Binoculars, BART, and Adversaries: Multi-Faceted AI Text Detection for PAN 2025

Notebook for PAN at CLEF 2025

Benjamin Ostrower[1], Poonam Dongare[1] and Mahitha Thekkinkattuvalappil Unnikrishnan[1]

[1] *Georgia Institute of Technology, 225 North Avenue, Atlanta, 30332, United States*

Abstract

With each passing year AI generated text methods continue to improve, blurring the line between human and machine. Our submissions recreate existing methods on competition data to achieve high scores on subtask 1 for the Voight-Kampff AI detection sensitivity task.

Keywords

PAN 2025, Adversarial fine tuning, AI text detection, BART, Binoculars

1. Introduction

The PAN competition is a part of the CLEF conference program [1] allowing contestants from across the globe to compete across an array of tasks related to natural language processing. This year the PAN workshop offered a generative ai text detection task [2], this paper focuses on subtask 1 where given a (potentially obfuscated) text decide whether it was written by a human or an ai where submissions are evaluated through the TIRA platform [3]. The committee provides a dataset that participants can use to train their own submissions. It is constructed of 13 LLMs that rephrased a human prompt so that each human authored prompt has atleast one LLM counterpart. This year the topics for the training set spanned essays, news, or fiction. A twist is also included that the evaluation set will include new models and other obfuscations not present in the provided dataset.

1.1. EDA

We began by exploring the dataset to look for patterns that might help distinguish human-written text from machine-generated text. First, we examined the word count distribution per author. Most texts ranged from 2,000 to 4,000 words, except for the Alpaca models, which were outliers (see Figure 1). However, this didn't reveal any clear signal for identifying human authorship.

Next, we looked at the usage of stop words using NLTK's stop word list (see Figure 3). The different authors—both human and machine—used stop words at very similar rates, again providing little insight for distinguishing authorship.

Finally, we applied a technique from [4] involving the "rank" of each word using GPT-2. For each text, we used a sliding window approach: starting with the first two words, then the first three, and so on, up to the model's maximum context window. At each step, we fed the current window into GPT-2 and checked how likely it thought the actual next word was. If the true next word was GPT-2's top prediction, we counted it as a match.

We found that human-written text had the lowest rate of top-ranked predictions by GPT-2, while texts from the GPT2-based instruct models had the highest overlap with GPT-2's predictions. This suggested that humans are less predictable by GPT-2, and that this kind of "next-word predictability" could be a useful signal for detecting machine-generated text. This insight helped guide our approach

CLEF 2025 Working Notes 9 – 12 September 2025, Madrid, Spain

bostrower3@gatech.edu (B. Ostrower); pdongare6@gatech.edu (P. Dongare); munnikris3@gatech.edu (M. T. Unnikrishnan)

to the competition, opting to incorporate fine-tuned language models and even utilizing the method in the token cohesiveness solution (section 2.2).

