PhantomHunter: Detecting Unseen Privately-Tuned LLM-Generated Text via Family-Aware Learning

Anonymous Author(s)
Submission Id: 933

Abstract

With the popularity of large language models (LLMs), undesirable societal problems like misinformation production and academic misconduct have been more severe, making LLM-generated text detection now of unprecedented importance. Though existing methods have made remarkable progress, they mostly consider publicly known LLMs when testing the performance, and a new challenge brought by text from privately-tuned LLMs is largely underexplored. Due to the rapid development of open-source models like LLaMA and Qwen series and efficient LLM training methods, even ordinary users can now easily possess private LLMs by fine-tuning an open-source one with private corpora. This could lead to a significant performance drop of existing detectors in practice, due to their poor capability of capturing the essential LLM traits robust to fine-tuning operations. Our preliminary examination reveals that fine-tuning an LLM with 11M tokens could make a detector's accuracy jump from 100% to only 59% at most. To address this issue, we propose PhantomHunter, an LLM-generated text detector specialized for detecting text from unseen privately-tuned LLMs, whose family-aware learning framework captures family-level traits shared across the base models and their derivatives, instead of memorizing individual characteristics. Specifically, PhantomHunter first extracts base model features and enhances the family-shared information using a contrastive family-aware learning module. The enhanced features are then fed into a mixture-of-experts module containing multiple experts for corresponding families for final predictions. Experiments on data from four widely-adopted LLM families (LLaMA, Gemma, Mistral, and Qwen) show PhantomHunter's superiority over 8 baselines and 11 industrial services. Our code is available at https://anonymous.4open.science/r/Phantomhunter-933/.

CCS Concepts

• **Computing methodologies** → **Natural language processing**; • **Information systems** → *Web and social media search.*

Keywords

LLM-generated Text Detection, LLM Family

ACM Reference Format:
Anonymous Author(s). 2026. *PhantomHunter:* Detecting Unseen Privately-Tuned LLM-Generated Text via Family-Aware Learning. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '26).* ACM, New York, NY, USA, 11 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Large language models (LLMs) have successfully revolutionized the way people organize and produce text and inspire productivity in a wide range of applications [22]. However, undesirable societal problems caused by the misuse of LLMs also rapidly emerge, such as cheating in academic writing [23], creating misinformation [5], and accelerating information system pollution [35]. As the front-line barrier against such threats, automatic detection techniques of LLM-generated text (LLMGT) are now of unprecedented importance and attract researchers from both academia and industry.

LLMGT detection generally distinguishes LLM-generated and human-written text via binary classification. Existing methods either learn common textual features (e.g., stylistic clues [13]) shared across LLMs using representation learning or design distinguishable metrics between human and LLM texts based on LLMs' internal signals (e.g., token probabilities [39]). For both categories, their tests were mostly conducted on data from publicly available LLMs, assuming that users generate text using public, off-the-shelf services. **We argue that this situation is being changed due to the recent development of the open-source LLM community.** With the help of platforms like HuggingFace[1] and the efficient LLM training techniques like low-rank adaptation (LoRA) [19], building fine-tuned LLMs with customized private datasets has become much easier than before. For instance, there have been over 200k Qwen-based and 60k Llama-based derivative models on HuggingFace [1, 31]. As shown in figure 1, after private fine-tuning on unknown corpora, the learned characteristics of base models could change, and the LLMGT detectors would fail (will empirically show in the following section), shaping a new risk that malicious users can generate harmful texts privately without being caught by LLMGT detectors. A new challenge arises: **How could we detect text generated by privately-tuned open-source LLMs?**

To address this issue, we first conduct an analysis of differences between the base LLMs and their derivative models via fine-tuning on data of different topics. The result reveals that compared to other non-homologous models, the token probability lists of text on the fine-tuned LLM have a higher similarity to the probability lists on the base LLM, exhibiting clear "family traits." Inspired by this finding, we propose to perform family-aware learning by extracting probabilistic features of base models and design **PhantomHunter**, a detector targeted at text generated by unseen, privately-tuned LLMs. PhantomHunter consists of three main components: First, it obtains probability features from widely-known base LLMs such as Qwen and LLaMA. The extracted features are then enhanced using

