Team A&A-detection at PAN 2026: A Hybrid Ensemble of Stylometric, Perplexity, and Neural Features for Voight-Kampff Generative AI Detection

Notebook for the PAN Lab at CLEF 2026

Amalia Voulgaraki[1,*], Arina Zamyshevskaya[1,*]

[1]University of Copenhagen, Copenhagen, Denmark

Abstract

This paper presents our system for the PAN-2026 Voight-Kampff Generative AI Detection Shared Task, which is a binary classification challenge in which systems assign a confidence score to indicate whether a text was written by a human or generated by AI. Our submitted approach is a compact hybrid ensemble comprising two components: a CatBoost classifier trained on stylometric and model-fingerprint features, and a fine-tuned Microsoft/Deberta-v3-Large sequence classifier. The feature-based branch captures surface-level writing patterns, including lexical diversity, sentence structure, punctuation, formatting and assistant-like artefacts, while the neural branch learns contextual patterns directly from the input text. At inference time, both components produce an AI probability, and the final prediction is computed as their arithmetic mean. The system is packaged as a standalone Docker submission for TIRA, outputting one continuous score in the range [0, 1] for each input document. On the official TIRA smoke test, the submitted system achieved a mean score of 0.906, with ROC-AUC of 0.962, Brier score complement of 0.893, C@1 of 0.882, F1 of 0.857 and F0.5u of 0.938. Official hidden test scores were not visible at the time of writing.

Keywords

AI-generated text detection, stylometry, DeBERTa, CatBoost, Binoculars, perplexity, fingerprint features, ensemble, PAN 2026

1. Introduction

The rapid spread of large language models (LLMs) has fundamentally changed the text generation field. Models such as GPT-4, Gemini, LMA, and DeepSeek can each produce fluent and coherent text that covers all genres. Therefore, raising concerns about academic integrity and authenticity. While the earlier neural network-based text generation system created relatively easy detections through repetitiveness and incoherence, the latest LLMs, which have been trained with reinforcement learning from human feedback (RLHF), therefore produce text often indistinguishable from human-authored texts.

This difficulty is not merely subjective. Jakesch et al. [1] provided experimental evidence that human readers who rely on intuitive heuristics struggle to identify AI-generated literature. Even automatic detectors have limitations. According to a study conducted by Ippolito et al. [2], detection is easiest when the generated text is of lower quality. Theoretically, Sadasivan et al. [3] argues that when LLMs approach human-level fluency, detection may become impossible. Therefore, a variety of complementary detection signals may prove to be more beneficial, rather than depending solely on one strategy.

The PAN 2026 Voight-Kampff Generative AI Detection task [4, 5, 6] formulates this problem as a binary classification problem. Given text of unknown origin, the system must calculate a confidence score in [0,1] intervals, indicating the probability that the text was generated by AI. The project is noteworthy in that it covers various genres such as literature, essays, and news, and that it intentionally includes AI texts generated through paraphrases or reverse translations to avoid detection, and texts

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

*Corresponding author(s).

amvoulgaraki@gmail.com (A. Voulgaraki); arinkazam@gmail.com (A. Zamyshevskaya)

from AI models that were not used in the training process. This design method directly reflects situations in which AI detectors have to operate in real competitive environments.

In this paper, we address this challenge by constructing a meta-ensemble of five complementary components that combine TF-IDF SVM, CatBoost classifier [7] trained with style measurements and model fingerprint features, Binoculars [8] and Fast-DetectGPT [9], a zero-shot perplexity detector, and DeBERTa-v3-large, a fine-tuned sequence classifier [10]. Each component captures different aspects that distinguish humans from AI. TF-IDF captures vocabulary-level patterns, style-measuring features favor surface-writing styles, fingerprint features favor known features of specific models, Perplexity signals favor token sequences of high probability, and DeBERTa captures potential distribution structures that are not revealed in manually extracted features. The basic idea is that individual signals can ignore or fail in certain models, but ensembles composed of various signals are harder to deceive overall.

2. Related Work

Prior work on AI-generated text detection falls into three broad families.

