Team threshdsnmj at PAN 2026: Lightweight Dual-Lens Evidence Modeling for Reasoning Trajectory Detection

Notebook for the PAN Lab at CLEF 2026

Jiacong He[1,*], Hui Chen[1]

[1]Foshan University, China

Abstract

Reasoning trajectories have become an observable part of many language-model outputs. They make intermediate processes easier to inspect, but also introduce two risks: the provenance of a trace can be unclear, and unsafe reasoning can appear before a seemingly benign final answer. This paper introduces *TraceLens*, a lightweight dual-lens system for the PAN 2026 Reasoning Trajectory Detection task. TraceLens does not treat reasoning trajectories as generic long texts. Instead, it selects evidence according to the decision being made: the origin lens fine-tunes RoBERTa-base on short solution-only inputs to capture authorship cues, while the safety lens uses XLM-RoBERTa-base with a binary trace head and a weakly weighted step head to capture risk evidence in query-trace context. The submission evaluation results show that the origin lens improves our Subtask 1 score from 0.55 to 0.65, and the safety lens improves our Subtask 2 score from 0.48 to 0.56 over an mBERT three-class baseline. In this shared-task setting, the experiments suggest that useful evidence in reasoning trajectory detection is decision-dependent: source detection benefits from compact solution-only representations, while longer problem+solution context can still provide compensation in some submission evaluations; safety detection depends more on query-trace context, moderate step-level auxiliary supervision, and robust threshold choice.

Keywords

Reasoning trajectory detection, AI-generated text detection, Safety detection, PAN 2026

1. Introduction

Large language models increasingly generate not only final answers but also visible reasoning trajectories. The practice is useful: a trace can make a solution easier to inspect, can reveal intermediate mistakes, and can support post-hoc debugging of complex answers. It also changes the forensic object. A final answer may be short, polished, and difficult to attribute, while the path that leads to it may contain longer stylistic, procedural, and safety-relevant signals. The reasoning trace therefore becomes evidence.

The PAN 2026 overview lists Reasoning Trajectory Detection as one of the shared tasks for studying source and safety signals in human- or LLM-written reasoning trajectories [1]. The task overview formalizes this setting over triplets consisting of a user query, a reasoning trajectory, and a final answer [2]; the task page and repository provide the released data, starter kit, and submission format [3, 4]. It asks two questions. The first is an authorship question: is the reasoning and answer human-written or AI-generated? The second is a safety question: is the reasoning trajectory safe or unsafe, and which individual steps are unsafe? These questions are connected but not identical. Source detection looks for traces of production; safety detection looks for traces of risk.

Based on this observation, we propose *TraceLens*, a two-lens trajectory-evidence model. Its central idea is that different decisions require different evidence from the same trace. For source detection, the problem statement is usually shared background for both human and AI solutions, and can dilute authorship cues. The solution body, especially its early reasoning style and formatting habits, is more diagnostic. For safety detection, the full query-trace context is necessary because a step can be unsafe only relative to the user intent; however, step-level supervision can overwhelm the trace-level objective

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

*Corresponding author.

2112453063@stu.fosu.edu.cn (J. He)

if it is weighted too strongly. These observations lead to two deliberately asymmetric modules: a short solution-only origin lens and a multilingual binary safety lens with weak step supervision.

The main contributions of this paper are as follows:

- We introduce TraceLens, which models reasoning trajectories as decision-specific evidence rather than as generic long-text inputs.
- We design two lightweight lenses for the two subtasks: a solution-only origin lens for source detection and a query-trace safety lens with weak step supervision.
- We analyze how context length, threshold calibration, and encoder choice affect system behavior through validation diagnostics, ablations, and submission evaluation results.

The rest of this paper is organized as follows. Section 2 reviews work on generated-text detection, authorship evidence, and safety evidence inside reasoning traces. Section 3 introduces the task definition and official datasets. Section 4 describes the origin and safety lenses of TraceLens. Section 5 presents the experimental setup, evaluation metrics, system configurations, and final hyperparameters. Section 6 reports the results and discusses the main empirical findings. Section 7 concludes the paper.

2. Related Work

2.1. Generated text and authorship evidence

AI-generated text detection has moved from surface statistics toward model-aware and representation-based approaches. Early visualization and statistical tools such as GLTR examine token likelihood patterns to expose generated text signatures [5]. DetectGPT uses probability curvature under a language model to perform zero-shot generated-text detection [6], while Binoculars compares the behavior of two language models to identify machine text without task-specific fine-tuning [7]. In shared-task settings, however, compact supervised encoders remain attractive because they are easy to train, fast to deploy, and directly optimized for the evaluation label space. Fine-grained systems also benefit from auxiliary signals when the target label is only one view of a richer generation process [8].

