Overview of the Reasoning Trajectory Detection Task at PAN 2026

Minh Ngoc Ta[1,*], Kaiyang Wan[2], Yan Lin[2,3], Yuxia Wang[2] and Preslav Nakov[1]

[1]*Mohamed bin Zayed University of Artificial Intelligence, United Arab Emirates*
[2]*INSAIT, Sofia University "St. Kliment Ohridski", Bulgaria*
[3]*Newcastle University, United Kingdom*

pan@webis.de *https://pan.webis.de*

Abstract

Reasoning-capable large language models increasingly expose intermediate reasoning trajectories before their final answers. These trajectories can help developers and auditors inspect model behavior, but they also create new origin and safety questions: Who produced the reasoning, a human or an AI system, and can unsafe or misaligned behavior be detected inside the reasoning process before or in spite of the final answer? The PAN 2026 Reasoning Trajectory Detection task addresses these questions with two subtasks. Subtask 1, Source Detection, asks systems to classify a problem solution as human-written or LLM-generated. Subtask 2, Safety Detection, asks systems to predict both a trajectory-level safety label and step-level unsafe reasoning labels.

Our task attracted 16 participating teams. The final leaderboards include nine evaluated systems for Subtask 1 and seven for Subtask 2, along with six notebook papers. In Subtask 1, the best system achieved 0.85 macro-F_1. In Subtask 2, the best system achieved a trajectory-level F_1 of 0.75 and a step-level score of 0.31 soft-F_1. Our analysis shows that the decisive factors were not model size alone but also evidence selection, robustness to distribution shift, decomposition of the decision, and calibration.

The data, baselines, and code are available in the task repository.[1]

Keywords

Reasoning Trajectory Detection, LLM-Generated Text Detection, LLM Safety, Chain-of-Thought Analysis, Reasoning Language Models, Text Forensics, PAN

1. Introduction

Large language models (LLMs) increasingly produce not only final answers, but also visible intermediate reasoning steps. Chain-of-thought prompting showed that explicit intermediate reasoning can improve performance on complex tasks [1]. Newer reasoning-oriented models go further by generating long trajectories that may include planning, self-correction, reflection, tool-use decisions, or safety deliberation. These traces can be useful evidence for auditing model behavior, but they should not be treated as transparent or necessarily faithful explanations of a model's internal computation. Rather, they are observable text artifacts that can be inspected, classified, and compared.

The availability of reasoning trajectories changes the object of study in both text forensics and AI safety. From a forensic perspective, a final answer can be short, normalized, or heavily edited, while a reasoning trace may reveal stronger signals about the production process: step organization, verbosity, formatting conventions, uncertainty handling, or consistency between intermediate derivations and the final response. From a safety perspective, a final response may look acceptable while its preceding steps contain unsafe instructions, harmful planning, or other forms of misaligned reasoning. Conversely, a trace may contain risky lexical cues while actually refusing the harmful request or explaining why it should not be fulfilled.

The PAN 2026 Reasoning Trajectory Detection task was introduced to study these two aspects jointly. The task has two complementary subtasks. Subtask 1, *Source Detection*, asks whether a reasoning trajectory and final answer were produced by a human or by an AI system [2, 3, 4, 5, 6]. Subtask 2,

[1] The task repository is available at GitHub.

CLEF 2026 Working Notes, 21 – 24 September 2026, Jena, Germany

*Corresponding author.

Safety Detection, asks whether the reasoning trajectory is safe, and where unsafe reasoning steps occur. This makes the task part of PAN's broader text forensics program, while connecting it to emerging work on reasoning-trace safety, guard models, and fine-grained AI-generated text detection [7, 8, 9].

This overview paper reports the full shared-task setup and the final results. We first define the two subtasks, then describe the datasets, metrics, baseline system, submission procedure, leaderboards, and participant approaches. We conclude with a discussion of empirical lessons that emerged across the six accepted notebook papers: source detection can be strong, but it is highly sensitive to domain shift; validation performance often overestimates test performance; compact evidence views can outperform longer inputs; trajectory-level safety classification is more tractable than step-level localization; and safety systems benefit from decomposition, multilingual robustness, and careful thresholding.

2. Task Definition

The task is defined over reasoning examples that contain a user query or problem, a reasoning trajectory, and a final answer. Depending on the subtask, systems use these fields to predict provenance or safety labels.

2.1. Subtask 1: Source Detection

Subtask 1 is a binary authorship verification task over reasoning-based solutions. Given a problem p and a solution s, where s includes the reasoning trajectory and final answer, the system predicts

$$y_{\text{src}} \in \{\text{human}, \text{LLM}\}.$$

The goal is not merely to detect answer correctness, but to infer whether the trajectory was produced by a human or by an LLM. The task therefore targets provenance cues in structured reasoning text, including formatting, verbosity, solution organization, symbolic style, and consistency between intermediate steps and conclusions.

Subtask 1 differs from ordinary AI-generated text detection in two ways. First, the texts are solution-like rather than open-ended prose. They contain formulas, enumerated derivations, code-like fragments, or financial calculations. Second, the hidden test set includes domains not present in the released training data, which makes simple memorization of generator-specific or math-specific patterns unreliable.

2.2. Subtask 2: Safety Detection

Subtask 2 is a multi-granularity safety classification task over segmented reasoning trajectories. Each instance contains a user query q, a pre-segmented reasoning trace

$$r = (r_1, \ldots, r_n),$$

and a final answer. Systems must predict a trajectory-level safety label and a sequence of step-level labels aligned with the reasoning steps. The trajectory-level label space in the released data is

$$y_{\text{safe}} \in \{\text{safe}, \text{potentially unsafe}, \text{unsafe}\},$$

while the detailed step label is binary:

$$z_j \in \{0, 1\}, \quad j \in \{1, \ldots, n\},$$

where 0 denotes a safe step and 1 denotes an unsafe step.

This setting reflects the fact that unsafe reasoning can be local. A single problematic step may be embedded in an otherwise benign trajectory, and unsafe behavior may arise from the interaction between the user intent, the intermediate plan, and the final response. The task therefore rewards systems that identify both the overall safety state and the location of unsafe reasoning behavior.

