Team Suspiciously Coherent at PAN 2026: We stacked two Qwens and hoped for the best

Notebook for the "Voight-Kampff" Generative AI Detection PAN Lab at CLEF 2026

Zuzanna Antoszewska[†], Natalia Maria Bryła[†] and Alexandra Victoria Szczerba[*,†]

University of Copenhagen, Nørregade 10, København K 1165, Denmark

Abstract

This paper describes our approach to the "Voight-Kampff" Generative AI Detection task from PAN at CLEF. Our system is based on a weighted ensemble of two classifiers: a lightweight frozen-embedding classifier built on Qwen3-4B-Base representations with a LightGBM backend, and a parameter-efficient LoRA fine-tuned Qwen3-14B-Base classifier. Additionally we constructed a training and evaluation pipeline combining the official PAN datasets with additional multilingual benchmarks, homoglyph attack, and self-generated obfuscations, to improve robustness of our model. The final system achieved a mean score of 0.975 on the official PAN 2026 test set. The results show that combining lightweight embedding-based approaches with large fine-tuned language models improves robustness and generalization in machine-generated text detection, particularly under out of the distribution conditions.

Keywords

PAN 2026, Voight-Kampff Generative AI Detection 2026, Machine-Generated Text Detection, Binary AI Detection task, Large Language Models, Ensemble Models

1. Introduction

Generative AI (GAI) has become an integral part of people's lives, with text generation being one of its most common applications. Large Language Models (LLMs) are now used as routine writing tools across many domains like journalism, education, and research [1, 2, 3].

As this technology continues to advance and becomes increasingly integrated into our daily lives, distinguishing machine-generated text from human-written content poses a great challenge. While LLMs can help produce texts and automate the writing process, the widespread expansion of machine-generated content carries notable risks, including the spread of misinformation, fake news, and biased outputs [4, 5]. Even beyond these concerns, the growing use of LLMs raises important questions around authorship, since the boundary between human and machine authors is increasingly blurred. In some cases, this ambiguity in authorship attribution can lead to plagiarism and academic fraud [6, 7]. These risks highlight the importance of developing and improving systems that verify human vs. machine authorship.

The "Voight-Kampff" Generative AI Detection shared task [8], as a part of PAN lab [9] at CLEF 2026 aims to contribute to research in this field and promote the development of systems capable of reliably detecting AI-generated text [10]. To better reflect real-world usage and stress-test detection systems, LLMs in this task were instructed to alter their style and mimic specific human authors. Participants are additionally required to build systems resilient to various obfuscation techniques.

In this work, we built an ensemble system based on two classifiers: an embedding classifier with frozen Qwen3-4B-Base embeddings and a parameter-efficient fine tuned Qwen3-14B-Base classifier. For replication purposes, we publish the source code of the our approach.

The contributions of this work are:

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

[*]Corresponding author.

[†]These authors contributed equally.

phj702@alumni.ku.dk (Z. Antoszewska); xml567@alumni.ku.dk (N. M. Bryła); xvc653@alumni.ku.dk (A. V. Szczerba)

0009-0007-4370-2036 (Z. Antoszewska); 0009-0009-5166-7941 (N. M. Bryła); 0009-0004-2214-5662 (A. V. Szczerba)

- We propose a robust ensemble-based approach for AI-generated text detection that combines a fine-tuned Qwen3 model with a lightweight frozen-embedding classifier.
- We design a training, fine-tuning, and evaluation pipeline that combines multiple benchmark datasets and incorporates adversarial augmentations to improve robustness against obfuscation attacks and out-of-distribution data.
- We construct a diverse set of synthetic obfuscations targeting stylistic and linguistic transformations in order to evaluate and improve detector robustness.

2. Related Work

Research on distinguishing machine-generated from human-written text is already extensive, and the systems submitted to PAN 2025 demonstrate how diverse the existing approaches are. Among the submitted systems you could find large fine-tuned encoder models, detectors based on stylometric and structural features, surprisal- and perplexity-based detectors, logit-based approaches, as well as hybrid methods combining multiple techniques [11, 12, 13, 14, 15].

