LOCAL ENTROPY FLUCTUATIONS FOR BLACK-BOX LLM-GENERATED TEXT DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Zero-shot detection of machine-generated text is vital for mitigating the risks of large language models (LLMs) misuse. While state-of-the-art dual-model and resampling-based detectors outperform single-model statistical detectors on unprompted continuations, they suffer a catastrophic breakdown on instruction-conditioned text, especially in the black-box setting where the source model is unknown. We analyze this failure and trace it to poor prompt conditioning for resampling-based methods and diminished cross-model discriminability for dual-model detectors. Motivated by recent study that human writing maintains a steady, uniform information flow, whereas machine-generated text exhibits sharp, localized entropy fluctuations, we propose a new metric, the Relative Peak Prominence Rate (RelPPR) to capture this structural contrast along the entropy trajectory and further leverage it as a signal to for LLM-generated text detection. Extensive evaluations across diverse model families and datasets demonstrate that RelPPR achieves state-of-the-art detection performance in the black-box setting.

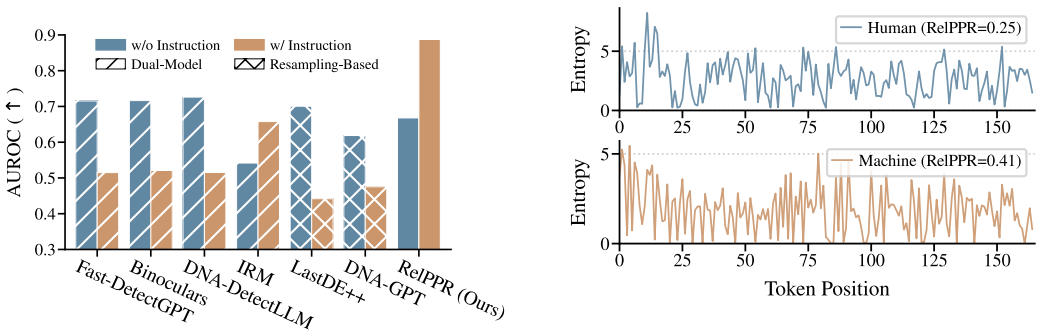
1 INTRODUCTION

The rapid development of large language models (LLMs) capable of producing fluent, human-like text at scale has raised growing concerns about misuse, making reliable detection of machine-generated text an urgent problem. We refer to the model that generate candidate text as the *source model*. Detection under the *black-box setting* assumes access only to the candidate text itself, with no knowledge of the source model's identity or parameters; this setting requires neither labeled training data nor access to the source model, making it a highly practical yet challenging paradigm. Detectors instead can only use one or more surrogate models to determine whether the candidate text is LLM-generated; we refer to such surrogate model as an *evaluator model*.

Existing zero-shot detectors broadly fall into three families: (i) *single-model statistical* criteria, which score a candidate text using only a single evaluator model's statistics, such as log-likelihood, log-rank, and entropy (Solaiman et al., 2019; Ippolito et al., 2020; Gehrmann et al., 2019); (ii) *resampling-based* detectors, which regenerate the text under an evaluator model to compare the candidate against this regeneration or normalize specific statistics (Mitchell et al., 2023; Yang et al., 2024; Xu et al., 2025); and (iii) *dual-model detectors*, which contrast the predictions of two evaluator models on the candidate text (Bao et al., 2024; Hans et al., 2024; Zhu et al., 2025). The latter two families currently define the state of the art, substantially outperforming single-model statistics.

We find, however, that this advantage holds mainly for *unprompted* continuations of a human prefix, i.e., text generated without user prompt or instruction. Once text is generated under the instruction-conditioned regime, dual-model and resampling-based detectors alike suffer a serious performance degradation as shown in Figure 1a. As our first contribution, we analyze this failure mode systematically for both detector families. 1) For resampling-based methods, when the user prompt is invisible at evaluation time, resampling-based detectors cannot generate text by conditioning on the original prompt as the source model did. This *prompt conditioning gap* drives the resampled text to drift away from the candidate text. 2) Dual-model detectors rely on the discrepancy between two evaluator models' probability estimates for the same candidate text, typically measured via statistics such as log-likelihood or entropy. However, we find in our experiment that, under instruction-conditioning, both evaluator models reach a stronger consensus with each other on machine-generated text (lower KL divergence), which shrinks the discrepancy signal the dual-model detectors rely on. In sum-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) AUROC under unprompted and instruction-conditioned generation.

(b) The entropy trajectory of human and machine text in response to the same question from ELI5 dataset.

Figure 1: (a) RelPPR is robust under instruction-conditioning where existing detectors degrade; (b) Machine-generated text exhibits stronger entropy fluctuations than human text.

mary, poor conditioning and diminished cross-model discriminability undermine predictive power in resampling-based and dual-model detectors, respectively.

Since all existing state-of-the-art (SOTA) detectors fail under instruction conditioning, our goal is to find a new detection signal that is effective under instruction-conditioning. Previous studies in psycholinguistics suggest that human writers tend to spread information evenly and avoid cramming too much information into one place (Jaeger & Levy, 2006; Jaeger, 2010; Meister et al., 2021). In contrast, LLMs exhibit high entropy at semantic branch points and key decision junctures, a pattern further accentuated by instruction-tuning (Wang et al., 2025; Ni et al., 2026; Zhu et al., 2026). Therefore, we hypothesize that machine text exhibits higher-amplitude local fluctuations in its entropy trajectory than human text .

Motivated by this observation, we propose the Peak Prominence Rate (PPR), defined as the average prominence of local peaks identified along the entropy trajectory, to quantify trajectory fluctuations for detection. However, a text with higher sequence-level mean entropy can attain a larger raw PPR even if its fluctuations are mild relative to its entropy level. To counteract this bias, we normalize PPR by the sequence-level mean entropy, yielding our final criterion, **RelPPR**. Notably, RelPPR is entirely parameter-free and requires only a single evaluator model during inference.

We evaluate RelPPR across seven datasets and five source models under instruction-conditioning. Beyond achieving state-of-the-art detection performance under the black-box setting, RelPPR demonstrates strong practical deployment value: it achieves superior recall under strict false-positive constraints, sustains reliability across varying sequence lengths and decoding configurations, withstands adversarial paraphrasing, and generalizes across diverse evaluator model scales and families.

Our contributions are threefold: (1) We show that dual-model and resampling-based detectors degrade catastrophically on instruction-conditioned text, and we trace this degradation to concrete mechanisms: diminished cross-model discriminability for dual-model detectors and the prompt conditioning gap for resampling-based detectors. (2) We propose RelPPR, a single-model, parameter-free detection criterion motivated by the observation that machine-generated text exhibits higher-amplitude fluctuations in entropy trajectory than human text. (3) We perform thorough emprical evaluations and extensive analysis on robustness across seven datasets and five source models under black-box setting. In these evaluations we show that RelPPR achieves the SOTA performance, and remains the most reliable detector across decision thresholds, sequence lengths, decoding configurations, evaluator model scales and families, and under paraphrasing attacks.

2 Related Work

Zero-Shot Detectors Zero-Shot Detectors Unlike supervised classifiers that struggle with out-of-distribution generalization (Pu et al., 2023; Doughman et al., 2025), zero-shot detectors require no model cooperation or labeled data, and fall into three paradigms. **Single-model statistical criteria** score text by sequence-level statistics under one surrogate model, from simple metrics such as Log-Likelihood (Solaiman et al., 2019), LogRank (Gehrmann et al., 2019), and Entropy (Ippolito

et al., 2020), to refinements using metric ratios (LRR (Su et al., 2023)), log-likelihood transition dynamics (LastDE (Xu et al., 2025)), or grammatical edit features (GECScore (Wu et al., 2025)). **Resampling-based detectors** instead evaluate fidelity by regenerating content from the observed context and comparing it against the candidate, e.g. via n-gram overlap (DNA-GPT (Yang et al., 2024)) or regenerated-sequence contrast (LastDE++ (Xu et al., 2025)). **Dual-model detectors** construct a dynamic baseline from a second model rather than a fixed statistic: DetectGPT (Mitchell et al., 2023) and Fast-DetectGPT (Bao et al., 2024) obtain this baseline via perturbation and token substitution respectively, Binoculars (Hans et al., 2024) and DNA-DetectLLM (Zhu et al., 2025) normalize surprisal against it, and IRM (Liu et al., 2025) instead baselines against a base (unaligned) model.

While dual-model and resampling-based detectors degrade under instruction conditioning, our proposed RelPPR performs exceptionally well in this challenging regime by capturing the fine-grained local fluctuation structure of the entropy trajectory.

Entropy Dynamics and Uniform Information Density. *Uniform Information Density* hypothesis from psycholinguistics (Jaeger, 2010; Jaeger & Levy, 2006; Meister et al., 2021) posits that humans intuitively spread out the information in communication, avoiding sudden bursts of hard-to-predict content so that listeners can follow along smoothly. In contrast, LLM generations concentrate uncertainty on a few pivotal "forking" tokens. Bigelow et al. (2025) locate such tokens by resampling from each prefix and show that a single token can redirect the whole generation; Wang et al. (2025) find that these high-entropy tokens are a small minority in chain-of-thought and that restricting RLVR updates to them suffices; Ni et al. (2026) show that diffusion LMs bypass them under arbitrary-order decoding, which hampers reasoning. These works, along with EDIS (Zhu et al., 2026), study entropy dynamics to understand or improve generation. We instead treat the resulting entropy spikes as a trace left in the output text: contrasted with the smoother profile of human writing, they provide a signal for black-box detection, which, to our knowledge, has not been explored.

