Context-Awakens at Touché: Generalizable Argument Identification with In-Context Learning

Touché at CLEF 2026

Johannes Widera[1,2,*]

[1]Heinrich Heine University Düsseldorf, Düsseldorf, Germany
[2]codecentric AG, Germany

Abstract

Argument identification is usually treated as sentence-level binary classification. State-of-the-art models perform well on the dataset they are trained on but generalize poorly out of distribution, in part because they rely on dataset-specific lexical shortcuts rather than on argumentative structure. The GAIC shared task at Touché 2026 targets this gap: the goal is a system that generalizes to unseen datasets by exploiting the context in which a sentence is labeled, instead of learning shortcuts. We approach this by keeping the model parameters fixed and moving the dataset-specific argument definition, annotation guidelines, and document context into the prompt. We evaluate three instruction-tuned decoder models in a cumulative context ladder and with input-manipulation diagnostics. The dataset-specific argument definition provides the largest performance gain, raising mean macro-F1 by 0.10 to 0.15 across all three models, while guidelines and document context help only when they match the scope of the annotation rule. Under content-removal manipulations comparable to those used for encoder systems, our models lose 0.16 to 0.19 macro-F1 on average, far more than the at most 0.05 reported for fine-tuned encoders; under a stronger word-order shuffle they drop 0.21 to 0.29, indicating that the decoder setup is considerably more sensitive to structure-disrupting transformations. The official Main evaluation scores the 340 sentences of the held-out TACO dataset re-annotated under three different guideline schemes (TACO, TAPE, and TAUS), so a system cannot succeed by reacting to the sentence surface alone. Our full-context submission reaches 0.7955 macro-F1 on this evaluation and ranks first among submitted systems.

Keywords

argument mining, argument identification, shortcut learning, in-context learning, large language models, shared task, generalization

1. Introduction

Argument identification is the binary task of deciding whether a given sentence is argumentative. It sits at the start of most downstream argument mining pipelines. Recent work commonly fine-tunes a pretrained encoder on one annotated dataset and evaluates it on the same dataset distribution [1]. This setup has produced strong results on familiar benchmarks, but it has also hidden a more basic generalization problem: models that perform well on the dataset they were trained on often fail when the annotation scheme, domain, or dataset-specific cues change. [1] show this directly: most cross-dataset transfer scores fall below the mean in-dataset score of 0.79 macro-F1. In their pairwise manipulation analysis, BERT and DistilBERT change by at most $\Delta = 0.02$, RoBERTa remains unchanged, and WRAP shows the largest drop with $\Delta = 0.05$. These results suggest that part of the apparent progress in encoder-based argument identification may reflect shortcut learning in the sense of [2], rather than robust sensitivity to argumentative structure. At the same time, [1] also show that shortcuts are not the only limitation: differences in argument definitions across datasets further restrict generalization. The GAIC shared task at Touché 2026 was designed around these findings [3, 4]. It collects ten argument mining datasets under one binary label space and provides the metadata that a human annotator would normally rely on: the dataset paper, the annotation guidelines where they exist, and the source document from which the sentence was drawn. This

CLEF 2026 Working Notes, 21 – 24 September 2026, Jena, Germany
*Corresponding author.
widera.johannes@gmail.com (J. Widera)

makes GAIC different from a standard sentence classification benchmark. The task does not only ask whether a sentence looks argumentative in isolation. It also asks whether a system can apply the rule under which this sentence should be labeled. We use GAIC to reverse the usual training-based setup. Instead of fitting a new classifier to argument labels, we keep the model parameters fixed and move the dataset-specific information into the prompt. The model receives the argument criterion, the available guidelines, and local document context. The question is therefore how far inference-time conditioning can carry a fixed model when the relevant annotation rule is made explicit. We study this setup in two steps. First, we run diagnostic experiments on the development data to analyze whether LLMs can use the provided context, whether their predictions remain sensitive to sentence structure, and whether benchmark contamination threatens the interpretation. Second, we submit the resulting full-context system to the official GAIC evaluation. The diagnostic part of the study is guided by three research questions:

RQ1. *What context is relevant for GAIC performance?* We test the contribution of the dataset criterion, the annotation guidelines, and the local document context separately, in a cumulative ladder.

RQ2. *Does the system rely on sentence structure, or mainly on surface cues?* We test this with two manipulations: removing discourse markers, function words, and punctuation, following [1], and shuffling word order while keeping the same words.

RQ3. *Does benchmark contamination threaten the conclusions?* We audit pre-training contamination per (model, dataset) pair using the Data Contamination Quiz of [5].

The diagnostic experiments and the official evaluation lead to three main findings. First, argument identification in GAIC is strongly rule-dependent: adding the dataset-specific argument definition produces the largest gain in the context ladder. Second, the manipulation experiments show that the decoder setup is more sensitive to content removal and word-order disruption than the encoder baselines reported by [1]. Third, the full-context system performs strongly in the official Main evaluation. On TACO, TAPE, and TAUS, where the same 340 sentences are labeled under three different annotation schemes, our submission reaches 0.7955 macro-F1 and ranks first among the submitted systems.[1]

2. Related Work

2.1. Encoder-Based Argument Mining

A common approach in argument mining is to use a pretrained bidirectional encoder with a task-specific classification head. Models such as BERT [6], RoBERTa [7], and DeBERTa [8] provide the encoder backbone, while the actual argument identification model is trained on one annotated dataset at a time. This setup has been used widely in previous argument mining work [9] and has produced strong results when training and test data come from the same dataset distribution [1]. The problem is that this success does not transfer cleanly across datasets. [1] show that encoder-based argument identification models generalize poorly when they are evaluated outside their training dataset. Their analysis suggests that this is not mainly a capacity problem. The encoders may contain information about argumentative structure, but the fine-tuned classifiers often rely on dataset-specific shortcut features instead. This matches the broader shortcut learning problem described by [2]: a model can perform well on familiar test data while using decision rules that break under distribution shift. Word-order insensitivity in BERT-style models has also been observed in natural language inference [10] and in text classification more generally [11]. This makes word-order manipulation a useful diagnostic for our setting.

[1] Code, prompts, and extraction pipeline: https://github.com/wideraHannes/GAIC-Thesis.

2.2. LLMs for Argument Mining

Recent work places this shift in a broader methodological development. [9] describe computational argumentation as moving from feature engineering, to model engineering, and finally to adaptation engineering. In the last stage, the model architecture is no longer the main object of change. Instead, researchers adapt large language models through prompts, instructions, retrieval, or task-specific context. [12] survey this development for argument mining and show how subtasks such as claim detection, evidence detection, and relation prediction are increasingly reformulated as prompt-based or retrieval-augmented problems. Our work follows this adaptation-engineering paradigm. We do not fine-tune the model, and the model parameters remain fixed across datasets. Adaptation happens through in-context learning: the dataset criterion, the annotation guidelines where available, the examples contained in those guidelines, and local document context. This lets us test whether an LLM can use the rule that defines the label, rather than learning that rule implicitly from dataset-specific training examples.