Although both the baseline and most of the submitted systems achieved high overall performance, last year's challenge revealed an important limitation: detectors performed well on regular patterns, but their performance degraded as soon as prompt-based obfuscations were introduced [10, 16] . The best-performing systems were those trained to anticipate such irregularities [11, 17]. This observation is consistent with broader research on machine-generated text detection, which shows that detectorsmust be resistant to adversarial attacks, topic shifts, and out-of-distribution data [18, 19].

One way to improve a model's predictions is to study attacks designed to weaken machine-generated text (MGT) detectors. Some of the most effective strategies are homoglyph attacks and other Unicode-based character replacements, which can significantly reduce detection accuracy [18, 20]. Incorporating these perturbations into the small portion of the training data will result in mitigating their impact on classifier performance [11, 21].

Another issue frequently highlighted in research on MGT detectors is the lack of diversity in the data the detectors are trained on. Existing datasets often cover only a limited range of topics and languages and the machine-generated samples are produced by a small group of models [19, 20, 22]. This limits the ability of detectors to generalize to unseen domains and generators.

Finally, last year's challenge results confirmed that stylistic obfuscations (for example instructing a model to write like a seven-year-old or to follow unusual syntactic constraints) pose a significant challenge for detection systems [10]. Adding obfuscated texts and prompts, that instruct models to change their writing style, in both training and validation data may therefore be a natural and effective preventive measure.

Regarding model architecture, our approach is inspired by techniques that have already been shown to perform well in AI-generated text detection. The standard and well working approach is fine-tuning large language models specifically for the detection taks. This has proven to be highly effective both in last year's challenge and previous research [11, 15, 17]. At the same time, we wanted to explore more lightweight approaches, such as the Unibuc-NLP system, which combined frozen LLM embeddings with traditional classifiers [13]. Despite its significantly lighter architecture, the system achieved competitive performance comparable to larger models.

Inspired by the strengths of both approaches, we decided to combine them in an ensemble-based system. Ensemble methods can improve prediction accuracy while reducing variance and increasing the reliability of predictions across different data samples. By combining multiple models, ensemble systems cancel out individual model errors and can achieve greater overall accuracy [23]. Ensemble methods have also been heavily used in machine-generated text detection [12, 24].

Our approach draws on the conclusions from PAN 2025 and is rooted in findings from the broader literature on machine-generated text detection. We propose an ensemble-based system trained on an augmented dataset designed to account for adversarial attacks and out-of-distribution data.

3. Dataset

The choice of training and testing data was a key design decision for our classifier. Our approach is an ensemble of two models: an embedding classifier with frozen Qwen3-4B-Base embeddings (4.1) and a parameter-efficient fine-tuned Qwen3-14B-Base model (4.2). Both models were trained and fine-tuned on the same dataset, which we describe in detail in this section.

3.1. Training & Fine Tuning

The dataset was built from texts from multiple sources in order to expose the models to multiple generation patterns and adversarial attacks. It consisted of the official PAN training and 95% of the PAN validation data augmented with a homoglyph attack applied to 10% of the training samples, following the approach described in [18]. 5 % of the validation data was reserved for internal validation.

The dataset was further boosted with a subset of the M4GT dataset containing both human- and machine-authored samples [22]. For all M4GT datasets, we used only the multilingual Subtask A split. Lastly, we expanded the dataset with self-generated obfuscations generated by our team using Gemini, OpenAI, and Claude models. A more detailed description of the obfuscations is provided below in Section 3.3. We used the combined dataset for training the frozen-embedding classifier and fine-tuning the parameterefficient classifier.

3.2. Validation

Our training, fine-tuning, and evaluation pipeline used three distinct validation sets, each serving a different purpose:

Internal (in-distribution): A small subset of the PAN data (5% of validation data). Used for fast feedback during development, like debugging, early stopping, and checking whether modifications improved performance on familiar data.

Internal OOD: Built from the M4GT dataset using only samples from a held-out generator (Llama2-fine-tuned) and a held-out language group (Italian). This was the primary validation set for model selection and choosing the best ensemble configuration.

External: Prepared of fully independent benchmarks, including RAID, a subset of M4GT, and MULTITuDEv2, with no overlap with the training data [20, 22, 25]. This set was used for final robustness evaluation across different domains, generators, languages, and adversarial attacks.