Feature-based approaches score texts using superficial statistical or style-measured signals. Gehrmann et al. [11] showed that AI-generated text focuses on high-probability tokens (GLTRs), and TF-IDF n-gram classifiers [12, 13] were competitive PAN baselines, despite their ease of overfitting known model vocabulary. Style-based features (e.g., measurement of jargon ratio, grammatical error rate, entropy-complexity) provide complementary, model-agnostic signals and improve F1 score when combined with neural confusion features [14, 15]. In particular, gradient boosting trees, such as CatBoost [7], are powerful and cost-effective baselines for learning these manually designed features. Recent work has also identified vocabulary 'fingerprints' of RLHF-aligned models, motivated by a sharp rise in words such as *delve* and *tapestry* [16] and near-deterministic reasoning-tag artefacts in DeepSeek-R1 [17].

The zero-shot Perplexity-based approach completely avoids manually designed features and instead scores the text using the probability topography of the reference language model. DetectGPT [18] and its much faster successor Fast-DetectGPT [9] exploit the observation that AI text sits in higher-probability, lower-curvature regions than human text; DetectLLM [19] further improves on this using log-rank rather than raw log-probability. Binoculars [8] compares the Perplexity of two related language models to achieve high zero-shot accuracy, but significantly degrades accuracy in a new and previously unseen family of models (e.g. only 8.9% detection on GPT-4.5-preview text), illustrating a general zero-shot transfer-failure pattern.

Neural classifier approaches fine-tune pre-trained transformers directly into detection tasks. Combining disentangled attention and ELECTRA-style pre-learning, DeBERTa-v3 [10] is a consistently strong baseline, and it has been the basis for several PAN 2024/2025 best systems, including 2024 winning systems (system combining binoculars and LoRA-fine-tuned LLMs [20] and chunk-level approaches [20] that mitigate long document dilution via multi-scale loss functions or graph-based structural features [21]. Throughout this literature, fine-tuned transformers clearly outperform methods using static features or Perplexity alone, but they lose an F1 score of 10 to 20 points when faced with a structurally novel generator. Therefore, we recommend using these transformers as a component of the ensemble rather than as standalone detectors.

Putting these studies together, we find a sharp drop in detection accuracy under distribution variations (paraphrases, inverse translations, mixed author texts, unknown generators), and Sadasivan et al. [3] made the theoretical argument that a single detector cannot remain robust for sufficiently aligned generators. The PAN 2026 task is explicitly designed to test these failure modes. This underlies our two key design choices: (1) combining diverse and complementary signals rather than relying on a single dominant detector, and (2) favoring fingerprint features that are expected to be transferred between model families rather than features tied to the vocabulary or style of a particular generator.

3. Methodology

3.1. Task Definition and Dataset

The PAN-2026 Voight-Kampff task is binary classification task. Given an input text, the system provides a confidence score between 0 and 1: values close to 0 indicate human-written text, while values close to 1 indicate AI-generated text. Each document is scored independently. The system does not use other test documents and does not adapt during testing.

The submitted system comprises two components: a hybrid ensemble. The first component is a CatBoost classifier that has been trained using stylometric and model fingerprint features. The second component is a fine-tuned Microsoft/Deberta-V3-Large sequence classifier. This architecture is small enough to run directly in the TIRA environment, with input in the form of a JSONL dataset and output in the form of a JSONL file containing one score per identifier.

3.2. System Overview

The pipeline uses late fusion. For each record, the raw text goes to two independent predictors. The CatBoost branch converts the text into a fixed-length feature vector and returns the probability of the AI class. The DeBERTa branch tokenizes the text, runs it through a fine-tuned classifier, and converts the logit into a probability via sigmoid. When both components are available, the final score is the arithmetic mean:

$$p_{\text{final}} = \frac{p_{\text{catboost}} + p_{\text{deberta}}}{2}.$$

If only one checkpoint is available, the system falls back to that component. This fallback exists for robustness; the official submission uses both.

The two branches capture different evidence. The neural model learns contextual and distributional regularities from text. The CatBoost model operates on interpretable surface features: text length, lexical diversity, punctuation, formatting, known model-like artifacts. We do not use Binoculars, Fast-DetectGPT, or TF-IDF SVM in the submission. GLTR-style features are implemented in the codebase but not enabled in the final CatBoost configuration.