3. Problem Definition

3.1. Why Models Fail to Generalize in Argument Identification

Argument identification is often framed as sentence-level binary classification. A standard classifier f_θ maps a sentence x to a binary label:

$$\hat{y} = f_\theta(x), \qquad \hat{y} \in \{\text{Argument, No-Argument}\}. \tag{1}$$

This formulation treats the sentence as if it were sufficient for the decision. In many argument mining datasets, however, the gold label is not produced from x alone. The annotator applies an explicit argument criterion q_d, follows annotation guidelines g_d, and may consider the surrounding context c. The labeling rule of dataset d is therefore better described as

$$y = R^{(d)}_{\text{arg}}\big((x, c)_d;\, q_d, g_d\big). \tag{2}$$

Within one dataset, this gap is easy to overlook. f_θ can absorb an approximation of $R^{(d)}_{\text{arg}}$ from the labels, because the training distribution carries traces of the rule. Across datasets, however, the missing inputs become important. A classifier trained on d approximates $R^{(d)}_{\text{arg}}$, not a universal notion of argumentativeness. On a new dataset d_{n+1}, the sentence on its own does not specify which definition or guideline should be applied. The same x may therefore be argumentative under one annotation scheme and non-argumentative under another.

Besides this underspecification problem, common encoder-head architectures face a second limitation for generalization: shortcut learning. To make the risk explicit, we describe the encoder representation schematically in terms of three possible sources of information:

$$h(x) \rightsquigarrow \{h_{\text{arg}}(x), h_{\text{short}}(x), \varepsilon\}.$$

This notation is conceptual: $h_{\text{arg}}(x)$ denotes information that would support a structurally grounded decision about the argumentative role of the sentence, while $h_{\text{short}}(x)$ denotes surface information that happens to correlate with the label in a particular dataset, such as topic words, lexical overlap, or annotation artifacts. The residual term ε covers the remaining variation in the representation.

During fine-tuning, the classification head is optimized for accuracy on the training distribution. This makes shortcut reliance an intrinsic risk of optimizing the loss on that distribution: if the shortcut signal is predictive, the objective gives the model no reason to prefer the structural signal instead. The dataset rule and the guidelines are therefore learned only implicitly, together with any shortcuts that help on the training set. This is consistent with the behavior reported by [1]: strong in-distribution performance, weak cross-dataset transfer, and little change under structure-disrupting input manipulations.

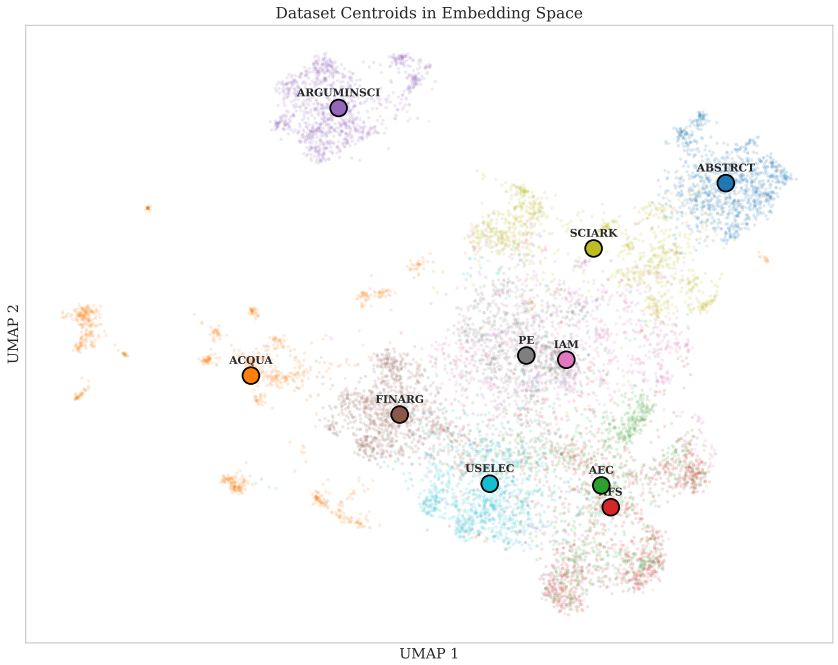

Figure 1: UMAP projection (text-embedding-3-small) of the 10,200 training sentences. Dataset centroids overlap meaningfully along the domain axis, which is part of why a single set of lexical cues will not work across the ten datasets.

3.2. The GAIC Shared Task

The GAIC shared task turns this problem into an explicit evaluation setting. It reframes argument identification as context-grounded classification [3, 4]: the input is a sentence together with optional metadata, such as the dataset definition, annotation guidelines, and source document, and the output is a binary label. We summarize the task's central objective as follows:

> **develop robust systems that use this context to generalize to unseen datasets instead of relying on dataset-specific lexical shortcuts.**

The benchmark consists of ten datasets from diverse domains and genres, including scientific writing, political and online debate, financial communication, educational essays, and web texts. This diversity is visible in Figure 1. The embedding projection shows that the datasets occupy different regions of the input space, with some overlap along broader domain clusters. GAIC therefore does not only test whether a system can classify sentences from a familiar distribution. It tests whether the system can remain stable when the domain, genre, and annotation rule change. The official Main evaluation makes this rule dependence especially clear: TACO, TAPE, and TAUS contain the same sentences but apply different annotation guidelines. A system cannot solve this case by relying only on the sentence surface. It has to condition its prediction on the rule of the target dataset. The methodology follows directly from this task structure. We keep the model fixed, move the dataset-specific rule into the prompt, and test which parts of the available context help the model apply that rule most effectively.

4. Methodology

4.1. System Overview

We treat argument identification as rule-conditioned inference under a fixed model. Let $(x, c)_d$ be a sentence and its (possibly empty) local context from dataset d. The prediction is

$$\hat{y} \;=\; f_\theta\big((x,c)_d;\, \tilde{q}_d, \tilde{g}_d\big), \qquad \theta \text{ fixed}, \tag{3}$$

where $\tilde{q}_d$ and $\tilde{g}_d$ are condensed approximations of q_d and g_d, extracted automatically from the dataset paper and the annotation guideline document. The true q_d and g_d exist only indirectly in long, noisy source material, so we extract approximations of them automatically, in an LLM-friendly form, and the fidelity of those approximations becomes part of what we measure.

4.2. Models

We evaluate three instruction-tuned decoder LLMs: Ministral 8B, Mistral Medium 3.1, and GPT-5.2. The selection gives us a small open-weight model, a mid-scale one, and a frontier one, and as a side effect, three different points on the contamination spectrum. All inferences run at temperature 0. The output is constrained to the binary label space via structured output[2].