3.3. Obfuscations

To improve robustness against detection evasion, we generated a set of obfuscated machine texts using a two-step pipeline similar to the one used for generating machine texts in last year's training and test sets. We first prompted a model to generate a 3–5 bullet point summary of a human text, then used those bullet points to generate a new 350–500 word text in the original text's style. Obfuscations were generated using GPT-4.1-mini, Claude, and Gemini for training, and GPT-4.1-mini, Gemini, and Llama-3.2-3B-Instruct for external validation.

The obfuscation prompts were intended to introduce a number of stylistic changes to the machine-generated texts. Training obfuscations included alternative word orders (Arabic VSO grammar), personality-based writing styles (Porky Pig, Victorian child, 7-year-old), and cross-lingual speech styles (Frenglish). Test-set obfuscations introduced similar, but unseen variants: Gen-Z-style, Hindi SOV word order, Italian-accented English, and an Elmer Fudd personality prompt.

The additional human source texts used for generating obfuscations were drawn from Project Gutenberg fiction, CNN news articles, and IELTS sample essays.

Table 1
Overview of all datasets across the experimental pipeline.

Split	Dataset	Labels	Size	Notes
Training & fine-tuning	PAN Training	Human & Machine	∼23,700	10% homoglyph attack applied to training data
Training & fine-tuning	PAN Validation (95% subset)	Human & Machine	∼3,300	10% homoglyph attack on 95% of official val pool
Training & fine-tuning	Self-generated obfuscations	Machine	∼1,240	Generated by team using Gemini, OpenAI, Claude
Training & fine-tuning	M4GT (subset)	Human & Machine	∼150,600	Subtask A multilingual
Internal val	PAN Validation (5% subset)	Human & Machine	∼300	In-distribution; used for early stopping and debugging
Internal OOD val	M4GT (held-out)	Human & Machine	6,075	Held-out Italian texts and Llama-fine-tuned outputs
External validation	RAID-extra	Human & Machine	4,416	
External validation	M4GT-Bench (held-out subset)	Human & Machine	∼5, 500	Subtask A multilingual
External validation	MULTITuDEv2	Human & Machine	∼6,000	Without obfuscated human samples
External validation	Self-generated obfuscations (Round 2)	Machine	∼1,200	Generated using Gemini, OpenAI, Llama
External validation	Additional human texts	Human	∼1,995	Texts not used for machine text generation
Obfuscation source	Project Gutenberg fiction	Human	∼1,500	
Obfuscation source	CNN News (Kaggle)	Human	∼1,500	News articles
Obfuscation source	IELTS Essays (Kaggle)	Human	∼1,435	Scored essays

4. The Model

Our system is a weighted ensemble of two complementary classifiers: an embedding-based classifier built on frozen language-model representations, and a parameter-efficient fine-tuned classifier. The two components look at the data in different ways. The embedding-based classifier uses a frozen language model to capture broad patterns in how a text is written. The fine-tuned classifier is trained directly on the detection task, so it learns to spot finer, more specific cues. Because text obfuscations affect these two kinds of signals differently, combining the classifiers gives more robust predictions than either one on its own.

4.1. Embedding-based Classifier

We obtain text representations from Qwen3-4B-Base [26], a 4-billion-parameter decoder-only language model. For each input text, we pass it through the model once and collect the internal representations produced by the last layer. Since texts have different lengths and shorter ones are padded with dummy tokens, we average only the real token representations, ignoring the padding, to obtain a single fixed-length vector per text. The model weights remain frozen during the later feature extraction, so no gradient updates are performed.

These embeddings serve as input features to a LightGBM classifier [27], a gradient-boosted decision-tree model. To make this decision we evaluated four classifiers on top of the frozen Qwen3-4B-Base embeddings on the validation set: Logistic Regression (Mean = 0.9952), Calibrated SVM (0.9890), XGBoost (0.9966), and LightGBM (0.9950). Although XGBoost achieved the highest validation score, we selected LightGBM for the final system because it trains significantly faster on large feature matrices

Table 2
Training configuration of the fine-tuned classifier.

Setting	Value
Training data	10 JSONL files (iterative, sequential)
Epochs per file	2
Learning rate	2×10^{-4}
LR schedule	linear (AdamW default)
Batch size	4
Gradient accumulation	4 steps (effective batch size 16)
LoRA rank (r)	8
LoRA alpha (α)	16
LoRA dropout	0.05
Target modules	`q_proj`, `v_proj`
Optimizer	AdamW [31]
Max sequence length	512 tokens
Precision	bfloat16
Seed	42

and produces smaller model files. That was an important practical consideration given the Docker image size constraints of the TIRA submission pipeline. The classifier outputs a probability score for the machine-generated class.