3 What Breaks Current Detectors under Instruction Conditioning

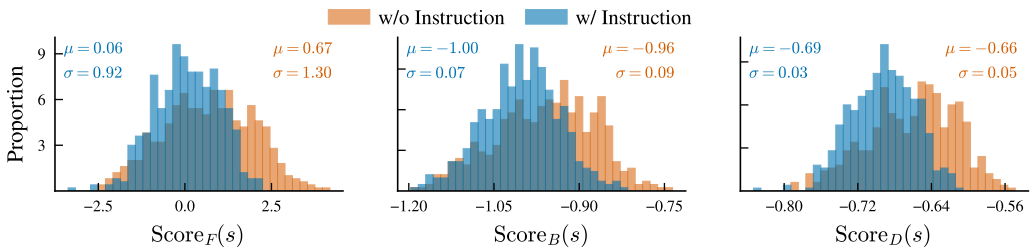
Empowered by post-training alignment, instruction-tuned LLMs produce highly fluent and task-conforming text under prompts with instructions, which makes detection more challenging than in unprompted continuation. We thus evaluate whether current zero-shot detectors remain effective in this demanding regime. Specifically, we contrast two generation settings on the *same* human texts: (i) *w/o instruction*, where the model performs unprompted continuation from a bare prefix, and (ii) *w/ instruction*, where the prefix is wrapped into task-specific prompt templates as shown in Table 4. We set the source model as OLMo-3.1-32B-Instruct and evaluate detectors under both settings. As shown in Figure 1a, current detectors degrade severely under instruction conditioning. Below, we uncover the underlying mechanisms driving this breakdown across both dual-model and resampling-based methods.

3.1 Preliminaries and Notations

Let $\mathcal{V}$ denote the token vocabulary. Given a candidate text sequence $s = (x_1, \ldots, x_L) \in \mathcal{V}^L$ of length L, we write the prefix preceding position i as $x_{<i} = (x_1, \ldots, x_{i-1})$. Given model p, we denote $p(\cdot \mid x_{<i})$ as the conditional distribution over the next token $x_i \in \mathcal{V}$ given the prefix $x_{<i}$. We denote the sequence-level surprisal $I_p(s)$, entropy $H_p(s)$, and cross-entropy $H(s; p_1, p_2)$ for p_1 and p_2 as follows, respectively:

$$I_p(s) = -\frac{1}{L}\sum_{i=1}^{L} \log p(x_i \mid x_{<i}), \quad H_p(s) = -\frac{1}{L}\sum_{i=1}^{L}\sum_{v\in\mathcal{V}} p(v \mid x_{<i}) \log p(v \mid x_{<i}),$$

$$H(s; p_1, p_2) = -\frac{1}{L}\sum_{i=1}^{L}\sum_{v\in\mathcal{V}} p_1(v \mid x_{<i}) \log p_2(v \mid x_{<i}).$$



(a) Fast-DetectGPT (b) Binoculars (c) DNA-DetectLLM

Figure 2: Figure 2: Machine text score distributions on ELI5 (OLMo-3.1-32B-Instruct). Scores shift lower under instruction conditioning. μ and σ denote mean and standard deviation.

3.2 Dual-Model Detectors

Fast-DetectGPT Fast-DetectGPT (Bao et al., 2024) builds on the probability curvature hypothesis of DetectGPT (Mitchell et al., 2023): machine-generated text tends to sit near a local maximum of the model's log-likelihood (negative curvature around it), while human text shows no such tendency. Instead of DetectGPT's costly mask-and-resample estimation, Fast-DetectGPT localizes this to the token level: at each position, the observed token x_i is contrasted with alternatives sampled from $p_1(\cdot \mid x_{<i})$ and scored by p_2, yielding a standardized score $\text{Score}_\text{F}(s) = N(s)/\sqrt{\sum_{i=1}^{L} \tilde{\sigma}_i^2}$ (full definition in Section A.1). Since Table 3 shows dropping the variance normalization has negligible impact on detection performance, we focus on the numerator, which admits a closed-form decomposition:

$$N(s) = L \cdot \big(H(s; p_1, p_2) - I_{p_2}(s)\big), \tag{1}$$

where $H(s; p_1, p_2)$ is the expected cross-entropy of tokens resampled from $p_1(\cdot \mid x_{<i})$ under p_2, serving as a dynamic baseline for the local neighborhood of s. $N(s)$ thus measures how far s stands above this baseline: large and positive for machine text near a local maximum ($I_{p_2}(s) \ll H$), near zero for human text ($I_{p_2}(s) \approx H$).

Binoculars and DNA-DetectLLM Rather than measuring a difference, Binoculars (Hans et al., 2024) and DNA-DetectLLM (Zhu et al., 2025) employ a ratio-based formulation of the same two statistics (full definitions in Section A.2):

$$\text{Score}_\text{B}(s) = -\frac{I_{p_2}(s)}{H(s; p_1, p_2)}, \qquad \text{Score}_\text{D}(s) = -\frac{I_{p_2}(s) + I_{p_2}(\tilde{s} \mid s)}{2 \cdot H(s; p_1, p_2)}, \tag{2}$$

where $\tilde{s}_t = \arg\max_{v \in \mathcal{V}} p_2(v \mid s_{<t})$ is the greedy repair of s.

The denominator $H(s; p_1, p_2)$ plays the same role as in Fast-DetectGPT, serving as the dynamic baseline obtained from the analytical resampling process. Combining with Equation (1), we have

$$\text{Score}_\text{B}(s) = \frac{N(s)}{L \cdot H(s; p_1, p_2)} - 1.$$

Therefore, Binoculars is essentially the Fast-DetectGPT numerator $N(s)$ normalized by the cross-entropy baseline instead of the token-wise variance. DNA-DetectLLM further adds the surprisal of the greedy repair $I_{p_2}(\tilde{s} \mid s)$ to the numerator, which measures the minimum attainable surprisal along the prefixes of s. Since machine-generated text largely follows the greedy path of p_2, this term stays close to $I_{p_2}(s)$ for machine text but falls well below it for human text, and the core comparison against $H(s; p_1, p_2)$ remains unchanged.

Unified Information-Theoretic Perspective

All three methods operate on the premise that machine-generated text yields higher scores than human text. However, as shown in Figure 2, their score distributions drift lower under instruction-conditioning, directly driving the performance degradation. We further analyze this shift from an information-theoretic perspective.

Comparing Equations (1) and (2), we find that these three methods are governed by the *exact same two information-theoretic quantities*: the sequence-level surprisal $I_{p_2}(s)$ and the sequence-level

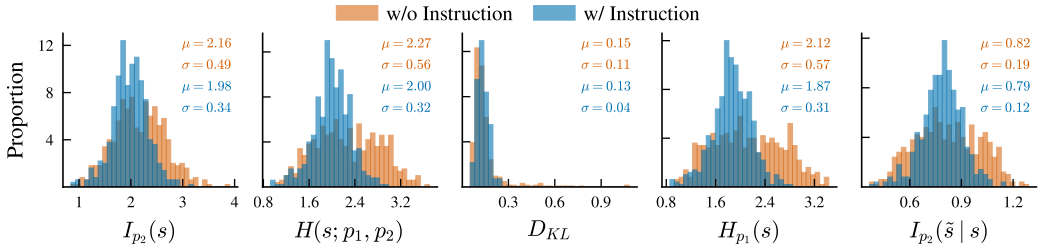


Figure 3: Distributions of the five information-theoretic components from Section 3.2 on ELI5, for machine text generated by OLMo-3.1-32B-Instruct. Under instruction conditioning, $H(s; p_1, p_2)$ shifts to lower values, compressing the gap that dual-model detectors rely on. μ and σ denote the mean and standard deviation.

cross-entropy $H(s; p_1, p_2)$ that can be further decomposed as:

$$H(s; p_1, p_2) = H_{p_1}(s) + D_{\mathrm{KL}}(s; p_1, p_2), \tag{3}$$

where $D_{\mathrm{KL}}(s; p_1, p_2)$ denotes the sequence-level KL divergence (derivation in Section A.3).

Taking the result on ELI5 (Figure 3) as a representative case study, we find that the distribution of $H(s; p_1, p_2)$ shifts toward lower values, while the distributions of $I_{p_2}(s)$ and $I_{p_2}(\tilde{s})$ remain largely unchanged. This asymmetric shift directly compresses the scores in Equations (1) and (2), causing performance degradation. Examining $H(s; p_1, p_2)$ at a finer level reveals that the distributions of $D_{\mathrm{KL}}(s; p_1, p_2)$ shift toward lower values; in particular, the standard variance of D_{KL} collapses dramatically ($0.11 \rightarrow 0.04$), demonstrating that the two models reach an exceptionally uniform consensus on text generated with human prompt. This pattern is also observed on other datasets, including SQuAD, ELI5, XSum, and WikiText (Figures 3, 8, 9 and 11).

