RepreGuard: Detecting LLM-Generated Text by Revealing Hidden Representation Patterns

Xin Chen[1,2*] **Junchao Wu**[1*] **Shu Yang**[3] **Runzhe Zhan**[1] **Zeyu Wu**[1] **Ziyang Luo**[4]
Di Wang[3] **Min Yang**[2] **Lidia S. Chao**[1] **Derek F. Wong**[1†]

[1]NLP [2]CT Lab, Department of Computer and Information Science, University of Macau, China
[2]Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
[3]Provable Responsible AI and Data Analytics Lab, KAUST, Saudi Arabia
[4]Hong Kong Baptist University, China
nlp2ct.{xinchen,junchao,runzhe,zeyu}@gmail.com,
{shu.yang,di.wang}@kaust.edu.sa
min.yang@siat.ac.cn, cszyluo@comp.hkbu.edu.hk, {derekfw,lidiasc}@um.edu.mo

Abstract

Detecting content generated by large language models (LLMs) is crucial for preventing misuse and building trustworthy AI systems. Although existing detection methods perform well, their robustness in out-of-distribution (OOD) scenarios is still lacking. In this paper, we hypothesize that, compared to features used by existing detection methods, the internal representations of LLMs contain more comprehensive and raw features that can more effectively capture and distinguish the statistical pattern differences between LLM-generated texts (LGT) and human-written texts (HWT). We validated this hypothesis across different LLMs and observed significant differences in neural activation patterns when processing these two types of texts. Based on this, we propose **RepreGuard**, an efficient statistics-based detection method. Specifically, we first employ a surrogate model to collect representation of LGT and HWT, and extract the distinct activation feature that can better identify LGT. We can classify the text by calculating the projection score of the text representations along this feature direction and comparing with a precomputed threshold. Experimental results show that RepreGuard outperforms all baselines with average 94.92% AUROC on both in-distribution and OOD scenarios, while also demonstrating robust resilience to various text sizes and mainstream attacks.[1]

[*] Equal contribution.
[†] Corresponding author.
[1]Data and code are publicly available at: https://github.com/NLP2CT/RepreGuard.

1 Introduction

Large language models (LLMs) demonstrate impressive language understanding and generation capabilities (OpenAI, 2022; Anthropic, 2023; Ghahramani, 2023; MetaAI, 2024), enabling them to produce creative and persuasive content that aligns with human preferences (Wu et al., 2023). These capabilities have raised concerns regarding future data regulation, particularly due to biases (Tjuatja et al., 2023) and hallucinations (Ji et al., 2023) in LLM-generated texts (LGT). Moreover, the potential misuse of LLMs, such as generating fake news (Pagnoni et al., 2022) or facilitating academic dishonesty (Cotton et al., 2024), poses significant risks and challenges to society. To defend against these usage cases, several algorithms have been developed to detect LGT. These primarily include fine-tuning-based classifiers, which involve training an model with extensive labeled data for classification, such as the OpenAI detector (Solaiman et al., 2019), and statistics-based methods, which identify LGT using feature metrics from specific distributions in a small training dataset, such as DetectGPT (Mitchell et al., 2023).

Fine-tuning-based classifiers generally offer higher accuracy than statistics-based detectors, but require large amounts of labeled data and often struggle to generalize across different generators, making updates costly for new models (Guo et al., 2023). In contrast, statistics-based methods provide better interpretability and only require setting a threshold based on observed distribution patterns in a small sample size, offering stonger reliability for real-world applications (Wu

Transactions of the Association for Computational Linguistics, vol. 13, pp. 1812–1831, 2025. https://doi.org/10.1162/TACL.a.61
Action Editor: Mohit Iyyer. Submission batch: 10/2024; Revision batch: 5/2025; Published 12/2025.

et al., 2023). However, current statistics-based methods perform poorly in both in-distribution (ID) and out-of-distribution (OOD) scenarios due to the insufficient robustness in classification feature metrics. For example, varying prompts could control the perplexity of generated text, rendering thresholds from training samples ineffective (Hans et al., 2024). This limitation is exacerbated in OOD scenarios, posing significant challenges to the usability of statistics-based detectors, with the growing number of new LLMs.

In response to this challenge, we propose a detection method based on hidden representation to identify texts generated by LLMs. This detection method is motivated by the following hypothesis: LLMs exhibit distinct hidden representation patterns when processing LGT compared to human-written texts (HWT), due to their different perceptions of statistical patterns in these text types.

These hidden representation patterns differences can be more explicitly observed in the model's representations, which are higher-dimensional features. This is also where the detection abilities of existing statistics-based detectors come into play, utilizing classification feature metrics such as likelihood, rank, and other variants. Thus, the model's representations may encompass more comprehensive and raw features that can enhance the ability to distinguish between text types, with a stronger potential for identifying LGT. To achieve this, we first employ a surrogate model as an ''observer'' to obtain the representations when processing texts generated by LLMs and those written by humans. We observed that the hidden representation patterns of the two types of texts show distinct differences, which serve as a strong signal for detecting LGT. Then, to filter out noisy features that might hinder LGT detection, we perform dimensionality reduction and modeling on both types of text representations to identify features that maximally retain the distinguishing information. By calculating the projection score of the representation of a given text along this feature direction, which we termed ''RepreScore'', and comparing it to the optimal classification threshold statistically derived, we can determine whether the text was generated by an LLM or written by a human. We call this detection method **RepreGuard**, which is an efficient detection method that combines the strengths of both statistics-based and fine-tuning-based approaches. RepreGuard exhibits zero-shot characteristics, requiring only a small number of training samples from one LLM source to generalize across texts generated by various types of LLMs.

Experiments on different setups show that RepreGuard outperforms the SOTA RoBERTa-based classifier and the statistics-based method Binoculars in both ID and OOD scenarios, with an average of 11.05% and 05.88% AUROC higher than RoBERTa-based classifier and Binoculars, respectively, and achieve better performance with fewer training samples. In addition, our method demonstrates strong robustness to the generalization on domains, texts with varied sizes, mainstream attacks including text paraphrasing and perturbation attacks and various sampling methods. It also achieves an excellent balance between effectiveness and resource efficiency.

2 Related Work

Statistics-based Detection Methods The statistics-based detection method detects LGT by examining the distribution difference of the specified feature metrics between LGT and HWT. This classification is achieved by extracting these feature metrics from the given text and comparing them to thresholds derived from statistics on training datasets, without the need to fine-tune a neural model as classifier. Early statistics-based methods focused on calculating feature metrics derived from the model output logits, such as Entropy, Log-Likelihood, and Log-Rank (Solaiman et al., 2019). Subsequently, Su et al. (2023) introduced the Log-Likelihood Log-Rank Ratio (LRR), offering a more comprehensive evaluation by calculating the ratio of Log-Likelihood to Log-Rank. Recently, perturbation-based methods have gained significant attention. Mitchell et al. (2023) and Su et al. (2023) used Log-Likelihood and Log-Rank curvature, respectively, to identify LGT, based on the hypothesis that LGT maintains higher Log-Likelihood and Log-Rank after semantic perturbation compared to HWT. Bao et al. (2024) enhanced this by replacing the perturbation step in DetectGPT with a more efficient sampling procedure, reducing the computational cost. Other methods, like DNA-GPT (Yang et al., 2024), use an iterative process where an LLM extends truncated text and assesses authorship via probability differences between the original and extended text. In contrast, GECScore (Wu et al., 2025) relies on the finding that LLMs are more likely to correct grammatical errors in human-written text and distinguishes text sources by measuring changes in similarity before and after grammatical error correction. Binoculars (Hans et al., 2024) is a novel method that employs a pair of LLMs to calculate the ratio of perplexity and cross-perplexity, measuring how one model's prediction of the next token surprises another one.

Fine-Tuning-Based Detection Methods Another approach is to fine-tune a neural model as a classifier for distinguishing between HWT and LGT, typically requiring a large amount of labeled data. Early efforts focused on fine-tuning pre-trained classifiers for detecting news articles (Zellers et al., 2019) and social media content (Fagni et al., 2020). Recent studies (Guo et al., 2023; Liu et al., 2023; Chen et al., 2023; Wang et al., 2023) further confirmed the strong performance of fine-tuned language model in identifying LGT. OpenAI's detector, for example, is a fine-tuned RoBERTa-based classifier to perform this task (Solaiman et al., 2019). However, fine-tuning-based classifiers tend to overfit to their training data or the training distribution of the source model, resulting in performance drops when encountering new LLMs or domains data (Sarvazyan et al., 2023).

3 RepreGuard

3.1 Preliminary

Hypothesis *The premise of RepreGuard is that LLMs perceive the statistical patterns of LGT and HWT differently. Their hidden representation patterns exhibit noticeable differences when observing and processing these two types of text.*

This hypothesis arises from the idea that there are behavioral differences in the writing processes between LGT and HWT, which can be captured through internal hidden representations, as the internal hidden representations of LLMs typically contain more information and raw features compared to external behaviours. This information can reveal the model's intrinsic understanding of the differences between the LGT and HWT.

Internal Representation of LLMs Recent research (Voita et al., 2024; Durrani et al., 2020; Xu et al., 2024b) suggests that the internal representation mechanisms of LLMs may capture more information. For instance, LLM-Check (Sriramanan et al., 2024) extracts the hidden representations from each layer during response generation, and calculates their covariance matrices (Hidden Score) as an indicator for hallucination detection. Zhang et al. (2024) indirectly reveal the spatial reasoning capabilities encoded in the hidden representations of vision-language models by systematically analyzing output probabilities under continuous variations in input space. Tang et al. (2024) proposed a method of extracting latent representations as specific concept directions, thereby enabling the effective guidance and control of LLMs towards the concept direction (eg, safety and honesty).

Therefore, RepreGuard is designed to capture the underlying hidden representations that characterise these processes based on the distinct behavioral processes in human writing and AI writing. These distinctions likely result from the statistical patterns internalized by LLMs during training and how these are leveraged in generative tasks. To validate this hypothesis, we follow Zou et al. (2023) and compare the neural activation patterns in Llama-3.1-8B when processing 1000 pairs of LGT and HWT from 4 different LLMs, as illustrated in Figure 1. Specifically, LGT and HWT exhibit largely similar hidden representations during the early stage of the sequence (approximately tokens 0–20). However, when the number of tokens exceeds 20, their hidden representations begin to diverge substantially, with LGT demonstrating a consistently higher overall activation level compared to HWT. Moreover, at the same sequence positions, LGT and HWT remain relatively similar in the lower layers (approximately layers 1–11), while exhibiting more pronounced differences in layers 11–32.

3.2 Detecting by Representation Comparisons

Based on this hypothesis, we propose **RepreGuard** for detecting LGT using the hidden representation patterns of LLMs. By analyzing the activation patterns of a surrogate model during the processing of LGT and HWT, we aim to capture subtle but systematic differences in the activation patterns when LLMs process them, and extract the most significant representation feature for the effective detection of LGT. The overview

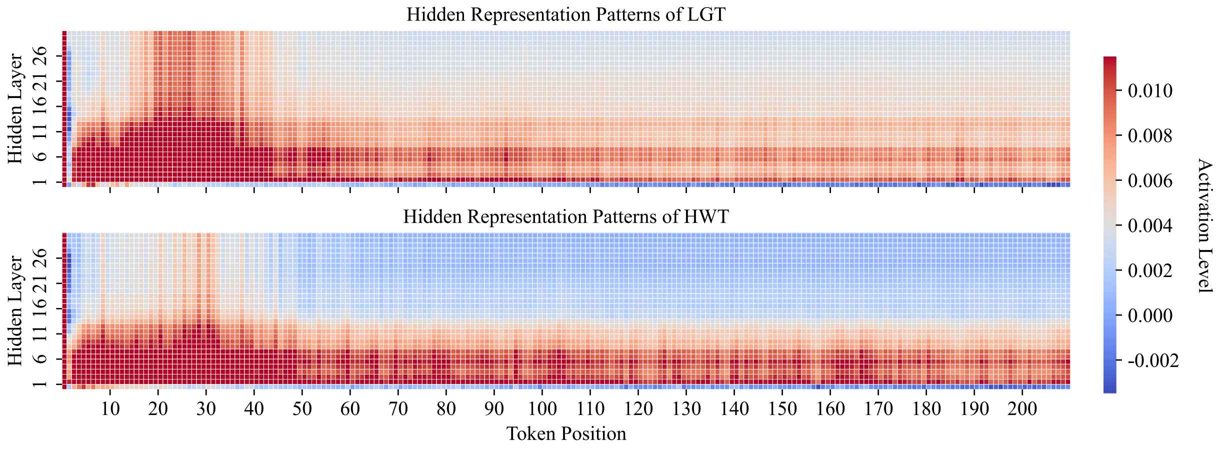

Figure 1: Comparison of Average Hidden Representation Distribution in Llama-3.1-8B Using 1000 Pairs of LGT and HWT from 4 Different LLMs. The red represents active representations and blue represents relatively inactive representations. The depth of the color represents the level of representation activation.