4.2. Fine-tuned Classifier

The second component of the full classifier is Qwen3-14B-Base [28] fine-tuned for AI text detection with the use of LoRA (Low-Rank Adaptation) [29]. An important aspect of this implementation is that the pretrained weights of the chosen large language model are kept frozen. Then instead of updating them, LoRA injects small trainable matrices into two parts of each attention layer. The layers are specifically the query and value projections. Those two are altered because thay are the components responsible for determining what information each token pays attention to.

We set the LoRA rank to $r = 8$, it controls the dimensionality of the injected update matrices. A rank of 8 gives a balance between expressiveness and parameter efficiency. In more detail, the matrices are large enough to capture task-specific adaptations for the given task binary classification, and small enough to keep the number of trainable parameters a small fraction of the full model size. The scaling factor $\alpha = 16$ controls how strongly LoRA updates influence the model's outputs. Setting it to twice the rank ($\alpha = 2r$) is a common default that keeps the updates small enough to not overwrite what the model have already learned during pretraining. A dropout rate of 0.05 is applied to the LoRA layers, which during training makes 5% of random LoRA updates being ignored. This is a light regularization measure to prevent the model from overfitting, which is a risk when fine-tuning a very large model (14 billion parameters) on a relatively simple binary classification task.

In addition, the model was fine-tuned using an iterative training procedure implemented with the HuggingFace Trainer API [30]. Instead of training on all the data at once, the training data was split into multiple files of the same size. Then the model was trained sequentially on each file, with every step initializing from the adapter saved in the previous step. This iterative approach allows the model to be exposed to different subsets of the data progressively, building on what it has already learned. Training uses the AdamW optimiser [31] with a constant learning rate of 2×10^{-4}. All training was performed in bfloat16 precision on GPU and the model was evaluated on a separate unseen dataset after each epoch using accuracy, F1, and ROC-AUC. All the other parameters used for fine-tuning are shown in Table 2.

4.3. Ensemble

In order to obtain the best possible performance on the task, we have decided to use a weighted-average ensemble. We arrived at this decision after comparing several alternatives: the frozen-embedding classifier (Section 4.1) and the fine-tuned Qwen classifier (Section 4.2) evaluated separately, a stacking ensemble, and a pipeline arrangement of the two models.

All the above evaluations were performed on our external validation dataset (Section 3.2) and the results can be found in Table 3 and Table 4 which presents the ROC-AUC and mean scores of ROC-AUC, Brier, C@1, F1 accordingly. Since our RAID validation set is only a subset of the full validation data, we gave more weight to the results on the complete validation set. Our primary evaluation metric was ROC-AUC, which measures the model's ability to rank human-written texts above machine-generated ones across all classification thresholds, making it well-suited for a system that outputs continuous probability scores rather than hard binary decisions. The stacked ensemble achieved the highest ROC-AUC on the validation set; however, stacking requires labeled data to train the meta-learner, which is not available at test time. We therefore selected the weighted average ensemble, which achieved the second-best scores on both ROC-AUC and mean metric, as our final approach.

The final prediction is a weighted average of the two classifier scores. The weights are derived from each model's individual ROC-AUC score on the full validation set. It means that each weight is computed as the model's ROC-AUC divided by the sum of both ROC-AUC scores, so that the better-performing model contributes more to the final prediction. This approach requires only validation labels and the weights were later hard-coded. The resulting weights are 0.47 for the frozen embedding with LightGBM classifier and 0.53 for the fine-tuned Qwen3-14B-Base classifier, reflecting the slightly stronger performance of the fine-tuned model on the validation set. The weighted average equation is shown below

$$s = w_{\mathrm{LGB}} \cdot p_{\mathrm{LGB}} + w_{\mathrm{ft}} \cdot p_{\mathrm{ft}}, \tag{1}$$

where p_{LGB} and p_{ft} are the machine-generated class probabilities from the frozen embeddings with LightGBM and fine-tuned Qwen3-14B-Base classifiers respectively. The weights are $w_i = \mathrm{AUC}_i / \sum_j \mathrm{AUC}_j$, giving $w_{\mathrm{LGB}} = 0.47$ and $w_{\mathrm{ft}} = 0.53$. Finally, the score $s \in [0, 1]$ is used directly as the predicted probability of the machine-generated class and our final prediction for a given input.