Reasoning trajectories add a new dimension to this problem. Chain-of-thought prompting demonstrated that intermediate reasoning can improve performance on complex tasks [9]. Once such reasoning is exposed, it becomes a candidate signal for authorship analysis. The source of a trajectory may be reflected in ordering, verbosity, symbolic formatting, or hesitation patterns rather than in the final answer alone.

2.2. Safety evidence inside reasoning traces

Safety detection differs from ordinary generated-text detection because harmfulness can be local and contextual. A trajectory may contain an unsafe intermediate instruction even if the final answer refuses the request, or it may contain benign-looking tokens in a harmful plan. Recent work on unsafe reasoning datasets emphasizes the need to evaluate safety along the reasoning path, not only at the output boundary [10]. The PAN 2026 task reflects this motivation by evaluating both trace-level and step-level safety labels [2, 4].

This setting creates a tension. Step labels are informative, but there are many more step examples than trace examples. If they dominate training, the model may learn local step regularities at the cost of the ranking metric, which is trace-level. TraceLens therefore treats step supervision as an auxiliary signal rather than as an equal target.

3. Task and Data

3.1. Problem formulation

Each instance consists of a query q, a reasoning trajectory $r = (s_1, \ldots, s_m)$ segmented into steps, and a final answer a. Subtask 1 predicts a source label $y^{src} \in \{\texttt{human}, \texttt{ai}\}$. Subtask 2 predicts a trace-level

Table 1
Dataset sample sizes.

Task	Training	Validation	Test
Subtask 1 source detection	87,556	497	2,617
Subtask 2 safety detection	6,998	2,200	2,835

Table 2
The two evidence lenses of TraceLens.

Lens	Decision	Evidence input	Model	Output
Origin lens	human vs. AI	solution only	RoBERTa-base	source
Safety lens	safe vs. unsafe + step labels	trace / step	XLM-R-base	trace + step labels

safety label and a sequence of step-level binary labels. The released data include three trace labels for Subtask 2: `safe`, `potentially unsafe`, and `unsafe`. The submission checker accepts the overall labels used by the competition and detailed step labels as a serialized sequence [4].

Subtask 1 uses macro F1 over the two source labels, `human` and `ai`. Subtask 2 considers both the overall safety label and detailed step labels: the official `s2_f1` metric is safe-vs.-unsafe macro F1 on the overall label, while `s2_soft_f1` is per-sample F1 on detailed step labels, macro-averaged across samples. The released trace labels for Subtask 2 contain three classes: `safe`, `potentially unsafe`, and `unsafe`. In our binary safety systems, `potentially unsafe` and `unsafe` are merged into a non-safe class. This is a modeling choice for aligning the training target with the official safe-vs.-unsafe decision; it does not imply that the released data are binary. As a comparison point, the three-class baseline still preserves the original trace-level supervision.

3.2. Dataset

We obtained the training, validation, and test data from the official GitHub repository [4], without using external training data, external safety corpora, or generative augmentation. Table 1 summarizes the dataset sizes. Subtask 1 contains mathematical problems and their solutions, labeled as `human` or `ai`. Subtask 2 contains queries, segmented reasoning traces, trace-level safety labels, and step-level safety labels.

4. TraceLens Method

4.1. Overall framework

TraceLens follows an *evidence before architecture* principle: it first identifies which evidence a decision needs, and then chooses the corresponding input view and training objective. The origin lens focuses on production style, whereas the safety lens focuses on risk expression. The two lenses do not share parameters. Table 2 summarizes the system, and Figure 1 illustrates the complete workflow.

4.2. Origin lens: short-context source detection

The source perspective uses a RoBERTa-base sequence classifier [11]. Let x^{src} denote the source evidence text, and let $y^{src} \in \{0, 1\}$ denote the source label, where 0 represents `human` and 1 represents `ai`. The source classification process is defined as