In summary, the degradation of dual-model detectors stems from diminished cross-model discriminability: explicit instructions induce strong inter-model consensus, shrinking the cross-entropy discrepancy signal necessary for detection.

3.3 Resampling-Based Detectors

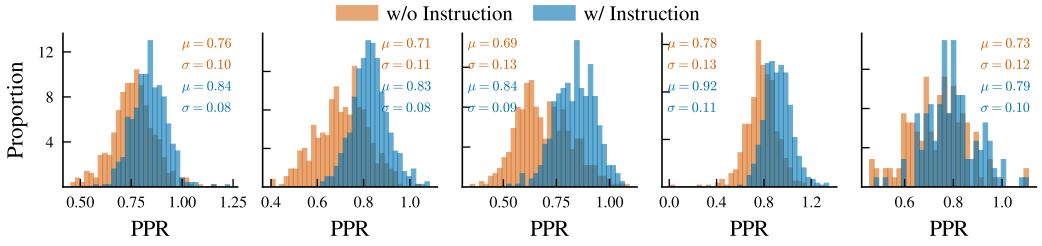
Resampling-based detectors judge whether s is machine-generated by checking whether the evaluator model p, continuing the same prefix, tends to reproduce it: given each prefix $x_{<i}$, they resample an alternative continuation $\tilde{x}_i \sim p(\cdot \mid x_{<i})$ and either compare it against s directly (e.g., DNA-GPT's n-gram overlap) or use it to calibrate a statistic on s (e.g., LastDE++). This premise holds *w/o instruction*, since the source model itself produced s as an unconditioned continuation, matching how the evaluator resamples. It breaks *w/ instruction*: the source model conditioned on instruction c, but the evaluator cannot access c at detection time and can only resample via $p(\cdot \mid x_{<i})$ instead of $p(\cdot \mid c, x_{<i})$. This *prompt conditioning gap* drifts the resampled continuation away from s, causing resampling-based detectors to lose their signal under instruction conditioning.

4 Our Method: Relative Peak Prominence Rate

To address the challenges posed by instruction conditioning, we draw on the difference in how humans and machines produce text and propose to detect machine-generated text by the difference of fluctuation in its entropy trajectory. We capture this fluctuation signal using *topological prominence*, which measures how sharply each local entropy peak stands out above its neighboring valleys.

4.1 Motivation: Human Text Flows, Machine Text Spikes

Human Writing: Smooth Information Flow As formalized in the *Uniform Information Density* (UID) principle (Jaeger, 2010; Meister et al., 2021), humans naturally regulate their writing to keep the rate of information transmission relatively steady (e.g., by cushioning complex concepts with preparatory context to facilitate reader comprehension). This communicative buffering yields a smooth information flow that inherently avoids sharp, localized entropy spikes. Extending this prin-



(a) XSum (b) CNN/DailyMail (c) WikiText (d) WritingPrompts (e) SQuAD

Figure 4: Distributions of PPR on machine-generated text from OLMo-3.1-32B-Instruct across five datasets. μ and σ denote the mean and standard deviation of the distribution. PPR shifts noticeably higher under instruction conditioning, indicating that instruction conditioning induces greater fluctuation in the entropy trajectory of generated text.

ciple to generation dynamics, we posit that the resulting entropy trajectory of human text exhibits smoother transitions with restrained local fluctuations.

paragraphMachine Text: High-Amplitude Local Fluctuations Autoregressive generation in LLMs, in contrast, is not globally planned for smooth communication. Recent studies on LLM reasoning (Wang et al., 2025; Ni et al., 2026; Bigelow et al., 2025) show that model uncertainty surges into sharp peaks at semantic branch points, referred to as *forking tokens*. This unevenness arises because LLMs balance strong lexical biases with on-the-fly adaptation of their semantic priors (McCoy et al., 2023; Shi et al., 2024). We posit that this alternation between forking tokens and predictable stretches yields an entropy trajectory with higher-amplitude local fluctuations than human text.

Instruction-Conditioned Text: Even Stronger Oscillations The forking-token perspective further suggests why this fluctuation should be even more pronounced when generation is conditioned on an instruction. Wang et al. (2025) show that reinforcement learning with verifiable rewards (RLVR) does not reshape a model's token-level entropy uniformly: after RLVR, the entropy of most low-entropy tokens stays close to its pre-RLVR level, while entropy at a small minority of high-entropy, decision-relevant forking tokens increases noticeably. Since instruction-tuned models are themselves shaped by such post-training, we conjecture that invoking their instruction-following behavior accentuates peaks at forking tokens.

Taken together, we conjecture that instruction conditioning accentuates the local entropy oscillations of machine text, creating a strong discriminative signal. Motivated by this insight, we formulate our detection criterion by directly quantifying these trajectory fluctuations.

4.2 Peak Prominence Rate of Entropy Trajectory

To capture the degree of local fluctuation as a detection signal, we adopt the concept of *topological prominence* from computational geometry and signal analysis (Perea & Harer, 2015). Given a text sequence s and a evaluator model M, we compute the token-level entropy trajectory $\mathbf{H}_M(s) = (h_1, \ldots, h_L) \in \mathbb{R}^L$, where $h_i = H\big(p(\cdot \mid x_{<i})\big)$ denotes the entropy at step i. A position $t \in \{2, \ldots, L-1\}$ is a peak if $h_t > h_{t-1}$ and $h_t > h_{t+1}$; we denote the set of all such peaks by $\mathcal{T}(s)$.

For each peak $t \in \mathcal{T}(s)$, its prominence $\mathrm{Prom}(t)$ is defined as the vertical distance between the peak and the highest contour line that connects it to an even higher peak:

$$\mathrm{Prom}(t) = h_t - \max\Big(\min_{i \in [l_t, t]} h_i,\ \min_{j \in [t, r_t]} h_j\Big),$$

where l_t and r_t are the nearest preceding and succeeding positions along the trajectory whose entropy exceeds h_t (defaulting to the sequence boundaries if no such position exists). Thus, $\mathrm{Prom}(t)$ measures the elevation of peak t above its local valley.

We define the sequence-level *Peak Prominence Rate* (PPR) as the cumulative prominence of all peaks normalized by the sequence length:

$$\mathrm{PPR}(s) = \frac{1}{L} \sum_{t \in \mathcal{T}(s)} \mathrm{Prom}(t).$$

In Figures 4 and 7 we show the distributions of PPR on machine-generated text with and without instruction conditioning across diverse datasets, where the human-written prefix is strictly held constant across both conditions. Across most datasets, PPR systematically shifts toward higher values when the text is generated with instructions, which empirically substantiates our hypothesis that instruction conditioning induces larger entropy trajectory fluctuations than unprompted continuation. We also visualize the entropy trajectories of a human sample and a machine text generated with instruction in Figure 1b, both are in response to the same question from the ELI5 dataset. The human text maintains a relatively steady trajectory with restrained fluctuations, while the machine text exhibits more frequent, high-amplitude fluctuations.

As human text generally exhibits higher entropy than machine-generated text under the evaluator model (Gehrmann et al., 2019), a high-entropy human text can attain a large raw PPR, even when its fluctuations are mild relative to its own entropy level. Therefore, we introduce two entropy-adjusted variants: (i) the Relative Peak Prominence Rate (RelPPR) score, which normalizes PPR by the sequence-level mean entropy $H_p(s)$ to isolate fluctuation relative to the sample's own entropy level; (ii) the Difference Peak Prominence Rate (DiffPPR) score, which treats PPR as an incremental signal to $H_p(s)$. The two criteria are defined as:

$$\mathrm{Score}_{\mathrm{RelPPR}}(s) = \frac{\mathrm{PPR}(s)}{H_p(s)}, \quad \mathrm{Score}_{\mathrm{DiffPPR}}(s) = \mathrm{PPR}(s) - H_p(s)$$

A higher RelPPR (or DiffPPR) score indicates a higher likelihood that the candidate text is machine-generated. Compared with dual-model and resampling-based detectors, our criteria require only a single forward pass and no sampling, making them computationally efficient, and, as shown in the next section, they also achieve superior detection performance.

5 Experiment

In this section, we evaluate RelPPR against twelve zero-shot baselines across seven datasets and six instruction-tuned source models, and further examine its robustness to decision thresholds, decoding configurations, generation length, paraphrasing attacks, and the size and family of evaluator model.

5.1 Experimental Setup

Benchmark Datasets We evaluate on seven benchmark datasets spanning diverse domains: news articles (XSum (Narayan et al., 2018) and CNN/DailyMail (Hermann et al., 2015)), encyclopedic contexts (WikiText-103 (Merity et al., 2016) and SQuAD (Rajpurkar et al., 2016)), creative story writing (WritingPrompts (Fan et al., 2018)), biomedical research question answering (PubMedQA (Jin et al., 2019)) and general-topic question answering (ELI5 (Fan et al., 2019)).

Human Prefix and Prompt Templates For datasets without a standalone question or opening sentence (XSum, CNN/DailyMail, WikiText, and SQuAD), following the Fast-DetectGPT protocol, the human prefix is formed by extracting the first 30 tokens from the source document, while the remaining tokens serve as the human completion. For WritingPrompts, ELI5, and PubMedQA, the human prefix is the clean opening sentence or question itself, and the corresponding human continuation is the original human-written story or answer. Each dataset is equipped with a tailored instruction prompt template (detailed in Table 4). Crucially, adhering to realistic zero-shot detection scenarios, the instruction prompt is strictly **invisible** to detectors during evaluation.