5. Our Results

We evaluate six detection configurations on two out-of-distribution evaluation sets. `Frozen Emb` is our LightGBM classifier trained on frozen Qwen3-4B-Base embeddings. `Qwen` is the LoRA fine-tuned Qwen3-14B-Base. `Stack` averages the two detector outputs with equal weight, while `Weighted Avg` uses optimized weights learned via meta-classification. `FE` $\rightarrow$ `Qwen` routes samples to the Qwen model only when the frozen-embedding detector is uncertain (probability in $[0.35, 0.65]$); `Qwen` $\rightarrow$ `FE` routes in the reverse direction. The `raid` row reports performance on our 4,416-sample RAID extra subset, while `full` reports on the team's external OOD validation set combining RAID, a held out M4GT subset, MULTITuDEv2, self-generated obfuscations, and additional human authored texts (Section 3.2). A more detailed explanation of our approach can be found in Section 4.

Table 3
ROC-AUC scores across detector configurations and evaluation sets.

Data	Frozen Emb	Qwen	Stack	FE → Qwen	Qwen → FE	Weighted Avg
RAID	0.790	0.706	0.844	0.780	**0.907**	0.848
Full	0.792	0.877	**0.888**	0.827	0.832	0.888

The results reveal that on RAID, where the frozen-embedding detector individually outperforms Qwen (0.790 vs. 0.706 ROC-AUC), the reverse routing strategy`Qwen` $\rightarrow$ `FE`, which trusts the fine-tuned model and falls back to the embedding classifier on uncertain samples gets the highest score (0.907). On

Table 4
Mean scores (average of ROC-AUC, Brier, C@1, F1) across detector configurations.

Data	Frozen Emb	Qwen	Stack	FE → Qwen	Qwen → FE	Weighted average
RAID	0.749	0.712	0.808	0.768	**0.851**	0.788
Full	0.756	0.84	**0.849**	0.791	0.745	0.847

the broader external OOD set, where Qwen individually outperforms the frozen embeddings (0.877 vs. 0.792), simple stacking via average is almost the same as with the weighted variant (0.888 vs. 0.888).

5.1. Comparison with Baseline Detectors

To contextualize our system's performance, we compare it against two reference detectors evaluated on our final external OOD set (18,711 samples): a TF-IDF SVM, representing the classical lexical-feature approach, and Binoculars, a zero-shot detector that requires no task-specific training. Table 5 reports the results.

Table 5
Mean-score comparison of baseline detectors against our final weighted-average ensemble on the external OOD validation set.

System	Mean
TF-IDF SVM	0.659
Binoculars (zero-shot)	0.830
Weighted Avg (ours)	**0.847**

The TF-IDF baseline performs barely above chance (0.659 mean), confirming the tendency of lexical-feature detectors to overfit and collapse under distributional shift. Binoculars, which exploits model-agnostic perplexity signals, is much more robust (0.830) and provides a strong zero-shot reference. Our weighted-average ensemble exceeds both (0.847), and demonstrates that combining a supervised embedding classifier with a fine-tuned model captures signals beyond what lexical features or zero-shot perplexity detection can do.

6. Final Test Set Results

Table 6 shows our submission's performance on the four evaluation datasets used by the PAN 2026 organizers. The scores were computed by the official TIRA evaluation pipeline. Our system, registered on TIRA under the name `lead-log`, achieves a mean score of 0.975 on the official PAN 2026 test set.

Table 6
Final results of our submitted system across the evaluation datasets. The PAN 2026 test set is the official ranking benchmark.

Dataset	ROC-AUC	Brier	C@1	F1	F05U	Mean
PAN 2026 test	0.989	0.965	0.965	0.976	0.980	**0.975**
PAN 2025 test	0.998	0.980	0.970	0.979	0.967	**0.979**
Smoke test	0.958	0.936	0.912	0.909	0.893	0.922
ELOQUENT 2026	0.955	0.831	0.823	0.896	0.954	0.892

The distributional proximity of the PAN test set to the training distribution is much closer than that of RAID or our own external OOD set. The result is that our system, which was designed and tuned to handle a strong distributional shift, scores *higher* on the relatively in-distribution PAN test set (0.975) than on the external validation set (0.847 in Table 4).