3.3. Stylometric Feature Extraction

The first feature group measures general stylometric properties. Features are computed with lightweight regex tokenization, no external parsers. The text is split into word tokens, raw-case tokens, and sentence-like units. From these, fourteen features are extracted: character count `n_chars`, word count `n_wrds`, sentence count `n_snts`, average word length `avg_wrd_ln`, average sentence length `avg_snt_ln`, type-token ratio `ttr`, moving average type-token ratio `mttr`, hapax legomenon ratio `hapax_rate`, sentence-length burstiness `burstiness`, bigram uniqueness `bigram_uniq`, approximate verb ratio `verb_ratio`, lowercase ratio `lowercase_ratio`, punctuation density `punct_dens`, and a surrogate repeated-character error rate `spelling_err_rate`.

These features are deliberately simple. They describe general writing behaviour. AI and human texts can differ in terms of rhythm, sentence variability, punctuation density and lexical repetition, and these features capture that. The moving average type-token ratio reduces the dependency of lexical diversity on document length. Burstiness captures the uneven structure that is typical of human writing.

3.4. Model Fingerprint Features

The second feature group targets surface artefacts that are common in instruction-tuned or reasoning-oriented language models. These are not intended to identify a specific generator; rather, they offer weak indications of model-like writing.

The CatBoost branch uses eleven fingerprint features. `think_block` detects `<think>` tags. `boxed_ans` detects LaTeX boxed answers via `\boxed{`. `sycoph_rate` measures the frequency of assistant-like

phrases per sentence ("great question", "of course"). `safety_pre` checks whether refusal or disclaimer expressions appear in the first 300 characters ("as an ai", "as a language model").

Lexical and formatting indicators: `rlhf_rate` counts words from a manually defined RLHF-like vocabulary (*delve*, *tapestry*, *nuanced*, *comprehensive*, *robust*). `downtoner_rate` measures hedging words (*somewhat*, *rather*, *arguably*). `em_dash_rate` counts em dashes per sentence. `md_density` measures Markdown-like formatting: headings, bold markers, inline code, list prefixes. `stop_ratio`, `para_burst`, and `stem_voc_rat` measure stop-word usage, paragraph-length variability, and suffix-stripped vocabulary ratio.

None of these are deterministic. A human can use Markdown or em dashes; an AI system can avoid them. They are weak statistical signals for the boosting model.

3.5. CatBoost Classifier

The fourteen stylometric features and eleven fingerprint features form the numeric input to CatBoost. The metadata field `genre` is appended as a categorical feature when present; otherwise the system assigns unknown. This makes inference compatible with datasets containing only `id` and `text`.

Hyperparameters: 1,000 boosting iterations, learning rate 0.05, tree depth 6, L2 leaf regularization 3.0, 128 borders for numeric quantization, log-loss objective, AUC evaluation metric, balanced automatic class weights, random seed 42, early stopping after 50 rounds without validation improvement. CatBoost was chosen because it handles heterogeneous feature types without explicit scaling and because gradient-boosted trees work well with sparse, manually constructed signals.

3.6. DeBERTa Classifier

The neural branch uses `microsoft/deberta-v3-large` as a binary classifier with one output logit. Each input text is tokenized, truncated to 256 tokens, and padded to this length. The fine-tuned checkpoint is loaded from `deberta_v2_best.pt`. The logit is converted to AI probability via sigmoid.

Inference batch size is 16. Device selection is automatic: CUDA, then Apple MPS, then CPU. The DeBERTa model does not receive stylometric features. It learns distributional patterns directly from text.

3.7. Submission Architecture

The entry point is `predict.py`. It loads all records from `dataset.jsonl`, runs the available branches, combines probabilities, and writes predictions to `predictions.jsonl`:

$$\{\texttt{"id"} : id, \texttt{"label"} : p_{\text{final}}\}.$$

The Docker image contains prediction code, both checkpoints, and Python dependencies. The Dockerfile preloads the DeBERTa tokenizer and base model during image construction, so no downloads happen at prediction time.

