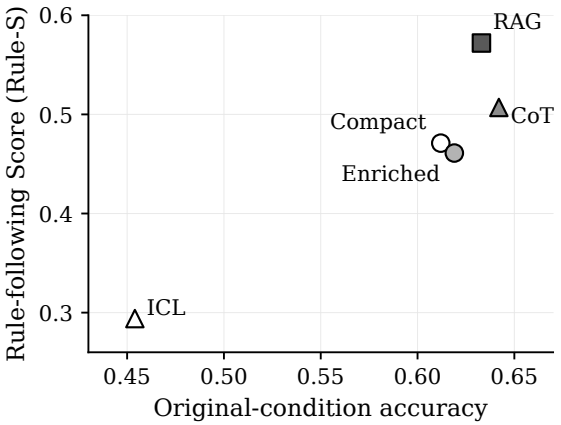

Figure 3: Prediction–behavioral reliability decoupling on PLV. Original-condition accuracy and Rule-S rank methods differently; RAG has the highest Rule-S among the evaluated variants.

the swapped mapping, the expected label becomes COOPERATE, but the model still outputs CONSULT. This suggests that the model preserves the familiar association between meetings and CONSULT rather than following the modified label–definition mapping.

For measurement applications, this distinction is consequential. Event labels are often treated as observed data in downstream analyses, but if a model changes labels under codebook-preserving perturbations, the resulting variables may reflect prompt artifacts rather than the intended political concepts. Failures in behavioral reliability at the coding stage therefore become measurement uncertainty and can threaten measurement validity (Adcock and Collier, 2001). Codebook-grounded LLMs should therefore be evaluated not only by predictive performance, but also by whether they preserve the coding logic specified by the codebook under controlled perturbations.

5 Conclusion

This work studies codebook-guided LLM event coding as a measurement problem rather than only a classification problem. Across predictive and behavioral evaluations, we find that richer codebook operationalization improves fine-grained event classification, but does not guarantee behavioral reliability under controlled codebook changes.

Our results show that enriched codebooks help LLMs apply definitions, examples, and boundary rules, especially for fine-grained event distinctions. However, stronger predictive performance does not fully translate into reliable codebook-following. Models can still change their outputs when label names, codebook order, or label–definition mappings are altered.

This gap matters because codebook outputs often become data. If a model relies on surface cues rather than the intended coding rules, failures in behavioral reliability become uncertainty in the resulting measurements. Our findings therefore suggest that codebook-guided LLMs should be evaluated not only by predictive performance, but also by whether they preserve the coding logic under controlled changes to the codebook.

Although our experiments focus on political event coding and a simpler binary Cooperation/Conflict setting, the same concern extends to other codebook-based domains, including armed-conflict event data, protest coding, gun violence coding, policy coding, and qualitative coding (Raleigh et al., 2010; Halterman and Radford, 2021; Pavlick et al., 2016; Than et al., 2025). A longer-term direction is to build auditable coding systems that return more than a label: they should identify the codebook rule being applied, report uncertainty, flag cases that fail behavioral reliability checks, and route them to human review.

Limitations

This study focuses on codebook-grounded LLM measurement in political event coding, using a fine-grained PLOVER-based relation-classification task and a binary Cooperation/Conflict task. These settings are useful for testing codebook guidance and behavioral reliability, but they cover only part of the broader space of codebook-based text measurement. Future work should examine whether the framework transfers to other codebooks and socio-political event coding settings, including protest coding, conflict event datasets, policy coding, and incident classification (Hurriyetoglu et al., 2021; Barker et al., 2021; Radford, 2021).

A second limitation concerns model coverage. We evaluate several open-source instruction-tuned models and selected prior baselines, but larger frontier models and domain-adapted models may behave differently. The retrieval and chain-of-thought results should therefore be interpreted as findings about the implementations tested here rather than as general conclusions about these method families.

The method comparisons also do not provide

full component-level ablations. In this study, we intentionally use unconstrained prompting settings in order to expose whether models rely on surface label semantics under ordinary codebook use. The Enriched setting combines manually selected examples, event-mode guidance, boundary notes, and disambiguation rules, so the experiments do not isolate the contribution of each component. Similarly, the RAG variants differ in embedding model, index design, example retrieval, preprocessing, and hierarchical filtering. These results therefore compare practical codebook-guidance configurations rather than identify the causal effect of any single design choice.