$$h^{src} = \mathrm{Enc}_{src}(x^{src})_{[\mathrm{CLS}]}, \tag{1}$$

$$\mathbf{p}^{src} = \mathrm{softmax}(W_{src} h^{src} + b_{src}). \tag{2}$$

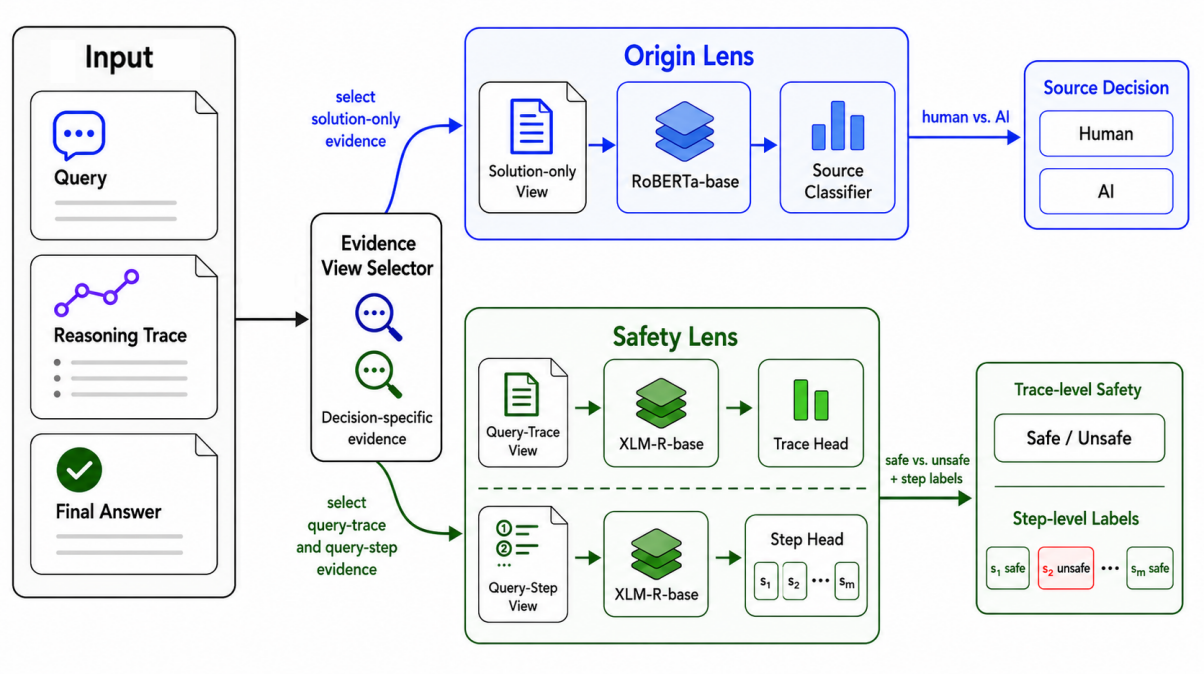

Figure 1: TraceLens workflow for the PAN 2026 Reasoning Trajectory Detection task. The origin lens selects solution-only evidence for source detection, while the safety lens uses query-trace and query-step evidence for overall safety prediction and step-level risk localization.

Here, Enc_{src} is the source detection encoder; h^{src} is the sequence-level semantic representation of the input text x^{src}; and W_{src} and b_{src} are the learnable parameters of the source classification head. $\mathbf{p}^{src}$ denotes the binary probability distribution predicted by the source classifier, which can be written as

$$\mathbf{p}^{src} = \left[p^{src}_{\texttt{human}}, p^{src}_{\texttt{ai}}\right], \tag{3}$$

where $p^{src}_{\texttt{human}}$ denotes the probability that the model assigns the source evidence text to the human-generated class, and $p^{src}_{\texttt{ai}}$ denotes the probability that the model assigns it to the AI-generated class. These two probabilities satisfy

$$p^{src}_{\texttt{human}} + p^{src}_{\texttt{ai}} = 1. \tag{4}$$

During training, the model is optimized by minimizing the cross-entropy between the ground-truth source label y^{src} and the predicted probability distribution $\mathbf{p}^{src}$:

$$\mathscr{L}_{src} = \mathrm{CE}(y^{src}, \mathbf{p}^{src}) = -\log p^{src}_{y^{src}}, \tag{5}$$

where $p^{src}_{y^{src}}$ denotes the probability assigned to the ground-truth class in the predicted distribution. Specifically, when $y^{src} = 0$, $p^{src}_{y^{src}} = p^{src}_{\texttt{human}}$; when $y^{src} = 1$, $p^{src}_{y^{src}} = p^{src}_{\texttt{ai}}$.

During inference, the model uses the AI-class probability component $p^{src}_{\texttt{ai}}$ in $\mathbf{p}^{src}$ for source prediction:

$$\hat{y}^{src} = \mathbb{I}\left[p^{src}_{\texttt{ai}} \geq \tau_{src}\right]. \tag{6}$$

Here, $\mathbb{I}[\cdot]$ is the indicator function, and τ_{src} is the source decision threshold. When $p^{src}_{\texttt{ai}} \geq \tau_{src}$, the model predicts the text source as `ai`; otherwise, it predicts the source as `human`.