Baselines and Evaluation Metric We evaluate our proposed method against 14 competitive zero-shot baselines: (1) *Single-model statistical criteria:* Likelihood (Solaiman et al., 2019), Entropy (Ippolito et al., 2020), LogRank (Gehrmann et al., 2019), LastDE (Xu et al., 2025), LRR (Su et al., 2023) and GECScore (Wu et al., 2025); (2) *dual-model detectors:* Fast-DetectGPT (Bao et al., 2024), Binoculars (Hans et al., 2024), DNA-DetectLLM (Zhu et al., 2025), and Implicit Reward Model (IRM) (Liu et al., 2025); (3) *Resampling-based detectors:* DNA-GPT (Yang et al., 2024) and LastDE++ (Xu et al., 2025). Detection performance is measured by the Area Under the Receiver Operating Characteristic Curve (AUROC) across all human-machine pairs. **Models** To ensure robust evaluation across diverse model capacities and structural paradigms, we evaluate on six instruction-tuned LLMs spanning both dense and sparse Mixture-of-Experts (MoE) architectures, with parameter scales ranging from 7B to 119B. We deliberately use the small Qwen3.5-2B (Qwen

Table 1: Detection performance (AUROC) by dataset (averaged across five source models) under black-box setting. The best and second-best results are shown in **bold** and underlined, respectively.

Methods	XSum	WritingP	SQuAD	WikiText	CNN/DM	ELI5	PubMedQA	Avg
Entropy	0.822	0.812	0.813	0.788	0.817	0.929	**0.703**	0.812
Likelihood	0.787	0.801	0.787	0.752	0.775	0.902	0.600	0.772
LogRank	0.786	0.764	0.782	0.767	0.782	0.894	0.595	0.767
LRR	0.739	0.613	0.731	0.781	0.756	0.831	0.550	0.714
GECScore	0.712	0.673	0.684	0.648	0.688	0.839	0.558	0.686
LastDE	0.755	0.767	0.783	0.785	0.767	0.890	0.590	0.762
LastDE++	0.406	0.545	0.432	0.317	0.433	0.673	0.444	0.464
DNA-GPT	0.401	0.509	0.372	0.441	0.335	0.568	0.486	0.445
Fast-DetectGPT	0.649	0.718	0.580	0.598	0.592	0.531	0.449	0.588
Binoculars	0.663	0.712	0.598	0.612	0.602	0.551	0.452	0.599
DNA-DetectLLM	0.704	0.731	0.590	0.646	0.614	0.420	0.451	0.594
IRM	0.867	0.807	0.707	0.887	0.818	0.662	0.581	0.761
PPR (Ours)	0.715	0.787	0.555	0.610	0.638	0.461	0.412	0.597
DiffPPR (Ours)	0.884	0.909	0.865	0.850	0.869	0.958	0.670	0.858
RelPPR (Ours)	**0.921**	**0.961**	**0.888**	**0.913**	**0.879**	**0.974**	0.563	**0.871**

Team, 2026) as the evaluator to simulate realistic black-box deployment scenarios where the target commercial model substantially exceeds the detector in capacity. Full model identities and evaluator configurations are detailed in Section B.1.

Further details regarding experimental setups are provided in Appendix B.

5.2 Results

We evaluate detectors under the black-box setting, where the detector has no access to the source model, and summarize their detection performance across datasets in Table 1 and across source models in Table 2. The results demonstrate that our proposed RelPPR achieves state-of-the-art performances across diverse source model architectures and domains.

Single-Model Statistical Criteria Outperform Dual-Model and Resampling Baselines Entropy attains the highest overall AUROC of 0.812 among all baselines, while Likelihood, LogRank and LastDE maintain robust stability across diverse source models and datasets. In contrast, dual-model and resampling-based detectors exhibit substantially inferior performance and severe instability across source models. The dramatic degradation from LastDE to LastDE++ isolates the adverse effect of resampling and corroborates our *prompt conditioning gap* analysis in Section 3.3.

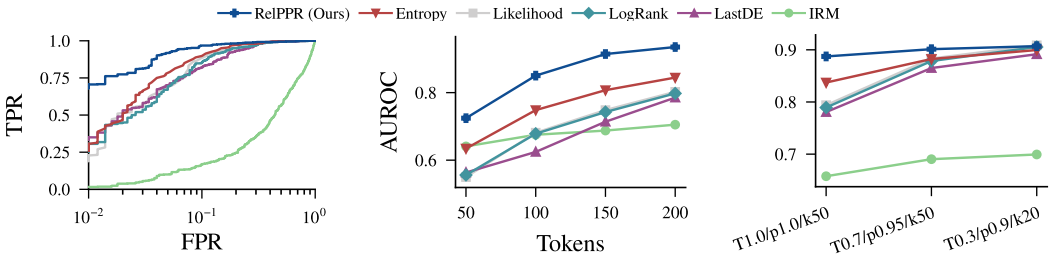
Table 2: AUROC across source models, averaged over seven datasets under the black-box setting. The best and second-best results are shown in **bold** and underlined, respectively.

	OLMo3 7B	OLMo3.1 32B	Gemma4 31B	Mistral 119B	Qwen3.5 35B	Avg
Entropy	0.780	0.837	0.811	0.841	0.790	0.812
Likelihood	0.693	0.794	0.803	0.848	0.723	0.772
LogRank	0.699	0.789	0.786	0.838	0.723	0.767
LRR	0.687	0.740	0.692	0.763	0.689	0.714
GECScore	0.646	0.679	0.743	0.631	0.731	0.686
LastDE	0.704	0.781	0.764	0.817	0.745	0.762
LastDE++	0.406	0.442	0.476	0.558	0.440	0.464
DNA-GPT	0.462	0.475	0.430	0.454	0.401	0.445
Fast-DetectGPT	0.420	0.515	0.693	0.712	0.601	0.588
Binoculars	0.436	0.521	0.699	0.718	0.618	0.599
DNA-DetectLLM	0.436	0.515	0.698	0.702	0.617	0.594
IRM	0.668	0.658	0.887	0.698	**0.895**	0.761
PPR (Ours)	0.586	0.575	0.638	0.568	0.616	0.597
DiffPPR (Ours)	0.824	0.877	0.874	0.875	0.839	0.858
RelPPR (Ours)	**0.837**	**0.887**	**0.897**	**0.879**	0.856	**0.871**

PPR and Entropy are Complementary Signals Combining PPR and entropy achieves the best performance. DiffPPR already surpasses every baseline, including Entropy itself, confirming that fluctuation magnitude and sequence-level entropy are complementary signals. Normalizing the PPR by the sequence-level entropy, as in RelPPR, yields a further improvement over DiffPPR on every source model and every dataset except PubMedQA. The outstanding performance of RelPPR substantiates our core hypothesis: machine-generated text exhibits a higher degree of relative fluctuation compared to human writing.

5.3 Robustness Analysis

Beyond overall detection accuracy, we examine whether RelPPR remains reliable under conditions that matter for real-world deployment: skewed decision thresholds, varying text length and decoding configurations, adversarial paraphrasing and evaluator choices.



(a) ROC curve (ELI5) (b) Sequence Length (c) Decoding Strategy

Figure 5: Detection Performance across decision thresholds, sequence length and decoding configurations on OLMo-3.1-32B-Instruct. Across all evaluations, RelPPR consistently outperforms baselines and exhibits notable robustness.

Robustness across Decision Thresholds, Length and Decoding Configurations Beyond overall AUROC, practical deployment also depends on how much recall (true positive rate, TPR) a detector retains at low false positive rate (FPR). Figure 5a plots the ROC curve (log-scale FPR) on dataset ELI5 for RelPPR and the top 5 baselines. RelPPR dominates the low-FPR regime, retaining substantially higher TPR than all baselines as FPR approaches 0.01, underscoring its practical deployment value. We further vary generation length (Figure 5b) and decoding determinism (Figure 5c). Shorter texts and more stochastic decoding are uniformly harder for all evaluated detectors, as both narrow the gap between machine and human text statistics. RelPPR leads every baseline at every length tier and every decoding setting, making it the most robust choice under uncertain deployment conditions. The detailed setup is provided in Section B.4

Robustness under Paraphrasing Attack A practical detector must withstand adversarial rewriting. We paraphrase machine text with a T5-based model (Alisetti, 2021), keeping human text untouched. As shown in Figure 6, RelPPR attains the highest post-attack AUROC and lowest drop among declining criteria. This indicates that entropy trajectory fluctuations survive rewriting far better than other single-model statistics. Although IRM's AUROC rises under paraphrasing, it still trails RelPPR by a wide margin.

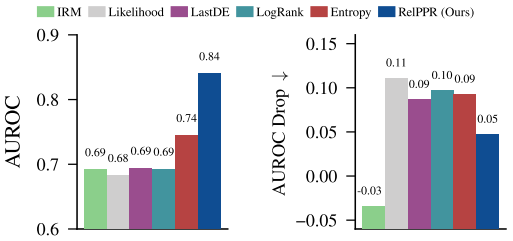


Figure 6: Post-attack AUROC (Left) and AUROC drop (Right) under paraphrasing attack, on OLMo-3.1-32B-Instruct.