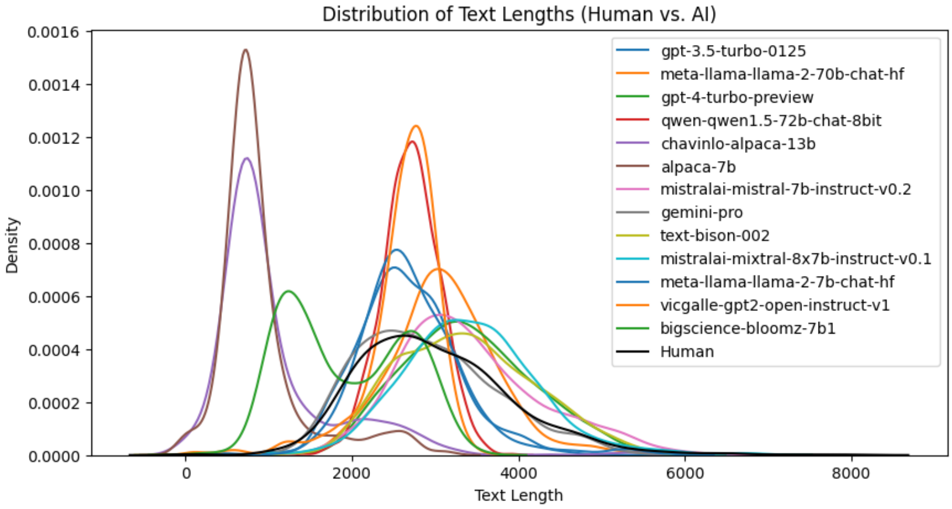

Figure 1: Distribution of text lengths for each author in the provided dataset

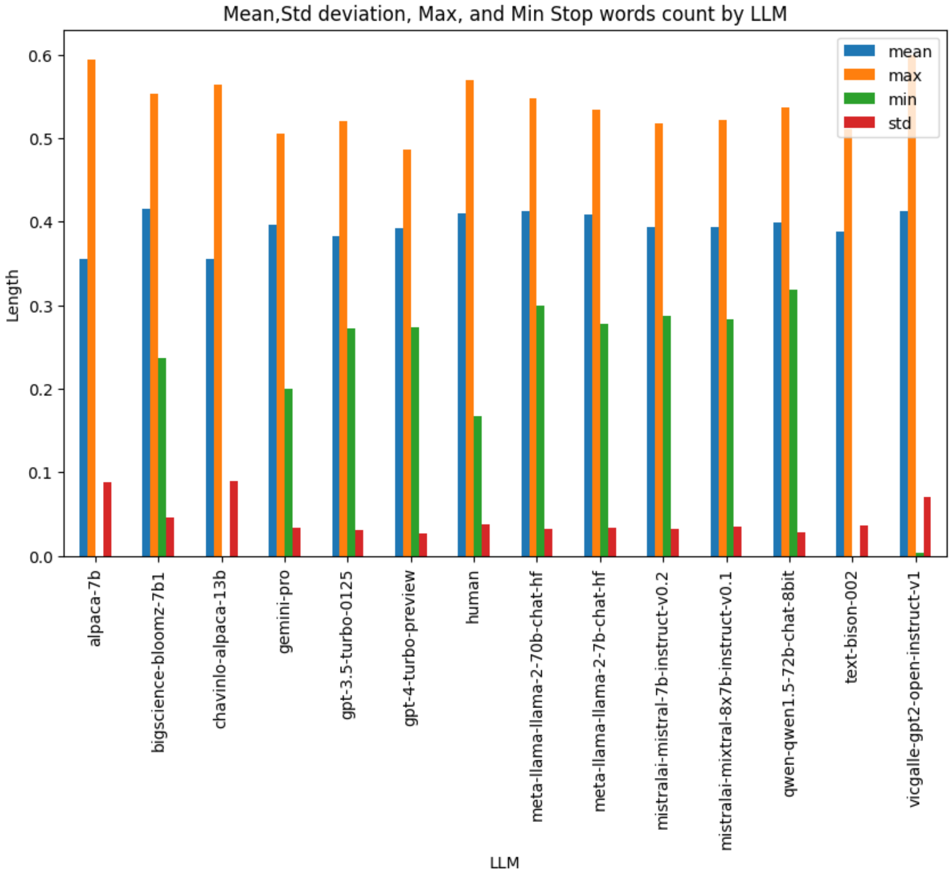

Figure 2: Counting statistics for stop word use by author

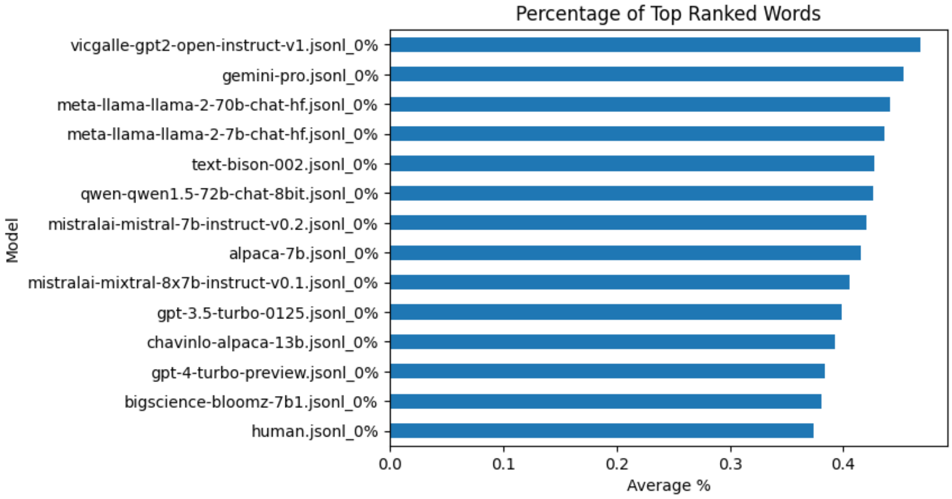

Figure 3: Percentage of words ranked in the top vocabulary choices for a GPT2 text completion