The swapped-mapping probe is intentionally diagnostic: it tests whether models follow a supplied codebook when label semantics conflict with familiar meanings, but it does not cover all realistic ontology revisions. Trained human coders operating strictly from operational definitions should in principle remain robust to such perturbations. Future probes could also test actor-role perturbations, such as reversing source and target roles or evaluating the same sentence under different actor pairs. Finally, although we report bootstrap confidence intervals for the main PLV predictive comparisons, the behavioral reliability probes are still reported mainly as averaged point estimates. Future work should add bootstrap or resampling-based uncertainty estimates for Rule-S and probe-level differences when per-example probe outputs are available.

Ethical Considerations

This work evaluates LLM-based political event coding as a measurement tool. Such systems should not be used to produce social-science data without auditing, uncertainty reporting, and human review for cases that fail behavioral reliability checks. Because event-coded data may inform claims about conflict, cooperation, and political behavior, errors or behavioral reliability failures in coding can affect downstream analysis. This concern is consistent with broader discussions of LLMs in computational social science, where validity, behavioral reliability, and annotation quality remain central issues (Ziems et al., 2024). We plan to release prompts, probe templates, and evaluation scripts where licensing permits. The behavioral tests proposed here are intended as diagnostic tools for identifying such risks, not as a replacement for expert validation.

References

Robert Adcock and David Collier. 2001. Measurement validity: A shared standard for qualitative and quantitative research. *American Political Science Review*, 95(3):529–546.

Ken Barker, Parul Awasthy, Jian Ni, and Radu Florian. 2021. IBM MNLP IE at CASE 2021 task 2: NLI reranking for zero-shot text classification. In *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text*, pages 193–202.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ACE) program – tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).

Deborah J. Gerner, Philip A. Schrodt, Omur Yilmaz, and Rajaa Abu-Jabr. 2002. Conflict and mediation event observations (CAMEO): A new event data framework for the analysis of foreign policy. Technical report, American Political Science Association.

Andrew Halterman and Katherine A. Keith. 2025. Codebook LLMs: Evaluating LLMs as measurement tools for political science concepts. *Political Analysis*.

Andrew Halterman and Benjamin J. Radford. 2021. Few-shot upsampling for protest size detection. In *Findings of the Association for Computational Linguistics*.

Yibo Hu, MohammadSaleh Hosseini, Erick Skorupa Parolin, Javier Osorio, Latifur Khan, Patrick T. Brandt, and Vito D'Orazio. 2022. ConfliBERT: A pre-trained language model for political conflict and violence. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5469–5482.

Yibo Hu, Erick Skorupa Parolin, Latifur Khan, Patrick T. Brandt, Javier Osorio, and Vito J. D'Orazio. 2024. Leveraging codebook knowledge with NLI and ChatGPT for zero-shot political relation classification. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 583–603.

Lifu Huang, Heng Ji, Kyunghyun Cho, Ido Dagan, Sebastian Riedel, and Clare Voss. 2018. Zero-shot transfer learning for event extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2160–2170.

Ali Hurriyetoglu, Hristo Tanev, Vanni Zavarella, Jakub Piskorski, Reyyan Yeniterzi, Deniz Yuret, and Aline Villavicencio. 2021. Challenges and applications of automated extraction of socio-political events from text (CASE 2021): Workshop and shared task report. In *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text*, pages 1–9.

Klaus Krippendorff. 2019. *Content Analysis: An Introduction to Its Methodology*, 4 edition. SAGE Publications.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474.

Sha Li, Heng Ji, and Jiawei Han. 2021. Document-level event argument extraction by conditional generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 894–908.

J. Lu and Joydeep Roy. 2017. Universal petrarch: Language-agnostic political event coding using universal dependencies. Available at https://github.com/openeventdata/UniversalPetrarch (2026/05/26).

Charles A. McClelland. 1999. World event/interaction survey (WEIS) codebook. ICPSR 5211.

Kimberly A. Neuendorf. 2018. *The Content Analysis Guidebook*, 2 edition. SAGE Publications.

Abiola Obamuyide and Andreas Vlachos. 2018. Zero-shot relation classification as textual entailment. In *Proceedings of the First Workshop on Fact Extraction and VERification*, pages 72–78.

Open Event Data Alliance. 2018. PLOVER: Political language ontology for verifiable event records. https://github.com/openeventdata/PLOVER.

Erick Skorupa Parolin, MohammadSaleh Hosseini, Yibo Hu, Latifur Khan, Patrick T. Brandt, Javier Osorio, and Vito D'Orazio. 2022. Multi-COPED: A multilingual multi-task approach for coding political event data on conflict and mediation domain. In *Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society*, pages 700–711.