4.3. Context Extraction Pipeline

For each dataset we build $\tilde{q}_d$, $\tilde{g}_d$, and c in three stages, summarized in Figure 2. Stage 1 pulls the raw GAIC metadata: dataset paper PDFs (10/10), guideline documents (4/10), and source documents (6/10). For document context, we use two different settings. In the diagnostic experiments on the development split, we use the two sentences immediately preceding each target sentence, extracted with a regex-based sentence splitter. This window was chosen without further tuning to keep the diagnostic grid tractable. For the official shared-task submission, we instead use the full context field provided per sample by GAIC. The diagnostic results on C3 therefore do not directly predict submission behavior on datasets where the GAIC-provided context differs substantially from a two-sentence window; we return to this point in Section 6.5. Stage 2 converts the PDFs to Markdown using Kreuzberg[3]. Stage 3 asks GPT-5.2 to summarize the Markdown into 3–10 sentences of dataset-specific labeling content, under a Pydantic schema. The extraction prompt tells the model to keep only dataset-specific criteria and to skip generic argumentation theory or dataset statistics. Full prompts are in Appendix C.

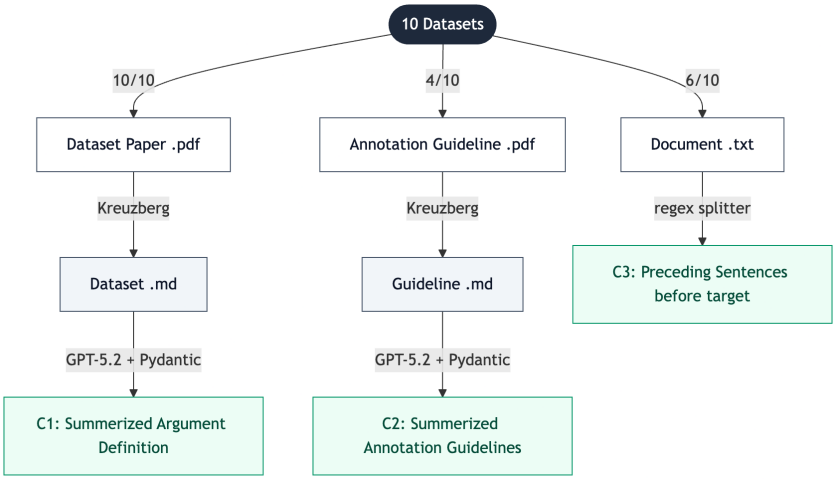

Figure 2: Context extraction pipeline. Stage 1 collects available GAIC metadata (definitions 10/10, guidelines 4/10, document context 6/10). Stage 2 converts PDFs to Markdown. Stage 3 summarizes the Markdown into 3–10 sentences under a Pydantic schema.

4.4. The Context Ladder

We stack the extracted components in a cumulative ladder of four conditions (Table 1). Each level includes everything below it. The full ladder runs on four datasets (ABSTRCT, ARGUMINSCI, PE, USELEC), because those are the ones that provide guidelines and document context. When both definition and

[2] https://developers.openai.com/api/docs/guides/structured-outputs
[3] https://github.com/kreuzberg-dev/kreuzberg

guideline are present, the prompt instructs the model to use the guideline as the operational rule and the definition as background. The full prompt structure for the ladder conditions is shown in Appendix B.

Table 1
The cumulative context ladder; each level adds one context source and keeps those below it. *Coverage* is how many of the ten datasets provide the required context.

Level	Prompt content	Formalization	Coverage
C0	generic instruction	$\hat{y} = f_\theta(x_d)$	10/10
C1	+ $\tilde{q}_d$	$\hat{y} = f_\theta(x_d;\, \tilde{q}_d)$	10/10
C2	+ $\tilde{g}_d$	$\hat{y} = f_\theta(x_d;\, \tilde{q}_d, \tilde{g}_d)$	4/10
C3	+ local context c_d	$\hat{y} = f_\theta((x, c)_d;\, \tilde{q}_d, \tilde{g}_d)$	4/10

4.5. Text Manipulation Experiments

To check whether the predictions actually depend on argumentative structure, we rerun the C1 setting under two manipulations of x. **M0** is the baseline: x unchanged. **M1** (content-only) removes stopwords, function words (ADP, AUX, CCONJ, SCONJ, DET, PART, PRON, INTJ in the spaCy tag set), and punctuation. Only content words remain. This is the manipulation [1] use, and the one for which their encoders changed only slightly. **M2** (shuffle) permutes the words of x at random with a fixed seed, keeping terminal punctuation but otherwise destroying word order. M2 is new relative to [1] and targets word-order sensitivity. Prior work shows that BERT-style models can be surprisingly insensitive to word-order perturbations [10, 11]. For decoder LLMs, shuffling also changes the sequence that the model conditions on during generation. For each manipulation we report the signed difference $\Delta = F_1^{Mx} - F_1^{M0}$. More negative values indicate larger performance drops and therefore stronger reliance on the removed or disrupted information.

4.6. Contamination Audit

We audit contamination using the Data Contamination Quiz of [5]. For 50 sentences per dataset, we ask GPT-4.1 to generate four synonym-preserving perturbations under a Pydantic schema. A first quiz (the Bias Detector) measures the model's position bias in a five-choice multiple-choice setting, presenting the four perturbations alongside a "none of the above" option. A second quiz (the Bias Compensator) then places the original, non-perturbed sentence at the positions where the model is least likely to pick by chance, alongside three of the perturbations. The raw recognition rate gets corrected via Cohen's κ against the measured position bias. The output is a contamination range per (model, dataset) pair. High values indicate that a model may recognize benchmark sentences from pre-training, so performance on those datasets should be interpreted less as clean rule application and more as potentially affected by prior exposure.

4.7. Experimental Setup

All context-ladder and manipulation experiments use 60 balanced sentences per dataset per condition, sampled deterministically from the development split. A full diagnostic grid over all possible combinations would contain 60 samples, 10 datasets, 3 models, 3 input variants (M0, M1, M2), and 4 context conditions (C0–C3), or $60 \times 10 \times 3 \times 3 \times 4 = 21{,}600$ inference calls. Because not all datasets provide all context types, the executed grid is smaller, but this upper bound guided the experimental budget. The test split is reserved for the official shared-task submission. For the diagnostic experiments, we compare against four reference points. The first two are taken directly from [1]: the in-distribution SOTA, where an encoder is trained and evaluated on the same dataset, and the cross-dataset transfer setting, where the encoder is trained on the remaining datasets and evaluated on the target dataset. The latter is the supervised baseline closest to our generalization setting.

The remaining two reference points are evaluated by us on the full development split ($N = 340$ per dataset), separately from the 60-sample ladder and manipulation runs. First, we report our fixed-parameter LLM setup prompted with the highest available context level per dataset (C3 where guidelines and document context exist, C1 otherwise). Second, we add a simple k-nearest-neighbor baseline: all training sentences are embedded with `text-embedding-3-small`, and each development sentence is assigned the majority label of its $k = 50$ nearest training neighbors.

Infrastructure. Experiments are fully parameterized via TOML configuration files specifying the model provider, model identifier, temperature, context sources, and enabled datasets; see Appendix A. At runtime, a unified OpenAI-compatible client dispatches requests to the appropriate backend by switching the `base_url`. Supported providers include OpenAI, Portkey, Mistral AI, Together AI, Groq, and Ollama for local inference. Prompts are assembled modularly: a system message combines the requested context sources (definition, guideline, document context) in fixed order, while the user message contains only the target sentence; see Appendix B. Classification uses structured output with a Pydantic schema constraining responses to `{Argument, No-Argument}`, eliminating post-hoc label normalization. All three text manipulations (M0, M1, M2) for a given sentence are classified in parallel via thread pooling.