2. Methodology

2.1. Binocular Score based XGBoost model

In the landscape of AI-generated text detection, text generated by modern LLMs is thought to be difficult to detect, as both humans and language models show a wide range of complex behaviors.

2.1.1. Binocular Score

This section provides a background on the definition of binocular score and its calculation. The binocular score [5], method introduces an innovative idea to identify AI generated content by leveraging the difference between two nearly similar language models. This method uses two language models viz. observer model and the performer model. Perplexity shows how confused a language model is when trying to predict the next word. The binocular score is calculated using this concept of perplexity. As stated by Hans et al. (2024), It is defined as the ratio between the perplexity of the performer model and the cross-perplexity, where the latter refers to how perplexed the performer model is when evaluated using the observer model. Because one language model can easily understand and predict the output of another language model as compared to human written text. This pattern helps us to identify if the content is Human written or AI generated.

Based on the approach discussed above, below are the calculations for the perplexity , cross perplexity and Binocular score. As described by Hans et al. (2024), A character string s can be divided into a sequence of tokens and mapped to a list of token indices $\vec{x}$ through a tokenizer T. Each token index x_i denotes the ID of the i-th token, where $x_i \in V = 1, 2, \ldots, n$, and V is the vocabulary of the language model. Given this input sequence, a language model M produces a probability distribution over the vocabulary to predict the next token

$$M(T(s)) = M(\vec{x}) = Y$$

$$Y_{ij} = P(v_j \mid x_0 : x_{i-1}) \quad \text{for all } j \in V.$$

For simplicity, the notation $M(s)$ instead of $M(T(s))$, assuming the tokenizer applied to the text s is the same as the one used to train the language model M. Log-perplexity (log PPL) is a metric used to evaluate how well a language model predicts a given sequence of text. It is calculated for each token in the input by taking the average of the negative log-likelihood values. Hence it is defined as below:

$$\log \mathrm{PPL}_M(s) = -\frac{1}{L}\sum_{i=1}^{L} \log(Y_{ix_i}),$$

where $\vec{x} = T(s)$, $Y = M(\vec{x})$, and L is the total number of tokens in the string s."

This approach [5], also evaluates how unexpected the output of one model is when interpreted by another model. To measure the relationship between two models' predictions on the same input, we adopt the concept of cross-perplexity, as described by Hans et al. (2024). This metric measures how different the outputs of two language models, M_1 and M_2, are by calculating the average cross-entropy for each token in the tokenized version of the input s.

$$\log \text{X-PPL}_{M_1,M_2}(s) = -\frac{1}{L}\sum_{i=1}^{L} M_1(s)_i \cdot \log(M_2(s)_i),$$

where the symbol $\cdot$ indicates the dot product between two vector-valued outputs corresponding to the i-th token prediction.

The Binoculars score B acts as a refined version of perplexity, offering a normalized perspective on text predictability. Specifically, it is defined as the ratio between standard perplexity and cross-perplexity. Given two models, M_1 and M_2, and an input string s, the Binoculars score is given by:

$$B_{M_1,M_2}(s) = \frac{\log \mathrm{PPL}_{M_1}(s)}{\log \text{X-PPL}_{M_1,M_2}(s)}$$

In this approach, the numerator indicates how surprising the input text is to model M_1, based on its perplexity score. The denominator shows how unexpected model M_1 finds the predictions made by model M_2. If M_1 and M_2 are more alike than either is to a human, we would expect a human's predictions to differ more from M_1's than M_2's predictions do.