Erick Skorupa Parolin, Latifur Khan, Javier Osorio, Patrick T. Brandt, Vito D'Orazio, and Jennifer Holmes. 2021. 3M-Transformers for event coding on organized crime domain. In *2021 IEEE 8th International Conference on Data Science and Advanced Analytics*, pages 1–10. IEEE.

Ellie Pavlick, Heng Ji, Xiaoman Pan, and Chris Callison-Burch. 2016. The gun violence database: A new task and data set for NLP. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1018–1024.

Benjamin J. Radford. 2021. CASE 2021 task 2: Zero-shot classification of fine-grained sociopolitical events with transformer models. In *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text*, pages 203–207.

Clionadh Raleigh, Andrew Linke, Håvard Hegre, and Joakim Karlsen. 2010. Introducing ACLED: An armed conflict location and event dataset. *Journal of Peace Research*, 47(5):651–660.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992. Association for Computational Linguistics.

Mattes Ruckdeschel. 2025. Just read the codebook! Make use of quality codebooks in zero-shot classification of multilabel frame datasets. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 6317–6337.

Oscar Stuhler, Cat Dang Ton, and Étienne Ollion. 2025. From codebooks to promptbooks: Extracting information from text with generative large language models. *Sociological Methods & Research*, 54(3):794–848.

Nga Than, Leanne Fan, Tina Law, Laura K. Nelson, and Leslie McCall. 2025. Updating "the future of coding": Qualitative coding with generative large language models. *Sociological Methods & Research*, 54(3):849–888.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 3914–3923.

Caleb Ziems, William Held, Omar Shaikh, Jiaao Chen, Zhehao Zhang, and Diyi Yang. 2024. Can large language models transform computational social science? *Computational Linguistics*, 50(1):237–291.

A Prompt Templates and Worked Examples

Throughout the appendix, we use the following running example: "Indonesian students said they would stage mass demonstrations to demand political reforms by President Suharto's government." The source is Indonesian students, the target is President Suharto's government, and the gold label is THREATEN.

A.1 ZSP Hypothesis Examples

ZSP Tiny constructs one hypothesis per Rootcode using only the label name:

> [S] threatened [T].
> [S] protested against [T].
> [S] provided aid to [T].

For the running example, REQUEST receives a higher entailment score than THREATEN in the flat pass. This is because the flat pass has no mechanism for event-mode disambiguation.

ZSP Full replaces label-name hypotheses with codebook-description hypotheses. For THREATEN, the hypothesis states that the source made coercive or forceful warnings with serious potential repercussions directed at the target.

ZSP Tree processes the example in three levels. Level 1 scores Past-mode hypotheses and keeps REQUEST and PROTEST as top candidates. Level 2 branches surviving candidates into Past and Future variants; the future protest variant maps to THREATEN. Level 3 applies the class-disambiguation rule and produces the final prediction THREATEN.

A.2 Prompting Conditions

No Codebook. The prompt asks the model to classify the relation between the marked source and target and choose one label from the valid PLOVER label set. No definitions or examples are provided.

Compact. Compact corresponds to the compact codebook setting used in the main text. Each entry contains the root label, quadcode assignment, and a short natural-language definition. A global disambiguation rule prioritizes Material Conflict over Verbal Conflict when both are plausible.

Enriched. Enriched keeps the same label space as Compact but adds quadcode section headers, worked examples, event-mode guidance, and boundary-case disambiguation notes. The global rules address future-tense cooperation, negated or halted cooperation, peacekeeping forces, CONSULT overuse, and Material Conflict priority. For example, future cooperation is coded as AGREE, completed aid as AID, and halted aid as SANCTION.

CoT. CoT uses the Enriched codebook with an added step-by-step reasoning instruction before the final label prediction. It asks the model to reason through the source, target, main action, event mode, cooperation/conflict status, and final root label before answering. In practice, this produces longer prompts and can amplify misleading keywords or output-format errors.

Representative CoT prompt template. The following template shows the reasoning format used for CoT. The placeholder `<DOCUMENT>` is replaced with the input sentence containing the marked source and target spans.