4.3. Safety lens: trace-level objective with step-level auxiliary evidence

The safety lens uses XLM-RoBERTa-base [12]. It separates overall safety prediction from step-level risk localization. Let q be the user query, and let $r = (s_1, \ldots, s_m)$ be the reasoning trace with m steps, where s_j is the j-th step. The trace-level and step-level inputs are

$$x^{tr} = \texttt{[QUERY]}\ q\ \texttt{[TRACE]}\ r, \tag{7}$$

$$x_j^{st} = \texttt{[QUERY]}\ q\ \texttt{[STEP]}\ s_j. \tag{8}$$

Here, x^{tr} is used for overall safety prediction and x_j^{st} is used for local risk prediction at step j; `[QUERY]`, `[TRACE]`, and `[STEP]` are field markers. Their representations are

$$h^{tr} = \mathrm{Enc}_{tr}(x^{tr})_{[\mathrm{CLS}]}, \tag{9}$$

$$h_j^{st} = \mathrm{Enc}_{st}(x_j^{st})_{[\mathrm{CLS}]}. \tag{10}$$

where Enc_{tr} and Enc_{st} are the trace-level and step-level encoders. h^{tr} represents the whole query-trace pair and h_j^{st} represents the j-th query-step pair.

The safety heads output binary probability distributions:

$$\mathbf{p}^{tr} = \mathrm{softmax}(W_{tr}h^{tr} + b_{tr}), \tag{11}$$

$$\mathbf{p}_j^{st} = \mathrm{softmax}(W_{st}h_j^{st} + b_{st}). \tag{12}$$

Here, $\mathbf{p}^{tr}$ is the trace-level distribution over `safe`/`unsafe`, $\mathbf{p}_j^{st}$ is the step-level distribution for step j, and $W_{tr}, b_{tr}, W_{st}, b_{st}$ are classifier parameters. We merge `potentially unsafe` and `unsafe` into the non-safe class for a safe-vs.-unsafe decision.

The training objective is

$$\mathscr{L}_{tr} = \mathrm{CE}(y^{tr}, \mathbf{p}^{tr}), \tag{13}$$

$$\mathscr{L}_{st} = \frac{1}{m}\sum_{j=1}^{m} \mathrm{CE}(y_j^{st}, \mathbf{p}_j^{st}), \tag{14}$$

$$\mathscr{L} = \lambda_{tr}\mathscr{L}_{tr} + \lambda_{st}\mathscr{L}_{st}. \tag{15}$$

where $y^{tr} \in \{0, 1\}$ is the binary trace-level label, $y_j^{st} \in \{0, 1\}$ is the label of step j, $\mathscr{L}_{tr}$ is the trace-level cross-entropy loss, $\mathscr{L}_{st}$ is the averaged step-level cross-entropy loss, and λ_{tr} and λ_{st} weight the two objectives. The concrete weights, thresholds, and sampling ratios are reported in Section 5.

5. Experimental Setup

5.1. Implementation

All systems were implemented with PyTorch [13], Hugging Face Transformers [14], and scikit-learn [15]. We used public pretrained encoders only as initialization. Fine-tuning used the PAN training set, while validation data were used for model selection, threshold sweeps, and diagnostic analysis. Experiments were run on a workstation with an NVIDIA GeForce RTX 5090 GPU.

5.2. Evaluation metrics

The metric definitions follow the task repository [4]. Validation metrics are computed on the released validation split; labels for the test inputs are not released, so results on test submissions are reported as Eval. score returned by the submission page. For any class c, precision, recall, and F1 are defined as

$$P_c = \frac{TP_c}{TP_c + FP_c}, \quad R_c = \frac{TP_c}{TP_c + FN_c}, \quad F_1(c) = \frac{2P_cR_c}{P_c + R_c}. \tag{16}$$

Here, TP_c, FP_c, and FN_c are true positives, false positives, and false negatives for class c; P_c, R_c, and $F_1(c)$ denote precision, recall, and F1 for that class. Subtask 1 uses macro F1 over the source labels:

$$\mathrm{MacroF1}_{src} = \frac{F_1(\texttt{human}) + F_1(\texttt{ai})}{2}, \tag{17}$$

where $F_1(\texttt{human})$ and $F_1(\texttt{ai})$ are the F1 scores of the two source classes.