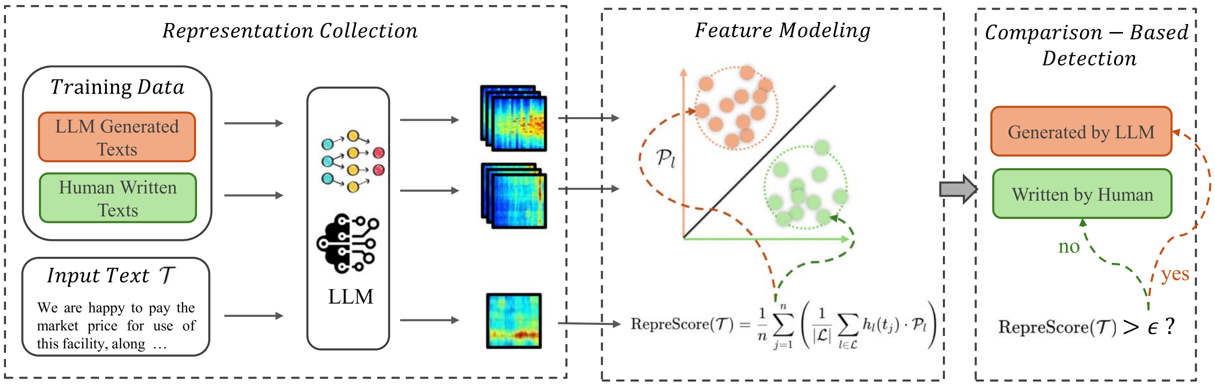
Representation Collection
Training Data
LLM Generated Texts
Human Written Texts
Input Text $\mathcal{T}$
We are happy to pay the market price for use of this facility, along …
LLM
Feature Modeling
$\mathcal{P}_l$
$\text{RepreScore}(\mathcal{T}) = \frac{1}{n}\sum_{j=1}^{n}\left(\frac{1}{|\mathcal{L}|}\sum_{l\in\mathcal{L}} h_l(t_j)\cdot\mathcal{P}_l\right)$
Comparison – Based Detection
Generated by LLM
Written by Human
no
yes
$\text{RepreScore}(\mathcal{T}) > \epsilon$?

Figure 2: The RepreGuard Framework Overview. The framework processes text through a surrogate model to capture hidden representation patterns and collect representation of LGT and HWT. It employs Principal Component Analysis model distinct activation features, filtering out noise and identifying key features that distinguish LGT. Next, the framework calculates an overall projection score, which we called ‘‘RepreScore’’, for the text to quantify how closely a text's activation pattern aligns with LGT representation features. A threshold is then set based on the RepreScores from the training data. In the application tion, if the given text's RepreScore exceeds this threshold, it is more likely to be generated by LLMs.

framework of RepreGuard is as illustrated in Figure 2.

Representation Collection We first introduce a small training set, formalized as $\{(\mathcal{T}^i_{\text{LGT}}, \mathcal{T}^i_{\text{HWT}}) \mid i \in [1, N]\}$, where N is the number of LGT and HWT pairs in the dataset. We use a surrogate model $\mathcal{M}$ as an ‘‘observer’’ to collect the corresponding representation distribution when processing LGT and HWT, to capture the difference in their activation patterns. Specifically, for each text sequence $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$ containing n tokens, we use $\mathcal{M}$ with $\mathcal{L}$ layers to ‘‘observe’’ it. For each token t_j in $\mathcal{T}$, we collect the neural activation of the model $\mathcal{M}$ at each layer l, represented as $h^l_j \in \mathbb{R}^d$, where d is the dimension of the model's hidden state. Based on this, the complete activation of the text $\mathcal{T}$ in model $\mathcal{M}$ can be formalized:

$$\mathcal{A}(\mathcal{T}) = \{h^l_j \mid j \in [1, n], l \in [1, \mathcal{L}]\} \quad (1)$$

For each text pair $(\mathcal{T}^i_{\text{LGT}}, \mathcal{T}^i_{\text{HWT}})$, we capture the model's representation activations from the end to the beginning of the token at all layers $\mathcal{L}$. We denote these activations as $\mathcal{A}(\mathcal{T}^i_{\text{LGT}})$, and $\mathcal{A}(\mathcal{T}^i_{\text{HWT}})$.

Feature Modeling The difference in activation patterns between LGT and HWT for each text pair can be denoted as $\Delta\mathcal{A}_i$, where:

$$\Delta\mathcal{A}_i = \mathcal{A}(\mathcal{T}^i_{\text{LGT}}) - \mathcal{A}(\mathcal{T}^i_{\text{HWT}}) \tag{2}$$

For each hidden layer l in $\mathcal{L}$, we capture this difference of neural activations from all text pairs:

$$\Delta\mathcal{A}^l = \{\Delta\mathcal{A}^l_i \mid i \in [1, N], l \in [1, \mathcal{L}]\} \tag{3}$$

where $\Delta\mathcal{A}^l_i = h^{l,i}_{\text{LGT}} - h^{l,i}_{\text{HWT}}$ represents the difference for the i-th text pair at layer l.

To filter out noise and identify key features distinguishing LGT, we perform Principal Component Analysis (PCA) on each $\Delta\mathcal{A}^l$:

$$\mathcal{P}_l = \text{PCA}(\Delta\mathcal{A}^l) \tag{4}$$

The resulting $\mathcal{P}_l$ represents the *probing vector* that differentiates between LGT and HWT at layer l, demonstrated that different types of text evoke distinct neural activation pattern in the LLM. We project the activations of each token t_j in text $\mathcal{T}$ onto the probing vector across all layers $\mathcal{L}$, defining this as the **RepreScore**:

$$\text{RepreScore}(t_j) = \frac{1}{|\mathcal{L}|}\sum_{l\in\mathcal{L}} h_l(t_j)\cdot\mathcal{P}_l \tag{5}$$

The overall projection score for the text $\mathcal{T}$ is the mean of its tokens' RepreScores:

$$\text{RepreScore}(\mathcal{T}) = \frac{1}{n}\sum_{j=1}^{n}\text{RepreScore}(t_j) \tag{6}$$

Comparison-Based Detection Based on the calculated RepreScore$(\mathcal{T})$ for each sample in the training dataset, we determine the optimal threshold θ to balance the true positive rate (TPR) and the false positive rate (FPR):

$$\theta = \arg\max_{\theta'}\left(\text{TPR}(\theta') + (1 - \text{FPR}(\theta'))\right) \tag{7}$$

In the application phase, RepreScore measures how closely the activation pattern of the input text aligns with the unique neural features of LGT identified in the training dataset. For the detection result $\mathcal{S}(\mathcal{T})$, we calculate the RepreScore$(\mathcal{T})$ and compare it to the optimal threshold θ. If RepreScore$(\mathcal{T})$ exceeds θ, the input text is more likely to be generated by the LLM:

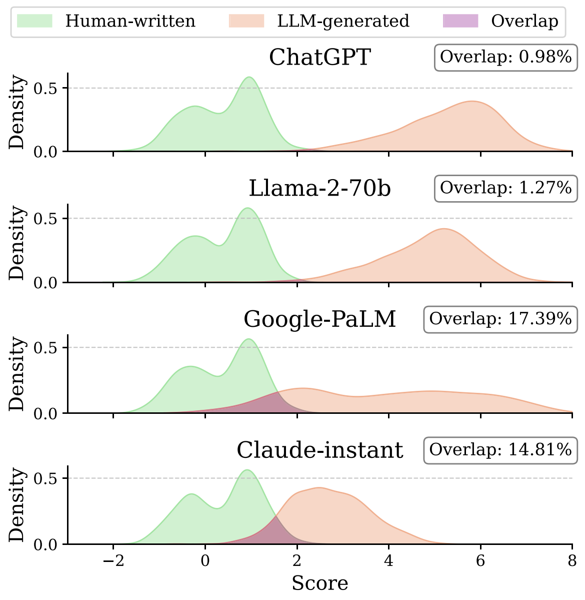

Figure 3: Comparison of RepreScores Distribution for Texts Generated by 4 Different LLMs (1000 Pairs LGT and HWT for Each LLM). The representation features used are modeling on Multi-LLMs generated texts. The overlap means the overlap probability.

$$\mathcal{S}(\mathcal{T}) = \begin{cases} \text{LGT} & \text{if RepreScore}(\mathcal{T}) > \theta \\ \text{HWT} & \text{otherwise} \end{cases} \tag{8}$$

Effectiveness and Generalization Our *RepreGuard* framework effectively detects LGT by extracting features derived from more fundamental and comprehensive representation information. This demonstrates a stronger and more adaptable detection capability. We further validate this capability on LGT generated by four different LLMs and their corresponding HWT. The results in Figure 3 clearly show a distinct separation between the RepreScore distributions of LGT and HWT, with a consistent trend across different LLMs. Specifically, the RepreScore for HWT primarily ranges from -2 to 2. Despite variations in text distributions generated by different LLMs, the RepreScore for LGT consistently falls between 0 and 8. This indicates a universal threshold that can be effectively applied across text generated by various types of LLMs, eliminating the need for individual adjustments and highlighting RepreGuard's strong generalization capability.

4 Experiment

4.1 Experiment Setup

Dataset We utilized the DetectRL benchmark (Wu et al., 2024), a dataset specifically designed to evaluate the detection capabilities of HWT and LGT in scenarios closely aligned with real-world applications. The dataset comprises data from four domains that are more prone to misuse: academic writing (ArXiv Archive[2]), news writing (XSum [Narayan et al., 2018]), creative writing (Writing Prompts [Fan et al., 2018]), and social media texts (Yelp Review [Zhang et al., 2015]). For each domain, the dataset includes 2,800 pairs of LGT and HWT samples, with LGT generated using four widely used LLMs: ChatGPT (OpenAI, 2022), Claude-instant (Anthropic, 2023), Google-PaLM (Ghahramani, 2023), and Llama-2-70B.[3]

To ensure robust evaluation, we applied the bootstrapping method five times to the dataset of each LLM, sampling a training set consisting of 512 pairs of samples and a test set consisting of 1,000 pairs of samples for each iteration. Additionally, to construct the multi-LLM dataset, we combined the data of four different LLMs and applied the bootstrap method five times equally on the training sets of different LLMs.

Baselines We compare RepreGuard with both fine-tuning-based and statistics-based methods to detect LGT:

- ***RoBERTa-based classifier*** (Liu et al., 2019): A supervised approach by fine-tuning pre-training language model (PLM) as a classifier. We use the RoBERTa-base consistent with the OpenAI detector (Solaiman et al., 2019) for training.
- ***LRR*** (Su et al., 2023): A statistics-based method based on the ratio of Log-likelihood and Log-rank. We uses GPT-neo-2.7B for scoring following Bao et al. (2024) experiments.
- ***DetectGPT*** (Mitchell et al., 2023): A statistics-based zero-shot method based on the Log-probability curvature of the perturbed text. We use T5-small (Raffel et al., 2020) to perturb text, and use GPT-Neo-2.7B (Black et al., 2021) for scoring following Bao et al. (2024).
- ***Fast-DetectGPT*** (Fast-Detect.; Bao et al. 2024): A statistics-based zero-shot method that using a sampling strategy to replace the perturbation strategy of DetectGPT. We use GPT-Neo-2.7B (Black et al., 2021) for scoring and GPT-J-6B (Wang and Komatsuzaki, 2021) for sample generation following the best setting reported.
- ***Straight-forward Detector*** (Str-Detect.): A zero-shot approach that directly asks LLM for both HWT and LGT. This detector is for better comparison with our method using the intrinsic mechanism of LLMs.
- ***Binoculars*** (Hans et al., 2024): A statistics-based zero-shot method that uses a pair of LLMs to calculate the ratio of perplexity and cross-perplexity for detection. We use Falcon-7B and Falcon-7B-Instruct (Penedo et al., 2023) for detection following the best setting reported.