> **System:** You are a helpful assistant. Follow the requested answer format.
>
> **User:** You are a political event classifier.
>
> **LABEL DEFINITIONS:**
> This block contains the Enriched codebook entries: root label, definition, clarification, examples, and boundary guidance.
>
> **Sentence:** `<DOCUMENT>`
>
> **Think step by step:**
> 1. Who is source, who is target?
> 2. What is the main action?
> 3. Verbal (statements/promises) or material (physical)?
> 4. Cooperative or conflictual?
> 5. Which label fits best?
>
> After reasoning, write final answer as:
>
> **ANSWER:** `<label>`
>
> **Allowed labels:** AGREE, CONSULT, SUPPORT, COOPERATE, AID, YIELD, REQUEST, ACCUSE, REJECT, THREATEN, PROTEST, SANCTION, MOBILIZE, COERCE, ASSAULT.

ICL. The in-context learning prompt gives one labeled demonstration per root class as an input–output pair with a one-line reason. It does not include full codebook definitions.

B RAG Configuration Details

RAG-v1 uses all-MiniLM-L6-v2 sentence embeddings (Reimers and Gurevych, 2019) and three FAISS indices: codebook definition chunks, disambiguation-rule chunks, and labeled examples.

It retrieves input-relevant definitions and examples instead of presenting the full codebook.

RAG-v2 uses all-mpnet-base-v2 embeddings, expands the example bank to 40 contrastive entries, and adds a confusable-neighbor table derived from the ZSP confusion matrix. Its hierarchical variant first predicts the quadcode and then retrieves root definitions within the predicted quadcode.

Both RAG variants use the Enriched codebook as the retrieval source. The main text reports RAG-v1 as the primary RAG result, while RAG-v2 is retained here as an additional diagnostic variant.

C Additional Codebook and Model Comparisons

C.1 JSON Codebook Experiment

The CAMEO manual (Gerner et al., 2002) was parsed into a per-label JSON structure with fields for definition, clarification, negative clarification, positive example, negative example, and disambiguation rules. Despite this richer structure, performance degraded relative to the compact natural-language codebook. The JSON codebook achieves 91.6 binary F1, 73.3 quad F1, and 55.7 root F1, compared with 92.8, 80.3, and 63.8 for the compact codebook.

This result should not be interpreted as evidence against structured representations in general. In this unconstrained prompting setup, JSON formatting may increase the burden of instruction following and output parsing, including the risk of non-label outputs. A cleaner test of structured codebook fields would combine structured inputs with schema-constrained or format-enforced generation.

C.2 Cross-Model PLV Predictive Checks

Table 7 reports the full cross-model PLV predictive results, including binary, quad-level, and root-level macro-F1.

The cross-model results show the same qualitative pattern as the main PLV table. Root-level performance improves from the no-codebook baseline to Compact and is highest under Enriched for all four open-source models. CoT does not consistently improve over the no-codebook baseline and remains below Enriched at the root level for all four models. These results are consistent with the main-text interpretation that enriched codebooks provide useful semantic guidance, while longer prompts may also increase the burden of instruction following and output formatting.

Table 7: Cross-model PLV predictive checks. Scores are macro-F1 on the PLV test set. Model names are shortened for space. Across all four models, root-level performance improves from the no-codebook baseline to Compact and is highest under Enriched; CoT does not consistently improve performance.

Model	Method	Binary F1	Quad F1	Root F1
Gemma2	No Codebook	90.9	65.7	52.0
	Compact	92.7	80.4	63.6
	CoT	90.7	65.5	52.0
	ICL	94.7	71.5	57.3
	Enriched	**96.0**	**86.9**	**71.8**
Qwen2.5	No Codebook	88.2	62.9	50.4
	Compact	89.5	76.7	57.9
	CoT	87.4	62.1	50.3
	ICL	90.3	64.9	51.4
	Enriched	90.0	79.9	61.0
Llama3.1	No Codebook	86.8	55.9	39.5
	Compact	90.8	75.8	52.6
	CoT	85.9	60.8	47.9
	ICL	88.9	59.6	42.9
	Enriched	89.5	76.8	58.7
Mistral	No Codebook	91.3	54.4	40.9
	Compact	89.2	79.7	55.5
	CoT	90.5	62.3	40.5
	ICL	86.3	62.5	44.4
	Enriched	90.0	79.4	61.7

Table 8: Direct binary and quadcode macro-F1 using Gemma2:9B. GPT baselines are from Hu et al. (2024).