Table 1
Dataset sizes for the PAN 2026 Reasoning Trajectory Detection task.

Subtask	Train	Validation	Test
Source Detection	87,556	497	2,617
Safety Detection	6,079	2,200	2,832

Table 2
Subtask 1 source distribution across the released splits.

Source / generator group	Train	Validation	Test	Details
Human	30,392 (34.7%)	100 (20.1%)	873 (33.4%)	Human-written reference solutions.
All LLMs	57,164 (65.3%)	397 (79.9%)	1,744 (66.6%)	Machine-generated solutions.
GPT-5 Nano	30,301 (34.6%)	100 (20.1%)	49 (1.9%)	
Gemini-3 Flash	24,655 (28.2%)	100 (20.1%)	221 (8.4%)	
K2-Think V2	1,677 (1.9%)	97 (19.5%)	220 (8.4%)	
DeepSeek R1	531 (0.6%)	100 (20.1%)	220 (8.4%)	
Other LLMs	–	–	1,034 (39.5%)	14 additional LLMs appear only in the test set.

3. Datasets

The shared task uses separate datasets for the two subtasks, released through the official task repository [10]. Table 1 summarizes the released training, validation splits and the test set which is used for the descriptive analysis in this paper. The figures for the training and validation splits follow the final release used by the official baseline.

Source Detection contains human and LLM solutions; training and validation focus on mathematical reasoning, while the test set introduces broader crowd-sourced, exam, finance, coding, and real-life problem domains. Safety Detection contains segmented reasoning traces with trajectory-level safety labels and binary step-level labels; queries include harmful requests, jailbreaks, and benign prompts with risky lexical cues.

3.1. Subtask 1 Dataset and Test Construction

The Source Detection data consist of problem-solution pairs. Each instance contains a problem statement, a solution, a coarse generator label, and a detailed generator label. The coarse label defines the binary decision space, `human` versus `LLM`. The detailed label records either `human` or the specific model used to generate the solution.

The human-written portion of the released training data is derived from mathematical reasoning corpora, including the AOPS subset of NuminaMath-CoT [11] and Nemotron [12]. The LLM-generated portion was produced by multiple reasoning models, including K2-Think V2, GPT-5 Nano, Gemini-3 Flash, and DeepSeek R1. Using multiple generators reduces dependence on a single model signature and encourages detectors to learn generalized indicators of machine-produced reasoning.

The test set was intentionally constructed to cover broader domains than the training and validation sets. We collected problems from crowd-sourced Internet sources and exam-style repositories, including high-school and university questions in multiple languages. The resulting problem pool covers mathematical reasoning, finance and business reasoning, common-sense and multiple-choice reasoning, coding or computing questions, and real-life quantitative word problems. We then used a larger and newer set of LLMs to generate solutions for the test set. This design makes the hidden evaluation a test of source robustness rather than memorization of a small training-generator set.

The Subtask 1 test set contains 2,083 unique problem statements and 348 problems with more than one associated solution, where the maximum number of solutions for the same problem is five. The coarse label balance is identical to the aggregate training set ratio: 873 human examples and 1,744 LLM examples. While the fine-grained generator distribution is more diverse than the training set. 14 of

Table 3
Subtask 1 test set LLM generator distribution.

LLM generator	Count	%	LLM generator	Count	%
Gemini 3 Flash	221	8.4	GPT 5.2	74	2.8
K2-Think V2	220	8.4	Qwen Turbo	72	2.8
DeepSeek R1	220	8.4	GPT 4.1	70	2.7
Gemini 2.5 Flash Lite	177	6.8	o3	70	2.7
Gemini 2.0 Flash	100	3.8	GPT 5.1	60	2.3
Claude Haiku 3.5	86	3.3	GPT-5 Nano	49	1.9
GPT 4.1 Mini	77	2.9	Gemini 2.0 Flash Lite	44	1.7
Claude Haiku 4.5	76	2.9	Llama 3.3 70B Instruct	43	1.6
DeepSeek V4 Flash	75	2.9	Claude 3.7 Sonnet	10	0.4

the 18 LLM identifiers in the test set are not among the four LLM generators reported in the training and validation splits, and these test-only LLMs account for 1034 examples – 59.29% of the test set. This directly targets the failure mode highlighted by participant analyses: features tied to a particular LLM generator, output marker, or prompt template can have high validation effect size but no support in the test distribution [13, 14].

Besides, the area of problem topics gives the following composition: 1,218 mathematics, exam, or advanced-theory examples (46.5%), 574 finance, business, or economics examples (21.9%), 320 other crowd-sourced examples (12.2%), 206 common-sense or general QA examples (7.9%), 188 real-life word problems (7.2%), and 111 coding or computing examples (4.2%). From the perspective of languages, 2,112 problems are English or mostly English (80.7%), while 505 examples (19.3%) are non-English or mixed. Dominant scripts include Latin, Cyrillic, Bengali, Japanese, Mandarin, and Korean. Therefore, the intended hidden test set construction is not only multi-generator, but also multi-domain and multilingual.

The text-length distribution also differs by source. In the test set, human solutions have a median length of 951 characters, while LLM solutions have a median length of 1,414 characters. The mean LLM solution length is much larger, 11,671 characters, because several reasoning models produce extremely long traces. The longest LLM solution has 215,703 characters. This heavy-tailed length distribution explains why truncation, chunking, and solution-only evidence selection became important design choices in participant systems [14, 15, 16, 17].

3.2. Subtask 2 Dataset and Test Construction

The Safety Detection data are based on ReasoningShield [8]. Each instance contains the user query, a segmented reasoning trace, the global safety label, and a detailed binary vector of step-level annotations. Unlike the Subtask 1 test set, the Subtask 2 test set does not expose generator identities, so the analysis below focuses on labels, steps, length, language/script, and query construction.