For Subtask 2, the primary ranking metric is `s2_f1`, macro F1 on the overall label under binary `safe` vs. `unsafe` evaluation:

$$\text{s2\_f1} = \frac{F_1(\texttt{safe}) + F_1(\texttt{unsafe})}{2}, \tag{18}$$

where $F_1(\texttt{safe})$ is the safe-class F1 and $F_1(\texttt{unsafe})$ is the unsafe-class F1 after mapping `potentially unsafe` to the unsafe side for the binary decision. The detailed-label metric is `s2_soft_f1`, which computes an F1 score for each sample and then averages across samples. For sample i, let $y_j \in \{0, 1\}$ and $\hat{y}_j \in \{0, 1\}$ be the gold and predicted labels of step j. Following the official notation,

$$\text{F1}^{(i)} = \frac{2 \cdot tp_{\text{soft}}}{2 \cdot tp_{\text{soft}} + fp_{\text{soft}} + fn_{\text{soft}}}, \tag{19}$$

$$tp_{\text{soft}} = \sum_j \hat{y}_j \cdot y_j, \tag{20}$$

$$fp_{\text{soft}} = \sum_j \hat{y}_j \cdot (1 - y_j), \tag{21}$$

$$fn_{\text{soft}} = \sum_j (1 - \hat{y}_j) \cdot y_j, \tag{22}$$

$$\text{s2\_soft\_f1} = \frac{1}{N} \sum_{i=1}^{N} \text{F1}^{(i)}. \tag{23}$$

Here, tp_{soft}, fp_{soft}, and fn_{soft} are computed within one sample, $\text{F1}^{(i)}$ is the per-sample detailed-label F1, and N is the number of samples. In the tables, Eval. score denotes the score returned by the corresponding submission page on the test inputs. For Subtask 1, it corresponds to source-detection macro F1; for Subtask 2, it corresponds to `s2_f1`, i.e., safe-vs.-unsafe macro F1 on the overall safety label [16, 17].

5.3. System configuration and ablation design

For Subtask 1, we design several groups of source-detection experiments. The first group varies the maximum length of the solution-only input among 64, 128, 256, and 512 tokens, testing whether source cues are concentrated in compact early solution fragments. The second group compares one epoch and three epochs under the same 128-token input, testing whether higher validation diagnostics also lead to better submission evaluation results. The third group changes the input into problem+solution concatenations at 128, 256, and 512 tokens, testing whether the problem statement contributes extra source evidence or dilutes the actual authorship signal. Together, these experiments ask whether source detection needs more context or cleaner solution evidence.

For Subtask 2, we design four groups of safety-detection experiments. The first group is an mBERT three-class trace baseline [18] that preserves the original trace-label structure. The second group trains binary trace models and threshold variants to test whether aligning the target with the official `s2_f1` metric is effective and whether validation-calibrated thresholds are robust. The third group changes the strength of step supervision to test whether step labels should act as a strong objective or weak auxiliary signal. The fourth group uses XLM-R variants with different context lengths, random seeds, and step-loss weights, testing whether encoder choice and training details improve submission evaluation results. Table 3 summarizes the experimental roles and comparison purposes.

5.4. Final hyperparameters

Table 4 gives the key hyperparameters of the two final systems. Safety-XLMR-Core uses $\lambda_{tr} = 1.0$ and $\lambda_{st} = 0.10$, together with a step balance ratio of 0.25 that limits the number of step-level samples drawn in each epoch. This keeps the local risk signal from step labels while preventing the more numerous step examples from pulling the model away from the trace-level objective.

Table 3
Main systems and ablation groups.

Experiment group	System	Input view	Purpose
S1 length group	Origin-Tiny64/Core128/Long256/Wide512	solution	Length
S1 depth group	Origin-Core128/Deep128	solution	Epochs
S1 context group	Origin-Context128/256/512	problem + solution	Context
S2 label-form group	Safety-Direct3C/Calibrated/Fixed050	trace / step	Labels / threshold
S2 step-supervision group	Safety-WeakStep/XLMR-Step005	trace / step	Step weight
S2 encoder group	Safety-XLMR128/Core/Seed	trace / step	Encoder

Table 4
Final-system hyperparameters.

System	Encoder	Input	Max len.	Epochs	LR	Main thr.	Auxiliary setting
Origin-Core128	RoBERTa-base	solution only	128	1	1×10^{-5}	0.78	–
Safety-XLMR-Core	XLM-R-base	trace / step	256	3	1×10^{-5}	0.50	$\lambda_{tr} = 1.0$, $\lambda_{st} = 0.10$; step bal.=0.25

Table 5
Main validation metrics and submission evaluation results. Val. metric denotes the metric computed on the released validation split; Val. score gives the corresponding validation score; Eval. score is returned after submitting predictions for the test inputs.