The ELOQUENT score of 0.892 (Table 6) is important as the set introduces variation that the PAN test set may not capture. The 0.083-point gap between the PAN test mean and the ELOQUENT mean provides a more conservative estimate of our system's robustness.

Acknowledgments

We would like to say thank you to our Msc IT and Cognition instructors at the University of Copenhagen. Special thanks to those who helped us in our NLP course and provided guidance with how to approach this challenge. Thanks to the developers of ACM consolidated LaTeX styles https://github.com/borisveytsman/acmart and to the developers of Elsevier updated LaTeX templates https://www.ctan.org/tex-archive/macros/latex/contrib/els-cas-templates.

Declaration on Generative AI

During the preparation of this work, the authors used AI for: Grammar and spelling check. Further, the authors used Claude to generate the LaTex syntax for the tables and paper structure. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] A. T. Neumann, Y. Yin, S. Sowe, S. Decker, M. Jarke, An LLM-Driven Chatbot in Higher Education for Databases and Information Systems, IEEE Transactions on Education 68 (2025) 103–116. URL: https://ieeexplore.ieee.org/document/10706931/. doi:10.1109/TE.2024.3467912.

[2] N. L. Rane, A. Tawde, S. P. Choudhary, J. Rane, Contribution and performance of ChatGPT and other Large Language Models (LLM) for scientific and research advancements: a double-edged sword, Open Access 05 (2023).

[3] S. Cheng, When Journalism Meets AI: Risk or Opportunity?, Digital Government: Research and Practice 6 (2025) 1–12. URL: https://dl.acm.org/doi/10.1145/3665897. doi:10.1145/3665897.

[4] S. Park, X. Nan, Generative AI and misinformation: a scoping review of the role of generative AI in the generation, detection, mitigation, and impact of misinformation, AI & SOCIETY 41 (2026) 1501–1515. URL: https://link.springer.com/10.1007/s00146-025-02620-3. doi:10.1007/s00146-025-02620-3.

[5] A. Loth, M. Kappes, M.-O. Pahl, Blessing or curse? A survey on the Impact of Generative AI on Fake News, 2024. URL: http://arxiv.org/abs/2404.03021. doi:10.48550/arXiv.2404.03021, arXiv:2404.03021 [cs.CL].

[6] D. J. Puche Villalobos, La inteligencia artificial y el fraude académico en el contexto universitario, Revista Digital de Investigación y Postgrado 6 (2025) 73–93. URL: https://redip.iesip.edu.ve/ojs/index.php/redip/article/view/140. doi:10.59654/kg944e15.

[7] M. Byrne, The Disruptive Impacts of Next Generation Generative Artificial Intelligence, CIN: Computers, Informatics, Nursing 41 (2023) 479–481. URL: https://journals.lww.com/10.1097/CIN.0000000000001044. doi:10.1097/CIN.0000000000001044.

[8] 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.