Metrics Following Mitchell et al. (2023), we utilized the AUROC to evaluate the performance of the detector as a binary classification model. In the LGT detection task, the most worrying harm usually comes from false positives, that is, HWT is incorrectly labeled as LGT. Therefore, we also focus on the TPR at low FPR and follow the experimental setting of Hans et al. (2024) to adopt a standard FPR threshold of 0.01% TPR (TPR@0.01).

ID and OOD Detection Setting Our study examines both the performance of ID and in a strict zero-shot detection scenario, where we use a purely ‘‘OOD’’ setting to distinguish LGT from HWT, consistent with the ‘‘out-of-domain’’ claims in Binoculars (Hans et al., 2024) and Ghostbuster (Verma et al., 2024). In fact, many previous zero-shot works such as DetectGPT (Mitchell et al., 2023) are in different experimental settings for zero-shot detection, where they use the test set to define the threshold and get the best performance on the test set. We argue this experimental setting may limit the development of truly effective statistics-based zero-shot detectors. Therefore, our experimental setting is strictly aligned with the real-world scenario to ensure that

[2]`https://www.kaggle.com/datasets/spsayakpaul/arxiv-paper-abstracts/data.`

[3]`https://huggingface.co/meta-llama/Llama-2-70b-chat-hf.`

Test→	**Detector↓**	ChatGPT		Llama-2-70b		Google-PaLM		Claude-instant		Avg.	
Train↓	**Metrics→**	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*
ChatGPT	Roberta	$\underline{98.38}_{\pm0.32}$	$\underline{90.52}_{\pm1.93}$	$81.71_{\pm2.27}$	$54.64_{\pm16.57}$	$74.56_{\pm0.89}$	$20.20_{\pm12.87}$	$\underline{66.74}_{\pm1.49}$	$\underline{22.36}_{\pm13.26}$	$80.35_{\pm1.24}$	$46.93_{\pm11.16}$
	LRR	$92.61_{\pm0.39}$	$26.20_{\pm0.00}$	$95.84_{\pm0.34}$	$86.20_{\pm0.00}$	$81.98_{\pm0.14}$	$39.80_{\pm0.00}$	$57.82_{\pm0.80}$	$0.10_{\pm0.00}$	$82.06_{\pm0.42}$	$38.07_{\pm0.00}$
	DetectGPT	$54.87_{\pm0.25}$	$0.11_{\pm0.14}$	$59.21_{\pm0.53}$	$0.66_{\pm1.77}$	$55.29_{\pm0.37}$	$0.08_{\pm0.06}$	$55.92_{\pm0.61}$	$0.00_{\pm0.00}$	$56.32_{\pm0.44}$	$0.21_{\pm0.49}$
	Fast-Detect.	$75.65_{\pm0.28}$	$11.60_{\pm0.00}$	$85.49_{\pm0.31}$	$2.50_{\pm0.00}$	$80.36_{\pm0.63}$	$17.70_{\pm0.00}$	$47.29_{\pm0.44}$	$0.00_{\pm0.00}$	$72.20_{\pm0.42}$	$7.95_{\pm0.00}$
	Str-Detect.	$52.05_{\pm0.00}$	$0.01_{\pm0.00}$	$52.30_{\pm0.00}$	$0.01_{\pm0.00}$	$56.05_{\pm0.00}$	$0.01_{\pm0.00}$	$57.50_{\pm0.00}$	$0.01_{\pm0.00}$	$54.47_{\pm0.00}$	$0.01_{\pm0.00}$
	Binoculars	$97.36_{\pm0.52}$	$40.70_{\pm0.00}$	$\underline{99.45}_{\pm0.44}$	$\underline{98.70}_{\pm0.00}$	$\mathbf{98.03}_{\pm0.34}$	$\mathbf{89.80}_{\pm0.00}$	$61.86_{\pm3.64}$	$3.40_{\pm0.00}$	$\underline{89.18}_{\pm1.24}$	$\underline{58.15}_{\pm0.00}$
	RepreGuard	$\mathbf{99.84}_{\pm0.12}$	$\mathbf{100.00}_{\pm0.00}$	$\mathbf{99.55}_{\pm0.12}$	$\mathbf{99.26}_{\pm0.11}$	$\underline{88.67}_{\pm0.65}$	$\underline{64.66}_{\pm1.26}$	$\mathbf{85.00}_{\pm1.40}$	$\mathbf{56.12}_{\pm2.20}$	$\mathbf{93.26}_{\pm0.57}$	$\mathbf{80.01}_{\pm0.89}$
Llama-2-70b	Roberta	$95.49_{\pm0.96}$	$71.16_{\pm4.59}$	$94.21_{\pm2.06}$	$63.66_{\pm10.40}$	$86.00_{\pm2.17}$	$43.60_{\pm2.56}$	$\underline{76.98}_{\pm4.04}$	$\underline{22.88}_{\pm5.13}$	$\underline{88.17}_{\pm2.31}$	$50.32_{\pm5.67}$
	LRR	$91.88_{\pm1.05}$	$26.20_{\pm0.00}$	$96.47_{\pm0.52}$	$86.20_{\pm0.00}$	$81.65_{\pm0.55}$	$39.80_{\pm0.00}$	$56.20_{\pm1.77}$	$0.10_{\pm0.00}$	$81.55_{\pm0.97}$	$38.07_{\pm0.00}$
	DetectGPT	$54.33_{\pm0.81}$	$0.51_{\pm0.80}$	$59.36_{\pm0.86}$	$1.28_{\pm2.18}$	$55.02_{\pm0.98}$	$0.02_{\pm0.06}$	$56.11_{\pm0.24}$	$0.00_{\pm0.00}$	$56.20_{\pm0.72}$	$0.45_{\pm0.76}$
	Fast-Detect.	$94.08_{\pm1.26}$	$71.90_{\pm0.00}$	$98.76_{\pm0.05}$	$94.20_{\pm0.00}$	$92.39_{\pm0.28}$	$81.80_{\pm0.00}$	$51.45_{\pm0.62}$	$0.40_{\pm0.00}$	$84.17_{\pm0.55}$	$62.08_{\pm0.00}$
	Str-Detect.	$52.05_{\pm0.00}$	$0.01_{\pm0.00}$	$52.30_{\pm0.00}$	$0.01_{\pm0.00}$	$56.05_{\pm0.00}$	$0.01_{\pm0.00}$	$57.50_{\pm0.00}$	$0.01_{\pm0.00}$	$54.47_{\pm0.00}$	$0.01_{\pm0.00}$
	Binoculars	$97.94_{\pm0.74}$	$\underline{85.90}_{\pm0.00}$	$\mathbf{99.63}_{\pm0.06}$	$\underline{98.70}_{\pm0.00}$	$\mathbf{97.23}_{\pm0.93}$	$\mathbf{89.80}_{\pm0.00}$	$57.49_{\pm3.57}$	$3.40_{\pm0.00}$	$88.07_{\pm1.32}$	$\underline{69.45}_{\pm0.00}$
	RepreGuard	$\mathbf{99.54}_{\pm0.08}$	$\mathbf{99.38}_{\pm0.14}$	$\underline{99.38}_{\pm0.18}$	$\mathbf{98.84}_{\pm0.11}$	$\underline{88.84}_{\pm1.28}$	$\underline{77.08}_{\pm0.45}$	$\mathbf{84.08}_{\pm3.52}$	$\mathbf{60.66}_{\pm1.66}$	$\mathbf{92.96}_{\pm1.27}$	$\mathbf{83.99}_{\pm0.59}$
Google-PaLM	Roberta	$88.72_{\pm1.35}$	$40.94_{\pm5.18}$	$85.36_{\pm5.00}$	$32.70_{\pm6.15}$	$82.09_{\pm3.99}$	$36.52_{\pm4.64}$	$\underline{72.98}_{\pm4.00}$	$\underline{21.54}_{\pm8.64}$	$82.29_{\pm3.58}$	$32.92_{\pm6.15}$
	LRR	$91.40_{\pm2.01}$	$26.20_{\pm0.00}$	$92.84_{\pm2.78}$	$86.20_{\pm0.00}$	$83.13_{\pm1.53}$	$39.80_{\pm0.00}$	$61.11_{\pm2.05}$	$0.10_{\pm0.00}$	$82.12_{\pm2.09}$	$38.07_{\pm0.00}$
	DetectGPT	$54.04_{\pm0.32}$	$0.00_{\pm0.00}$	$59.21_{\pm0.22}$	$0.00_{\pm0.00}$	$55.30_{\pm0.35}$	$0.02_{\pm0.06}$	$55.53_{\pm0.46}$	$0.96_{\pm1.63}$	$56.02_{\pm0.34}$	$0.24_{\pm0.42}$
	Fast-Detect.	$74.62_{\pm0.99}$	$11.60_{\pm0.00}$	$86.47_{\pm0.50}$	$2.50_{\pm0.00}$	$80.64_{\pm0.22}$	$17.70_{\pm0.00}$	$48.46_{\pm0.47}$	$0.00_{\pm0.00}$	$72.55_{\pm0.54}$	$7.95_{\pm0.00}$
	Str-Detect.	$52.05_{\pm0.00}$	$0.01_{\pm0.00}$	$52.30_{\pm0.00}$	$0.01_{\pm0.00}$	$56.05_{\pm0.00}$	$0.01_{\pm0.00}$	$57.50_{\pm0.00}$	$0.01_{\pm0.00}$	$54.47_{\pm0.00}$	$0.01_{\pm0.00}$
	Binoculars	$\mathbf{98.56}\pm0.35$	$\underline{85.90}_{\pm0.00}$	$\mathbf{99.58}\pm0.20$	$\underline{98.70}_{\pm0.00}$	$\mathbf{97.98}\pm0.38$	$\mathbf{89.80}_{\pm0.00}$	$61.15_{\pm2.49}$	$3.40_{\pm0.00}$	$\underline{89.32}_{\pm0.86}$	$\underline{69.45}_{\pm0.00}$
	RepreGuard	$\underline{98.39}_{\pm0.31}$	$\mathbf{99.36}_{\pm0.07}$	$\underline{98.53}_{\pm0.28}$	$\mathbf{99.16}_{\pm0.07}$	$\underline{93.36}_{\pm0.27}$	$\underline{79.88}_{\pm3.43}$	$\mathbf{90.57}_{\pm1.06}$	$\mathbf{56.90}_{\pm2.20}$	$\mathbf{95.21}_{\pm0.48}$	$\mathbf{83.82}_{\pm1.44}$
Claude-instant	Roberta	$88.63_{\pm1.34}$	$28.50_{\pm11.74}$	$77.45_{\pm4.73}$	$23.70_{\pm10.19}$	$74.90_{\pm1.76}$	$14.58_{\pm13.14}$	$\underline{88.07}_{\pm3.89}$	$\underline{36.56}_{\pm4.83}$	$82.26_{\pm2.93}$	$25.84_{\pm9.98}$
	LRR	$86.33_{\pm4.23}$	$26.20_{\pm0.00}$	$87.46_{\pm4.43}$	$86.20_{\pm0.00}$	$81.19_{\pm3.34}$	$39.80_{\pm0.00}$	$63.74_{\pm1.07}$	$0.10_{\pm0.00}$	$79.68_{\pm3.27}$	$38.07_{\pm0.00}$
	DetectGPT	$53.95_{\pm0.84}$	$0.02_{\pm0.06}$	$57.90_{\pm1.09}$	$0.00_{\pm0.00}$	$54.33_{\pm1.13}$	$0.06_{\pm0.07}$	$55.97_{\pm0.38}$	$0.00_{\pm0.00}$	$55.54_{\pm0.86}$	$0.02_{\pm0.03}$
	Fast-Detect.	$81.27_{\pm5.40}$	$71.90_{\pm0.00}$	$82.35_{\pm4.48}$	$94.20_{\pm0.00}$	$80.85_{\pm4.70}$	$81.80_{\pm0.00}$	$61.35_{\pm0.35}$	$0.40_{\pm0.00}$	$76.46_{\pm3.73}$	$62.08_{\pm0.00}$
	Str-Detect.	$52.05_{\pm0.00}$	$0.01_{\pm0.00}$	$52.30_{\pm0.00}$	$0.01_{\pm0.00}$	$56.05_{\pm0.00}$	$0.01_{\pm0.00}$	$57.50_{\pm0.00}$	$0.01_{\pm0.00}$	$54.47_{\pm0.00}$	$0.01_{\pm0.00}$
	Binoculars	$\underline{92.36}_{\pm1.95}$	$\underline{85.90}_{\pm0.00}$	$\underline{93.18}_{\pm1.01}$	$\mathbf{98.70}_{\pm0.00}$	$\mathbf{92.83}_{\pm1.18}$	$\mathbf{89.80}_{\pm0.00}$	$73.92_{\pm0.17}$	$3.40_{\pm0.00}$	$\underline{88.07}_{\pm1.08}$	$\underline{69.45}_{\pm0.00}$
	RepreGuard	$\mathbf{97.20}_{\pm1.39}$	$\mathbf{96.92}_{\pm0.49}$	$\mathbf{97.51}_{\pm0.99}$	$\underline{97.16}_{\pm0.40}$	$\underline{87.77}_{\pm2.37}$	$\underline{63.04}_{\pm0.73}$	$\mathbf{92.76}_{\pm0.49}$	$\mathbf{56.22}_{\pm5.22}$	$\mathbf{93.81}_{\pm1.31}$	$\mathbf{78.34}_{\pm1.71}$
Multi-LLMs	Roberta	$92.13_{\pm2.80}$	$60.42_{\pm10.23}$	$87.84_{\pm4.06}$	$45.90_{\pm15.30}$	$82.07_{\pm3.28}$	$31.16_{\pm5.57}$	$\underline{82.99}_{\pm5.42}$	$\underline{25.64}_{\pm4.52}$	$86.26_{\pm3.89}$	$40.78_{\pm8.91}$
	LRR	$92.78_{\pm0.27}$	$26.20_{\pm0.00}$	$94.98_{\pm0.27}$	$86.20_{\pm0.00}$	$81.94_{\pm0.05}$	$39.80_{\pm0.00}$	$59.93_{\pm0.41}$	$0.10_{\pm0.00}$	$82.41_{\pm0.25}$	$38.07_{\pm0.00}$
	DetectGPT	$54.92_{\pm0.13}$	$0.11_{\pm0.14}$	$59.05_{\pm0.22}$	$0.02_{\pm0.06}$	$55.29_{\pm0.37}$	$0.08_{\pm0.06}$	$55.98_{\pm0.35}$	$0.00_{\pm0.00}$	$56.31_{\pm0.27}$	$0.05_{\pm0.06}$
	Fast-Detect.	$94.98_{\pm1.72}$	$71.90_{\pm0.00}$	$95.97_{\pm2.65}$	$94.20_{\pm0.00}$	$93.19_{\pm1.48}$	$81.80_{\pm0.00}$	$56.57_{\pm3.33}$	$0.40_{\pm0.00}$	$85.18_{\pm2.30}$	$62.08_{\pm0.00}$
	Str-Detect.	$52.05_{\pm0.00}$	$0.01_{\pm0.00}$	$52.30_{\pm0.00}$	$0.01_{\pm0.00}$	$56.05_{\pm0.00}$	$0.01_{\pm0.00}$	$57.50_{\pm0.00}$	$0.01_{\pm0.00}$	$54.47_{\pm0.00}$	$0.01_{\pm0.00}$
	Binoculars	$\underline{97.95}_{\pm0.89}$	$\underline{85.90}_{\pm0.00}$	$\underline{98.05}_{\pm1.17}$	$\underline{98.70}_{\pm0.00}$	$\mathbf{97.58}_{\pm0.70}$	$\mathbf{89.80}_{\pm0.00}$	$68.59_{\pm3.55}$	$3.40_{\pm0.00}$	$\underline{90.54}_{\pm1.58}$	$\underline{69.45}_{\pm0.00}$
	RepreGuard	$\mathbf{98.00}_{\pm0.43}$	$\mathbf{99.26}_{\pm0.11}$	$\mathbf{98.44}_{\pm0.24}$	$\mathbf{99.20}_{\pm0.12}$	$\underline{92.35}_{\pm0.46}$	$\underline{74.74}_{\pm4.52}$	$\mathbf{93.40}_{\pm0.74}$	$\mathbf{56.52}_{\pm1.04}$	$\mathbf{95.55}_{\pm0.47}$	$\mathbf{82.43}_{\pm1.45}$