Robustness Across Evaluator Size and Family, and White-Box Detection RelPPR also generalizes well across evaluator architectures and scales, ranking first on seven of nine evaluators (Table 6); cross-family shifts affect performance more than scaling within a family. Under the white-box setting, dual-model detectors instead achieve the best performance, as access to the source model restores cross-model discriminability, while resampling-based methods remain limited by the prompt conditioning gap (Tables 7 and 8; details in Section B.6).

6 Conclusion

We study zero-shot detection of LLM-generated text in the black-box, instruction-conditioned setting, where state-of-the-art detectors break down. Our analysis traces this failure to two mechanisms: the prompt conditioning gap, which drives resampled text away from the candidate for resampling-based detectors, and a stronger cross-model consensus that shrinks the discriminability dual-model detectors rely on. Building on the contrast between the uniform information flow of human writing and the sharp local entropy peaks of machine text, we propose RelPPR, a parameter-free criterion that quantifies the peak prominence rate of entropy trajectory relative to sequence-level entropy and uses this relative fluctuation magnitude as the detection signal, requiring only a single evaluator model and one forward pass. We evaluate RelPPR across seven datasets and five source models, where it achieves state-of-the-art black-box performance and remains robust across decision thresholds, text lengths, decoding configurations, paraphrasing attacks, and evaluator choices.

AI USE STATEMENT

In this work, we used generative AI tools to polish writing. We have not used generative AI tools to generate synthetic datasets, help develop theoretical models or conceptual frameworks, propose or refine hypotheses, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, design or provide feedback on research methodology or experiments, implement the proposed RelPPR method, assist with translation, clean and reformat the dataset, or support qualitative or thematic data analysis. The machine-generated texts in our experiments are produced by the source models listed in Section B.1 as the object of study, **not** as AI assistance to the research. We have reviewed all AI-assisted work. The authors carefully revised all AI-polished or AI-drafted text to ensure it accurately reflects our methods and findings; the authors checked all figures and processed data against the raw experimental outputs; and the authors reviewed and tested all AI-assisted code before use. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies zero-shot detection of LLM-generated text; we discuss its main ethical considerations below.

Data and human subjects All human-written text is drawn from publicly available benchmark datasets (XSum, CNN/DailyMail, WikiText-103, SQuAD, WritingPrompts, ELI5, and PubMedQA) that are widely used in prior detection work, and we use them for research purposes in accordance with their respective licenses. WritingPrompts and ELI5 consist of user-generated content collected from Reddit; we use the text only as released, collect no additional personal information, and make no attempt to identify its authors. Machine-generated text was produced by publicly available open-weight models. The study involves no human participants or annotators.

Risk of false positives Errors of a text detector fall on people: a human author whose writing is flagged as machine-generated may face unwarranted accusations. Our results indicate that this risk is not uniform across domains. On PubMedQA, human text dense in specialized terminology exhibits entropy fluctuations resembling those of machine text (Section 5.2), suggesting elevated false-positive rates on technical writing. Prior work has also shown that detectors can be biased against non-native English writers (Liang et al., 2023); our evaluation is limited to English and does not assess this dimension. We therefore caution that RelPPR, like other detectors, should not be used as the sole basis for consequential decisions (e.g., academic misconduct findings or content removal), and that its scores should be treated as probabilistic evidence subject to human review.

REPRODUCIBILITY STATEMENT

Our code is available at the anonymous GitHub `https://anonymous.4open.science/r/entropy-llm-detector-998D/`. RelPPR is fully specified in Section 4.2 and has no hyperparameters. All source and evaluator models are publicly available open-weight checkpoints, and all datasets are public benchmarks. Appendix B details the full experimental setup: Section B.1 discloses evaluator configurations for the black-box and white-box settings; Table 4 includes instruction prompt templates; decoding parameters, the length-control and trimming protocol are in Section B.3; Section B.4 describes the setups of the length and decoding ablations.

REFERENCES

Sai Vamsi Alisetti. Paraphrase generator with t5, 2021.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, et al. The falcon series of open language models. *arXiv preprint arXiv:2311.16867*, 2023.

Guangsheng Bao, Yanbin Zhao, Zhiyang Teng, Linyi Yang, and Yue Zhang. Fast-DetectGPT: Efficient zero-shot detection of machine-generated text via conditional probability curvature. In *International Conference on Learning Representations (ICLR)*, 2024.

Eric Bigelow, Ari Holtzman, Hidenori Tanaka, and Tomer Ullman. Forking paths in neural text generation. In *International Conference on Learning Representations*, volume 2025, pp. 17134–17167, 2025.

Jad Doughman, Osama Mohammed Afzal, Hawau Olamide Toyin, Shady Shehata, Preslav Nakov, and Zeerak Talat. Exploring the limitations of detecting machine-generated text. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 4274–4281, 2025.

Falcon-LLM Team. Introducing the Technology Innovation Institute's Falcon 3: Making advanced AI accessible and available to everyone, everywhere, 2024.

Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 889–898, 2018.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. Eli5: Long form question answering. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 3558–3567, 2019.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander 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*, pp. 111–116, 2019.

Abhimanyu Hans, Avi Schwarzschild, Valeriia Cherepanova, Hamid Kazemi, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Spotting LLMs with binoculars: Zero-shot detection of machine-generated text. In *International Conference on Machine Learning (ICML)*, pp. 17519–17537, 2024.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28, 2015.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. Automatic detection of generated text is easiest when humans are fooled. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 1808–1822, 2020.

T. Jaeger and Roger Levy. Speakers optimize information density through syntactic reduction. In B. Schölkopf, J. Platt, and T. Hoffman (eds.), *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006.

T. Florian Jaeger. Redundancy and reduction: Speakers manage syntactic information density. *Cognitive Psychology*, 61(1):23–62, 2010.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. Pubmedqa: A dataset for biomedical research question answering. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 2567–2577, 2019.

Weixin Liang, Mert Yuksekgonul, Yining Mao, Eric Wu, and James Zou. Gpt detectors are biased against non-native english writers. *Patterns*, 4(7), 2023.

Alexander H Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, et al. Ministral 3. *arXiv preprint arXiv:2601.08584*, 2026.

Runheng Liu, Heyan Huang, Xingchen Xiao, and Zhijing Wu. Zero-shot detection of LLM-generated text via implicit reward model. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

R Thomas McCoy, Shunyu Yao, Dan Friedman, Matthew Hardy, and Thomas L Griffiths. Embers of autoregression: Understanding large language models through the problem they are trained to solve. *arXiv preprint arXiv:2309.13638*, 2023.

Clara Meister, Tiago Pimentel, Patrick Haller, Lena Ann Jäger, Ryan Cotterell, and Roger Levy. Revisiting the uniform information density hypothesis. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 963–980, 2021.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Mistral AI. Introducing Mistral Small 4, March 2026.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. DetectGPT: Zero-shot machine-generated text detection using probability curvature. In *International Conference on Machine Learning (ICML)*, pp. 24950–24962, 2023.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 1797–1807, 2018.

Zanlin Ni, Shenzhi Wang, Yang Yue, Tianyu Yu, Weilin Zhao, Yeguo Hua, Tianyi Chen, Jun Song, Cheng Yu, Bo Zheng, and Gao Huang. The flexibility trap: Rethinking the value of arbitrary order in diffusion language models. In *Forty-third International Conference on Machine Learning*, 2026.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025.

Jose A Perea and John Harer. Sliding windows and persistence: An application of topological methods to signal analysis. *Foundations of computational mathematics*, 15(3):799–838, 2015.

Xiao Pu, Jingyu Zhang, Xiaochuang Han, Yulia Tsvetkov, and Tianxing He. On the zero-shot generalization of machine-generated text detectors. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 4799–4808, 2023.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 2383–2392, 2016.

Zhenmei Shi, Junyi Wei, Zhuoyan Xu, and Yingyu Liang. Why larger language models do in-context learning differently? In *Forty-first International Conference on Machine Learning*, 2024.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, et al. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*, 2019.

Jinyan Su, Terry Zhuo, Di Wang, and Preslav Nakov. Detectllm: Leveraging log rank information for zero-shot detection of machine-generated text. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 12395–12412, 2023.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, et al. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for LLM reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Junchao Wu, Runzhe Zhan, Derek F Wong, Shu Yang, Xuebo Liu, Lidia S Chao, and Min Zhang. Who wrote this? the key to zero-shot llm-generated text detection is gecscore. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 10275–10292, 2025.

Yihuai Xu, Yongwei Wang, Yifei Bi, Huangsen Cao, Zhouhan Lin, Yu Zhao, and Fei Wu. Training-free llm-generated text detection by mining token probability sequences. In *International Conference on Learning Representations*, volume 2025, pp. 19072–19098, 2025.

Xianjun Yang, Wei Cheng, Yue Wu, Linda Petzold, William Wang, and Haifeng Chen. Dna-gpt: Divergent n-gram analysis for training-free detection of gpt-generated text. In *International Conference on Learning Representations*, volume 2024, pp. 48572–48597, 2024.