The test set was constructed by prompting multiple LLMs under a range of safety-stressing conditions. We attempted to collect both safe and unsafe reasoning traces by using direct harmful requests, role-play and hypothetical framings, educational or fictional reframings, multilingual prompts, and other jailbreak-style transformations that obscure or soften unsafe intent. We also included benign prompts that contain risky lexical cues but are not harmful themselves. The resulting traces were then manually annotated with a trajectory-level label and binary safe/unsafe labels for each reasoning step. This construction forces systems to reason over the relation among user intent, intermediate reasoning, and response posture rather than relying on isolated keywords.

The Subtask 2 test set is short and multilingual. Test set traces have a median of four annotated steps. 2,491 traces of 2,832 (88.0%) have at most five annotated steps. The median query length is 82 characters and the median reasoning-trace length is 314 characters. There are 1,140 queries in English or mostly in English (40.3%), 966 in non-English or mixed Latin (34.1%), 257 in Cyrillic (9.1%), 253 in Mandarin

Table 4
Subtask 2 trajectory-level label distribution across splits.

Trajectory label	Train	Validation	Test	Shift pattern
Safe	5,150 (84.7%)	1,238 (56.3%)	1,281 (45.2%)	Safe examples dominate training but are less than half of valid test.
Potentially unsafe	41 (0.7%)	334 (15.2%)	491 (17.3%)	Boundary cases are uncommon in training but common in validation and test.
Unsafe	888 (14.6%)	628 (28.5%)	1,060 (37.4%)	Unsafe examples become substantially more common in test.

Table 5
Subtask 2 Test set step-level label distribution.

Trajectory label	Rows	Total steps	Mean steps	% unsafe	Rows with unsafe step
Safe	1,281	5,591	4.36	1.4%	73 (5.7%)
Potentially unsafe	491	2,150	4.38	50.3%	474 (96.5%)
Unsafe	1,060	2,571	2.43	78.9%	1,056 (99.6%)
All valid rows	2,832	10,312	3.64	30.9%	1,603 (56.6%)

(8.9%), 81 each in Japanese and Arabic (2.9% each), and 37 in Korean (1.3%). This confirms participant observations that the test set is much more multilingual and much shorter than the released training and validation sets [18, 13, 14].

The step statistics clarify why Subtask 2 is hard. The global label is not simply a function of whether any unsafe step appears: safe trajectories are almost always clean but still contain a small number of unsafe-step annotations, while both potentially unsafe and unsafe trajectories usually contain at least one unsafe step. The distinction between potentially unsafe and unsafe is therefore closer to a severity, density, and context judgment: unsafe traces have a much higher unsafe-step density (78.9%) and are shorter on average, while potentially unsafe traces contain a mixture of safe and unsafe reasoning steps. This annotation pattern makes trajectory-level classification non-trivial.

4. Submission and Evaluation

Submissions were file-based. Participants trained and ran their systems on their own hardware or services and submitted prediction files in the required CSV format. This design made it possible to use open-weight models, local GPUs, cloud infrastructure, external APIs, or rule-based systems, while evaluating all predictions with the same scripts.

4.1. Subtask 1 Metric

For Subtask 1, systems are ranked by macro-averaged F_1 over the two source classes:

$$\mathrm{MacroF1}_{\mathrm{src}} = \frac{F_1(\mathrm{human}) + F_1(\mathrm{LLM})}{2}.$$

Accuracy is reported as a secondary metric. Macro-F_1 was selected as the primary metric because it gives equal importance to the human and LLM classes even when their frequencies differ.

4.2. Subtask 2 Metrics

Subtask 2 uses two complementary metrics. The primary ranking metric is trajectory-level macro-F_1 after mapping the task to binary classification between safe and unsafe outputs. The "potentially unsafe"

Table 6
Accepted notebook papers and their main modeling choices.

Team / notebook	S1	S2	Main approach
Writerslogic [13]	✓	✓	Feature robustness analysis, frontier LLM agreement for source detection, and query-refusal decomposition with multi-signal union for safety.
DUAN [15]	✓	✓	Qwen3.5-2B full fine-tuning for source detection and LoRA-based multi-task Qwen adaptation for trace and step safety labels.
TUKE-AI [14]	✓	✓	RoBERTa chunked long-context classification for source detection and XLM-RoBERTa step embeddings with BiLSTM-attention trace modeling for safety.
Asdkkllk [16]	✓		ModernBERT-base, solution-only input, validation threshold calibration, and grouped analysis by generator for source detection.
threshdsnmj [17]	✓	✓	Decision-specific evidence views: RoBERTa short solution-only source lens and XLM-RoBERTa query-trace safety lens with weak step supervision.
Bit-by-Bit [18]		✓	Zero-shot Granite Guardian multi-criterion trace ensemble and LoRA-tuned Qwen3Guard step expert with aggressive thresholding.

label is mapped to the unsafe side for this primary score because these examples were designed as difficult borderline unsafe cases. This yields a conservative binary safety evaluation: systems should flag such cases rather than treat them as safe.

$$\mathrm{F}_{1\mathrm{trace}} = \frac{F_1(\mathrm{safe}) + F_1(\mathrm{unsafe})}{2}.$$

The secondary metric evaluates step-level localization. For each sample i, an F_1 score is computed between the predicted and gold step labels, and the final score is the average across samples:

$$F_1^{(i)} = \frac{2 \cdot \mathrm{TP}_{\mathrm{soft}}}{2 \cdot \mathrm{TP}_{\mathrm{soft}} + \mathrm{FP}_{\mathrm{soft}} + \mathrm{FN}_{\mathrm{soft}}}.$$

Here,

$$\mathrm{TP}_{\mathrm{soft}} = \sum_j \hat{y}_j y_j, \quad \mathrm{FP}_{\mathrm{soft}} = \sum_j \hat{y}_j (1 - y_j), \quad \text{and} \quad \mathrm{FN}_{\mathrm{soft}} = \sum_j (1 - \hat{y}_j) y_j,$$

where y_j is the gold label for step j and $\hat{y}_j$ is the corresponding prediction. The metric supports soft predictions, but many participant systems submitted hard step labels. This score rewards systems that locate unsafe steps rather than only predicting the trajectory-level label.