Table 1: ID and OOD Performance Comparison of Detection Algorithms on Different Train and Eval Settings. RepreGuard shows the strongest detection performance on all settings, with an average of $96.34_{\pm0.27}\%$ and $93.49_{\pm1.13}\%$ AUROC (*AUR.*), and $83.74_{\pm1.56}\%$ and $81.13_{\pm2.11}\%$ TPR@0.01 (*TPR.*) in ID and OOD respectively. We conduct 5 rounds of bootstrapping and report the mean, standard deviation, and 95% confidence interval. Here, the subscript represents the standard deviation (e.g., 99.84 ± 0.12 indicates a mean value of 99.84 with a standard deviation of 0.12). The blue background or **bold** indicates the best performance and the grey background or underline indicates the second best.

the detector is robust and not overly dependent on any particular dataset. We use the training data to make the decision thresholds to detect text generated by unknown LLMs.

4.2 ID and OOD Performance

Our experiments evaluate both ID and OOD performance of the detectors, as shown in Table 1. The results demonstrate that RepreGuard achieves the best overall performance, excelling in both ID and OOD scenarios, achieving an average of $94.92_{\pm0.70}\%$ and $82.44_{\pm1.84}\%$ in AUROC and TPR@0.01, respectively.

ID Performance ID performance refers to the detectors' ability to detect instances within the same distribution as the training dataset. The results show that RepreGuard outperforms other detectors in both AUROC and TPR@0.01 in ID setting. Specifically, it achieves $96.34_{\pm0.27}\%$ AUROC and $83.74_{\pm1.56}\%$ TPR@0.01, demonstrating its strength in detecting data from the same distribution as the training datasets. Binoculars is the second-best performing method in most settings, particularly in the AUROC metric, with an average AUROC of 92.16% while its TPR@0.01 is only achieving an average of 58.15%. Fine-tuning classifiers based on RoBERTa generally perform well to some extent, especially in terms of AUROC, with values rarely approaching RepreGuard (e.g., $98.38_{\pm0.32}\%$ AUROC on ChatGPT generated text), while its overall performance can be unstable and may experience significant drops (e.g., $82.09_{\pm3.99}\%$ AUROC on on Google-PaLM generated text), and its average TPR@0.01 is $56.82_{\pm5.45}\%$. Additionally, LRR, DetectGPT, and Fast-Detect perform poorly in TPR@0.01 across most datasets, suggesting they struggle in achieving accurate LGT detection within the distribution.

Train→	ChatGPT		Llama-2-70b		Google-PaLM		Claude-instant		Multi-LLMs		Avg.	
Surrogate Model↓ Metrics→	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*
LLama-3.1-8B	93.20	**79.70**	94.35	**81.80**	95.05	84.80	94.80	**77.10**	96.30	81.20	**94.74**	**80.92**
LLama-3.1-8B-Instruct	90.20	77.10	93.85	76.50	93.40	79.40	94.75	63.10	95.75	77.20	93.59	74.66
Llama-3-8B	92.45	69.70	90.70	70.30	94.35	73.50	93.70	65.70	94.95	69.70	93.23	69.78
Llama-3-8B-Instruct	91.45	65.10	92.30	56.80	95.05	69.10	94.60	57.60	95.45	65.30	93.77	62.78
Llama-2-7B	87.85	66.20	92.30	78.70	**95.40**	**85.20**	94.45	65.70	**96.85**	**85.20**	93.37	76.20
Llama-2-7B-Instruct	90.60	65.70	90.70	80.10	92.10	45.90	96.10	57.50	95.95	84.00	93.09	66.64
Mistral-7B	85.95	61.80	85.35	46.80	78.70	27.80	88.65	63.90	85.85	47.80	84.90	49.62
Mistral-7B-Instruct	**93.85**	55.80	92.35	40.70	89.05	32.90	95.25	69.80	93.35	45.40	92.77	48.92
Falcon-7B	67.55	2.30	86.50	46.00	89.70	50.50	89.80	27.90	69.15	2.10	80.54	25.76
Falcon-7B-Instruct	74.60	0.00	92.20	65.40	90.05	67.10	92.45	76.20	72.60	0.00	84.38	41.74
Gemma-7B	78.50	2.90	78.20	0.60	77.70	0.50	75.30	0.00	77.55	0.00	77.45	0.80
Gemma-7B-Instruct	86.50	69.60	90.95	68.40	78.95	18.10	74.70	3.40	81.70	32.00	82.56	38.30
Gemma-2B	86.55	70.20	90.80	74.60	89.80	64.80	91.40	36.90	92.40	72.50	90.19	63.80
Gemma-2B-Instruct	93.35	75.30	**94.70**	81.10	91.35	72.10	90.00	65.80	94.35	76.70	92.75	74.20
Phi-3-Mini-4K-Instruct	85.35	36.10	89.90	23.00	89.40	2.10	89.30	1.00	91.85	3.70	89.16	13.18
Phi-2	92.80	75.80	93.55	69.20	94.85	68.40	**96.10**	54.50	93.40	58.80	94.14	65.34
GPT-J-6B	83.10	9.50	89.45	64.80	89.75	55.80	86.95	24.40	85.30	15.50	86.91	34.00
GPT-Neo-2.7B	48.90	0.10	50.00	0.00	50.00	0.10	50.00	0.10	49.75	0.00	49.73	0.06

Table 2: Performance Comparison of RepreGuard Using Different Surrogate Models on 1000 ‘‘HWT-LGT’’ Pairs from 4 Different LLMs. The blue background or **bold** indicates the best performance and the grey background or underline indicates the second best.

OOD Performance OOD performance measures how well a model detects unseen data that differs from the training set. RepreGuard demonstrates exceptional performance in OOD scenarios, with significantly smaller drops in both AUROC and TPR@0.01 compared to other detectors. This is particularly evident in the testing on Claude-instant. For instance, when trained on Google-PaLM and tested on Claude-instant, RepreGuard achieves an AUROC of $90.57_{\pm 1.06}\%$ and a TPR@0.01 of $56.22_{\pm 5.22}\%$. In contrast, Binoculars experiences a drop to an AUROC of $61.15_{\pm 2.49}\%$ and a TPR@0.01 of $3.40_{\pm 0.00}\%$, while the RoBERTa classifier drops to an AUROC of $72.98_{\pm 4.00}\%$ and a TPR@0.01 of $21.56_{\pm 8.64}\%$. Notably, Binoculars performs particularly well on the Google-PaLM test set, highlighting its preference for certain models. Furthermore, although the RoBERTa-based classifier demonstrates relatively high AUROC performance in certain scenarios, such as achieving $95.49_{\pm 0.96}\%$ when trained on Llama-2-70b and tested on ChatGPT, its overall performance appears relatively unstable. The combined AUROC and TPR@0.01 are only $81.27_{\pm 1.33}\%$ and $41.36_{\pm 3.87}\%$ respectively, highlighting the instability of its performance across different scenarios.

4.3 Ablation Study

Surrogate Model We evaluated the performance of RepreGuard in detection using surrogate models of different sizes and structures on 1000 random sampled ‘‘HWT-LGT’’ pairs from 4 different LLMs. The results in Table 2 show that LLama-3.1-8B achieved the best performance with 94.74% AUROC and 80.92% TPR@0.01, and was chosen as the example surrogate model in our paper. We find that LLMs with large sizes (7B or above), such as Llama-2-7B and Mistral-7B, consistently performed well. However, smaller models do not necessarily mean poor performance. For instance, phi-2 (2.7B) and Gemma-2B-Instruct outperformed most 7B models, achieving an AUROC of 94.14% and 92.75%, suggesting that our approach can be effectively deployed even with limited computational resources. Additionally, instruction-tuned models generally performed better than their non-instruction version, though this is not always the case (e.g., LLama-3.1-8B). Therefore, the performance of surrogate models may relate to their architecture and training data, requiring proper evaluation to determine model effectiveness.