5. Results

This section first reports our diagnostic experiments on the development split, then the official submission result on the held-out test set. The context-ladder (Section 5.1) and manipulation (Section 5.2) experiments use a sample size of $N = 60$ per dataset, chosen for faster inference and less computation. The baseline comparison in Table 3 instead uses the full development split ($N = 340$ per dataset), since gold labels were available to us and it serves as our diagnostic reference for the held-out test. The contamination audit (Section 5.3) uses 50 sentences per dataset, as described in Section 4. The final submission (Section 5.4) is evaluated on the official held-out test set.

5.1. RQ1: What context is relevant?

The clearest effect appears at the first context step. Adding the dataset-specific argument criterion already changes the task substantially: mean macro-F1 across the ten public datasets increases from 0.567 to 0.721 for GPT-5.2 ($+0.154$), from 0.606 to 0.709 for Ministral 8B ($+0.103$), and from 0.606 to 0.747 for Mistral Medium ($+0.141$). In comparison, the later context levels have a much smaller effect. Guidelines and document context can still help in individual cases, but they do not produce a second jump of the same magnitude.

On the four datasets where the full ladder is available, the C0 $\rightarrow$ C1 jump averages around $+0.10$. C2 (adding the annotation guideline) gives smaller and uneven improvements. C3 (adding two sentences of document context) is the least reliable step. Sometimes it helps, sometimes it does not. On ABSTRCT, GPT-5.2 climbs from 0.883 at C2 to 0.917 at C3. On PE, more context tends to hurt: every model drops at each step from C1 onward, and C3 falls close to random for some configurations. No other dataset shows this pattern.

AEC is worth a separate mention. It is the easiest dataset in our experiments, because the labels can largely be recovered from four sentence-initial connectives: "But", "If", "So", and "First". The automatically extracted $\tilde{q}_d$ missed this cue, and all three models scored near chance. After we added the connective rule to the definition and reran the condition as a diagnostic check, all three models jumped above 0.9 macro-F1. This shows a simple failure mode of the extraction pipeline: performance depends on whether $\tilde{q}_d$ captures the rule that actually separates the labels.

Averaged across the ten public datasets, our best LLM configuration reaches 0.72 macro-F1. The cross-dataset encoder baseline of [1] averages 0.67, and the KNN retrieval baseline at $k = 50$ averages 0.68. We outperform the encoder baseline on five datasets (ABSTRCT, ACQUA, AEC, AFS, ARGUMINSCI),

while still remaining below the in-dataset Encoder average of 0.82. This is the expected gap for a fixed-parameter system that is not trained on the target datasets. At the same time, the comparison shows that the supplied context is not merely decorative: the prompt carries enough dataset-specific information to close part of the transfer gap.

Table 2
Macro-F1 scores by model, context level, and dataset. Best per column in **bold**.

Model	Ctx	ABSTRCT	ACQUA	AEC	AFS	ARGUMINSCI	FINARG	IAM	PE	SCIARK	USELEC	Avg
gpt_5_2_openai	c0	0.692	0.618	0.562	0.569	0.524	0.405	0.563	0.666	0.601	0.472	0.567
	c1	0.663	0.749	**0.967**	**0.767**	**0.917**	0.438	0.764	0.600	0.717	0.630	0.721
	c2	0.883	–	–	–	0.883	–	–	0.583	–	0.649	0.750
	c3	0.917	–	–	–	0.847	0.540	–	0.524	0.748	**0.738**	**0.757**
ministral_8b	c0	0.738	0.748	0.451	0.514	0.496	0.426	0.764	0.676	0.729	0.520	0.606
	c1	0.833	0.744	0.950	0.699	0.733	0.542	0.681	**0.697**	0.633	0.580	0.709
	c2	**0.950**	–	–	–	0.668	–	–	0.603	–	0.679	0.725
	c3	0.900	–	–	–	0.733	0.563	–	0.474	0.619	0.619	0.682
mistral_medium	c0	0.764	0.732	0.444	0.451	0.576	0.365	0.661	0.641	**0.867**	0.562	0.606
	c1	0.900	**0.764**	0.917	0.766	0.757	0.583	**0.782**	0.603	0.783	0.614	0.747
	c2	0.917	–	–	–	0.753	–	–	0.569	–	0.710	0.737
	c3	0.933	–	–	–	0.753	**0.611**	–	0.547	0.762	0.698	0.733

Table 3
Comparison of argument identification approaches on the GAIC benchmark. Encoder (in-dataset) and Encoder (cross-dataset) are both taken from [1]: the former trains and evaluates an encoder on the same dataset (in-distribution supervised performance), the latter trains on the remaining datasets and evaluates on the target (cross-dataset transfer). KNN and LLM are our own results, computed on the full development split ($N = 340$ per dataset). KNN is a k-nearest neighbor retrieval baseline: each development sentence is embedded, its $k = 50$ nearest training sentences are retrieved, and it is assigned the majority label among those neighbors. LLM is our best fixed-parameter decoder-based approach across the available context conditions. Best result per row excluding the in-dataset encoder in **bold**.

Dataset	**Encoder** *(in-dataset)*	**Encoder** *(cross-dataset)*	**KNN** *(k=50)*	**LLM** *(context-based)*
ABSTRCT	0.89	0.74	0.87	**0.89**
ACQUA	0.84	0.66	0.66	**0.82**
AEC	0.96	0.57	0.60	**0.93**
AFS	0.84	0.60	**0.76**	0.71
ARGUMINSCI	0.84	0.59	0.69	**0.77**
FINARG	0.68	**0.66**	0.61	0.55
IAM	0.76	**0.73**	0.60	0.71
PE	0.78	**0.69**	0.57	0.55
SCIARK	0.83	**0.75**	0.71	0.70
USELEC	0.74	**0.70**	**0.70**	0.58
Mean	0.82	0.67	0.68	**0.72**

5.2. RQ2: Does the system attend to argumentative structure?

Both manipulations cause substantial drops. Under content-only reduction (M1), where [1] report $\Delta \leq 0.02$ for Encoder based models, our three models lose 0.16, 0.19, and 0.17 macro-F1 on average. Under shuffle (M2), the corresponding numbers are 0.21, 0.29, and 0.24. The largest single drop is GPT-5.2 on ARGUMINSCI, where performance falls from 0.917 to roughly chance under both manipulations. We did not see anything like that in the encoder baselines. One detail that surprised us: Ministral 8B, the smallest model in our set, is the most robust to shuffle on average. One possible explanation is that smaller models follow the prompt more literally, whereas larger models may compensate more strongly for degraded input by relying on background knowledge or semantic priors. We treat this interpretation as tentative.