4.3. Baseline

We supervised fine-tuned RoBERTa [19] as baselines. For Subtask 1, the baseline was fine-tuned as a sequence classifier. For Subtask 2, the segmented trace was decomposed into step-level examples, and the same classifier was applied to obtain binary step predictions. The step-level predictions were aggregated back to the original trace format. The baseline reached 0.55 macro-F_1 and 0.57 accuracy on Subtask 1, and 0.57 trajectory-F_1 with 0.24 soft-F_1 on Subtask 2.

5. Participant Systems and Results

The Reasoning Trajectory Detection task attracted 16 participating teams. The final leaderboards contain nine evaluated systems for Source Detection and seven for Safety Detection. Seven teams submitted notebook papers and six were accepted. Table 6 gives a compact overview of these notebooks.

The accepted systems cover a broad methodological range. Some teams relied on encoder fine-tuning with RoBERTa, XLM-RoBERTa, ModernBERT, or Qwen-style models [19, 20, 21, 22]. Others used guard models, frontier LLMs, or feature-based ensembles. Several notebooks explicitly report that high

Table 7
Subtask 1 Source Detection leaderboard: Systems are ranked by macro-F_1.

Rank	Team	Notebook	Macro-F_1	Accuracy	Approach summary
1	Writerslogic	[13]	**0.85**	**0.85**	Claude Opus and Sonnet agreement with LightGBM fallback.
2	srikarkashyap	No submission	0.83	0.82	–
3	DUAN	[15]	0.80	0.83	Qwen3.5-2B full fine-tuning with binary head.
4	TUKE-AI	[14]	0.75	0.74	RoBERTa-base with chunked problem-solution processing.
5	23020073	No submission	0.72	0.72	–
6	Asdkkllk	[16]	0.71	0.75	ModernBERT-base with solution-only input and calibrated threshold.
7	threshdsnmj	[17]	0.65	0.67	RoBERTa-base short solution-only source lens.
8	sajayrrr	No submission	0.43	0.43	–
9	krrag	No submission	0.30	0.28	–
–	Baseline	–	0.55	0.57	RoBERTa sequence-classification baseline.

validation scores did not reliably predict high performance in hidden test, making domain shift and calibration recurring themes.

5.1. Subtask 1: Source Detection

Table 7 shows the final Source Detection leaderboard. The best system Writerslogic achieved 0.85 macro-F_1 and 0.85 accuracy. The next two systems, srikarkashyap and DUAN, achieved 0.83 and 0.80 macro-F_1. The baseline score of 0.55 was exceeded by seven of the nine ranked submissions.

The Source Detection leaderboard is top-heavy rather than uniformly spread. The mean macro-F_1 across submitted systems is 0.67, but the median is 0.72, and six systems reach at least 0.70. The top three are separated by only 0.05 absolute points, suggesting that several different design choices can recover strong provenance evidence. At the same time, the full range is large, from 0.30 to 0.85. This indicates that the task is not solved by a trivial artifact: systems that select non-transferable cues can fail badly, while systems that select robust cues perform well.

The best score is 0.30 absolute points above the baseline, a relative improvement of roughly 55%. Accuracy and macro-F_1 are closely aligned in this leaderboard, so the ranking is not mainly an artifact of class skew. The important question is therefore not whether source information exists in reasoning traces, but which evidence remains useful after the hidden test set moves beyond the math-heavy training distribution.

The notebook papers identify three recurring sources of robustness. First, source evidence is often concentrated in the solution text rather than in the problem statement. Asdkkllk, DUAN, and threshdsnmj all report that solution-only variants were competitive or more robust than problem-plus-solution variants [16, 15, 17]. This is intuitive because the problem statement is shared input, while the solution contains the authored reasoning style. Second, models that fit the validation distribution too tightly often overestimated their final performance. ModernBERT, RoBERTa chunking, and feature-based systems all obtained high validation scores that did not fully transfer [16, 14, 13]. Third, robustness did not require a single model family. The top system used frontier-model agreement, and the following systems used full Qwen fine-tuning, chunked RoBERTa, ModernBERT, or compact RoBERTa. The common factor is not architecture alone, but the choice of source-pure, shift-tolerant evidence.

Table 8
Subtask 2 Safety Detection leaderboard: Systems are ranked by trajectory-level F_1.

Rank	Team	Notebook	F_1	Soft-F_1	Approach summary
1	Bit-by-Bit	[18]	**0.75**	**0.31**	Granite Guardian seven-criterion ensemble plus Qwen3Guard step expert.
2	srikarkashyap	No submission	0.70	0.17	–
3	Writerslogic	[13]	0.66	0.27	Query harmfulness, multilingual refusal detection, and multi-signal union.
4	DUAN	[15]	0.65	0.16	Qwen3.5-2B LoRA multi-task classifier.
5	threshdsnmj	[17]	0.56	0.09	XLM-RoBERTa trace classifier with weak step supervision.
6	TUKE-AI	[14]	0.44	0.12	XLM-RoBERTa step classifier and BiLSTM-attention trace classifier.
7	krrag	No submission	0.43	0.07	–
–	Baseline	–	0.57	0.24	RoBERTa baseline with step-level decomposition.

Table 9
Aggregate patterns across the official leaderboards. "Best–base" is the absolute gap between the best submitted score and the official baseline on the corresponding metric.

Metric	n	Mean	Median	Best	Best–base	Systems above baseline
S1 macro-F_1	9	0.67	0.72	0.85	+0.30	7/9
S2 trajectory-F_1	7	0.60	0.65	0.75	+0.18	4/7
S2 soft-F_1	7	0.17	0.16	0.31	+0.07	2/7

5.2. Subtask 2: Safety Detection

Table 8 shows the Safety Detection leaderboard. The primary ranking metric is trajectory-level F_1; soft-F_1 is reported as the secondary step-level metric. The Bit-by-Bit system achieved the best trajectory-F_1 score of 0.75, and also the best soft-F_1 of 0.31. The srikarkashyap system ranked second on trajectory-F_1, while Writerslogic ranked third on trajectory-F_1 and second on soft-F_1. The baseline was competitive on step localization: its 0.24 soft-F_1 would place third among the submitted systems on the secondary metric.