2.1.2. Model architecture and feature integration

We employ an ensemble learning approach based on the XGBoost algorithm to enhance predictive performance for the AI generated text detection task. The ensemble integrates a rich and heterogeneous set of features designed to capture lexical, semantic, syntactic, and stylistic aspects of the text. Specifically, we include the Binoculars score, a normalized metric that quantifies the divergence between language models by comparing perplexity with cross-perplexity, thereby capturing differences in model interpretation of text.

To Calculate the Binocular score we utilized GPT-2[6] model from Hugging face library as both observer and performer model. While we explored other models during our primary experimentation such as "falcon-7b"[7], "TinyDeepSeek-1.5b" [8] etc. Due to limitations in memory resources, the GPT-2 [6] model was chosen as the final architecture for calculating the Binocular score.

In addition, Term Frequency–Inverse Document Frequency (TF-IDF) is utilized to measure term significance across the entire corpus, producing a sparse vector format that highlights informative and distinguishing terms. To complement this with richer semantic understanding, we extract sentence-level embeddings from BERT, which provides dense, pre-trained representations encoding both syntactic and semantic information.

We used the "bert-base-uncased" [9, 10] checkpoint from Hugging Face to extract the BERT embeddings. The BERT base uncased model is not specifically optimized for sentence level tasks. In this architecture, the [CLS] token is mainly pre-trained to support classification task. Hence,to capture more richer sentence semantics we decided to use mean pooling over last hidden layer's token embeddings.

We integrated stylometric features such as the average length of sentences and the cumulative length of the text to reflect patterns inherent in the author's writing style. Furthermore, linguistic features—most notably various readability metrics (e.g., Flesch-Kincaid)—are computed to assess the complexity and accessibility of the language used.

By combining these diverse feature categories, the XGBoost ensemble utilizes both lexical-level attributes and high-level linguistic representations, thereby improving the precision and contextual relevance of its predictions.

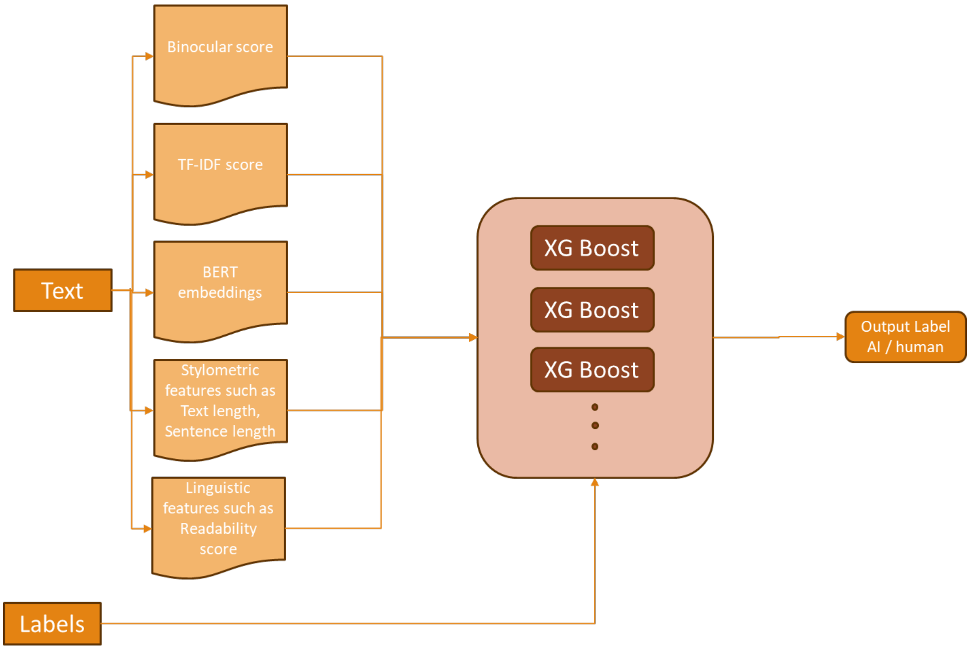
Binocular score
TF-IDF score
BERT embeddings
Stylometric features such as Text length, Sentence length
Linguistic features such as Readability score
Text
Labels
XG Boost
XG Boost
XG Boost
Output Label AI / human

Figure 4: Binocular score based XGBoost model architecture

2.2. Token Cohesiveness

[11] presents Token Cohesiveness as a novel feature, defined as the expected semantic variation between a given input text (x) and its modified version ($\tilde{x}$), generated by randomly removing a small subset of tokens. Due to the causal self-attention mechanism used by large language models (LLMs) in text generation, each token maintains a strong contextual relationship with preceding tokens. As a result, AI-generated text exhibits greater token cohesiveness compared to human-written content. The study incorporates token cohesiveness alongside several established zero-shot detection methods to enhance text classification and AI-authorship verification.