Table 4
Performance degradation under the two input manipulations, measured in the C1 context (definition only) as the signed change $\Delta = F_1^{Mx} - F_1^{M0}$. Each cell reports the M1 / M2 drop: M1 removes function words and punctuation, leaving only content words; M2 shuffles word order. More negative values indicate stronger reliance on the removed or disrupted structure. **Bold** marks the largest drop per dataset within each manipulation type.

Dataset	GPT-5.2	Ministral 8B	Mistral Medium
ABSTRCT	-0.02 / -0.16	**-0.25** / **-0.45**	-0.18 / -0.35
ACQUA	-0.14 / -0.14	**-0.23** / **-0.22**	-0.16 / -0.11
AEC	**-0.33** / **-0.19**	-0.32 / -0.10	-0.28 / -0.07
AFS	-0.07 / -0.18	**-0.11** / **-0.37**	-0.04 / -0.36
ARGUMINSCI	**-0.46** / **-0.48**	-0.31 / -0.40	-0.32 / -0.39
FINARG	-0.08 / -0.10	-0.21 / -0.21	**-0.25** / **-0.25**
IAM	**-0.12** / -0.23	-0.02 / **-0.28**	-0.10 / -0.27
PE	-0.08 / -0.07	**-0.26** / **-0.36**	-0.01 / -0.19
SCIARK	**-0.21** / **-0.30**	-0.09 / -0.30	-0.20 / -0.26
USELEC	-0.08 / **-0.23**	-0.09 / -0.17	**-0.12** / -0.14
Mean	-0.16 / -0.21	-0.19 / -0.29	-0.17 / -0.24

5.3. RQ3: Does benchmark contamination threaten the conclusions?

GPT-5.2 and Mistral Medium show the highest contamination ranges, with several datasets falling into the 40–80% band. Ministral 8B remains below 30% on every dataset; Figure 3. This ordering roughly follows model scale, which is plausible: larger models are more likely to have memorized parts of public benchmark data. We therefore treat contamination as a validity threat for absolute performance scores. High macro-F1 on a contaminated dataset should not be read as clean evidence that the model inferred the annotation rule from the prompt. At the same time, contamination does not by itself explain the main diagnostic patterns we report. The C0 $\rightarrow$ C1 gains are measured as changes within the same model and dataset, and the manipulation experiments use altered inputs rather than the original benchmark sentences. Large drops after content removal or word shuffling therefore suggest that the models still respond to the presented input, not only to memorized labels. However, this does not eliminate contamination as a possible influence: a model that has memorized a sentence may still recognize degraded or partially preserved variants of it. The held-out TACO dataset is especially relevant for the official task evaluation. In our audit, TACO shows one of the lowest contamination levels across all datasets. This does not prove that the final score is free from contamination effects, but it strengthens the interpretation of the official evaluation relative to highly contaminated public datasets. The audit also has a methodological limitation. The Data Contamination Quiz tests whether a model recognizes original benchmark sentences among perturbed alternatives. It does not directly test whether the model has seen the corresponding labels, the annotation guidelines, or the exact GAIC task formulation. For this reason, we use the audit as a conservative warning signal rather than as a correction of the reported macro-F1 scores. Future work should combine this sentence-recognition test with contamination methods that are more directly targeted at label memorization and task-specific exposure.

5.4. Official Submission Results

The development-set diagnostics analyzed the effect of added context, input-manipulation sensitivity, and possible benchmark contamination. They motivated a simple final system design: for each dataset in the official GAIC submission, we used the maximum context available for that dataset. Depending on the dataset, this includes the dataset-specific definition, annotation guidelines where provided, and the available document context. All official test-set predictions were generated with GPT-5.2, see Appendix D for more details. Table 5 shows the final leaderboard scores on the evaluation only datasets. Our submission, *Full GAIC Testset*, reaches a Main score of 0.7955 macro-F1 and ranks first

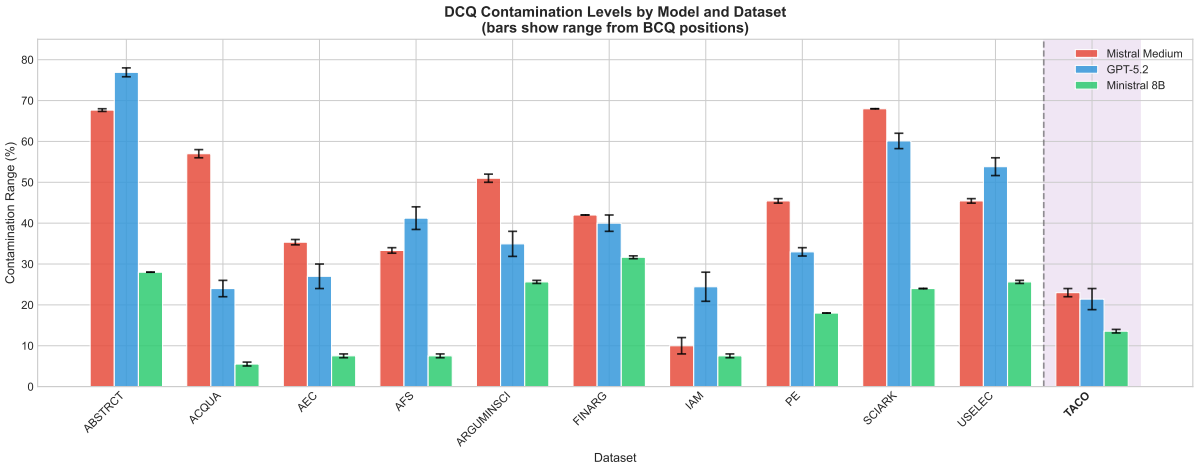

Figure 3: Data Contamination Quiz (DCQ) estimates per (model, dataset) pair. Bar height is the central estimate; whiskers show the range across Bias-Compensation-Quiz (BCQ) positions. The shaded band marks TACO, the held-out evaluation dataset.

overall. The Main score is particularly informative because it is computed over TACO, TAPE, and TAUS. These datasets contain the same 340 sentences, but label them under different annotation guidelines. A system that only reacts to the surface form of the sentence should therefore struggle in this setting. The result is not driven by one dataset alone: the system scores 0.8408 on TACO, 0.7604 on TAPE, and 0.7853 on TAUS. This supports the central assumption of our approach. The model has to condition its prediction on the dataset rule, and the prompt provides a workable way to expose that rule at inference time. Figure 4 gives another view of this evaluation setting. It projects the TACO test samples into the embedding space of the public GAIC training data. The TACO instances do not form a cleanly separated cluster. Instead, they lie mostly in the central region of the benchmark space, with overlap around debate style, political, and mixed domain datasets. This makes the result more interesting. TACO is not an obvious outlier that can be solved by treating it as a completely new domain, but it is also not identical to one public training dataset. Surface similarity is available, but it is not enough, because the same sentences are evaluated again under the TAPE and TAUS rules.