The Safety Detection results are qualitatively different from the Source Detection results. As Table 9 shows, the winning trajectory-level score improves on the baseline by 0.18, but the winning step-level score improves on the baseline by only 0.07. Four of seven systems exceed the baseline on trajectory-level F_1, while only two exceed it on soft-F_1. This asymmetry is the central result of Subtask 2: deciding whether a trace is globally unsafe is substantially easier than identifying the exact unsafe steps.

The two safety metrics are related, but they are not interchangeable. Bit-by-Bit ranks first on both metrics, but the next systems shift order: srikarkashyap ranks second on trajectory-F_1 but third on soft-F_1, while Writerslogic ranks third on trajectory-F_1 but second on soft-F_1. threshdsnmj beats TUKE-AI on trajectory-F_1, but TUKE-AI scores higher on soft-F_1. These rank differences show that a system can learn useful global risk cues without localizing the unsafe step well, and conversely that more local predictions can improve the secondary score without guaranteeing a strong trajectory label.

The winning Bit-by-Bit system is especially informative because it separates the two decisions. The trace label is produced by a seven-criterion zero-shot Granite Guardian ensemble, while the step labels come from a LoRA-tuned Qwen3Guard step expert. The notebook reports a trace-F_1 progression from 0.64 for a single criterion, to 0.66 for four criteria, to 0.75 for seven criteria [18]. This suggests that broad safety coverage mattered more than ordinary supervised fine-tuning for the global label. For the step label, the same notebook reports that lowering the unsafe threshold from the default operating

Table 10
Validation-test gaps reported by accepted notebooks. Values compare each notebook's local validation result with the official test score under the same configuration.

Notebook	Task	Configuration	Validation	Test	Gap
DUAN [15]	S1	Qwen full fine-tuning	0.930	0.800	0.130
DUAN [15]	S2	Qwen LoRA trace head	0.671	0.650	0.021
Asdkkllk [16]	S1	ModernBERT calibrated	0.943	0.707	0.236
threshdsnmj [17]	S1	RoBERTa Core128	0.916	0.650	0.266
threshdsnmj [17]	S2	XLM-R Core	0.833	0.560	0.273
TUKE-AI [14]	S1	RoBERTa-base chunked	0.970	0.750	0.220
TUKE-AI [14]	S2	BiLSTM trace classifier	0.858	0.444	0.414
Writerslogic [13]	S1	LightGBM feature model	0.886	0.690	0.196
Writerslogic [13]	S2	XLM-R supervised model	0.834	0.448	0.386
Bit-by-Bit [18]	S2	Granite plus Qwen final system	0.782	0.747	0.036
Bit-by-Bit [18]	S2-step	Incremental Qwen step expert	0.310	0.313	-0.004

point improved step-F_1 from 0.20 to 0.31 [18]. Thus, the winning system combined semantic coverage at the trace level with recall-oriented calibration at the step level.

Writerslogic provides a contrasting but compatible lesson. Rather than directly classifying every reasoning step, the system decomposes safety into query harmfulness and refusal detection, then assigns step labels uniformly from the trace-level decision [13]. This strategy achieved the second-best soft-F_1. The result should not be read as evidence that localization is unnecessary. Instead, it reveals a property of the benchmark and metric: many examples are sufficiently trace-coherent that a strong parent-level decision can be surprisingly competitive for step labels. Future systems should therefore beat a strong trace-propagation baseline, not only a token-level or step-level classifier.

6. Analysis and Discussion

6.1. Robustness Under Train-Test Shift

The most important empirical pattern is the validation-test gap. Table 10 summarizes representative gaps reported in the accepted notebook papers. These values should be interpreted cautiously because teams used slightly different local validation objectives, but the pattern is consistent: most supervised systems looked much stronger on the released validation split than on the hidden test set.

The gaps are large enough to change the interpretation of model selection. For Subtask 1, several encoders reached validation macro-F_1 above 0.90, yet their official test scores ranged from 0.65 to 0.80. For Subtask 2, supervised safety systems show even sharper drops: threshdsnmj reports 0.833 validation trace F_1 but 0.56 on test, TUKE-AI reports 0.858 validation trace F_1 but 0.444 on test, and Writerslogic reports 0.834 validation F_1 using an XLM-R classifier but 0.448 on test [17, 14, 13].

This pattern is not just ordinary overfitting. It reflects systematic distribution shift. The test statistics in Tables 2–5 make this shift concrete. For Subtask 1, 59.8% of test LLM examples come from generator identifiers that do not appear among the training and validation sets, and more than half of the test problems fall outside the math/exam category under our topic heuristic. This supports the Writerslogic analysis that generator-specific and domain-anchored features can die at test time [13]. For Subtask 2, the label prior changes sharply: the safe class falls from 84.7% in training to 45.2% in the test set, while potentially unsafe examples rise from 0.7% to 17.3%. The test traces are also short and multilingual, with a median of four annotated steps and about 40% English or mostly English queries. Bit-by-Bit and Writerslogic independently report the same qualitative shift toward shorter, multilingual test traces [18, 13]. Under this shift, high validation F_1 can reward features that are specific to the released domains, generators, lengths, label priors, or languages.

The notable exception is Bit-by-Bit. Its final trace system drops only 0.036 from validation to test, and its step score is essentially unchanged. The reason appears to be methodological rather than accidental: the trace classifier uses safety-specialized external judgment rather than fitting the released trace distribution, and the step classifier deliberately subsamples at most five steps per training trace, matching the short test-trace regime [18]. This does not prove that guard models always generalize better, but it shows that matching the evaluation regime can matter more than maximizing validation performance.

6.2. Evidence Selection Differed by Subtask

The two subtasks require opposite evidence-selection strategies. For Source Detection, the best-performing systems often benefited from removing context. The problem statement is common to both human and AI solutions, so it can dilute authorship cues. Solution-only models focus on formatting habits, derivational style, verbosity, equation handling, final-answer conventions, and other production traces. This explains why Asdkkllk, DUAN, and threshdsnmj all converged on solution-only variants [16, 15, 17].