Chenghua Zhu, Siyan Wu, Xiangkang Zeng, Zishan Xu, Zhaolu Kang, Yifu Guo, Yuquan Lu, Junduan Huang, and Guojing Zhou. EDIS: Diagnosing LLM reasoning via entropy dynamics. *arXiv preprint arXiv:2602.01288*, 2026.

Xiaowei Zhu, Yubing Ren, Fang Fang, Qingfeng Tan, Shi Wang, and Yanan Cao. DNA-detectLLM: Unveiling AI-generated text via a DNA-inspired mutation-repair paradigm. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

A Failure Analysis

A.1 Fast-DetectGPT

In the general dual-model formulation where p_1 serves as the sampling model and p_2 as the scoring model, Fast-DetectGPT (Bao et al., 2024) considers drawing candidate counterfactual tokens $\tilde{x} \sim p_1(\cdot \mid x_{<i})$ at each position i. It computes their conditional expectation $\tilde{\mu}_i$ and variance $\tilde{\sigma}_i^2$ under p_2:

$$\tilde{\mu}_i = \mathbb{E}_{\tilde{x}\sim p_1(\cdot|x_{<i})}\big[\log p_2(\tilde{x} \mid x_{<i})\big], \qquad \tilde{\sigma}_i^2 = \mathrm{Var}_{\tilde{x}\sim p_1(\cdot|x_{<i})}\big[\log p_2(\tilde{x} \mid x_{<i})\big].$$

The detection score is then defined as the normalized sum of conditional likelihood deviations:

$$\mathrm{Score}_{\mathrm{F}}(s) = \frac{N(s)}{\sqrt{\sum_{i=1}^{L} \tilde{\sigma}_i^2}}, \qquad N(s) = \sum_{i=1}^{L} \Big[\log p_2(x_i \mid x_{<i}) - \tilde{\mu}_i\Big],$$

where the numerator $N(s)$ accumulates conditional likelihood deviations, and the denominator normalizes them by the conditional variance. By definition, $\tilde{\mu}_i$ is the negative of the per-position cross-entropy, so summing over i gives:

$$\sum_{i=1}^{L} \tilde{\mu}_i = -L \cdot H(s; p_1, p_2), \qquad \sum_{i=1}^{L} \log p_2(x_i \mid x_{<i}) = -L \cdot I_{p_2}(s).$$

Substituting both into $N(s)$ yields:

$$N(s) = L \cdot \big(H(s; p_1, p_2) - I_{p_2}(s)\big).$$

Table 3 compares $\mathrm{Score}_{\mathrm{F}}(s)$ against $N(s)$ across generation conditions and datasets, to test whether the variance normalization denominator drives the performance collapse on instruction-tuned generations. As shown in the table, removing the denominator produces virtually identical detection performance across all settings, validating our analytical focus on decomposing $N(s)$ in Section 3.2.

A.2 Binoculars and DNA-DetectLLM

Rather than measuring a difference, Binoculars (Hans et al., 2024) and DNA-DetectLLM (Zhu et al., 2025) employ a ratio-based formulation. Following their official code implementations, Binoculars defines its score as the negative ratio of log-perplexity to log cross-perplexity given model p_1 and model p_2,:

$$\mathrm{Score}_{\mathrm{B}}(s) = -\frac{\log \mathrm{PPL}_{p_2}(s)}{\log \text{X-PPL}_{p_1,p_2}(s)} = -\frac{I_{p_2}(s)}{H(s; p_1, p_2)}.$$

DNA-DetectLLM additionally repairs s into a greedily re-decoded sequence $\tilde{s}$ under the scoring model, $\tilde{s}_t = \arg\max_{v\in\mathcal{V}} p_2(v \mid s_{<t})$, and defines the conditional surprisal of $\tilde{s}$ given s as the surprisal of the repaired sequence evaluated position-by-position against the original prefixes,

$$I_{p_2}(\tilde{s} \mid s) = -\frac{1}{L}\sum_{i=1}^{L} \log p_2(\tilde{x}_i \mid x_{<i}).$$

The score is defined as:

$$\mathrm{Score}_{\mathrm{D}}(s) = -\frac{\log \mathrm{PPL}_{p_2}(s) + \log \mathrm{PPL}_{p_2}(\tilde{s} \mid s)}{2 \cdot \log \text{X-PPL}_{p_1,p_2}(s)} = -\frac{I_{p_2}(s) + I_{p_2}(\tilde{s} \mid s)}{2 \cdot H(s; p_1, p_2)}.$$

A.3 Cross-Entropy Decomposition

Section 3.2 decomposes the sequence-level cross-entropy $H(s; p_1, p_2)$ into the sequence-level entropy $H_{p_1}(s)$ and KL divergence $D_{\mathrm{KL}}(s; p_1, p_2)$. We derive this decomposition here. At each position i, the cross-entropy, entropy, and KL divergence between the sampling distribution $p_1(\cdot \mid x_{<i})$ and the scoring distribution $p_2(\cdot \mid x_{<i})$ satisfy the standard identity

$$H(p_1, p_2)_i = H_{p_1,i} + D_{\mathrm{KL}}(p_1, p_2)_i,$$

Table 3: Per-dataset ROC AUC comparison between $\mathrm{Score}_{\mathrm{F}}(s)$ and $N(s)$ for OLMo-3-32B-generated text. $\Delta = \mathrm{Avg}_N - \mathrm{Avg}_{\mathrm{Score}_{\mathrm{F}}}$. Removing the variance denominator produces virtually no change in AUROC ($|\Delta| \leq 0.007$).

Condition	Variant	XSum	SQuAD	WritingP	CNN/DM	ELI5	WikiText	PubMedQA	Average	Δ
w/o Instruction	$\mathrm{Score}_{\mathrm{F}}(s)$	0.766	0.753	0.769	0.708	0.683	0.609	0.719	0.715	+0.007
	$N(s)$	0.794	0.749	0.774	0.707	0.687	0.615	0.728	0.722	
w/ Instruction	$\mathrm{Score}_{\mathrm{F}}(s)$	0.569	0.514	0.494	0.497	0.556	0.509	0.465	0.515	0.000
	$N(s)$	0.569	0.502	0.501	0.500	0.552	0.510	0.469	0.515	

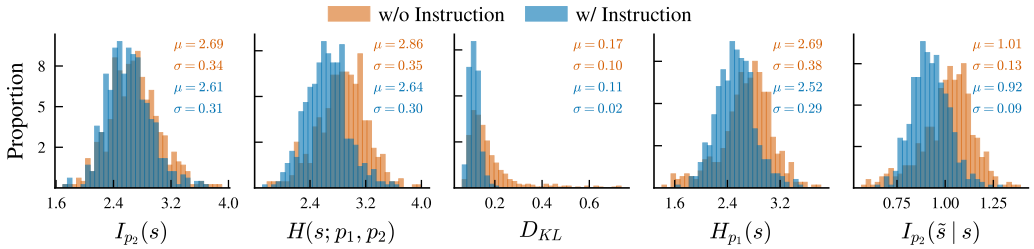


Figure 8: Distributions of the five information-theoretic components from Section 3.2 on XSum, for machine text generated by OLMo-3.1-32B-Instruct without (orange) and with (blue) the instruction prompt. The y-axis shows the percentage of samples per bin. μ and σ denote the mean and standard deviation.

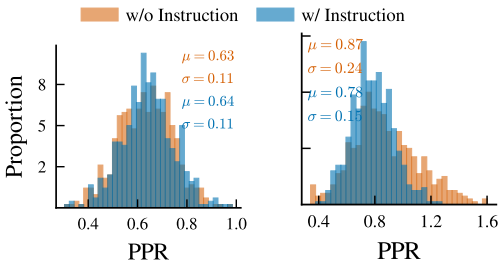
which holds for any pair of distributions over the same support $\mathcal{V}$. Averaging both sides over positions $i = 1, \ldots, L$ preserves the identity by linearity:

$$H(s; p_1, p_2) = \frac{1}{L}\sum_{i=1}^{L} H_{p_1,i} + \frac{1}{L}\sum_{i=1}^{L} D_{\mathrm{KL}}(p_1, p_2)_i = H_{p_1}(s) + D_{\mathrm{KL}}(s; p_1, p_2),$$

where $D_{\mathrm{KL}}(s; p_1, p_2) = \frac{1}{L}\sum_{i=1}^{L} D_{\mathrm{KL}}\big(p_1(\cdot \mid x_{<i}) \,\|\, p_2(\cdot \mid x_{<i})\big)$ denotes the sequence-level KL divergence, which is Equation (3) in Section 3.2.

A.4 Statisctic Distributions on Additional Datasets

Here we provide the distributions of the five statistics mentioned in Section 3.2 for the remaining five datasets in Figures 9 to 12. XSum, CNN/DailyMail, SQuAD, and WikiText reproduce the ELI5 pattern: instruction conditioning shrinks H_{p_1} and D_{KL}, collapsing cross-entropy $H(p_1, p_2)$ to lower value while surprisal I_{p_2} stays largely stable. However, WritingPrompts is an exception: the scores in Equations (1) and (2) decrease because $H(s; p_1, p_2)$ remains virtually unchanged while $I_{p_2}(s)$ increases noticeably. This behavior is intuitively expected on this creative writing task, since stories generated with instruction prompts exhibit higher narrative branching and diversity; however, the prompts are invisible to evaluator models, which naturally incurs higher surprisal for the evaluator model.