Method	Binary F1	Quad F1
Direct No Codebook	90.5	64.3
Direct Compact	94.0	78.4
Direct ICL	90.5	76.2
Direct CoT	52.4	19.2
GPT-4 baseline	93.4	76.7
GPT-3.5 baseline	90.1	66.2

C.3 Direct Binary and Quadcode Classification

Direct binary and quadcode experiments show that codebook information is most useful when the task requires finer distinctions. CoT does not consistently improve performance in this setting, which supports interpreting its predictive failure as partly an output-format problem rather than only a classification error.

C.4 Full Behavioral Probe Details

The main text reports summarized behavioral diagnostics. Tables 9 and 10 provide fuller probe-level details averaged across the four open-source models.

Metric computation. Orig. Acc. is original-condition probe accuracy, computed as the share of examples whose original-codebook prediction matches the gold label. Rev. Acc. and Shuf. Acc.

Table 9: Full PLV behavioral probe details averaged across four open-source models. PLV is evaluated at the root-label level.

Method	Orig.	Rev.	Rev. Chg.	Shuf.	Shuf. Chg.	κ	Gen.	Swap
Compact	0.612	0.615	0.255	0.637	0.209	0.743	0.566	0.105
Enriched	0.619	0.625	0.212	0.640	0.188	0.782	0.519	0.081
ICL	0.454	0.474	0.354	0.491	0.306	0.668	0.152	0.064
CoT	0.642	0.640	0.256	0.667	0.236	0.745	0.625	0.152
RAG-v1	0.633	0.633	0.000	0.633	0.000	1.000	0.522	0.193
RAG-v2	0.516	0.516	0.000	0.516	0.000	1.000	0.044	0.142

Table 10: Full AW behavioral probe details averaged across four open-source models. AW is evaluated as a binary Cooperation/Conflict task.

Method	Orig.	Rev.	Rev. Chg.	Shuf.	Shuf. Chg.	κ	Gen.	Swap
Compact	0.869	0.854	0.153	0.854	0.153	0.755	0.859	0.141
Enriched	0.841	0.869	0.136	0.869	0.136	0.737	0.865	0.109
ICL	0.807	0.824	0.083	0.824	0.083	0.854	0.029	0.148
CoT	0.834	0.825	0.104	0.825	0.104	0.776	0.823	0.143
RAG-v1	0.837	0.872	0.000	0.837	0.000	1.000	0.009	0.137
RAG-v2	0.781	0.781	0.000	0.781	0.000	1.000	0.084	0.157

are accuracies under reversed and shuffled codebook orders. Rev. Chg. and Shuf. Chg. measure the share of examples whose predictions change relative to the original-order prediction. Order κ is Fleiss' κ over the original, reversed, and shuffled prediction sets. Generic F1 is computed against the expected neutral label in the generic-label probe. Swap F1 is computed against the expected label under the perturbed label–definition mapping. Codebook Alignment (**CB-Align.**) is the average of valid-label compliance and definition-recovery accuracy. The Rule-following Score (**Rule-S**) is the average of order κ, Generic F1, and Swap F1. These summary scores are diagnostics and should be interpreted together with the individual probe results.

The fuller behavioral tables reinforce the main-text interpretation. In PLV, Enriched improves order agreement relative to Compact but does not improve Generic F1 or Swap F1. In AW, the binary label space yields stronger original-condition accuracy, but the meaning-level probes still show behavioral reliability failures, especially for ICL and retrieval variants. Both RAG variants show perfect order agreement because retrieval selects local chunks rather than presenting the full ordered codebook; therefore, their order-based agreement should be interpreted together with Generic F1 and Swap F1.

C.5 Qualitative Examples of Swapped-Mapping Failures

The following examples illustrate representative swapped-mapping failures. Original denotes the gold label and original prediction. Swapped denotes the expected label under the perturbed label–definition mapping and the model prediction. These cases show that models often preserve the original semantic label after the mapping is changed.

Table 11: Representative swapped-mapping failures on PLV.

Method	Example
CoT	*Text:* Merkel met Erdogan for talks on a migration deal. Original: CONSULT → CONSULT. Swapped: COOPERATE → CONSULT.
ICL	*Text:* Merkel met Erdogan for talks on a migration deal. Original: CONSULT → CONSULT. Swapped: COOPERATE → CONSULT.
RAG-v1	*Text:* North Korea threatened to shell the Christmas tree. Original: THREATEN → THREATEN. Swapped: AID → THREATEN.
RAG-v2	*Text:* North Korea threatened to shell the Christmas tree. Original: THREATEN → THREATEN. Swapped: AID → THREATEN.