Activation Token Ratio The activation token ratio refers to the proportion of tokens selected from end to the beginning in the samples used to collect neural activation representations for HWT and LGT. Each token’s representation is influenced by all preceding tokens in the text, making this parameter critical for optimizing LGT detection. By focusing on the tokens with the most informative representations, the activation token

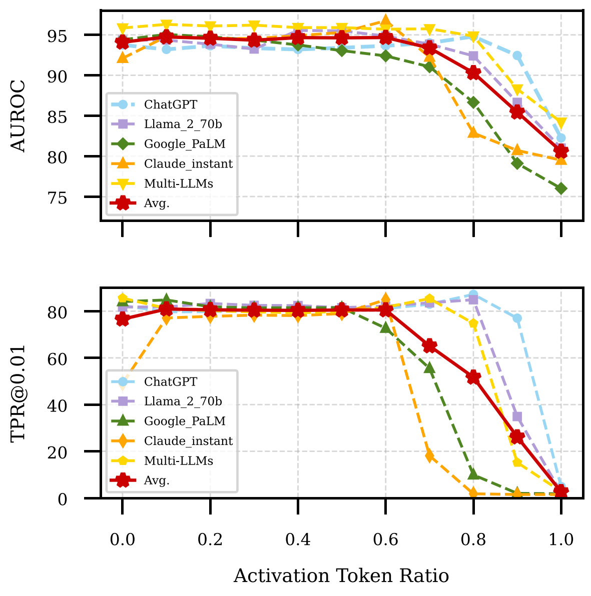

Figure 4: Effect of Activation Token Ratio in Terms of AUROC and TPR@0.01 (%) by 4 Different LLMs (250 ‘‘HWT-LGT’’ Pairs for Each LLM). The model name in the legend refers to the model-generated text used for representation features modeling and threshold setting.

ratio ensures a better balance between signal and noise. The results in Figure 4 indicate that the average AUROC and TPR@0.01 remain relatively stable as the activation token ratio increases, peaking at a ratio of 0.1, after which they stay consistent until approximately 0.6, followed by a sharp decline. This pattern suggests that not all token positions contribute equally to feature modeling. Specifically, the tokens at the end of the text are more distinctive in differentiating between LGT and HWT texts. As we extend toward the beginning of the sentence, noise gradually increases, leading to a decline in performance. Thus, an optimal activation token ratio is essential for balancing the modelling of important detection features while minimizing noise.

Shots of Training Dataset We evaluated the impact of the number of samples used to set the detection threshold, as shown in Table 3. The results indicate that RepreGuard is significantly more effective with limited data compared to other detectors. Specifically, RepreGuard demonstrates optimal performance in the 16-shot setting across most training datasets, achieving an average AUROC of 90.21% and TPR@0.01 of 77.36%, surpassing the best statistics-based baseline Binoculars by 6.61% and 4.76%, respectively. In addition, while the RoBERTa-based classifier achieved an average AUROC of 91.77%, which is lower than RepreGuard’s average AUROC of 94.79% in the 1024-shot setting, its TPR@0.01 is significantly lower, averaging only 59.1%. This indicates that the RoBERTa-based classifier struggles to reliably identify positive cases under stringent false positive constraints, highlighting a limitation in its ability to balance sensitivity and specificity in such scenarios. In contrast, other detectors, such as LRR, DetectGPT and Fast-DetectGPT, demonstrate unstable performance, and consistently fail to exceed a 90% AUROC even with 1024-shot settings. These findings emphasize the strong and robust detection capabilities of RepreGuard, especially when limited data is available for training.

5 Reliability in the Wild

To evaluate our method in real-world scenarios, we conducted experiments from multiple perspectives, including performance in OOD domains, sensitivity to text length, robustness against paraphrase and perturbation attacks, and the impact of different sampling strategies.

5.1 Generalization on Domains

We evaluated our method on four domain datasets derived from different sources: ArXiv, XSum, Writing Prompt, and Yelp Review. As illustrated in Figure 5, most detectors exhibited poor performance in the OOD domain setting, especially with a low TPR@0.01. However, RepreGuard consistently achieves the highest average AUROC and TPR@0.01 on the OOD domain tasks, with 91.60% and 85.63%, respectively. Specifically, RepreGuard achieves the best performance in terms of AUROC and TPR@0.01 under the OOD domain settings for the Arxiv, Writing Prompt, and Yelp Review datasets. While its AUROC on the XSum dataset is slightly lower, RepreGuard still attains the highest TPR@0.01. In contrast, although the Roberta-based classifier, LRR, Fast-DetectGPT, and Binoculars demonstrate strong performance in terms of AUROC, their TPR@0.01 values are quite low. For instance, the second-best detector, Binocular, achieved an average AUROC of 88.82% but a TPR@0.01 of 76.21%. This indicates that these detectors struggle to identify positive samples when operating at

Shots→	**Detector↓**	16		32		64		128		256		512		1024	
Train↓	**Metrics→**	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*
ChatGPT	Roberta	60.55	0.40	42.05	0.00	64.10	0.00	68.05	0.90	84.30	5.90	86.40	48.00	88.55	44.70
	LRR	80.55	31.50	81.10	31.50	82.30	31.50	82.85	31.50	83.35	31.50	83.45	31.50	82.75	31.50
	DetectGPT	53.68	0.15	54.07	0.04	54.02	0.19	55.89	0.08	56.22	0.13	57.88	0.06	57.78	0.17
	Fast-Detect.	**84.80**	58.80	**84.85**	58.80	84.75	58.80	84.70	58.80	84.75	58.80	84.70	58.80	84.70	58.80
	Binoculars	84.70	72.60	84.70	72.60	88.00	72.60	88.00	72.60	90.25	72.60	88.45	72.60	90.15	72.60
	RepreGuard	83.20	**81.00**	82.55	**83.70**	**91.05**	**81.60**	**90.50**	**81.00**	**93.15**	**78.60**	**93.20**	**79.70**	**93.85**	**79.40**
Llama-2-70b	Roberta	52.55	0.20	52.15	0.20	41.65	0.00	60.45	0.00	83.85	27.20	85.85	30.30	**95.50**	73.40
	LRR	75.40	31.50	81.10	31.50	82.80	31.50	82.95	31.50	82.45	31.50	82.45	31.50	82.80	31.50
	DetectGPT	54.12	0.11	53.89	0.07	54.33	0.18	56.01	0.02	55.76	0.16	57.23	0.09	58.09	0.13
	Fast-Detect.	79.35	58.80	84.10	58.80	84.20	58.80	84.50	58.80	85.00	58.80	84.70	58.80	84.70	58.80
	Binoculars	83.00	72.60	83.00	72.60	86.10	72.60	86.10	72.60	87.25	72.60	87.25	72.60	87.25	72.60
	RepreGuard	**89.55**	**79.20**	**89.55**	**78.70**	**96.35**	**77.90**	**91.85**	**82.00**	**95.50**	**81.70**	**94.35**	**81.80**	94.05	**82.70**
Google-PaLM	Roberta	55.45	0.50	61.10	0.10	66.30	0.00	71.25	2.40	77.50	2.80	89.40	22.90	**95.45**	64.50
	LRR	78.95	31.50	81.15	31.50	81.35	31.50	82.15	31.50	82.30	31.50	83.35	31.50	83.20	31.50
	DetectGPT	54.12	0.08	54.45	0.19	54.89	0.12	55.76	0.04	56.34	0.17	57.21	0.06	57.95	0.15
	Fast-Detect.	76.20	58.80	83.50	58.80	85.35	58.80	85.35	58.80	85.25	58.80	85.25	58.80	85.30	58.80
	Binoculars	86.00	72.60	88.95	72.60	88.95	72.60	88.95	72.60	88.95	72.60	88.45	72.60	89.85	72.60
	RepreGuard	**91.70**	**76.30**	**92.00**	**80.40**	**94.25**	**85.30**	**94.95**	**85.10**	**95.05**	**85.50**	**95.05**	**84.80**	95.30	**82.60**
Claude-instant	Roberta	51.90	0.00	62.30	0.00	50.85	0.00	62.10	0.20	74.45	9.30	84.85	43.90	83.70	66.00
	LRR	82.50	31.50	82.15	31.50	83.50	31.50	72.75	31.50	79.95	31.50	79.00	31.50	79.05	31.50
	DetectGPT	53.71	0.14	54.15	0.06	54.09	0.19	55.92	0.03	56.30	0.17	57.95	0.09	57.82	0.12
	Fast-Detect.	79.45	58.80	79.45	58.80	80.80	58.80	80.55	58.80	78.85	58.80	80.45	58.80	79.60	58.80
	Binoculars	82.15	**72.60**	82.15	**72.60**	82.15	**72.60**	82.15	72.60	82.15	72.60	81.90	72.60	85.80	72.60
	RepreGuard	**92.40**	63.20	**93.25**	72.50	**94.55**	69.90	**94.60**	**76.30**	**95.40**	**77.00**	**94.80**	**77.10**	**94.75**	**77.30**
Multi-LLMs	Roberta	57.25	0.00	50.45	1.60	59.80	0.70	59.60	1.30	64.95	3.60	93.50	34.80	95.65	46.90
	LRR	78.25	31.50	78.25	31.50	83.30	31.50	81.90	31.50	82.35	31.50	83.50	31.50	83.50	31.50
	DetectGPT	55.48	0.04	54.56	0.02	54.43	0.15	54.97	0.18	55.93	0.01	56.31	0.03	57.11	0.07
	Fast-Detect.	84.80	58.80	84.80	58.80	85.35	58.80	85.20	58.80	85.35	58.80	85.30	58.80	85.30	58.80
	Binoculars	82.15	72.60	82.15	72.60	89.30	72.60	89.30	72.60	89.45	72.60	90.45	72.60	90.45	72.60
	RepreGuard	**94.20**	**87.10**	**95.85**	**86.90**	**95.70**	**81.90**	**94.20**	**79.90**	**95.80**	**79.80**	**96.30**	**81.20**	**96.00**	**80.20**

Table 3: Performance Comparison of RepreGuard on Various Training Data Shots in Terms of AUROC (%) on 1000 ‘‘HWT-LGT’’ Pairs from 4 Different LLMs. The blue background or **bold** indicates the best performance and the grey background or underline indicates the second best.

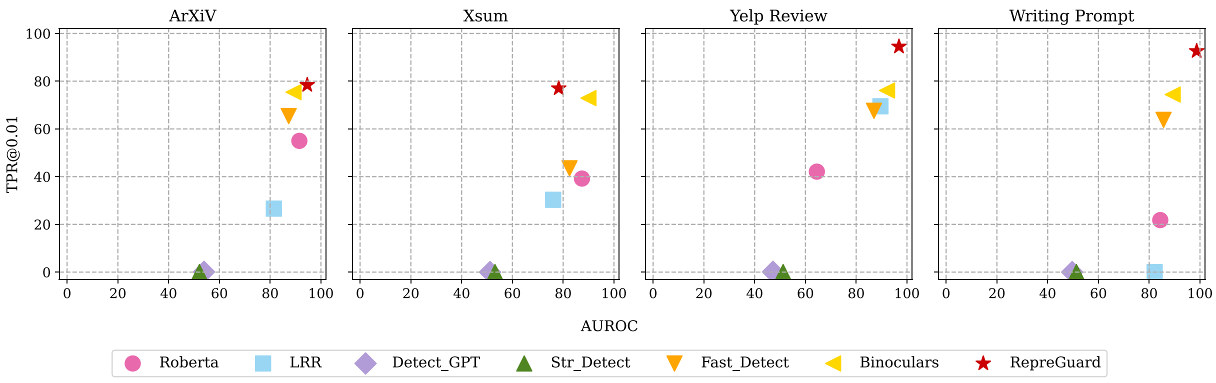

Figure 5: Performance Comparison of Various Detection Methods under OOD Domain Settings across Four Domain in Terms of AUROC (%) and TPR@0.01 (%) on a Test Set with 1000 ‘‘HWT-LGT’’ Pairs from 4 Different LLMs. The name of each subgraph corresponds to the test domain, while training is conducted on the other three domains. In each domain, the data consists of LGT from four LLMs.

extremely low false positive rates, resulting in a higher risk of misdetections.