The scores on the remaining test datasets are broadly in line with the development results in Table 2. The strong Main result therefore seems to come less from a general jump in test performance and more from the structure of the held out evaluation itself. TACO, TAPE, and TAUS provide a comparatively clean version of the problem we target: the sentence set is fixed, the annotation rules are explicit, and the relevant definitions and guidelines can be extracted without much ambiguity. This is exactly the setting in which a context conditioned system should work best. If the rule is clearly specified, the model can use it. If the rule is incomplete, vague, or poorly captured by the extracted context, the same prompting strategy has much less to rely on.

This also gives a broader lesson for generalizable argument identification. The official result is not only evidence for the model and prompt design. It also shows the value of carefully specified annotation guidelines and consistently labeled data. Context can only help if it actually describes the rule that produced the labels. In this sense, the Main evaluation supports both sides of the paper's argument: fixed LLMs can use dataset-specific context effectively, but the quality of that context depends on the quality of the underlying annotation process.

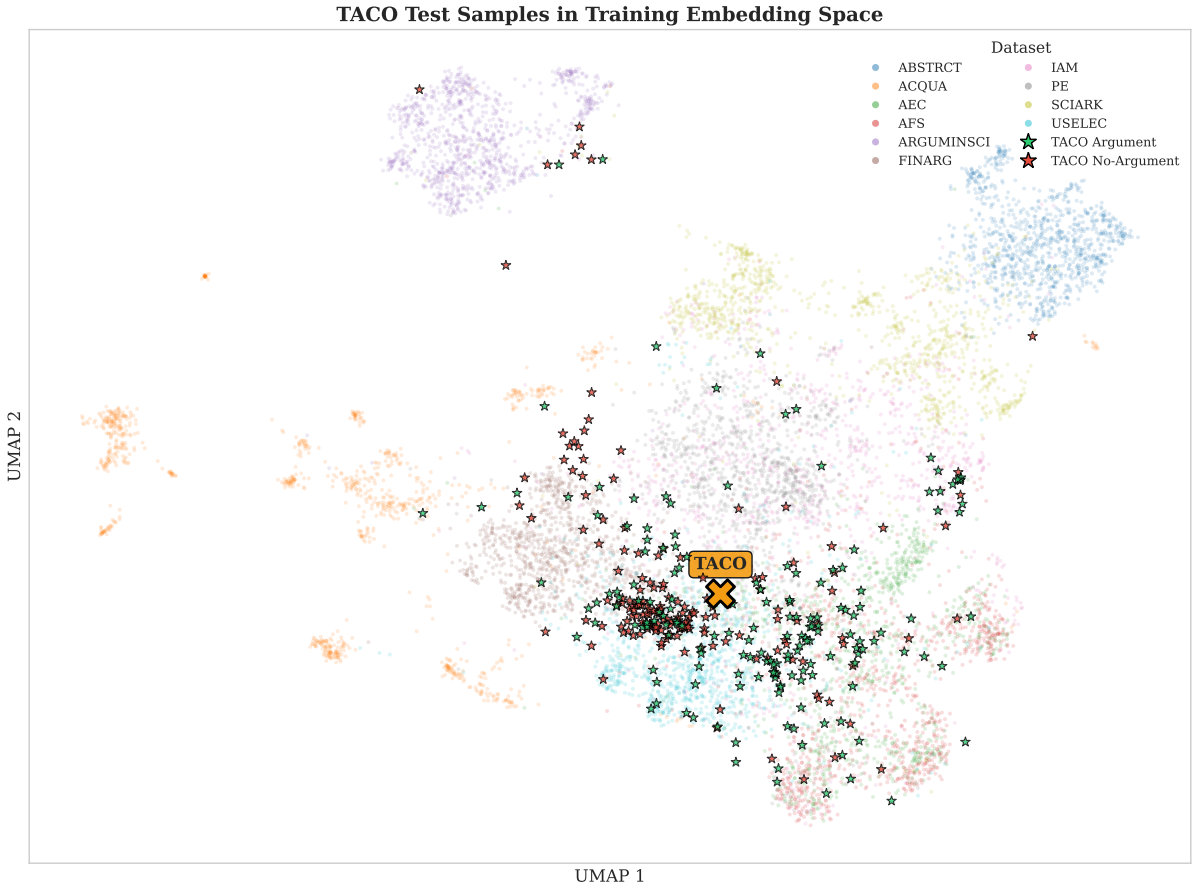

Figure 4: UMAP projection of the TACO test samples in the embedding space of the GAIC training data. Public training sentences are shown by dataset, while TACO test instances are overlaid by their gold label. The TACO centroid lies in the central region of the benchmark space, with substantial overlap around debate and mixed-domain datasets.

Table 5
Official GAIC held-out test set results. The table reports the exact macro-F1 scores on the three evaluation-only datasets used for the official ranking. Main is the average over TACO, TAPE, and TAUS.

Team	Name	TACO	TAPE	TAUS	Main
context-awakens	**Full GAIC Testset**	**0.8408**	0.7604	**0.7853**	**0.7955**
arginvariant	arginvariant_1	0.8133	**0.7757**	0.7612	0.7834
arginvariant	arinvariant_2	0.8133	**0.7757**	0.7598	0.7829
the-wildcards	hybrid	0.8265	0.7465	0.7735	0.7822
the-wildcards	local	0.7912	0.7506	0.7660	0.7693
arginvariant	arginvariant_3	0.8101	0.7204	0.7377	0.7561
code-doctors	run_1	0.6025	0.6748	0.6734	0.6502
the-wildcards	solo	0.5098	0.6216	0.6499	0.5938

6. Discussion

6.1. Effect of Context

The context ladder shows a clear hierarchy. The argument definition has by far the largest impact: adding $\tilde{q}_d$ alone lifts average macro-F1 by 0.10 to 0.15 across all three models, and accounts for almost all of the gain over the C0 baseline. Annotation guidelines add a smaller, less uniform improvement on top. Document context is the most situational of the three. It helps where the rule explicitly demands it, and can hurt where it does not.

The TAPE and TAUS guidelines make this concrete. Both state that a tweet lacking necessary context, for instance one referencing a missing parent tweet, must be labeled No-Argument. A system that sees only the isolated sentence cannot satisfy this rule. It has no way to check whether the necessary context is missing, because it has no surrounding context to check against. This is the strongest form of rule-dependence in the benchmark: the rule does not just move the decision boundary, it makes c a required input. Our strong TAPE and TAUS results are consistent with this interpretation, because these settings require information that is not available from the isolated sentence alone. Where the rule does not demand it, as on PE with its hierarchical essay-scale annotation, the same document window can hurt by implying a scope the binary label cannot express.

The Main evaluation makes this point clearer: TACO, TAPE, and TAUS use the same 340 sentences, but apply different annotation rules. Any system whose prediction depends only on x would score identically on all three. Ours does not. The prompt is the only moving part across the three runs, so it has to account for the differences.