(a) ELI5 (b) PubMedQA

Figure 7: Distributions of PPR on machine-generated text from OLMo-3.1-32B-Instruct for datasets ELI5 and PubMedQA.

A.5 PPR Distributions on Additional Datasets

Figure 4 shows the raw PPR (s) distribution shift under instruction conditioning on four datasets (XSum, CNN/DailyMail, WikiText, WritingPrompts). Figure 7 shows the corresponding distribu-

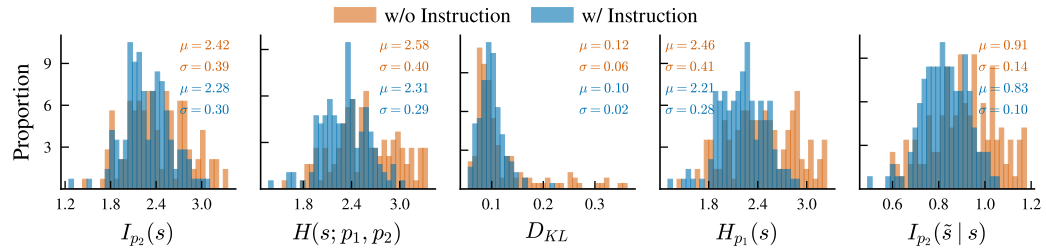


Figure 9: Distributions of the five information-theoretic components from Section 3.2 on SQuAD, for machine text generated by OLMo-3.1-32B-Instruct without (orange) and with (blue) the instruction prompt. The y-axis shows the percentage of samples per bin. μ and σ denote the mean and standard deviation.

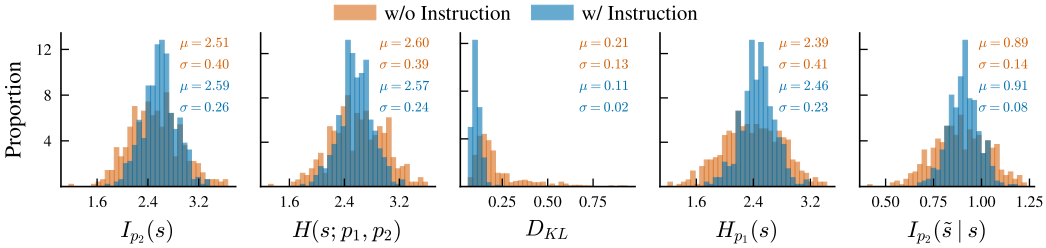


Figure 10: Distributions of the five information-theoretic components from Section 3.2 on CNN/DailyMail, for machine text generated by OLMo-3.1-32B-Instruct without (orange) and with (blue) the instruction prompt. The y-axis shows the percentage of samples per bin. μ and σ denote the mean and standard deviation.

tions on the remaining two datasets, ELI5 and PubMedQA. ELI5 shows only a small shift, consistent with RelPPR's more modest margin over baselines on this dataset (Table 1); PubMedQA is the sole dataset where the shift reverses direction, matching the PubMedQA outlier discussed in Section 5.2.

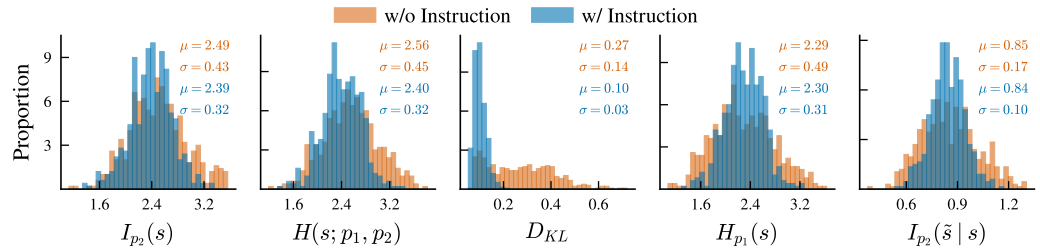


Figure 11: Distributions of the five information-theoretic components from Section 3.2 on WikiText, for machine text generated by OLMo-3.1-32B-Instruct without (orange) and with (blue) the instruction prompt. The y-axis shows the percentage of samples per bin. μ and σ denote the mean and standard deviation.

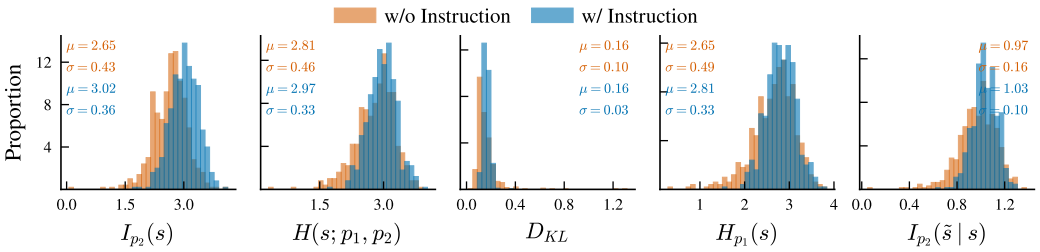


Figure 12: Distributions of the five information-theoretic components from Section 3.2 on WritingPrompts, for machine text generated by OLMo-3.1-32B-Instruct without (orange) and with (blue) the instruction prompt. The y-axis shows the percentage of samples per bin. μ and σ denote the mean and standard deviation.

Table 4: Instruction prompt templates across all seven benchmark datasets. {`prefix`} represents the conditioning context, and {`word_target`} denotes the expected word count (150 words, except PubMedQA at 30 words).

Dataset	**Instruction Prompt Template**
XSum	"*Continue the following text for about* {`word_target`} *words. Output ONLY the continuation – do not repeat or restate any of the given text:* {`prefix`}"
CNN/DM	"*Continue the following text for about* {`word_target`} *words. Output ONLY the continuation – do not repeat or restate any of the given text:* {`prefix`}"
WikiText	"*Continue the following text for about* {`word_target`} *words. Output ONLY the continuation – do not repeat or restate any of the given text:* {`prefix`}"
SQuAD	"*Continue the following text for about* {`word_target`} *words. Output ONLY the continuation – do not repeat or restate any of the given text:* {`prefix`}"
WritingPrompts	"*Please write a story of about* {`word_target`} *words based on the following writing prompt:* {`prefix`}. *Output ONLY the story – do not repeat or restate any of the given text:* {`prefix`}"
ELI5	"*Please explain the following question in about* {`word_target`} *words:* {`prefix`}. *Output ONLY the explanation – do not repeat or restate any of the given text:* {`prefix`}"
PubMedQA	"*Please answer the question in about* {`word_target`} *words. Output ONLY the answer – do not repeat or restate any of the given text:* {`prefix`}"

B EXPERIMENTAL SETUP AND IMPLEMENTATION DETAILS

In this section, we present the complete experimental setup, including prompt templates, text generation protocols, length-control mechanisms, and evaluator model pairings used across our experiments.

B.1 DETECTION SETTINGS AND EVALUATOR MODELS

Source Models. We use six instruction-tuned LLMs as source models: OLMo-3-7B-Instruct, OLMo-3.1-32B-Instruct (Olmo et al., 2025), Qwen3.5-35B-A3B (Qwen Team, 2026), Gemma-4-31B-it (Team et al., 2026), Mistral-Small-4-119B (Mistral AI, 2026).

Black-Box Setting. In the black-box setting, the detector operates without knowledge of the source model. Single-model statistical criteria employ Qwen3.5-2B-Base as the evaluator (and CoEdit-Large for GECScore). dual-model baselines Qwen3.5-2B-Base, Qwen3.5-2B-Instruct as p_1 and p_2, respectively.

White-Box Setting. In the white-box regime, the detector directly utilizes the generative source model ($M = M_{\text{source}}$). For dual-model detectors, the source instruction-tuned model serves as p_2, while its same-lineage foundation base counterpart at the same parameter scale serves as the reference/observer model p_1 (e.g., OLMo-3-7B-Base for OLMo-3-7B-Instruct, Qwen3.5-35B-A3B-Base for Qwen3.5-35B-A3B). Mistral-Small-4-119B is excluded from the white-box setting, as no same-lineage base checkpoint is available for it; GECScore is likewise excluded, since it always scores against a fixed off-the-shelf grammar-correction model and therefore has no white-box variant.

B.2 PROMPT TEMPLATES AND DECODING CONFIGURATION

Table 4 details the instruction prompt template for each benchmark dataset. In our main experiments, all text generations are carried out using the vLLM engine with $T = 1.0$, $p = 1.0$, and top-$k = 50$.

B.3 TEXT LENGTH CONTROL AND SYMMETRICAL TRIMMING

Systematic length discrepancies between human and machine text can introduce trivial shortcut features into detection. To ensure rigorous, length-unbiased evaluation, we enforce a multi-stage length-control protocol:

1. *Document pre-filtering:* Human documents shorter than a minimum word threshold (200 words for standard datasets, and 150 words for ELI5) are filtered out to guarantee sufficient text length (PubMedQA is not filtered by length, since the sequence in it is short).

2. *Generation constraint:* The prompt parameter {`word_target`} is uniformly set to 150 words across six datasets, and to 30 words for PubMedQA to match its considerably shorter human answers. During generation, we set `min_tokens` to 150 for datasets except PubMedQA, and to 30 for PubMedQA, alongside an upper bound of `max_new_tokens` = 200 for all datasets.