5.2 Detecting Texts with Varied Sizes

We evaluate the impact of text size on the performance of our detector. The results are shown in Figure 6. Overall, RepreGuard achieved the best performance on both short and long texts. Specifically, it attained an AUROC of 84.22% and a TPR@0.01 of 57.74% on short texts (64 tokens), while achieving an AUROC of 92.94% and a TPR@0.01 of 81.70% on long texts (256 tokens). Although RepreGuard demonstrates slightly lower AUROC performance on 64-token texts compared to other detectors when trained on ChatGPT dataset, its TPR@0.01 consistently outperforms that of other detectors. Furthermore, as the text length increases, the performance advantage of RepreGuard becomes increasingly evident. On 256-token texts, its AUROC and TPR@0.01 significantly surpass those of other detectors,

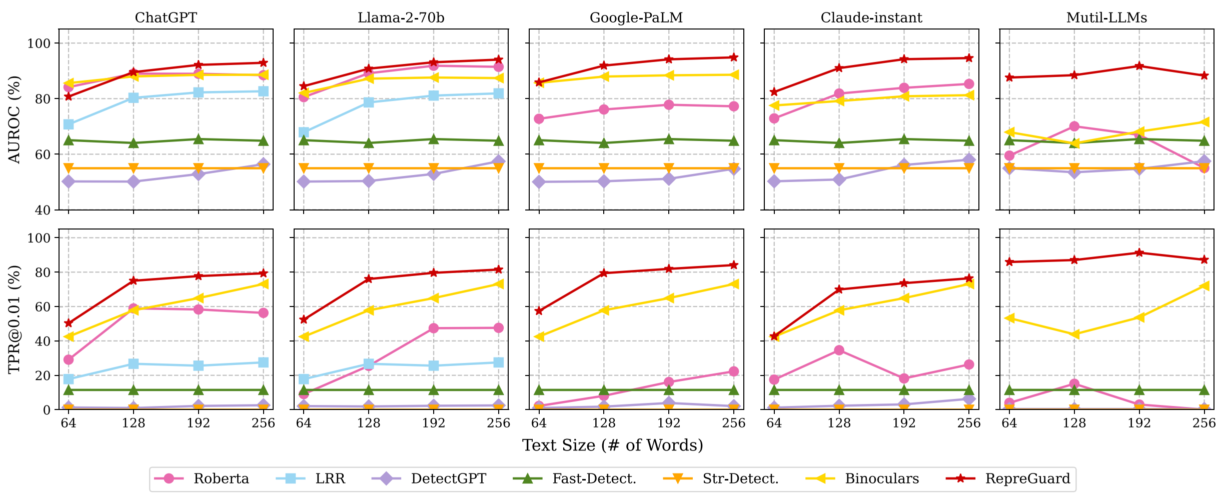

Figure 6: Performance Comparison of RepreGuard on Texts with Varied Sizes in Terms of AUROC (%) and TPR@0.01 (%) on a Test Set with 1000 ‘‘HWT-LGT’’ Pairs from 4 Different LLMs. The model name on each sub-graph refers to the LGT from different models used for representation features modeling and threshold setting.

showcasing its exceptional capability in handling long texts. This indicates that RepreGuard remains effective in accurately identifying HWT, minimizing the risk of misclassifying it as LGT, even with shorter text sizes.

5.3 Robustness on Paraphrase & Perturbation Attack

We also evaluate the robustness of RepreGuard on mainstream attack methods, including paraphrase attacks and adversarial perturbation attacks. In practical applications, humans often make semantically equivalent revisions to LGT in line with their preferences. In addition, humans might intentionally introduce adversarial noise into LGT to evade detection, creating challenges for the detector. We used DIPPER Paraphraser (Krishna et al., 2023) and TextBugger (Li et al., 2019) to simulate these realistic scenarios, respectively. The results presented in Figure 7 and Figure 11 (see Appendix A.5) demonstrate that RepreGuard is the most robust detection method against both paraphrase and perturbation attacks. Significantly, this phenomenon is particularly evident under perturbation attacks, where the AUROC and TPR@0.01 reach 89.65% and 88.63%, respectively, exceeding the second-best detector, Binocular, which achieves 69.45% and 58.54%. Additionally, although Roberta classifier performs well in certain aspects of AUROC, its TPR@0.01 is extremely poor, dropping as low as 0.10%, which highlights its significant limitations in identifying positive samples under strict thresholds. In contrast, while certain detectors exhibit strong performance on specific datasets and attacks, such as Fast-DetectGPT achieving an AUROC of 89.70% on the Google-PaLM dataset under the perturbation attack, and LRR attaining an AUROC of 83.80% in paragraph attacks on the ChatGPT dataset, their overall performance remains poor, indicating that their robustness is significantly compromised.

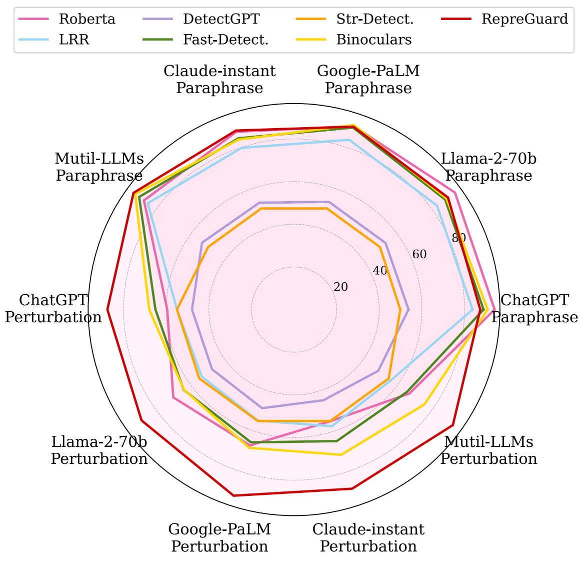

Figure 7: Performance Comparison of RepreGuard on Paraphrase and Perturbation Attack in Terms of AUROC on 1000 ‘‘HWT-LGT’’ Pairs from 4 Different LLMs. The raw text generated by each model is used to model representation features and set thresholds.

	Chat Models (llama-chat, mistral-chat, mpt-chat)								Non-Chat Models (mistral, mpt, gpt2)								Avg.	
Dec. Strategy	greedy				sampling				greedy				sampling					
Rep. Penalty?	✗		✓		✗		✓		✗		✓		✗		✓			
Metrics	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*	*AUR.*	*TPR.*
Roberta	88.32	65.97	83.58	41.02	88.57	46.81	71.66	13.27	93.11	40.22	74.45	14.87	80.24	<u>34.03</u>	<u>77.45</u>	<u>5.69</u>	82.17	32.74
LRR	90.17	49.00	80.23	12.69	86.13	24.25	67.17	4.19	94.86	87.43	83.43	34.33	77.25	0.40	50.00	0.20	78.66	26.56
Fast-Detect.	97.80	95.21	87.56	68.03	97.06	91.12	71.86	30.04	<u>98.65</u>	<u>96.21</u>	77.50	<u>48.10</u>	<u>84.03</u>	31.44	50.00	0.00	83.06	<u>57.52</u>
Str-Detect.	55.99	0.01	55.14	0.01	55.24	0.01	55.34	0.01	56.94	0.01	53.84	0.01	56.64	0.01	55.69	0.01	56.98	0.01
Binoculars	**99.50**	**98.70**	<u>91.52</u>	<u>71.26</u>	**99.15**	<u>94.41</u>	<u>77.69</u>	<u>31.24</u>	**99.50**	**99.30**	<u>79.54</u>	33.43	**88.12**	1.70	50.05	0.00	<u>85.63</u>	53.76
RepreGuard	<u>98.30</u>	<u>96.61</u>	**97.16**	**94.81**	<u>97.55</u>	**94.61**	**94.86**	**85.73**	98.50	92.22	**92.47**	**75.55**	72.55	**34.43**	**81.99**	**46.31**	**92.05**	**77.53**

Table 4: AUROC and TPR@0.01 for All Detectors Across Model Groups and Sampling Strategies. Sampling with a repetition penalty consistently makes most detectors difficult to detect, while RepreGuard maintains the best performance. The **Bold** indicates the best performance and <u>underline</u> indicates the second best.

5.4 Various Sampling Methods

Holtzman et al. (2020) pointed out that sampling strategies with maximum likelihood (such as beam search) often lead to text degeneration. Nucleus sampling addresses these issues by dynamically truncating the long tail of the probability distribution and sampling only from the ‘‘nucleus’’. This approach effectively avoids degeneration, producing higher-quality and more diverse text, thereby making LGT closer to HWT. To investigate whether different sampling strategies would impact RepreGuard, we utilized the RAID dataset (Dugan et al., 2024) to evaluate the robustness of various sampling strategies. This dataset encompasses multiple domains and generative models and was constructed using diverse sampling approaches. Following the RAID setting, we also divided the data into Chat Models and Non-Chat Models and evaluated the AUROC metric. The results on Table 4 demonstrate that RepreGuard achieves the best performance across both models under the different sampling strategies, with an average of 6.42% in AUROC and 23.77% in TPR@0.01 higher than Binoculars. Noteworthily, most detectors, like LRR, Fast-Detect, and Binoculars, perform well when the repetition penalty mechanism is disabled. However, their AUROC showed a significant decline under the setting of the repetition penalty, whereas RepreGuard demonstrates strong robustness, with its performance only slightly decreasing and even improving on non-chat models. In contrast, most detectors experience a significant performance drop after enabling the penalty mechanism, especially in the sampling scenario of Non-Chat Models, where their detection capability almost completely deteriorates (AUROC approaching 50%). These indicate that RepreGuard can effectively detect the diversity and complexity of LGT by capturing internal representations, whereas LRR, Fast-Detect, and Binoculars only capture information based on output probabilities, leading to the uncertainty introduced by different sampling strategies.

Detector ↓	*AUR.*	*TPR.*	Cost of Space	Cost of Time (Per sample)
Roberta	84.85	43.90	**2.0GB**	**0.016s**
Fast-Detect.	80.45	58.80	40.0GB	0.390s
Binocular	81.90	<u>72.60</u>	58.0GB	0.653s
RepreGuard (Phi-2)	**96.10**	54.50	<u>16.0GB</u>	<u>0.072s</u>
RepreGuard (Llama-3.1-8B)	<u>94.80</u>	**77.10**	38.0GB	0.359s

Table 5: Comparison of Effectiveness and Resource Costs on A Test Set with 1,000 ‘‘HWT-LGT’’ Pairs from Four Different LLMs. These were trained on the Claude-Instant dataset with 512 ‘HWT-LGT’ pairs under the setting of NVIDIA A100 80GB using Float32 Precision. The **bold** indicates the best performance and <u>underline</u> indicates the second best.

5.5 Costs of Space and Time

When examining the costs of Methodology, we particularly focus on their balance between effectiveness and resource consumption in real-world applications. To evaluate our method in terms of effectiveness and resource cost, we compared RepreGuard with three other detectors: Roberta, Fast-Detect, and Binocular. In the comparative experiments, we set the batch size to 1 to measure the performance of each method when processing a single sample in the setting of float32 under the A100 80GB GPU. The results in Table 5 shown that RepreGuard demonstrates the best overall performance with relatively low resource consumption. Specifically, RepreGuard (Phi-2) achieves the highest AUROC of 96.10% and relatively low resource consumption (16.0 GB, 0.072 seconds per sample), striking an effective balance between accuracy and efficiency. Meanwhile, RepreGuard (Llama-3.1-8B) achieves an AUROC of 94.80 and the highest TPR@0.01 of 77.10%, showcasing exceptional capability in positive case detection. It is noteworthy that Roberta lies in its extremely low resource consumption (2.0 GB, 0.016 seconds per sample), making it suitable for cost-constrained scenarios. However, its detection performance (84.85% in AUROC, 43.90% in TPR@0.01) is significantly inferior to that of RepreGuard.

In addition, we assess whether our approach is affected by memorization (see Appendix A.2), and examine how the performance as the increase of the LGT used in LLMs (see Appendix A.3).

6 Conclusion

In this paper, we introduce **RepreGuard**, a novel and reliable method based on hidden representation features for detecting text generated by LLMs. Experimental results on both ID and OOD demonstrate RepreGuard's strong detection capabilities and zero-shot proficiency. It requires only a small number of training samples to achieve impressive OOD generalization, effectively handling diverse real-world application scenarios and challenge from newly emerging LLMs. Furthermore, we verify the effectiveness, robustness, and generalization ability of RepreGuard in detecting texts of varied sizes, as well as texts that have undergone paraphrasing attack, perturbation attack, and various sampling methods.