[9] 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.
[10] J. Bevendorff, Y. Wang, J. Karlgren, M. Wiegmann, M. Fröbe, A. Tsivgun, J. Su, Z. Xie, M. 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 (2025).
[11] D. Macko, mdok of KInIT: Robustly Fine-tuned LLM for Binary and Multiclass AI-Generated Text Detection, 2025. URL: https://arxiv.org/abs/2506.01702. doi:10.48550/ARXIV.2506.01702, version Number: 2.
[12] Enhancing AI Text Detection with Frozen Pretrained Encoders and Ensemble Learning (2025).
[13] T. Marchitan, C. Creanga, L. P. Dinu, Unibuc - NLP at "Voight-Kampff" Generative AI Detection PAN 2025 (2025).
[14] M. Seeliger, P. Styll, M. Staudinger, A. Hanbury, Human or Not? Light-Weight and Interpretable Detection of AI-Generated Text (2025).
[15] J. Yang, K. Yan, Genre-Aware Contrastive Learning for AI Text Detection: A RoBERTa-Based Approach (2025).
[16] G. P. Georgiou, What Distinguishes AI-Generated from Human Writing? A Rapid Review of the Literature, Big Data and Cognitive Computing 10 (2026) 55. URL: https://www.mdpi.com/2504-2289/10/2/55. doi:10.3390/bdcc10020055.
[17] J. Liu, L. Kong, Z. Peng, F. Chen, Generative AI Authorship Verification Based on Contrastive-Enhanced Dual-Model Decision System (2025).
[18] D. Macko, R. Moro, A. Uchendu, I. Srba, J. S. Lucas, M. Yamashita, N. I. Tripto, D. Lee, J. Simko, M. Bielikova, Authorship Obfuscation in Multilingual Machine-Generated Text Detection, in: Findings of the Association for Computational Linguistics: EMNLP 2024, Association for Computational Linguistics, Miami, Florida, USA, 2024, pp. 6348–6368. URL: https://aclanthology.org/2024.findings-emnlp.369. doi:10.18653/v1/2024.findings-emnlp.369.
[19] Z. Ahmad, M. Torres-Ruiz, A. Mahmood, R. Quintero, I. Ameer, N. Bölücü, Human or Machine? A Survey on Machine-Generated Text Detection, IEEE Access 14 (2026) 34113–34136. URL: https://ieeexplore.ieee.org/document/11404135/. doi:10.1109/ACCESS.2026.3666781.
[20] L. Dugan, A. Hwang, F. Trhlik, J. M. Ludan, A. Zhu, H. Xu, D. Ippolito, C. Callison-Burch, RAID: A Shared Benchmark for Robust Evaluation of Machine-Generated Text Detectors, 2024. URL: http://arxiv.org/abs/2405.07940. doi:10.48550/arXiv.2405.07940, arXiv:2405.07940 [cs.CL].
[21] D. Macko, R. Moro, I. Srba, Increasing the Robustness of the Fine-tuned Multilingual Machine-Generated Text Detectors, 2025. URL: http://arxiv.org/abs/2503.15128. doi:10.48550/arXiv.2503.15128, arXiv:2503.15128 [cs.CL].
[22] Y. Wang, J. Mansurov, P. Ivanov, J. Su, A. Shelmanov, A. Tsvigun, O. M. Afzal, T. Mahmoud, G. Puccetti, T. Arnold, A. F. Aji, N. Habash, I. Gurevych, P. Nakov, M4GT-Bench: Evaluation Benchmark for Black-Box Machine-Generated Text Detection, 2024. URL: http://arxiv.org/abs/2402.11175. doi:10.48550/arXiv.2402.11175, arXiv:2402.11175 [cs.CL].
[23] F. Acito, Predictive Analytics with KNIME: Analytics for Citizen Data Scientists, Springer Nature Switzerland, Cham, 2023. URL: https://link.springer.com/10.1007/978-3-031-45630-5. doi:10.1007/978-3-031-45630-5.
[24] K. Aggarwal, S. Singh, Parul, V. Pal, S. S. Yadav, A Framework for Enhancing Accuracy in AI Generated Text Detection Using Ensemble Modelling, in: 2024 IEEE Region 10 Symposium (TENSYMP), IEEE, New Delhi, India, 2024, pp. 1–8. URL: https://ieeexplore.ieee.org/document/10752173/. doi:10.1109/TENSYMP61132.2024.10752173.
[25] D. Macko, R. Moro, A. Uchendu, J. S. Lucas, M. Yamashita, M. Pikuliak, I. Srba, T. Le, D. Lee, J. Simko, M. Bielikova, MULTITuDE: Large-Scale Multilingual Machine-Generated Text Detection Benchmark, in: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, 2023, pp. 9960–9987. URL: http://arxiv.org/abs/2310.13606. doi:10.18653/v1/2023.emnlp-main.616, arXiv:2310.13606 [cs.CL].
[26] Q. Team, Qwen3-4b-base, https://huggingface.co/Qwen/Qwen3-4B-Base, 2025. Hugging Face

model repository.
[27] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, T.-Y. Liu, Lightgbm: A highly efficient gradient boosting decision tree, in: Advances in Neural Information Processing Systems (NeurIPS), volume 30, 2017.
[28] Q. Team, Qwen3-14b-base, https://huggingface.co/Qwen/Qwen3-14B-Base, 2025. Hugging Face model repository.
[29] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, Lora: Low-rank adaptation of large language models, 2021. arXiv:2106.09685.
[30] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, et al., 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.
[31] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, in: International Conference on Learning Representations, 2019. URL: https://arxiv.org/abs/1711.05101.