Based on the approach described in [11], we computed token cohesiveness as a feature in our analysis. For a given input text, we generated multiple perturbed versions by systematically removing a fixed proportion of tokens. The token cohesiveness score is determined by calculating the average negative BARTScore [12] between the original text and its modified copies. BARTScore [12] measures the sum of weighted probabilities for each token in (y), conditioned on the preceding tokens in (y) and another reference text (x). In our framework, (y) represents the perturbed text, while (x) corresponds to the original input text. Unlike the original weighting scheme, we assign equal weight to all tokens in the evaluation process.

$$BARTScore : p(y|x, \theta) = \prod_{t=1}^{m} p(y_t|y_{<t}, x, \theta) \tag{1}$$

$$TokenCohesiveness : u(x) = \frac{\sum_{i=1}^{n} \text{DIFF}(x, \tilde{x}_i)}{n} \tag{2}$$

DIFF is a semantic difference metric, here we are using negative BARTScore. The paper proposes a dual channel paradigm for text classification by passing input text to upper channel to calculate token cohesiveness and lower channel make prediction using existing zero shot detector. The two scores are combined together as follows

$$w(x) = \begin{cases} e^{u(x)} \times v(x), & \text{if } v(x) \geq 0 \\ e^{-u(x)} \times v(x), & \text{if } v(x) < 0 \end{cases} \tag{3}$$

We are using 3 existing zero shot detectors for our work and they are Likelihood[13], LRR[14] and FastDetectGpt[15]. Token cohesiveness scores combining each of the zero shot detectors and negative BARTScore are the features created using this approach.

In addition to token cohesiveness, the total number of tokens in a given text is also considered as a distinguishing feature. [16] explores how stylistic and complexity-based attributes can be leveraged to differentiate AI-generated text from human-authored content. Based on the findings of this study, the complexity characteristics examined include the word count, representing the total number of words, and the type-token ratio (TTR), which measures the proportion of unique vocabulary words relative to the total word count.

The stylistic features analyzed encompass several linguistic characteristics: the frequency of stop words, determined using the NLTK predefined stopword list; the frequency of words absent from the SentiWordNet dictionary; and the distribution of specific parts of speech (POS) - including nouns (POS tags: NNP, NNPS, NN, NNS), verbs (POS tags: VB, VBD, VBG, VBN, VBP, VBZ), and past-tense verbs (POS tags: VBD, VBN). These features collectively contribute to the identification of different patterns between AI-generated and human-authored texts.

In [17], the concept of intrinsic dimensionality is introduced as a means of distinguishing between AI-generated and human-generated text. The paper suggests that human-authored text exhibits a higher intrinsic dimensionality compared to text produced by AI models. Although the original study employs Persistent Homology Dimension Estimation for this calculation, we opt instead for Maximum Likelihood Estimation (MLE) using the Scikit-learn library to determine the intrinsic dimension of the given text and incorporate it as a feature in our analysis.

Unfortunately, this approach resulted in too large of a submission to comply with the competitions 15GB container format and was unable to be submitted, it did reach accuracy levels over 90% on the provided validation datasets.

2.3. Adversarial Training

Our Adversarial approach closely mirrors that found in [18] and can be summarized in 3 steps:

- **Data preparation**: We take the human prompt, randomly sample an AI generated prompt, and paraphrased the AI prompt with an off the shelf paraphraser to create 3 separate texts.
- **Paraphraser update**: We collect the reward (correctness of the detector) on only the paraphrased texts and use this reward to update the paraphraser
- **Detector update**: We use all 3 samples (human, AI, paraphrased) to update the detector with a logistic loss function.
  We collect a tuple of (human,AI, and paraphrased) 10 at a time to store in our memory buffer due to memory constraints. In our case the paraphraser we used was LLama 3.2-1B and the detector was a roberta-large model for sequence classification with 2 labels. Both models used an AdamW optimizer with a lr of 5e-6.