Acknowledgments

This work was supported in part by the Science and Technology Development Fund of Macau SAR (grant no. FDCT/0007/2024/AKP), the Science and Technology Development Fund of Macau SAR (grant no. FDCT/0070/2022/AMJ, China Strategic Scientific and Technological Innovation Cooperation Project grant no. 2022YFE0204900), the Science and Technology Development Fund of Macau SAR (grant no. FDCT/060/2022/AFJ, National Natural Science Foundation of China grant no. 62261160648), the UM and UMDF (grant nos. MYRG-GRG2023-00006-FST-UMDF, MYRG-GRG2024-00165-FST-UMDF, EF2024-00185-FST), and the National Natural Science Foundation of China (grant no. 62266013). This work was also supported in part by National Key Research and Development Program of China (2022YFF0902100), National Natural Science Foundation of China (grant no. 62376262), the Natural Science Foundation of Guangdong Province of China (2024A1515030166, 2025B1515020032), and the Shenzhen Science and Technology Innovation Program (KQTD20190929172835662).

References

Anthropic. 2023. Releasing claude instant 1.2.

Amos Azaria and Tom Mitchell. 2023. The internal state of an LLM knows when it‘s lying. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 967–976, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.findings-emnlp.68

Guangsheng Bao, Yanbin Zhao, Zhiyang Teng, Linyi Yang, and Yue Zhang. 2024. Fast-detectGPT: Efficient zero-shot detection of machine-generated text via conditional probability curvature. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7–11, 2024*.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-NEO: Large scale autoregressive language modeling with mesh-tensorflow.

Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B. Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium, USENIX Security 2021, August 11–13, 2021*, pages 2633–2650. USENIX Association.

Yutian Chen, Hao Kang, Vivian Zhai, Liangze Li, Rita Singh, and Bhiksha Raj. 2023. GPT-sentinel: Distinguishing human and chatGPT generated content. *CoRR*, abs/2305.07969.

Debby R. E. Cotton, Peter A. Cotton, and J. Reuben Shipway. 2024. Chatting and cheating: Ensuring academic integrity in the era of chatgpt. *Innovations in Education and Teaching International*, 61(2):228–239. https://doi.org/10.1080/14703297.2023.2190148

Liam Dugan, Alyssa Hwang, Filip Trhlík, Andrew Zhu, Josh Magnus Ludan, Hainiu Xu, Daphne Ippolito, and Chris Callison-Burch. 2024. RAID: A shared benchmark for robust evaluation of machine-generated text detectors. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11–16, 2024*, pages 12463–12492. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.acl-long.674

Nadir Durrani, Hassan Sajjad, Fahim Dalvi, and Yonatan Belinkov. 2020. Analyzing individual neurons in pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16–20, 2020*, pages 4865–4880. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.395

Tiziano Fagni, Fabrizio Falchi, Margherita Gambini, Antonio Martella, and Maurizio Tesconi. 2020. TweepFake: About detecting deepfake tweets. *CoRR*, abs/2008.00036. https://doi.org/10.1371/journal.pone.0251415, PubMed: 33984021

Angela Fan, Mike Lewis, and Yann N. Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15–20, 2018, Volume 1: Long Papers*, pages 889–898.

Zoubin Ghahramani. 2023. Introducing palm 2.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatGPT to human experts? Comparison corpus, evaluation, and detection. *CoRR*, abs/2301.07597.

Abhimanyu Hans, Avi Schwarzschild, Valeriia Cherepanova, Hamid Kazemi, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2024. Spotting LLMs with binoculars: Zero-shot detection of machine-generated text. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21–27, 2024*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26–30, 2020*. OpenReview.net.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):248:1–248:38. https://doi.org/10.1145/3571730

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. 2023. Paraphrasing evades detectors of AI-generated text, but retrieval is an effective defense. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 – 16, 2023*.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2019. Textbugger: Generating adversarial text against real-world applications. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24–27, 2019*. https://doi.org/10.14722/ndss.2019.23138

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Yikang Liu, Ziyin Zhang, Wanyang Zhang, Shisen Yue, Xiaojing Zhao, Xinyuan Cheng, Yiwen Zhang, and Hai Hu. 2023. ArguGPT: Evaluating, understanding and identifying argumentative essays generated by GPT models. *CoRR*, abs/2304.07666.

MetaAI. 2024. Introducing meta Llama 3: The most capable openly available LLM to date.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and

Chelsea Finn. 2023. DetectGPT: Zero-shot machine-generated text detection using probability curvature. In *International Conference on Machine Learning, ICML 2023, 23–29 July 2023, Honolulu, Hawaii, USA*, pages 24950–24962.

Giovanni Monea, Maxime Peyrard, Martin Josifoski, Vishrav Chaudhary, Jason Eisner, Emre Kiciman, Hamid Palangi, Barun Patra, and Robert West. 2024. A glitch in the matrix? Locating and detecting language model grounding with fakepedia. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6828–6844, Bangkok, Thailand. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.acl-long.369

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of GPT-4. *CoRR*, abs/2306.02707.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. 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, Brussels, Belgium, October 31 – November 4, 2018*, pages 1797–1807. https://doi.org/10.18653/v1/D18-1206

OpenAI. 2022. Introducing chatGPT.

Artidoro Pagnoni, Martin Graciarena, and Yulia Tsvetkov. 2022. Threat scenarios and best practices to detect neural fake news. In *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12–17, 2022*, pages 1233–1249.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Hamza Alobeidli, Alessandro Cappelli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon LLM: Outperforming curated corpora with web data only. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 – 16, 2023*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:140:1–140:67.

Areg Mikael Sarvazyan, José Ángel González, Paolo Rosso, and Marc Franco-Salvador. 2023. Supervised machine-generated text detectors: Family and scale matters. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction - 14th International Conference of the CLEF Association, CLEF 2023, Thessaloniki, Greece, September 18–21, 2023, Proceedings*, pages 121–132. https://doi.org/10.1007/978-3-031-42448-9_11

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, and Jasmine Wang. 2019. Release strategies and the social impacts of language models. *CoRR*, abs/1908.09203.

Gaurang Sriramanan, Siddhant Bharti, Vinu Sankar Sadasivan, Shoumik Saha, Priyatham Kattakinda, and Soheil Feizi. 2024. LLM-check: Investigating detection of hallucinations in large language models. In *Advances in Neural Information Processing Systems*, volume 37, pages 34188–34216. Curran Associates, Inc. https://doi.org/10.52202/079017-1077

Jinyan Su, Terry Yue Zhuo, Di Wang, and Preslav Nakov. 2023. DetectLLM: Leveraging log rank information for zero-shot detection of machine-generated text. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6–10, 2023*, pages 12395–12412. https://doi.org/10.18653/v1/2023.findings-emnlp.827

Zhen Sun, Zongmin Zhang, Xinyue Shen, Ziyi Zhang, Yule Liu, Michael Backes, Yang Zhang, and Xinlei He. 2025. Are we in the AI-generated text world already? Quantifying and monitoring AIGT on social media. https://doi.org/10.18653/v1/2025.acl-long.1120

Tianyi Tang, Wenyang Luo, Haoyang Huang, Dongdong Zhang, Xiaolei Wang, Xin Zhao, Furu Wei, and Ji-Rong Wen. 2024. Language-specific neurons: The key to multilingual capabilities in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11–16, 2024*, pages 5701–5715. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.acl-long.309

Lindia Tjuatja, Valerie Chen, Sherry Tongshuang Wu, Ameet Talwalkar, and Graham Neubig. 2023. Do LLMs exhibit human-like response biases? A case study in survey design. *CoRR*, abs/2311.04076.

Vivek Verma, Eve Fleisig, Nicholas Tomlin, and Dan Klein. 2024. Ghostbuster: Detecting text ghostwritten by large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1702–1717, Mexico City, Mexico. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.naacl-long.95

Elena Voita, Javier Ferrando, and Christoforos Nalmpantis. 2024. Neurons in large language models: Dead, n-gram, positional. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and Virtual Meeting, August 11–16, 2024*, pages 1288–1301. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.findings-acl.75

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 billion parameter autoregressive language model. https://github.com/kingoflolz/mesh-transformer-jax

Zecong Wang, Jiaxi Cheng, Chen Cui, and Chenhao Yu. 2023. Implementing BERT and fine-tuned roBERTa to detect AI generated news by chatGPT. *CoRR*, abs/2306.07401.

Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Derek F. Wong, and Lidia S. Chao. 2023. A survey on LLM-generated text detection: Necessity, methods, and future directions. *CoRR*, abs/2310.14724.

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

Junchao Wu, Runzhe Zhan, Derek F. Wong, Shu Yang, Xinyi Yang, Yulin Yuan, and Lidia S. Chao. 2024. Detectrl: Benchmarking LLM-generated text detection in real-world scenarios. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10–15, 2024.*

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2024a. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7–11, 2024*. OpenReview.net.

Haoyun Xu, Runzhe Zhan, Derek F. Wong, and Lidia S. Chao. 2024b. Let's focus on neuron: Neuron-level supervised fine-tuning for large language model. *CoRR*, abs/2403.11621.

Xianjun Yang, Wei Cheng, Yue Wu, Linda Ruth Petzold, William Yang Wang, and Haifeng Chen. 2024. DNA-GPT: Divergent n-gram analysis for training-free detection of GPT-generated text. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7–11, 2024.*

Weichen Yu, Tianyu Pang, Qian Liu, Chao Du, Bingyi Kang, Yan Huang, Min Lin, and Shuicheng Yan. 2023. Bag of tricks for training data extraction from language models. In *International Conference on Machine Learning, ICML 2023, 23–29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 40306–40320. PMLR.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems 32: Annual Conference* *on Neural Information Processing Systems 2019, NeurIPS 2019, December 8–14, 2019, Vancouver, BC, Canada*, pages 9051–9062.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7–12, 2015, Montreal, Quebec, Canada*, pages 649–657.

Zheyuan Zhang, Fengyuan Hu, Jayjun Lee, Freda Shi, Parisa Kordjamshidi, Joyce Chai, and Ziqiao Ma. 2024. Do vision-language models represent space and how? Evaluating spatial frame of reference under ambiguities. In *The Thirteenth International Conference on Learning Representations*.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. 2023. Representation engineering: A top-down approach to AI transparency. *CoRR*, abs/2310.01405.

A Appendix

A.1 Analysis of Activation Token

To investigate whether activation tokens contain specific tokens or which parts of speech enable the model to distinguish between HWT and LGT, We conducted an analysis of the word frequency and part-of-speech (POS) tags of Activation Tokens (the last 10% of tokens) and their relationship with RepreScore. The results are presented in Figure 8 and Figure 9. In general, the RepreScores for LGT are generally higher than those for HWT, with significant differences observed particularly in adjectives (ADJ), adverbs (ADV), and pronouns (PRON), while the differences for symbols (SYM) are the smallest. From the frequency distribution of the top 50 tokens, it is evident that the same token does not have identical RepreScore values in HWT and LGT; the scores for LGT are generally higher. This suggests that the RepreScore is not directly determined by the token itself. To explain this phenomenon, it is necessary to analyze from the perspective of Equation 1. Since the activation value of each token is computed based on the inputs of its preceding tokens, this implies that when the model processes a token t_n, it has already accounted for the contextual information from $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$. Therefore, when we calculate the hidden representation changes of a token, these changes are essentially based on an analysis of the complete context rather than an isolated computation of the token itself.

A.2 Analysis of Model Memorization

Previous research (Carlini et al., 2021) has demonstrated the potential to extract substantial portions of text from the training data of LLMs by employing carefully designed prompting techniques. This finding has been further substantiated by subsequent work (Yu et al., 2023), which introduced advanced strategies for extracting training data. As a result, when text generated by LLMs is sourced directly from their training data, it becomes virtually indistinguishable from human-written text, rendering efforts to differentiate between LGT and HWT content effectively futile. Additionally, recent studies (Sun et al., 2025) have revealed that a significant portion of contemporary textual data now contains LGT while Reddit has exhibited relatively slower growth in this trend. To ensure that newly collected data consists exclusively of HWT, we utilize the latest Reddit dataset[4] released in 2025, which contains content written after the training cut-off dates of the Llama-3.1-8B.[5] The results in Table 6 demonstrate that the RepreGuard achieved exceptionally high precision across all model datasets when evaluating new HWT, with an average precision of 95.81%. This suggests that the RepreGuard is not influenced by model memorization, as the models do not simply recall the texts but can accurately identify the distinguishing features of LGT and HWT texts.