However, "solution-only" should not be confused with "surface-only". Writerslogic's feature analysis shows that some very strong surface features had no support in the test set. For example, generator-specific and math-format features that were reliable in training became useless when the domain and generator mix changed [13]. The robust source detector therefore needs a narrow evidence field, but the features inside that field must be broad enough to survive new domains and generators.

For Safety Detection, the opposite is true. Removing context can destroy the label. A step can be unsafe because it advances a harmful user intent, or safe because it refuses that intent. The same phrase may be benign in an educational explanation, dangerous in an operational plan, or safe inside a refusal. This is why the stronger safety systems use query-trace relations, multi-criterion judgments, query-refusal decomposition, or trace-plus-step multi-task learning rather than only isolated step text [18, 13, 15, 17].

This contrast is one of the clearest lessons of the task. Reasoning trajectories are structured evidence, not merely longer text. For provenance, the useful signal is often source-pure and compact. For safety, the useful signal is relational and depends on user intent, intermediate reasoning, and response posture.

6.3. Decomposition Was More Reliable Than Monolithic Classification

The highest-scoring systems rarely treated the task as a single flat classification problem. Writerslogic decomposed source detection into independent LLM judgments with a feature-based fallback, and decomposed safety into query harmfulness, refusal detection, and multi-signal union [13]. Bit-by-Bit decomposed safety into a trace expert and a step expert, using different model families and different calibration strategies for each level [18]. threshdsnmj used separate evidence lenses for source and safety, and weakened step supervision to avoid overwhelming the trace-level objective [17]. DUAN used full-model adaptation for source but parameter-efficient multi-task adaptation for safety, reflecting different compute and data-expansion constraints [15].

This convergence suggests that the task has multiple latent decisions inside each official label. In Source Detection, systems must distinguish human-vs.-AI authorship while ignoring topic, problem difficulty, and generator-specific formatting artifacts. In Safety Detection, systems must infer user intent, detect refusal or compliance, judge risk severity, and localize unsafe steps. A monolithic classifier can succeed if the validation and test distributions match, but under shift it can entangle these factors and fail. Decomposition makes the factors explicit and allows each component to be made more invariant.

The decomposition lesson also explains why model size alone is insufficient. Qwen full fine-tuning performed well in Subtask 1, but a frozen Qwen representation was much weaker [15]. ModernBERT and RoBERTa reached very high validation scores but lower test scores [16, 14]. Conversely, guard-model prompting with carefully chosen criteria outperformed several supervised safety systems [18]. The

decisive factor was not simply more parameters, but whether the system matched the causal structure of the decision.

6.4. Step-Level Localization Is a Distinct Unsolved Difficulty

The low soft-F_1 scores should not be interpreted only as weak modeling. They expose a difficult evaluation setting. Unsafe steps are sparse, the metric is computed per sample, and correct safe-step predictions do not compensate for missed unsafe steps in the same way that accuracy would. Bit-by-Bit explicitly notes that the step metric focuses on unsafe-class recovery, which motivated recall-oriented thresholding [18]. This helps explain why a threshold shift from the default to a low unsafe threshold substantially improved the winning step expert.

There is also a hierarchy problem. A trace-level label can often be predicted from high-level evidence such as a harmful query plus lack of refusal. Step labels require finer distinctions: which exact step crosses from analysis to enablement, whether a caveat converts compliance into refusal, and whether a risky phrase is descriptive or operational. When traces are short, multilingual, or segmented inconsistently, the local decision becomes even less stable.

The leaderboard shows that local modeling and global modeling can disagree. The baseline is weak on trajectory-F_1 compared with the best systems, but competitive on soft-F_1. Writerslogic's uniform step propagation performs well despite being intentionally coarse [13]. TUKE-AI's hierarchical step-to-trace architecture reports strong local validation performance but weak hidden-test scores [14]. Together, these results imply that future work should evaluate step localization against strong parent-label propagation baselines, not only against independent step classifiers.

A practical path forward is to model step labels as structured evidence rather than independent examples. Better systems could jointly predict trace and step labels with constraints such as: a globally safe refusal trace should not receive many unsafe step labels unless those steps are genuinely operational; an unsafe trace may contain both safe setup and unsafe enablement; and potentially unsafe behavior may require calibrated uncertainty rather than a hard unsafe label. The present task makes these issues visible, but the best soft-F_1 of 0.31 shows that localization remains far from solved.

6.5. Calibration Was a Modeling Decision, Not a Post-processing Detail

Thresholding repeatedly changed system behavior. Asdkkllk improved validation macro-F_1 from 0.925 to 0.943 by calibrating the AI probability threshold, but the official test macro-F_1 was 0.707 [16]. threshdsnmj reports that a validation-calibrated safety threshold performed worse on test than a fixed threshold in one of its safety variants [17]. Bit-by-Bit achieved the best step score by using a deliberately low unsafe threshold, increasing recall on unsafe steps [18]. These cases point in the same direction: thresholds are part of the model, and thresholds tuned on a shifted validation split can be harmful.

For future editions, calibration should be reported as a first-class component. Notebook papers should state not only the model architecture and validation score, but also the threshold-selection objective, the predicted class distribution, and the sensitivity of the final score to plausible thresholds. For safety detection in particular, precision-recall curves may be more informative than a single threshold because deployments often value missed unsafe steps differently from over-flagged benign steps.

6.6. Implications for Future Benchmarks and Systems

The results suggest several design principles for future reasoning-trajectory benchmarks. First, if the goal is robust evaluation, an intentionally shifted hidden test set is valuable, but the benchmark should make the shift explicit enough that participants know validation is a pipeline check rather than an IID proxy. Second, result reports should include per-domain, per-generator, per-language, and per-trace-length analyses whenever labels permit. The strongest explanations in the notebooks came from exactly these slices: support overlap for source features, trace length for safety, and multilinguality for refusal detection [13, 18].

Third, the step-level metric deserves continued study. Soft-F_1 is useful because it rewards localization, but the current results show that trace-propagation baselines can be surprisingly strong. A future edition could report additional diagnostics, such as unsafe-step recall, over-flagging rate on safe traces, performance by number of unsafe steps, and consistency between trace and step predictions. These diagnostics would make it easier to tell whether a system truly localizes unsafe reasoning or merely inherits a good parent decision.