Task	System	Val. metric	Val. score	Eval. score	Gain
S1	Origin-Long256	MacroF1_{src}	0.8728	0.55	ref.
S1	Origin-Core128	MacroF1_{src}	0.9163	0.65	+0.10
S2	Safety-Direct3C	`s2_f1`	0.6076	0.48	ref.
S2	Safety-XLMR-Core	`s2_f1`	0.8330	0.56	+0.08

6. Results and Discussion

6.1. Submission evaluation overview

Table 5 shows the main submitted systems. Origin-Core128 improves the Subtask 1 Eval. score by 0.10 over the long RoBERTa baseline. Safety-XLMR-Core improves the Subtask 2 Eval. score by 0.08 over the direct mBERT three-class baseline. These gains come from different forms of restraint: the origin lens removes prompt context and shortens the input, while the safety lens avoids letting step labels or validation thresholds dominate the trace-level objective. Because the two subtasks use different submission scoring programs, Eval. score should be interpreted within each subtask rather than as a direct comparison between S1 and S2.

6.2. Compact solution evidence for source detection

The Subtask 1 experiments address three questions: how long the solution-only input should be, whether longer training helps, and whether the problem statement should be included in the source detector. Table 6 shows that the 128-token Origin-Core128 model reaches 0.65 in the submission evaluation, making it one of the most compact and strongest submissions in our source-detection runs. Origin-Deep128 achieves a higher validation MacroF1 but drops to 0.57 on Eval. score, suggesting that additional training can reinforce validation-specific patterns. Origin-Wide512 reaches only 0.59 in the submission evaluation and does not outperform the short-context system.

The context group gives a more nuanced conclusion. Adding the problem statement clearly lowers

Table 6
Subtask 1 source-detection ablations using RoBERTa-base. Val. MacroF1 denotes validation macro F1 for source detection.

System	Input	Length / epochs	Val. MacroF1	Eval. score
Origin-Tiny64	solution	64 / 1	0.9088	0.57
Origin-Core128	solution	128 / 1	0.9163	0.65
Origin-Deep128	solution	128 / 3	0.9223	0.57
Origin-Long256	solution	256 / 3	0.8728	0.55
Origin-Wide512	solution	512 / 3	0.8854	0.59
Origin-Context128	problem + solution	128 / 1	0.7982	0.43
Origin-Context256	problem + solution	256 / 3	0.8396	0.42
Origin-Context512	problem + solution	512 / 3	0.8344	0.65

Table 7
Subtask 2 safety-detection ablations. Val. `s2_f1` denotes validation macro F1 on the overall safety label, Val. `s2_soft_f1` denotes validation per-sample F1 on detailed step labels, and Eval. score reports the score returned after submitting predictions for the test inputs.

System	Setting	Val. `s2_f1`	Val. `s2_soft_f1`	Eval. score
Safety-Direct3C	mBERT, three-class trace + binary step	0.6076	0.6716	0.48
Safety-Calibrated	mBERT, binary trace, validation-calibrated	0.8134	0.7052	0.42
Safety-Fixed050	same checkpoint, fixed trace threshold 0.50	0.7528	0.7052	0.46
Safety-WeakStep	mBERT, binary trace, weaker step supervision	0.8045	0.6713	0.48
Safety-XLMR128	XLM-R, 128-token trace-only variant	0.8093	0.6735	0.40
Safety-XLMR-Core	XLM-R, 256 tokens, step loss 0.10	0.8330	0.6793	0.56
Safety-XLMR-Seed46	same as core, seed 46	0.8337	0.6787	0.55
Safety-XLMR-Step005	XLM-R, very weak step loss 0.05	0.8317	0.6749	0.51

validation MacroF1, and Origin-Context128 and Origin-Context256 obtain only 0.43 and 0.42 in the submission evaluation. This suggests that short problem+solution inputs can shift the representation toward problem semantics or truncation noise rather than authorship cues. Origin-Context512 reaches 0.65 in the submission evaluation, matching Origin-Core128, which means that a sufficiently long concatenated input can sometimes recover useful solution evidence. However, it does not outperform the Origin-Core128, which is shorter, simpler, and stronger in validation MacroF1. The conclusion is therefore not that the problem statement is always useless, but that more context does not reliably provide better source evidence. We select Origin-Core128 because it reaches the same best Eval. score with a much shorter input and less dependence on prompt context and truncation strategy.

The confusion matrices further explain the improvement. Origin-Long256 obtains perfect AI recall on validation but misclassifies 35 of 100 human examples as AI. Origin-Core128 reduces human false positives to 21 while keeping AI recall high. The final model is therefore not simply more aggressive; it is less biased toward calling every trajectory machine-generated.

6.3. Threshold, step-supervision, and encoder effects in safety detection

The Subtask 2 experiments correspond to the four design groups in Section 5: the three-class baseline preserves the original label structure, binary models align the objective with `s2_f1`, threshold experiments test whether threshold calibration is robust, and the step-supervision and XLM-R variants test whether auxiliary labels and encoder choice improve submission evaluation behavior. Table 7 reports validation diagnostics and submission evaluation results.