3.8. Evaluation

The official PAN metrics are ROC-AUC, Brier score complement, C@1, F1, $F0.5_u$, and their arithmetic mean. The system outputs continuous scores, so both ranking quality and calibration matter. ROC-AUC measures how well AI-generated texts are ranked above human texts. The Brier component penalizes poor calibration. C@1 and $F0.5_u$ make the treatment of uncertain cases relevant. The script always returns the averaged probability from the available components — it does not force uncertain examples to 0.5.

4. Results

At the time of writing, official scores for the hidden PAN 2026 test sets were not visible in the TIRA interface. We report the smoke-test result as evidence that the software executes correctly in the official environment. The run completed successfully (Table 1).

Table 1
TIRA smoke-test results of the submitted hybrid system.

Dataset	ROC-AUC	Brier	C@1	F1	$F0.5_u$	Mean
PAN 2026 smoke test	0.962	0.893	0.882	0.857	0.938	0.906

The highest component is $F0.5_u$ (0.938), followed by ROC-AUC (0.962) — the model ranks AI-generated examples above human-written examples well. Lower F1 (0.857) and C@1 (0.882) suggest that decision-boundary cases remain difficult when continuous probabilities are converted to hard labels.

We do not report official test performance because the hidden test metrics were not available. We also do not report ablation scores for CatBoost-only or DeBERTa-only variants unless produced by the same submitted code under the same evaluation setting.

5. Discussion

The system is simpler than much of the related work. Although perplexity-based methods (Binoculars) and curvature-based methods (Fast-DetectGPT) are relevant to the task, they are not part of our submission. The detector is a two-branch ensemble comprising one neural classifier and one feature-based classifier. This makes the system easier to package, faster to run and less dependent on multiple large model checkpoints.

The CatBoost branch contributes interpretability and surface-level robustness. Its features describe text length, sentence structure, punctuation, lexical diversity, paragraph variability, Markdown usage, assistant-like phrases and model-fingerprint artefacts. These features are inexpensive to compute and do not require a GPU. However, they are limited in that a human can produce the same markers and an AI system can avoid them through paraphrasing or prompt instructions. These are weak statistical signals, not detection rules.

The DeBERTa branch is the stronger contextual component. It captures patterns that are not represented in hand-crafted features. However, texts are truncated to 256 tokens, meaning evidence from later in long documents is lost. While the CatBoost branch still receives features from the full text, this only partly compensates for the fact that the neural branch itself is not a long-document detector.

Late fusion is a conservative approach. The final score is the arithmetic mean of the two probabilities. This avoids the need for a second-stage meta-classifier, the behaviour of which would be difficult to validate under distribution shift. Simple averaging assumes that both branches are useful and reasonably calibrated, which is practical but not guaranteed. Since PAN evaluation includes a Brier component, future work should test the use of explicit calibration or validation-tuned weighting.

The system does not include an abstention band. It always outputs the raw averaged probability. While this preserves ranking information for ROC-AUC, it may be suboptimal for C@1 and F0.5_u. One possible extension would be to tune a narrow interval around 0.5 on the validation set and map those cases to exactly 0.5.

The system's main strength is that its two branches make different kinds of errors. However, it relies on supervised training data and surface artefacts that may change as generators and obfuscation strategies evolve, which is its main weakness.

Acknowledgments

This work was carried out as part of the Language Processing 2 course at the University of Copenhagen

under the supervision of Manex Agirrezabal and Patrizia Paggio. We thank the PAN organisers for providing the shared task infrastructure and the TIRA evaluation platform [22].

Declaration on Generative AI

During the preparation of this work, the authors used Claude (Anthropic) to assist with the editing of this notebook paper. Having used this tool, the authors reviewed and edited the content as required, taking full responsibility for the publication's content.

References

[1] M. Jakesch, J. T. Hancock, M. Naaman, Human heuristics for AI-generated language are flawed, Proceedings of the National Academy of Sciences 120 (2023) e2208839120.

[2] D. Ippolito, D. Duckworth, C. Callison-Burch, D. Eck, Automatic detection of generated text is easiest when humans are fooled, in: Proceedings of ACL 2020, Association for Computational Linguistics, 2020, pp. 1808–1822.

[3] V. S. Sadasivan, A. Kumar, S. Balasubramanian, W. Wang, S. Feizi, Can AI-generated text really be detected?, arXiv preprint arXiv:2303.11156 (2023).