For system builders, the main recommendation is to treat reasoning trajectories as structured forensic objects. A robust source detector should isolate authored solution evidence and avoid features tied to a specific generator or mathematical format. A robust safety detector should model the relation between query, trace, and answer, and should separate global risk judgment from local unsafe-step detection. Across both subtasks, the most reliable systems were those that selected evidence intentionally, decomposed the decision, and calibrated for the expected operating regime.

6.7. Limitations of the Shared Task

This first edition of the task has several limitations. The Source Detection training data focus heavily on mathematical reasoning, while the test set includes additional domains. This is intentional, but it makes validation less representative. The Safety Detection data include strong class imbalance, especially for the potentially unsafe class in the final training split. The step-level metric is informative, but its behavior under short traces and trace-coherent labels requires further study. Another limitation concerns the difference between annotation granularity and evaluation granularity: three labels were mapped to two. Finally, file-based submissions enabled flexibility, but they limit exact reproducibility for systems that rely on closed APIs or external services.

Despite these limitations, the task establishes a useful benchmark for two emerging questions: reasoning provenance and reasoning safety. It also provides a common evaluation protocol for comparing supervised encoders, open LLMs, guard models, prompting systems, thresholding strategies, and decomposition-based safety pipelines.

7. Conclusion

The PAN 2026 Reasoning Trajectory Detection task introduced a benchmark for source and safety detection of reasoning traces. Subtask 1 identifies whether a reasoning trajectory and final answer is human-written or LLM-generated. Subtask 2 recognizes whether the trajectory is safe and where unsafe reasoning steps occur. The final leaderboards included nine Source Detection systems and seven Safety Detection systems, with six accepted notebook papers among seven submissions.

The results show that source detection over reasoning trajectories is feasible: the best system reached 0.85 macro-F_1, seven of nine submitted systems exceeded the baseline, and several different modeling families produced competitive results. The deeper lesson is that robust provenance detection depends on source-pure evidence and shift-tolerant cues. High validation scores based on math-specific, generator-specific, or threshold-specific patterns did not reliably transfer to the hidden test set.

For safety detection, the best system reached 0.75 trajectory-F_1, but step-level localization remained difficult, with the best soft-F_1 at 0.31 and only two systems above the baseline on the secondary metric. The strongest systems decomposed the problem into trace-level judgment and step-level evidence, or into query harmfulness and refusal detection. This suggests that unsafe reasoning is not a property of isolated tokens alone, it depends on the relation between user intent, intermediate reasoning, and response posture.

Overall, the task suggests that reasoning trajectories are valuable forensic evidence, but fragile. They reveal useful signals about source and safety, yet these signals are affected by domain shift, multilinguality, trace length, label imbalance, metric design, and calibration. Future work could focus on developing stronger out-of-domain system, better safety localization baselines, multilingual guard models, and architectures that jointly model query, trajectory, answer, and step-level evidence.

Acknowledgments

We thank all PAN 2026 participants for their submissions and notebook papers. We also thank the broader PAN team and CLEF organizers for maintaining the shared-task infrastructure and publication process.

References

[1] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, D. Zhou, Chain-of-thought prompting elicits reasoning in large language models, in: Proceedings of the 36th International Conference on Neural Information Processing Systems, NIPS '22, Curran Associates Inc., New Orleans, LA, USA, 2022, pp. 24824–24837. doi:10.5555/3600270.3602070.

[2] S. Gehrmann, H. Strobelt, A. Rush, GLTR: Statistical detection and visualization of generated text, in: M. R. Costa-jussà, E. Alfonseca (Eds.), Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, ACL '2019, Association for Computational Linguistics, Florence, Italy, 2019, pp. 111–116. URL: https://aclanthology.org/P19-3019/. doi:10.18653/v1/P19-3019.

[3] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, C. Finn, DetectGPT: Zero-shot machine-generated text detection using probability curvature, in: Proceedings of the 40th International Conference on Machine Learning, ICML'23, JMLR.org, Honolulu, Hawaii, USA, 2023, pp. 24950 – 24962. doi:10.5555/3618408.3619446.

[4] M. Abassy, K. Elozeiri, A. Aziz, M. N. Ta, R. V. Tomar, B. Adhikari, S. E. D. Ahmed, Y. Wang, O. Mohammed Afzal, Z. Xie, J. Mansurov, E. Artemova, V. Mikhailov, R. Xing, J. Geng, H. Iqbal, Z. M. Mujahid, T. Mahmoud, A. Tsvigun, A. F. Aji, A. Shelmanov, N. Habash, I. Gurevych, P. Nakov, LLM-DetectAIve: a tool for fine-grained machine-generated text detection, in: D. I. Hernandez Farias, T. Hope, M. Li (Eds.), Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Association for Computational Linguistics, Miami, Florida, USA, 2024, pp. 336–343. URL: https://aclanthology.org/2024.emnlp-demo.35/. doi:10.18653/v1/2024.emnlp-demo.35.

[5] J. Bevendorff, Y. Wang, J. Karlgren, M. Wiegmann, M. Fröbe, A. Tsivgun, J. Su, Z. Xie, M. T. Abassy, J. Mansurov, R. Xing, M. N. Ta, K. A. Elozeiri, T. Gu, R. V. Tomar, J. Geng, E. Artemova, A. Shelmanov, N. Habash, E. Stamatatos, I. Gurevych, P. Nakov, M. Potthast, B. Stein, Overview of the "Voight-Kampff" generative ai authorship verification task at PAN and ELOQUENT 2025, in: Conference and Labs of the Evaluation Forum, CEUR-WS, 2025. URL: https://ceur-ws.org/Vol-4038/paper_277.pdf.

[6] M. N. Ta, D. C. Van, D.-A. Hoang, M. Le-Anh, T. Nguyen, M. A. T. Nguyen, Y. Wang, P. Nakov, D. V. Sang, FAID: Fine-grained AI-generated text detection using multi-task auxiliary and multi-level contrastive learning, in: V. Demberg, K. Inui, L. Marquez (Eds.), Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), EACL '2026, Association for Computational Linguistics, Rabat, Morocco, 2026, pp. 3275–3296. URL: https://aclanthology.org/2026.eacl-long.151/. doi:10.18653/v1/2026.eacl-long.151.