The first group shows that the direct three-class baseline is a useful reference point: although its validation `s2_f1` is only 0.6076, it reaches 0.48 in the submission evaluation. The binary objective gives much higher validation `s2_f1`, but Safety-Calibrated obtains only 0.42 as Eval. score. This configuration uses a trace threshold of 0.84 selected on the validation split; in contrast, the same checkpoint with a fixed 0.50 threshold improves to 0.46, but still remains below the direct three-class baseline. This

shows that threshold calibration is a sensitive factor and should not be judged only from validation `s2_f1`. This pattern is consistent with the broader calibration problem of neural networks [19].

The step-supervision experiments show that weakening the step loss alone is not sufficient to improve the submission evaluation result. Safety-WeakStep reaches validation `s2_f1` 0.8045 and obtains 0.48 as Eval. score, matching the direct three-class baseline. This suggests that step labels provide useful local risk information, but a higher `s2_soft_f1` diagnostic score does not necessarily lead to a better submission evaluation result. The final system therefore treats the step-level objective as an auxiliary constraint rather than the dominant training signal.

The XLM-R experiments show that encoder choice and context length must be considered together. Safety-XLMR128 reaches a validation `s2_f1` of 0.8093 but only 0.40 as Eval. score, indicating that simply switching to a multilingual encoder with a shorter trace context is not reliable. Safety-XLMR-Core extends the trace length to 256 tokens, uses a fixed 0.50 threshold, and keeps weak step supervision with $\lambda_{st} = 0.10$, reaching 0.56. Safety-XLMR-Seed46 uses the same main configuration as Core and changes only the random seed, obtaining an Eval. score of 0.55, which suggests that the effectiveness of this configuration is not an accident of a single random seed. Safety-XLMR-Step005 further reduces the step-loss weight to 0.05, but its Eval. score drops to 0.51, suggesting that overly weak step supervision can hurt the final safety detection performance. We cannot claim that language coverage is the only cause of the improvement, but XLM-R's evaluation advantage is consistent with a setting where mixed-language or cross-domain trace patterns may appear.

These comparisons should be interpreted as configuration-level evidence rather than a fully controlled factor isolation. The reported ablations vary encoder choice, input length, label mapping, thresholding, and auxiliary step-loss weight, but they do not exhaustively disentangle every interaction among these factors. We therefore treat Safety-XLMR-Core as the strongest submitted configuration in our runs, not as evidence that any single design choice alone explains the improvement.

Our validation-side error inspection suggests that the remaining safety errors are concentrated near the boundary between safe and non-safe traces. Some queries contain risky words, but the trace actually explains, refuses, or avoids harm. Other examples have a comparatively safe final tendency while one intermediate step already describes an actionable unsafe plan. This suggests that safety detection cannot rely only on keywords or only on the final answer. It must jointly consider user intent, reasoning steps, and the trace-level conclusion. Step labels provide useful localization, but excessive step weight can overfit local step patterns.

6.4. Trace evidence is decision-dependent

The combined results suggest that reasoning traces are not uniformly useful long texts. For provenance, the useful evidence can be compact and local. For safety, the evidence is contextual and may be distributed across steps. This distinction explains why a single generic input recipe is suboptimal. It also gives the paper its main story: exposing intermediate reasoning is useful only when the model knows which part of the trace is evidential for which decision.

7. Conclusion

We presented TraceLens, a compact two-lens approach to reasoning trajectory detection. The origin lens treats source detection as a short-context authorship problem and achieves the best Subtask 1 Eval. score among our submissions. The safety lens treats unsafe reasoning as contextual evidence and improves Subtask 2 in our submitted runs by combining XLM-R with binary trace modeling and weak step supervision. Beyond the specific scores, the main finding is methodological: validation diagnostics alone can select the wrong amount of context, the wrong threshold, or the wrong auxiliary balance. In this shared-task setting, we found it more useful to model reasoning traces as structured evidence whose useful fields depend on the decision being made, rather than as generic long text.

Acknowledgments

We thank the PAN 2026 organizers for preparing the task, data, starter kit, and evaluation platform.

Declaration on Generative AI

During the preparation of this work, the authors utilized ChatGPT and Grammarly for grammar and spelling checks. After employing these tools, the authors independently reviewed and edited the content as necessary, taking full responsibility for the final publication.