6.2. Extraction Fidelity Is the Bottleneck

C2 and C3 do not behave like C1. Sometimes they help, sometimes nothing changes, and on PE they consistently hurt. Two diagnoses keep recurring. First, extraction fidelity matters more than how much we add. AEC was random until we hand-corrected $\tilde{q}_d$ to include the extractability criterion the automatic pipeline had quietly dropped, after which all three models jumped above 0.9. FINARG looks faithful to the paper-stated definition, but the actual annotation rule seems to encode finer decisions the paper does not document, so $\tilde{q}_d$ is a good approximation of the wrong thing. PE has a hierarchical rule that operates above the sentence level, and a two-sentence window is the wrong unit to approximate it. In each case the bottleneck is the gap between q_d and $\tilde{q}_d$, or between the available window and the scope of the rule, not the model. A simple strategy of adding all available metadata assumes a level of fidelity that the documentation does not always provide, and it also assumes that the model uses long context uniformly. Both assumptions are fragile. [13] show across 18 frontier models that performance degrades non-uniformly as input length grows, even on simple tasks, and that context can contain confusing artifacts, so-called distractors, that interfere with information retrieval.

6.3. Relation to Shortcut Learning

The drops under M1 and M2 create a clear contrast with the encoder results reported by [1]. This contrast has a plausible architectural explanation. Our decoder models are not fine-tuned on GAIC labels and therefore do not receive a task-specific training signal that could turn dataset-level lexical correlations into a classification rule. In addition, causal decoders process tokens in sequence, so word order directly affects the input available to the model. Shuffling the sentence therefore changes more than the surface form of the input. It changes the information available for applying the supplied criterion.

6.4. Implications and Future Work

The shortcut problem identified by [1] therefore remains relevant, but it appears in a different place. By avoiding task-specific training, we also avoid the encoder-head mechanism that can turn dataset correlations into decision rules. The new bottleneck is the prompt: the system can only apply the intended annotation rule if that rule, and the relevant context around the sentence, are represented faithfully enough in the input. Context extraction therefore stops being a preprocessing chore and becomes part of the system. The AEC and TACO cases make this point from opposite directions: a missing criterion can sink performance even with a frontier model, and a well-matched document window can be the largest single contributor to the final score. We do not yet have a clean way to check whether $\tilde{q}_d$ matches q_d before running experiments rather than after.

The architectural prior also matters more than usually admitted. The shuffle experiment draws a sharp empirical line between encoders and decoders on a property the two families treat differently by

design. This is not a recommendation to always use decoders. Encoders remain the right answer for in-distribution performance on a single dataset. A natural next step is guideline-conditioned fine-tuning in the style of [14], which keeps annotation rules explicit in the input rather than hiding them in a classification head, and could improve robustness while preserving the system's central idea: the model should apply the rule it is given, not infer one from the data.

6.5. Limitations

A few limitations qualify these results. We do not measure extraction fidelity directly. The AEC case shows that an incorrectly extracted $\tilde{q}_d$ can dominate performance, but for the other datasets we infer extraction quality only from downstream scores. The DCQ audit measures recognition of original benchmark sentences, not whether the model has seen the corresponding labels, the guidelines, or the GAIC task format, and should be read as a warning signal rather than a complete account of leakage. The diagnostic experiments and the official submission use different document-context settings. The ladder experiments use a fixed two-sentence window, while the submission uses the full context field provided by GAIC. C3 results on the development split therefore characterize the two-sentence setting, not the submission setting. We evaluate three instruction-tuned decoder models and deliberately avoid fine-tuning. Whether fine-tuning would reintroduce shortcut behavior remains open. The diagnostic experiments use 60 balanced samples per dataset and condition, which keeps the grid tractable but means small dataset-level differences should not be overinterpreted.

7. Conclusion

We approached GAIC as a rule-conditioned classification problem. Instead of training a new classifier for each dataset, we kept the model fixed and supplied the dataset-specific rule through the prompt. This framing matches the structure of the task: the sentence alone is often not enough, because the meaning of *Argument* changes with the annotation scheme. The results show that this rule matters. Adding the dataset-specific argument definition produces the largest performance gain across the context ladder. Guidelines and document context are more fragile. They can help when they sharpen the applicable rule, but they can also hurt when they introduce the wrong scope or when the extracted context does not match the actual labeling process. The AEC case makes this especially clear: performance was not limited by the model, but by whether the extracted criterion captured the central annotation rule.

The manipulation experiments point in the same direction. Unlike the encoder results reported by [1], the decoder setup drops substantially when content words are isolated or word order is destroyed. This does not prove that the model has a full structural understanding of arguments, but it does show that the predictions are not as stable under shortcut-preserving manipulations as the encoder baselines were. The official Main evaluation gives the strongest task-level evidence: on TACO, TAPE, and TAUS, where the same sentences are labeled under different rules, our system ranks first with 0.7955 macro-F1.

The shortcut problem therefore does not disappear. It changes form. In the encoder setting, the risk is that the model learns dataset-specific correlations during training. In our setting, the risk is that the prompt does not faithfully capture the rule that produced the labels. The next step is to close this extraction gap: definitions and guidelines need to be recovered more reliably, without relying on manual inspection when the automatic summary misses the decisive criterion.

A second direction is to revisit fine-tuning under this constraint. We deliberately avoided the encoder-head setup, because it is exactly the setting in which dataset-specific shortcuts can become attractive to the loss. But fine-tuning does not have to mean hiding the task rule in a classification head. Approaches such as [14], fine-tune models to follow structured guidelines and output schemas, keeping the annotation rules explicit in the input. This kind of guideline-conditioned fine-tuning could improve robustness while preserving the main idea of our system: the model should apply the rule it is given, not infer a shortcut from the dataset distribution.

Declaration on Generative AI

During the preparation of this work, the author used generative AI tools to support language editing, grammar correction, and formulation of draft passages. The author reviewed and edited all generated suggestions and takes full responsibility for the final content of the paper.

References

[1] M. Feger, K. Boland, S. Dietze, Limited generalizability in argument mining: State-of-the-art models learn datasets, not arguments, in: W. Che, J. Nabende, E. Shutova, M. T. Pilehvar (Eds.), Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Vienna, Austria, 2025, pp. 23900–23915. URL: https://aclanthology.org/2025.acl-long.1164/. doi:10.18653/v1/2025.acl-long.1164.

[2] R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, F. A. Wichmann, Shortcut learning in deep neural networks, Nat. Mach. Intell. 2 (2020) 665–673.

[3] J. Kiesel, M. Feger, T. Hagen, S. Heineking, M. Heinrich, M. Fröbe, K. Boland, C. Mathews, W. Pertsch, J. Romberg, I. Zelch, S. Dietze, M. Hagen, M. Potthast, B. Stein, Overview of touché 2026 — argumentation systems, in: M. Hagen, M. Potthast, B. Stein, P. Schaer, E. Zangerle, S. MacAvaney, J. M. Struß, E. S. Salido, A. Barrón-Cedeño, A. G. S. Herrera (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. 17th International Conference of the CLEF Association (CLEF 2026), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2026.

[4] J. Kiesel, M. Feger, T. Hagen, S. Heineking, M. Heinrich, M. Fröbe, K. Boland, C. Mathews, W. Pertsch, J. Romberg, I. Zelch, S. Dietze, M. Hagen, M. Potthast, B. Stein, Overview of touché 2026 — argumentation systems — extended version, in: E. S. Salido, A. Barrón-Cedeño, A. G. S. Herrera, S. MacAvaney, J. M. Struß (Eds.), CLEF 2026 Working Notes, CEUR Workshop Proceedings, CEUR-WS.org, 2026.