[4] J. Bevendorff, et al., Overview of the voight-kampff generative AI detection task at PAN 2026, in: Working Notes of CLEF 2026, 2026. (to appear).

[5] 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.

[6] J. Bevendorff, R. R. Gunti, J. Karlgren, M. Fröbe, M. Potthast, B. Stein, Overview of the third "Voight-Kampff" Generative AI / LLM Detection Task at PAN and ELOQUENT 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.

[7] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, A. Gulin, CatBoost: Unbiased boosting with categorical features, in: Advances in Neural Information Processing Systems 31 (NeurIPS 2018), 2018, pp. 6638–6648.

[8] A. Hans, A. Schwarzschild, V. Cherepanova, H. Kazmi, A. Saha, M. Goldblum, J. Geiping, T. Goldstein, Spotting LLMs with binoculars: Zero-shot detection of machine-generated text, arXiv preprint arXiv:2401.12070 (2024).

[9] G. Bao, Y. Zhao, Z. Teng, L. Yang, Y. Zhang, Y. Cui, Fast-DetectGPT: Efficient zero-shot detection of machine-generated text via conditional probability curvature, in: Proceedings of ICLR 2024, 2024. URL: https://arxiv.org/abs/2310.05130.

[10] P. He, J. Gao, W. Chen, DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing, in: International Conference on Learning Representations (ICLR 2023), 2023.

[11] S. Gehrmann, H. Strobelt, A. Rush, GLTR: Statistical detection and visualization of generated text, in: Proceedings of ACL 2019, System Demonstrations, Association for Computational Linguistics, 2019, pp. 111–116.

[12] J. Lorenz, V. Egger, C. Schöch, TF-IDF SVM for AI text detection: Simple yet robust, in: Working Notes PAN @ CLEF 2025, 2025.

[13] R. Biswas, LLM detect AI generated text — 1st place solution, Kaggle Competition, 2024. URL: https://github.com/rbiswasfc/llm-detect-ai.

[14] H. Abburi, J. Bhanu, Y. Sharma, Ensemble methods for AI-generated text detection, 2025. URL: https://arxiv.org/abs/2505.11550.

[15] V. A. Gromov, A. S. Kogan, Spot the bot: Coarse-grained partition of semantic paths, in: Pattern Recognition and Machine Intelligence (Lecture Notes in Computer Science), Springer, 2023, pp. 348–355.
[16] T. S. Juzek, N. Ward, ChatGPT's language: RLHF-induced overuse of 21 specific words, arXiv preprint arXiv:2412.11385 (2025).
[17] DeepSeek-AI, D. Guo, D. Yang, H. Zhang, et al., DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning, arXiv preprint arXiv:2501.12948 (2025).
[18] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, C. Finn, DetectGPT: Zero-shot machine-generated text detection using probability curvature, in: Proceedings of ICML 2023, 2023. URL: https://arxiv.org/abs/2301.11305.
[19] J. Su, T. Y. Zhuo, D. Wang, P. Nakov, DetectLLM: Leveraging log rank information for zero-shot detection of machine-generated text, arXiv preprint arXiv:2306.05540 (2023).
[20] E. Tavan, M. Najafi, Binoculars and LoRA fine-tuned language models for AI text detection, in: Working Notes PAN @ CLEF 2025, 2025.
[21] Y. Huang, W. Chen, J. Liu, Chunk-level AI text detection with multi-scale positive-unlabelled loss, in: Working Notes PAN @ CLEF 2025, 2025.
[22] M. Fröbe, M. Wiegmann, N. Kolyada, B. Grahm, T. Elstner, F. Loebe, M. Hagen, B. Stein, M. Potthast, Continuous Integration for Reproducible Shared Tasks with TIRA.io, in: J. Kamps, L. Goeuriot, F. Crestani, M. Maistro, H. Joho, B. Davis, C. Gurrin, U. Kruschwitz, A. Caputo (Eds.), Advances in Information Retrieval. 45th European Conference on IR Research (ECIR 2023), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2023, pp. 236–241. URL: https://link.springer.com/chapter/10.1007/978-3-031-28241-6_20. doi:10.1007/978-3-031-28241-6_20.