References

[1] J. Bevendorff, M. Fröbe, A. Greiner-Petter, A. Jakoby, M. Mayerl, P. Nakov, H. Plutz, M. Potthast, B. Stein, M. N. Ta, Y. Wang, E. Zangerle, in: M. Hagen, M. Potthast, B. Stein, P. Schaer, E. Zangerle, A. Barrón-Cedeño, A. G. S. de Herrera, E. S. Salido, S. MacAvaney, J. M. Struß (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Seventeenth International Conference of the CLEF Association (CLEF 2026), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2026.

[2] M. N. Ta, K. Wan, Y. Lin, Y. Wang, P. Nakov, Overview of the first Reasoning Trajectory Detection Task at PAN 2026, in: A. Barrón-Cedeño, A. G. S. de Herrera, E. S. Salido, S. MacAvaney, J. M. Struß (Eds.), Working Notes of CLEF 2026 – Conference and Labs of the Evaluation Forum, CEUR Workshop Proceedings, CEUR-WS.org, 2026.

[3] PAN at CLEF 2026: Reasoning trajectory detection, https://pan.webis.de/clef26/pan26-web/reasoning-trajectory-detection.html, 2026. Accessed: 2026-05-24.

[4] PAN-CLEF 2026: Reasoning trajectory detection, https://github.com/mbzuai-nlp/PAN-CLEF2026-Reasoning-Trajectory-Detection, 2026. GitHub repository. Accessed: 2026-05-24.

[5] S. Gehrmann, H. Strobelt, A. M. Rush, GLTR: Statistical detection and visualization of generated text, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, 2019, pp. 111–116. URL: https://aclanthology.org/P19-3019/. doi:10.18653/v1/P19-3019.

[6] 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, volume 202 of *Proceedings of Machine Learning Research*, PMLR, 2023, pp. 24950–24962. URL: https://proceedings.mlr.press/v202/mitchell23a.html.

[7] A. Hans, A. Schwarzschild, V. Cherepanova, H. Kazemi, A. Saha, M. Goldblum, J. Geiping, T. Goldstein, Spotting LLMs with binoculars: Zero-shot detection of machine-generated text, in: Proceedings of the 41st International Conference on Machine Learning, volume 235 of *Proceedings of Machine Learning Research*, PMLR, 2024, pp. 17519–17537. URL: https://proceedings.mlr.press/v235/hans24a.html.

[8] 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: Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), 2026, pp. 3275–3296. URL: https://aclanthology.org/2026.eacl-long.151/. doi:10.18653/v1/2026.eacl-long.151.

[9] J. Wei, X. Wang, D. Schuurmans, M. Bosma, b. ichter, F. Xia, E. Chi, Q. V. Le, D. Zhou, Chain-of-thought prompting elicits reasoning in large language models, in: Advances in Neural Information Processing Systems, volume 35, 2022, pp. 24824–24837. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

[10] R. V. Tomar, P. Nakov, Y. Wang, UnsafeChain: Enhancing reasoning model safety via hard cases, in: 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,

2025, pp. 1233–1247. URL: https://aclanthology.org/2025.findings-ijcnlp.75/. doi:10.18653/v1/2025.findings-ijcnlp.75.

[11] 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, arXiv preprint arXiv:1907.11692.

[12] 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: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 8440–8451. URL: https://aclanthology.org/2020.acl-main.747/. doi:10.18653/v1/2020.acl-main.747.

[13] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, PyTorch: An imperative style, high-performance deep learning library, in: Advances in Neural Information Processing Systems, volume 32, 2019. URL: https://papers.neurips.cc/paper_files/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

[14] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, A. M. Rush, Transformers: State-of-the-art natural language processing, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, 2020, pp. 38–45. URL: https://aclanthology.org/2020.emnlp-demos.6/. doi:10.18653/v1/2020.emnlp-demos.6.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, Édouard Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830. URL: https://jmlr.org/papers/v12/pedregosa11a.html.

[16] PAN-CLEF 2026 subtask 1: Source detection submission evaluation page, https://www.codabench.org/competitions/15502/, 2026. Accessed: 2026-05-24.

[17] PAN-CLEF 2026 subtask 2: Safety detection submission evaluation page, https://www.codabench.org/competitions/16085/, 2026. Accessed: 2026-05-24.

[18] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: 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), 2019, pp. 4171–4186. URL: https://aclanthology.org/N19-1423/. doi:10.18653/v1/N19-1423.

[19] C. Guo, G. Pleiss, Y. Sun, K. Q. Weinberger, On calibration of modern neural networks, in: Proceedings of the 34th International Conference on Machine Learning, volume 70 of *Proceedings of Machine Learning Research*, PMLR, 2017, pp. 1321–1330. URL: https://proceedings.mlr.press/v70/guo17a.html.