[5] S. Golchin, M. Surdeanu, Data contamination quiz: A tool to detect and estimate contamination in large language models, arXiv [cs.CL] (2023).

[6] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: J. Burstein, C. Doran, T. Solorio (Eds.), Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. URL: https://aclanthology.org/N19-1423/. doi:10.18653/v1/N19-1423.

[7] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, RoBERTa: A robustly optimized bert pretraining approach, 2019. URL: https://arxiv.org/abs/1907.11692. arXiv:1907.11692.

[8] P. He, X. Liu, J. Gao, W. Chen, DeBERTa: Decoding-enhanced bert with disentangled attention, 2021. URL: https://arxiv.org/abs/2006.03654. arXiv:2006.03654.

[9] L. Shi, F. Giunchiglia, H. Wang, Y. Cheng, R. Song, D. Shi, X. Diao, H. Xu, From text mining to intelligent debate: Task frameworks and technological evolution in computational argumentation, Inf. Process. Manag. 63 (2026) 104465.

[10] K. Sinha, P. Parthasarathi, J. Pineau, A. Williams, UnNatural language inference, arXiv [cs.CL] (2020).

[11] J. Hessel, A. Schofield, How effective is BERT without word ordering? implications for language understanding and data privacy, in: C. Zong, F. Xia, W. Li, R. Navigli (Eds.), Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), Association for Computational Linguistics, Online, 2021, pp. 204–211. URL: https://aclanthology.org/2021.acl-short.27/. doi:10.18653/v1/2021.acl-short.27.

[12] H. Li, V. Schlegel, Y. Sun, R. Batista-Navarro, G. Nenadic, Large language models in argument mining: A survey, arXiv [cs.CL] (2025).
[13] K. Hong, A. Troynikov, J. Huber, Context Rot: How Increasing Input Tokens Impacts LLM Performance, Technical Report, Chroma, 2025. URL: https://trychroma.com/research/context-rot.
[14] O. Sainz, I. García-Ferrero, R. Agerri, O. L. de Lacalle, G. Rigau, E. Agirre, Gollie: Annotation guidelines improve zero-shot information-extraction, 2024. URL: https://arxiv.org/abs/2310.03668. arXiv:2310.03668.

A. Experiment Configuration Schema

Each experiment is defined by a TOML configuration file. Listing 1 shows a representative configuration for the full-context condition (C3) using GPT-5.2.

Listing 1: Example TOML configuration for the C3 (full context) condition.

```toml
[experiment]
experiment_name = "c3"
sample_size = 60
output_dir = "experiments/v3/gpt_5_2_openai"

[llm]
provider = "openai"
model = "gpt-5.2-2025-12-11"
temperature = 0.0

[context]
sources = ["definition", "guideline", "document_context"]

[datasets]
enabled = ["ABSTRCT", "ARGUMINSCI", "PE", "USELEC"]
```

[experiment] Metadata for the run. `experiment_name` identifies the context condition (here C3). `sample_size` specifies the number of balanced samples per dataset (30 Argument + 30 No-Argument). `output_dir` sets the destination for result files.

[llm] Model configuration. `provider` selects the API backend (e.g., `openai`, `mistral`, `portkey`). `model` specifies the exact model identifier. `temperature` is set to 0.0 for deterministic outputs.

[context] Defines which context sources to include in the prompt. Options are `definition` (generic argument definition), `guideline` (dataset-specific annotation rules), and `document_context` (preceding sentences from the source document). Context sources are assembled in this fixed order.

[datasets] Lists the datasets to evaluate. Here, only the four datasets with full context availability (ABSTRCT, ARGUMINSCI, PE, USELEC) are enabled.

B. Prompt Structure

Figure 5 shows the prompt structure used for the context ladder. C0 uses only the task instruction and the target sentence. C1 adds the dataset-specific argument definition, C2 adds the annotation guideline, and C3 adds local document context.

System Message

```
You are an expert in argumentation analysis.

C1+  ## Argument Definition
     [dataset-specific definition]

C2+  ## Annotation Guideline
     [operational rules; overrides definition]

C3   ## Document Context
     [preceding sentences, per-sample]

## Task
Classify whether the following sentence is an argument
based on the criteria above.
```

User Message

```
{sentence}
```

Figure 5: Prompt structure by context level. C0 omits all colored blocks and has a simplified task instruction. Each subsequent level adds context cumulatively.

C. Context Extraction Prompts

This appendix lists the prompts used to extract dataset-specific argument definitions and annotation criteria from the GAIC metadata. The placeholders `{dataset_name}`, `{paper_text}`, and `{guidelines_text}` are filled automatically during preprocessing.

C.1. Argument Definition Extraction

System prompt.

```
You are an academic text analyst. Read the following dataset paper
and extract the definition used to distinguish argumentative from
non-argumentative sentences.

Note: The paper may define multiple argument component types
(e.g., claims, premises, evidence). Treat ALL argumentative
component types as "Argument".

Output a single concise paragraph (3-5 sentences) that describes
what counts as an argument in this dataset and, only if the paper
explicitly states it, what does not.

Rules:
- Use the paper's own terminology but avoid annotation-tool jargon
- Extract ONLY from this paper's own definition, not from cited
  related work
- Do not invent or infer criteria for non-argumentative text if
  the paper does not explicitly describe them
- No examples, no elaboration, no hedging
```

User prompt.

```
Dataset: {dataset_name}
```

```
--- PAPER TEXT START ---
{paper_text}
--- PAPER TEXT END ---

Synthesize the argument definition according to the schema.
```

C.2. Annotation Guideline Extraction

System prompt.

```
You are an annotation guideline analyst. Read the following
annotation guidelines and extract operational decision rules
for annotators.

Output a concise paragraph (maximum 8 sentences) that describes
how annotators decide whether a sentence is argumentative or not.
Focus on practical decision rules, signal phrases, and boundary
cases, not on restating the theoretical definition. If the
guidelines contain worked examples showing annotation decisions,
include 2-3 boundary examples with their labels and brief
reasoning.

Rules:
- Focus on how to decide ambiguous cases
- No elaboration, no hedging
- If the guidelines contain no operational rules beyond the
  theoretical definition, write "No additional decision rules
  found in guidelines."
```

User prompt.

```
Dataset: {dataset_name}

--- GUIDELINES TEXT START ---
{guidelines_text}
--- GUIDELINES TEXT END ---

Synthesize the annotation criteria according to the schema.
```

D. Official Submission Config

Listing 2: Official Submission config

```
[llm]
provider = "openai"
model = "gpt-5.2-2025-12-11"
temperature = 0.0
reasoning = false

[submission]
context_strategy = "dynamic"  # c0, c1, or dynamic
input_file = "test.jsonl"     # test.jsonl or dev.jsonl for validation
output_dir = "submissions/gpt5.2_experiment"
max_workers = 8               # parallel API calls
```