2.3.1. Training the paraphraser

The reward is defined on the left side as the predicted output of the Detector model on the paraphrased text $D_\phi(x_p)$ and the log probability of the text x_p is defined as the right hand side of equation 4.

$$R(x_p, \phi) = D_\phi(x_p) \in [0, 1]; \quad \log P_\phi(x_p \mid x_m) = \sum_{i=1}^{N} \log P_\phi(x_p^i \mid x_m, x_p^{1:i-1}) \tag{4}$$

The authors from [18] designate equation 5 as Clipped PPO with Entropy Penalty (cppo-ep) to optimize the paraphraser. The importance sampling ratio, $r_\phi(x_m, x_p)$, is defined as the ratio between the new policy and the old policy in this instance our policies being the paraphrasers. The authors also added an entropy regularization term (eq 7) that is "introduced to encourage the paraphraser to explore a more diverse generation policy" [18] in this case γ is a coefficient to control the weights of the advantage term and generation term for our experiment we set this equal to 0.2. Prior to completion of the paraphraser update step we save the current paraphraser ($P_{\phi\prime}$) before applying equation 5 to update to the new policy (P_ϕ). After normalizing each individual reward with the mean and std dev of the entire collection in the memory buffer we trained and updated the model on each individual collection in the memory buffer. For the clipping we set our epsilon to 0.002.

$$L_G(\phi) = \mathbb{E}_{(x_m,x_p)\sim P_\phi'} - \min\left(\text{clip}\left(r_\phi(x_m,x_p), 1-\epsilon, 1+\epsilon\right), r_\phi(x_m,x_p)\right) A(x_p,\phi) - \gamma S(x_p) \quad (5)$$

$$r_\phi(x_m, x_p) = \frac{P_\phi(x_p \mid x_m)}{P_{\phi\prime}(x_p \mid x_m)} \quad (6)$$

$$S(x_p) = \mathbb{E}_{(x_m,x_p)\sim P_{\phi\prime}} - \; P_\phi(x_p \mid x_m) \log P_\phi(x_p \mid x_m) \quad (7)$$

2.3.2. Training the detector

Our loss function for the detector is a triple weighted logistic loss defined in equation 8. Like in training the paraphraser and due to memory constraints our batch size is 1. For each observation we have 1 human and 2 non-human texts to combat the model overweighting the non-human texts they added a λ term to account for this, in our experiments we set this to 0.5.

$$L_D(\phi) = -\mathbb{E}_{x_h\sim\mathcal{H}} \log D_\phi(x_h) + \lambda\mathbb{E}_{x_m\sim\mathcal{M}} - \log\left(1 - D_\phi(x_m)\right) + \lambda\mathbb{E}_{x_p\sim\mathcal{P}} - \log\left(1 - D_\phi(G_\phi(x_m))\right) \quad (8)$$

3. Results

The following metrics were used in evaluation of approaches:

- ROC-AUC: The area under the ROC (Receiver Operating Characteristic) curve.
- Brier: The complement of the Brier score (mean squared loss).
- C@1: A modified accuracy score that assigns non-answers (score = 0.5) the average accuracy of the remaining cases.
- F1: The harmonic mean of precision and recall.
- F0.5u: A modified F0.5 measure (precision-weighted F measure) that treats non-answers (score = 0.5) as false negatives.
- The arithmetic mean of all the metrics above.
- A confusion matrix for calculating true/false positive/negative rates.

The adversarial method performed best on the validation set obtaining a score of at least 0.992 across each category.

References

[1] J. Bevendorff, D. Dementieva, M. Fröbe, B. Gipp, A. Greiner-Petter, J. Karlgren, M. Mayerl, P. Nakov, A. Panchenko, M. Potthast, A. Shelmanov, E. Stamatatos, B. Stein, Y. Wang, M. Wiegmann, E. Zangerle, Overview of PAN 2025: Voight-Kampff Generative AI Detection, Multilingual Text Detoxification, Multi-Author Writing Style Analysis, and Generative Plagiarism Detection, in: J. C. de Albornoz, J. Gonzalo, L. Plaza, A. G. S. de Herrera, J. Mothe, F. Piroi, P. Rosso, D. Spina, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Sixteenth International Conference of the CLEF Association (CLEF 2025), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2025.