A.3 Performance After LGT Pretraining

As LLMs continue to evolve, an increasing proportion of their training data (Mukherjee et al., 2023; Xu et al., 2024a) is likely to consist of LGT, as the internet becomes increasingly saturated with content produced by these systems. This raises

[4]`https://huggingface.co/datasets/tensorshield/reddit_dataset_157`.

[5]`https://huggingface.co/meta-llama/Llama-3.1-8B`.

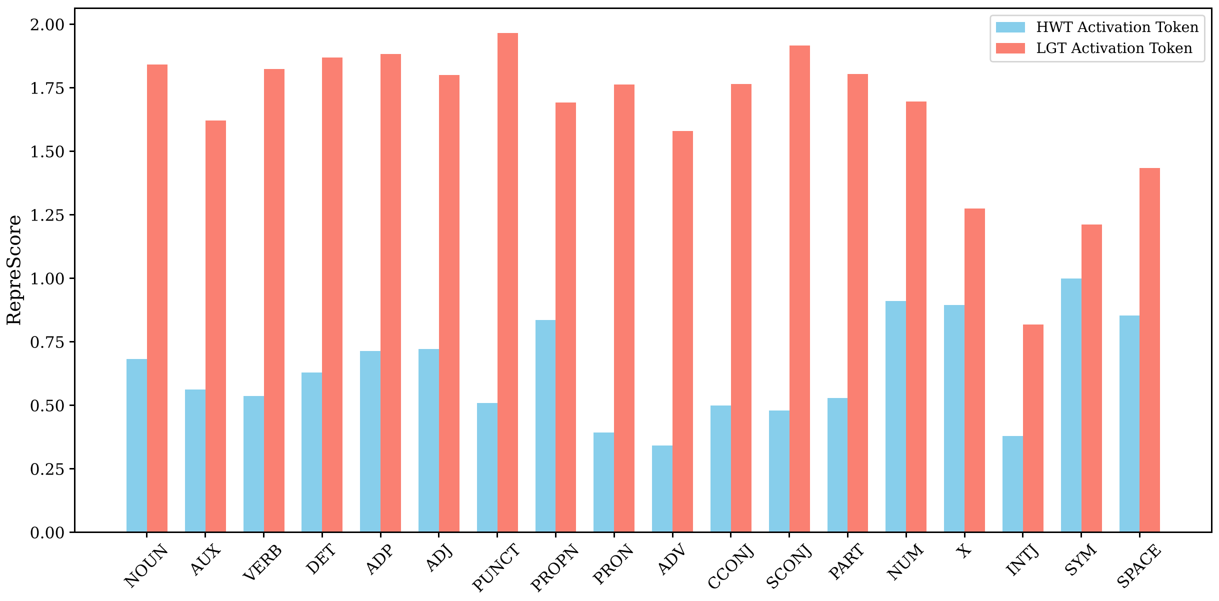

Figure 8: Average RepreScore Values Across Different Parts of Speech (POS) Tags for Activation Tokens.

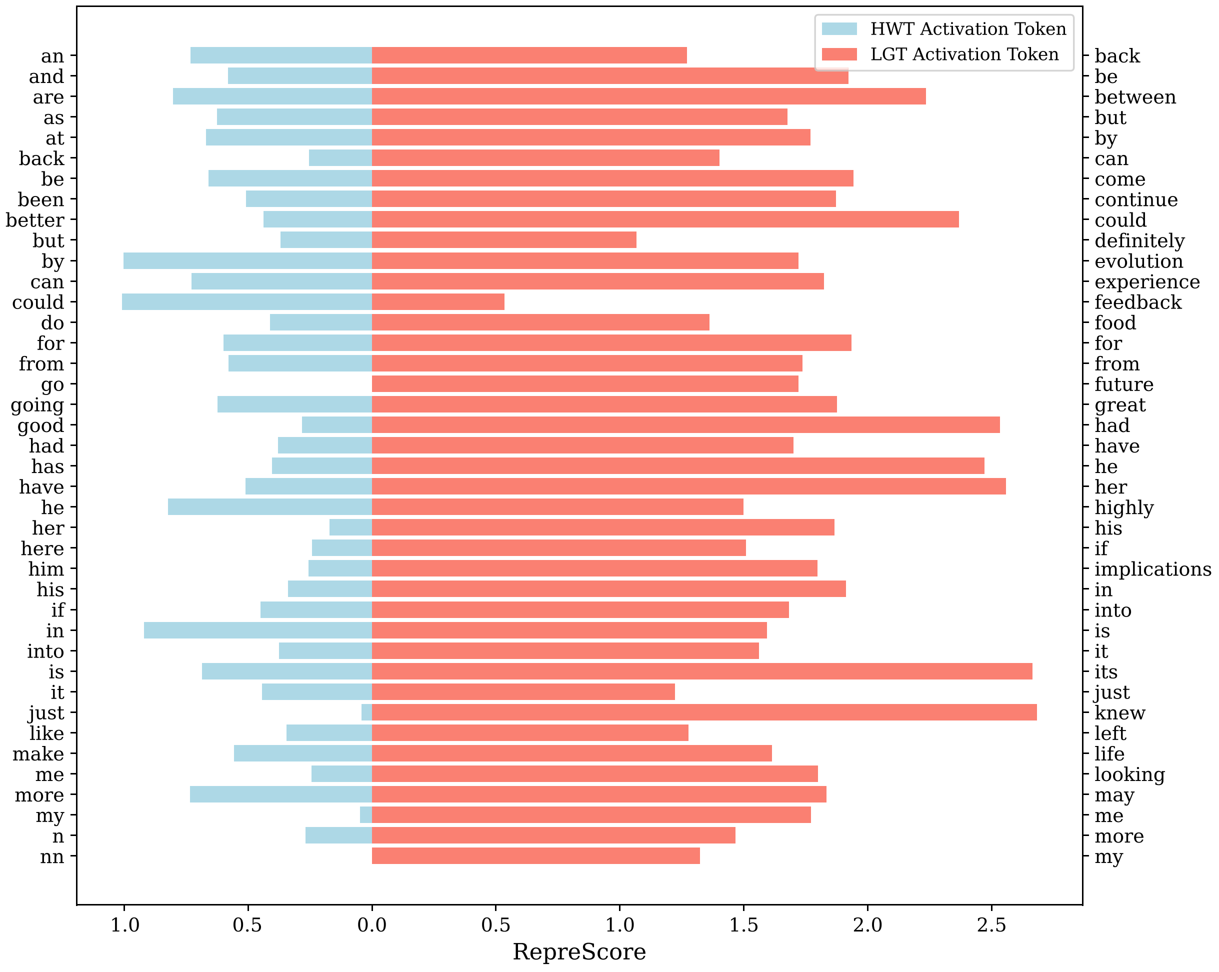

Figure 9: Frequency Distribution of the Top 50 Activation Tokens and Their Corresponding RepreScore Values for HWT and LGT.

Train ↓	Precision
ChatGPT	96.50
Llama-2-70b	96.50
Google-PaLM	95.55
Claude-instant	96.45
Multi-LLMs	94.05
AVG.	95.81

Table 6: Precision of Different LLMs in Identifying HWT on 2,000 Samples from the Newly Collected Reddit Dataset Released in 2025. Precision is used as the dataset contains only a single class (HWT).

critical questions about the sustained effectiveness of RepreGuard when applied to large-scale datasets in such a scenario. To explore this, we curate a dataset comprising 1 million LGT to pre-train our surrogate model and systematically record the corresponding checkpoints, shown on the figure Figure 10. The results shown that As the proportion of LGT in pre-training increases from 0% to 100%, the AUROC of the RepreGuard exhibits a slight decline, yet overall performance remains robust. Specifically, the average AUROC decreases from 94.76% to 94.68%, demonstrating good overall robustness. In contrast, TPR@0.01 experiences a notable reduction, with the average TPR@0.01 decreasing from 80.92% to 73.72%. This suggests that pre-training with LGT diminishes the model's detection capability under extremely low false positive rates.

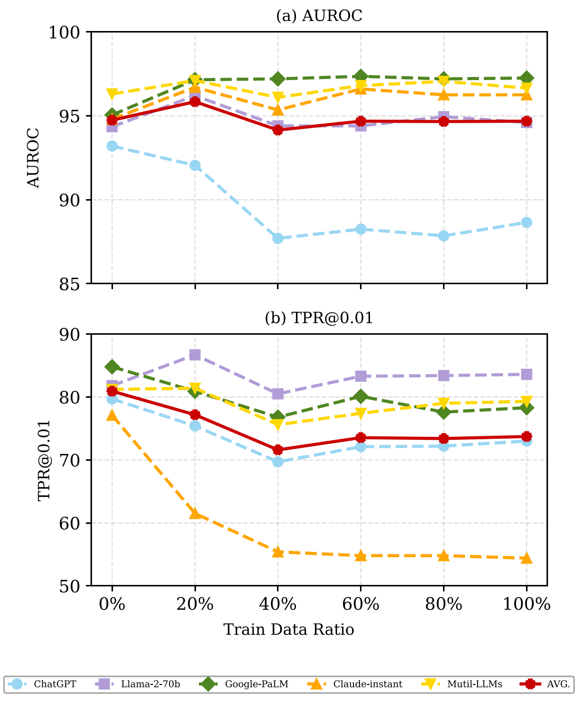

Figure 10: Performance of RepreGuard in Terms of AUROC and TPR@0.01 on 1000 ''HWT-LGT'' Pairs from 4 Different LLMs after LGT Pretraining. The raw text generated by each model is used to model representation features and set thresholds.

A.4 Discussion on Hallucination Detection and LGT Detection Using the Hidden Representation

Recent hallucination detection research has gradually shifted from focusing on the external performance to the internal hidden representation from LLMs. For instance, Masked Grouped Causal Tracing (MGCT) (Monea et al., 2024) reveals the internal mechanisms underlying grounded and ungrounded behaviors by selectively perturbing and restoring hidden activations. Azaria and Mitchell (2023) used the hidden representation as feature inputs to train an external feedforward neural network classifier, enabling the automatic determination of statement veracity. The LLM-Check (Sriramanan et al., 2024) method further extracts hidden representations during LLM response generation and calculates the covariance matrix (Hidden Score) as a quantitative metric for hallucination detec tion. These methods collectively validate that the hidden representation contains rich information, offering significant advantages in hallucination detection tasks.

However, our method, RepreGuard, systematically identifies differences in hidden states between LGT and HWT to distinguish them. Similar to MGCT, RepreGuard focuses on disparities within the hidden space, while MGCT (Monea et al., 2024) places greater emphasis on causal intervention and mechanistic explanation. In contrast to Azaria and Mitchell (2023), RepreGuard does not require training additional networks, enabling efficient detection within an unsupervised framework. Furthermore, compared to LLM-Check (Sriramanan et al., 2024), which relies on the covariance features of generation, RepreGuard is designed to capture the hidden representations underlying behavioral processes, allowing the model to simulate the writing process and thereby discern differences in hidden representations between HWT and LGT.

A.5 Figure of TPR@.01 on Paraphrase & Peturbation Attack

Figure 11 illustrates the performance comparison of RepreGuard under paraphrase and perturbation attacks in terms of TPR@0.01.

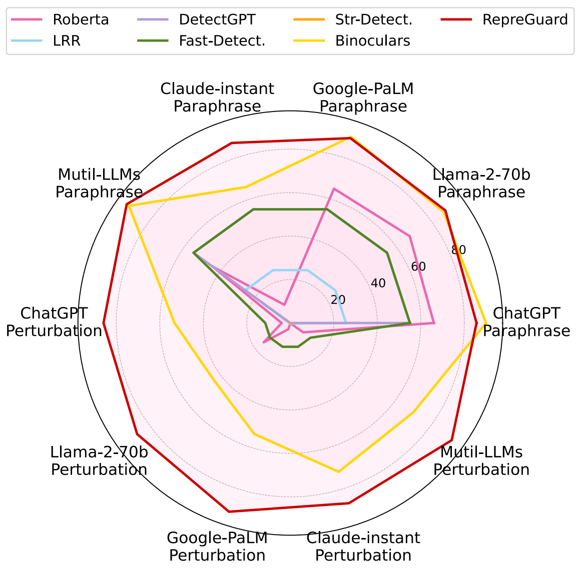

Figure 11: Performance Comparison of RepreGuard on Paraphrase and Perturbation Attack in Terms of TPR@0.01 on 1000 ‘‘HWT-LGT’’ Pairs from 4 Different LLMs. The raw text generated by each model is used to model representation features and set thresholds.