3. *Post-trimming:* Following Bao et al. (2024), after generating machine continuations, both the human completion and machine text are encoded into token sequences under the generative model's tokenizer and trimmed down to their shared minimum token length: $L = \min(|s_{\text{human}}|, |s_{\text{machine}}|)$. Both trimmed sequences are then decoded back to text.

This guarantees that for every candidate pair, the human and machine texts have the *exact same sequence length* under the evaluation tokenizer. We draw 500 paired samples (1,000 texts total) per dataset and model, except for SQuAD where 143 pairs are yielded due to pre-filtering length constraint. The resulting sample lengths for XSum, WritingPrompts, SQuAD, and CNN/DailyMail are uniformly 200 tokens, with detailed token length distributions on ELI5 and PubMedQA provided in Table 5.

Table 5: Token length distributions (Mean ± Std) on the ELI5 and PubMedQA across all evaluated instruction models ($n = 500$ pairs per model), measured with each source model's own tokenizer.

	OLMo-7B	**OLMo-32B**	**Gemma-4-31B**	**Qwen-35B-MoE**	**Mistral-119B**
ELI5	178.9 ± 12.8	174.4 ± 13.6	171.2 ± 9.7	170.2 ± 11.8	182.8 ± 14.6
PubMedQA	33.3 ± 6.1	34.3 ± 8.9	32.0 ± 4.7	32.9 ± 6.1	32.8 ± 6.2

Table 6: Zero-shot detection AUROC on OLMo-3.1-32B-Instruct (black-box, avg. over all 7 datasets) as the evaluator model varies in size and family (Qwen3.5 (Qwen Team, 2026), Gemma 4 (Team et al., 2026), Ministral 3 (Liu et al., 2026), and Falcon (Almazrouei et al., 2023; Falcon-LLM Team, 2024)). Best result per column is shown in **bold**, second-best is underlined.

Methods	**Qwen3.5 2B**	**Qwen3.5 9B**	**Qwen3.5 35B-MoE**	**Gemma4 12B**	**Gemma4 31B**	**Gemma4 26B-MoE**	**Falcon3 7B**	**Ministral3 8B**	**Falcon 7B**	**Avg**
Entropy	0.837	0.847	0.788	0.804	0.780	0.765	**0.761**	0.788	0.414	0.754
Likelihood	0.794	0.814	0.746	0.722	0.728	0.720	0.671	0.772	0.526	0.721
LogRank	0.789	0.814	0.749	0.732	0.731	0.723	0.687	0.770	0.535	0.726
LastDE	0.781	0.812	0.755	0.761	0.769	0.760	0.684	0.796	0.657	0.753
Fast-DetectGPT	0.515	0.513	0.456	0.507	0.230	0.301	0.653	0.625	0.825	0.514
Binoculars	0.521	0.526	0.472	0.683	0.652	0.523	0.649	0.642	0.827	0.611
DNA-DetectLLM	0.515	0.502	0.459	0.678	0.630	0.518	0.729	0.566	**0.855**	0.606
RelPPR (Ours)	**0.887**	**0.863**	**0.826**	**0.805**	**0.804**	**0.804**	0.759	**0.835**	0.602	**0.798**

B.4 Length-Ablation and Decoding-Ablation Experimental Setup

For each tier, we re-generate machine continuations with the prompt's {`word_target`} set to a length-matched word count (40, 80, 120 words), set `min_tokens` to this same value, and cap `max_new_tokens` at the token tier itself, forcing a continuation of approximately the target length. Human and machine texts are then symmetrically trimmed as in Section B.3. All other settings are unchanged from the main experiments. This procedure is applied in the black-box setting, across the XSum, WritingPrompts, SQuAD, WikiText, and CNN/DailyMail datasets; ELI5 and PubMedQA are excluded since their human documents are shorter.

For the decoding ablation, we re-generate machine continuations under three decoding configurations spanning the sampling-randomness spectrum: $(T{=}1.0, p{=}1.0, k{=}50)$, $(T{=}0.7, p{=}0.95, k{=}50)$, and $(T{=}0.3, p{=}0.9, k{=}20)$, with all other generation settings (length control, datasets, source models) unchanged from the main experiments. Human and machine texts are again symmetrically trimmed as in Section B.3, and each baseline is re-scored at every decoding configuration across all seven datasets.

B.5 Robustness Across Evaluator Size and Family

We assess detection performance across varying evaluator models while fixing the source model to OLMo-3.1-32B-Instruct. As shown in Table 6, RelPPR demonstrates superior generalizability across diverse evaluator architectures and parameter scales, securing the top performance on seven of the nine evaluators and ranking second on another. Notably, while scaling the evaluator size within the same family yields marginal impact on detection performance, cross-family shifts induce substantially more pronounced variations. Additionally, within the Qwen3.5 family, transitioning from dense architectures to the MoE variant visibly degrades the detection efficacy of all baselines.

B.6 White-Box Detection Results

Single-model criteria employ the source model itself as the evaluator, whereas dual-model configurations follow Section B.1. In addition to PPR, DiffPPR, and RelPPR, we test whether PPR can offer complementary signals over standard likelihood by pairing it with sequence-level surprisal, yielding the variant DiffPPR-S:

$$\mathrm{Score}_{\mathrm{DiffPPR-S}}(s) = \mathrm{PPR}(s) - I_p(s).$$

Tables 7 and 8 report the detection results under white-box setting. Unlike the black-box setting, where RelPPR is state-of-the-art, dual-model detectors achieves the best performance under white-box setting, and IRM attains the best or second-best score in nearly every source model and dataset.

Table 7: Detection performance (AUROC) by dataset (averaged across four source models) under white-box setting. The best and second-best results are shown in **bold** and underlined, respectively.

Baseline	XSum	WritingP	SQuAD	WikiText	CNN/DM	ELI5	PubMedQA	Avg
Entropy	0.803	0.856	0.589	0.683	0.724	0.855	0.667	0.740
Likelihood	0.931	0.911	0.708	0.913	0.920	0.907	0.574	0.838
LogRank	0.926	0.901	0.697	0.909	0.915	0.896	0.601	0.835
LRR	0.798	0.797	0.614	0.767	0.777	0.811	0.620	0.741
LastDE	0.887	0.882	0.794	0.941	0.908	0.890	0.534	0.834
LastDE++	0.416	0.555	0.245	0.414	0.485	0.381	0.426	0.418
DNA-GPT	0.465	0.561	0.431	0.439	0.401	0.622	0.519	0.491
Fast-DetectGPT	0.817	0.892	0.708	0.835	0.823	0.767	0.482	0.760
Binoculars	0.956	0.990	0.849	0.952	0.948	0.932	0.671	0.900
DNA-DetectLLM	0.964	**0.992**	0.845	0.946	0.949	0.915	0.584	0.885
IRM	**0.966**	0.962	**0.941**	**0.974**	**0.958**	**0.972**	**0.679**	**0.922**
PPR (Ours)	0.382	0.461	0.481	0.405	0.405	0.291	0.405	0.404
DiffPPR (Ours)	0.825	0.893	0.608	0.700	0.737	0.887	0.648	0.757
RelPPR (Ours)	0.769	0.883	0.617	0.663	0.689	0.903	0.567	0.727
DiffPPR-S (Ours)	0.935	0.920	0.716	0.925	0.926	0.913	0.548	0.840

Table 8: Detection performance (AUROC) by source model (averaged across seven datasets) under white-box setting. The best and second-best results are shown in **bold** and underlined, respectively.

Baseline	OLMo3 7B	OLMo3.1 32B	Gemma4 31B	Qwen3.5 35B	Avg
Entropy	0.791	0.805	0.552	0.810	0.740
Likelihood	0.901	0.870	0.776	0.803	0.838
LogRank	0.906	0.866	0.758	0.809	0.835
LRR	0.791	0.791	0.590	0.790	0.741
LastDE	0.893	0.884	0.745	0.813	0.834
LastDE++	0.461	0.414	0.359	0.439	0.418
DNA-GPT	0.502	0.536	0.396	0.531	0.491
Fast-DetectGPT	0.934	0.913	0.466	0.729	0.760
Binoculars	**0.947**	0.924	**0.949**	0.779	0.900
DNA-DetectLLM	**0.947**	0.913	0.926	0.753	0.885
IRM	0.928	**0.952**	0.869	**0.938**	**0.922**
PPR (Ours)	0.317	0.300	0.571	0.429	0.404
DiffPPR (Ours)	0.774	0.801	0.625	0.826	0.757
RelPPR (Ours)	0.678	0.744	0.677	0.811	0.727
DiffPPR-S (Ours)	0.897	0.870	0.780	0.814	0.840

This is expected: white-box access lets the evaluator model that matches the source model assign very high confidence to the machine text, while the other evaluator model cannot reach a comparable confidence, which restores cross-model discriminability between the two evaluator models. Notably, since detectors still cannot access the user prompt under the white-box setting, the prompt conditioning gap described in Section 3.3 still persists, so resampling-based baselines remain ineffective under the white-box setting. While RelPPR itself is not competitive in this regime, DiffPPR-S ranks best among all single-model criteria, confirming that PPR can still serve as a complementary signal under the white-box setting.