Table 1
Comparison of performance metrics between Adversarial and Binoculars + XGB methods on the validation dataset.

Method	Dataset	ROC-AUC	Brier	C@1	F1	F05U	Mean
Adversarial	val	0.993	0.992	0.992	0.994	0.997	0.994
Binoculars+XGB	val	0.978	0.980	0.980	0.985	0.983	0.981

4. Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT, Grammarly in order to: Grammar and spelling check, Paraphrase and reword. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

[2] 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, in: G. Faggioli, N. Ferro, P. Rosso, D. Spina (Eds.), Working Notes of CLEF 2025 – Conference and Labs of the Evaluation Forum, CEUR Workshop Proceedings, CEUR-WS.org, 2025.

[3] 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: 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.

[4] S. Gehrmann, H. Strobelt, A. M. Rush, Gltr: Statistical detection and visualization of generated text, arXiv preprint arXiv:1906.04043 (2019).

[5] 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, 2024. URL: https://arxiv.org/abs/2401.12070. arXiv:2401.12070.

[6] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language models are unsupervised multitask learners, OpenAI Blog 1 (2019). https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

[7] E. Almazrouei, H. Alobeidli, A. Alshamsi, A. Cappelli, R. Cojocaru, M. Debbah, Goffinet, D. Hesslow, J. Launay, Q. Malartic, D. Mazzotta, B. Noune, B. Pannier, G. Penedo, The falcon series of open language models, arXiv preprint arXiv:2311.16867 (2023). Describes Falcon-7B, Falcon-40B, and Falcon-180B open-source LLMs under Apache2.0 license.

[8] EQUES / TinyDeepSeek, TinyDeepSeek-1.5B, https://huggingface.co/EQUES/TinyDeepSeek-1.5B, ???? Accessed July 2025; 1.54B parameters, Apache-2.0 license.

[9] 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, Association for Computational Linguistics, 2019, pp. 4171–4186.

[10] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Brew, 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.

[11] S. Ma, Q. Wang, Zero-shot detection of llm-generated text using token cohesiveness, 2024. URL: https://arxiv.org/abs/2409.16914. arXiv:2409.16914.

[12] W. Yuan, G. Neubig, P. Liu, Bartscore: Evaluating generated text as text generation, 2021. URL: https://arxiv.org/abs/2106.11520. arXiv:2106.11520.

[13] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, C. Finn, Detectgpt: Zero-shot machine-generated text detection using probability curvature, in: International Conference on Machine Learning,

PMLR, 2023, pp. 24950–24962.

[14] J. Su, T. Y. Zhuo, D. Wang, P. Nakov, Detectllm: Leveraging log rank information for zero-shot detection of machine-generated text, 2023. URL: https://arxiv.org/abs/2306.05540. arXiv:2306.05540.

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

[16] A. Aich, S. Bhattacharya, N. Parde, Demystifying neural fake news via linguistic feature-based interpretation, in: N. Calzolari, C.-R. Huang, H. Kim, J. Pustejovsky, L. Wanner, K.-S. Choi, P.-M. Ryu, H.-H. Chen, L. Donatelli, H. Ji, S. Kurohashi, P. Paggio, N. Xue, S. Kim, Y. Hahm, Z. He, T. K. Lee, E. Santus, F. Bond, S.-H. Na (Eds.), Proceedings of the 29th International Conference on Computational Linguistics, International Committee on Computational Linguistics, Gyeongju, Republic of Korea, 2022, pp. 6586–6599. URL: https://aclanthology.org/2022.coling-1.573/.

[17] E. Tulchinskii, K. Kuznetsov, L. Kushnareva, D. Cherniavskii, S. Barannikov, I. Piontkovskaya, S. Nikolenko, E. Burnaev, Intrinsic dimension estimation for robust detection of ai-generated texts, 2023. URL: https://arxiv.org/abs/2306.04723. arXiv:2306.04723.

[18] X. Hu, P.-Y. Chen, T.-Y. Ho, Radar: Robust ai-text detection via adversarial learning, Advances in neural information processing systems 36 (2023) 15077–15095.