[7] R. V. Tomar, P. Nakov, Y. Wang, UnsafeChain: Enhancing reasoning model safety via hard cases, in: K. Inui, S. Sakti, H. Wang, D. F. Wong, P. Bhattacharyya, B. Banerjee, A. Ekbal, T. Chakraborty, D. P. Singh (Eds.), Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, AACL '2025, The Asian Federation of Natural Language Processing and The Association for Computational Linguistics, Mumbai, India, 2025, pp. 1233–1247. URL: https://aclanthology.org/2025.findings-ijcnlp.75/. doi:10.18653/v1/2025.findings-ijcnlp.75.

[8] C. Li, J. Wang, X. Pan, G. Hong, M. Yang, ReasoningShield: Safety detection over reasoning traces of large reasoning models, arXiv preprint arXiv:2505.17244 (2025). URL: https://arxiv.org/abs/2505.17244.

[9] Y. Zhang, S. Zhang, Y. Huang, Z. Xia, Z. Fang, X. Yang, R. Duan, D. Yan, Y. Dong, J. Zhu, STAIR:

Improving safety alignment with introspective reasoning, arXiv preprint arXiv:2502.02384 (2025). URL: https://arxiv.org/abs/2502.02384.

[10] MBZUAI NLP, PAN CLEF 2026: Reasoning trajectory detection, GitHub repository, 2026. URL: https://github.com/mbzuai-nlp/PAN-CLEF2026-Reasoning-Trajectory-Detection.

[11] J. LI, E. Beeching, L. Tunstall, B. Lipkin, R. Soletskyi, S. C. Huang, K. Rasul, L. Yu, A. Jiang, Z. Shen, Z. Qin, B. Dong, L. Zhou, Y. Fleureau, G. Lample, S. Polu, NuminaMath, https://huggingface.co/AI-MO/NuminaMath-CoT, 2024.

[12] W. Du, B. Kisacanin, G. Armstrong, S. Toshniwal, I. Moshkov, A. Ayrapetyan, S. Mahdavi, D. Zhao, S. Diao, D. Masulovic, M. Stanean, A. Avadhanam, M. Wang, A. Dutta, S. Govil, S. Yanamandara, M. Tandon, S. Ananthakrishnan, V. Rathi, D. Zhang, J. Kang, L. Luo, T. Andreescu, B. Ginsburg, I. Gitman, The challenge of teaching reasoning to LLMs without rl or distillation, arXiv preprint arXiv:2507.09850 (2025). URL: https://arxiv.org/abs/2507.09850.

[13] D. Condrey, Writerslogic at PAN 2026: Process over content for robust detection under domain shift, in: Working Notes of CLEF 2026 - Conference and Labs of the Evaluation Forum, CEUR-WS.org, 2026.

[14] R. Trehub, M. Grus, I. Popovych, Team TUKE-AI at PAN 2026: Reasoning trajectory detection, in: Working Notes of CLEF 2026 - Conference and Labs of the Evaluation Forum, CEUR-WS.org, 2026.

[15] Team RTD-Qwen, Team RTD-Qwen at PAN 2026: Fine-tuning Qwen3.5-2B for reasoning trajectory source and safety detection, in: Working Notes of CLEF 2026 - Conference and Labs of the Evaluation Forum, CEUR-WS.org, 2026.

[16] Z. Liang, Z. Han, X. Duan, Z. Zeng, C. Liu, Team Asdkklkk at PAN 2026: ModernBERT-based source detection for mathematical reasoning trajectories at PAN 2026, in: Working Notes of CLEF 2026 - Conference and Labs of the Evaluation Forum, CEUR-WS.org, 2026.

[17] J. He, H. Chen, Team threshdsnmj at PAN 2026: Lightweight dual-lens evidence modeling for reasoning trajectory detection, in: Working Notes of CLEF 2026 - Conference and Labs of the Evaluation Forum, CEUR-WS.org, 2026.

[18] I. Branescu, L.-S. Dragne, T.-I. Becheru, M. Dascalu, Team Bit-by-Bit at PAN 2026: Multi-criterion guardian ensemble and step expert for reasoning trajectory safety, in: Working Notes of CLEF 2026 - Conference and Labs of the Evaluation Forum, CEUR-WS.org, 2026.

[19] 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, arXiv preprint arXiv:1907.11692 (2019). URL: https://arxiv.org/abs/1907.11692.

[20] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, V. Stoyanov, Unsupervised cross-lingual representation learning at scale, in: D. Jurafsky, J. Chai, N. Schluter, J. Tetreault (Eds.), Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL '2020, Association for Computational Linguistics, Online, 2020, pp. 8440–8451. URL: https://aclanthology.org/2020.acl-main.747/.

[21] B. Warner, A. Chaffin, B. Clavié, O. Weller, O. Hallström, S. Taghadouini, A. Gallagher, R. Biswas, F. Ladhak, T. Aarsen, G. T. Adams, J. Howard, I. Poli, Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference, 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), ACL '2025, Association for Computational Linguistics, Vienna, Austria, 2025, pp. 2526–2547. URL: https://aclanthology.org/2025.acl-long.127/. doi:10.18653/v1/2025.acl-long.127.

[22] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv, C. Zheng, D. Liu, F. Zhou, F. Huang, F. Hu, H. Ge, H. Wei, H. Lin, J. Tang, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Zhou, J. Lin, K. Dang, K. Bao, K. Yang, L. Yu, L. Deng, M. Li, M. Xue, M. Li, P. Zhang, P. Wang, Q. Zhu, R. Men, R. Gao, S. Liu, S. Luo, T. Li, T. Tang, W. Yin, X. Ren, X. Wang, X. Zhang, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Zhang, Y. Wan, Y. Liu, Z. Wang, Z. Cui, Z. Zhang, Z. Zhou, Z. Qiu, Qwen3 technical report, arXiv preprint arXiv:2505.09388 (2025). URL: https://arxiv.org/abs/2505.09388.