[1]https://huggingface.co/models

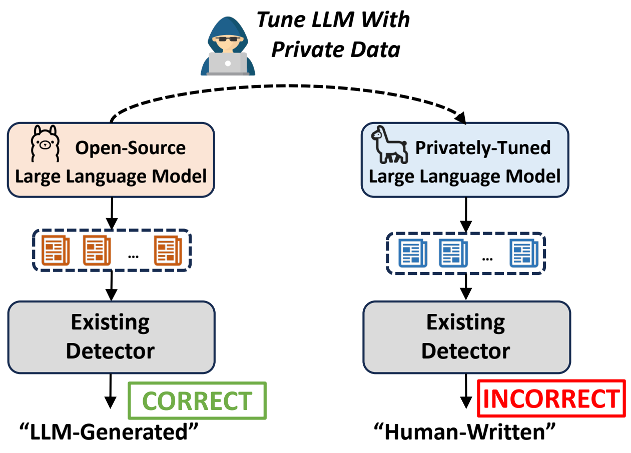
Tune LLM With Private Data
Open-Source Large Language Model
Privately-Tuned Large Language Model
Existing Detector
Existing Detector
CORRECT
INCORRECT
"LLM-Generated"
"Human-Written"

Figure 1: Illustration of how private fine-tuning LLMs affects LLM-generated text detection. Existing detectors may successfully identify text generated by the original open-source LLM, but fail to detect text generated by the privately fine-tuned one due to its reliance on non-shared features among the base and fine-tuned LLMs, causing missed detection.

a contrastive family-aware learning to better preserve the family-level traits shared across the text pieces from both the base model and the fine-tuned derivative models. The enhanced features are fed into a Mixture-of-Experts (MoE) module that contains multiple expert modules specialized in detecting texts from specific LLM families. Finally, the collective judgment of the given text being LLM-generated or not is made by the gate mechanism controlled by the family prediction results. Experiments against baseline methods from both previous literature and the real-world industrial services demonstrate that our proposed PhantomHunter could better capture family-level traits and effectively detect generated text from unseen privately-tuned LLMs. Our main contributions are summarized as follows:

- **New Risk:** We empirically expose the new risk that current LLM-generated text detectors may fail to detect text from privately-tuned LLMs.
- **New Method:** We propose PhantomHunter, which is trained via family-aware learning to learn common traits shared in typical open-source LLM families.
- **High Performance:** We construct a new dataset containing texts in academic abstracts and Q&A genres generated by humans and four widely-adopted LLM families (LLaMA, Gemma, Mistral, and Qwen) for evaluation. Experiments show that PhantomHunter outperforms 7 baselines and 11 industrial services.

2 How Does Fine-tuning Affect LLM-Generated Texts Detection?

2.1 Preliminary Experiment

To explore the extent to which fine-tuning affects the performance of LLM-generated text detectors, we fine-tuned LLaMA-2 7B using the arXiv abstract corpus and saved checkpoints at different fine-tuning steps, which were then used to generate test samples. We

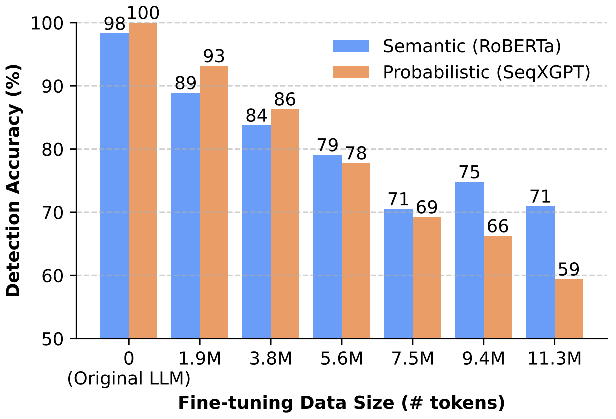

Figure 2: Detection accuracy for LLM-generated text with increasing amounts of fine-tuning data.

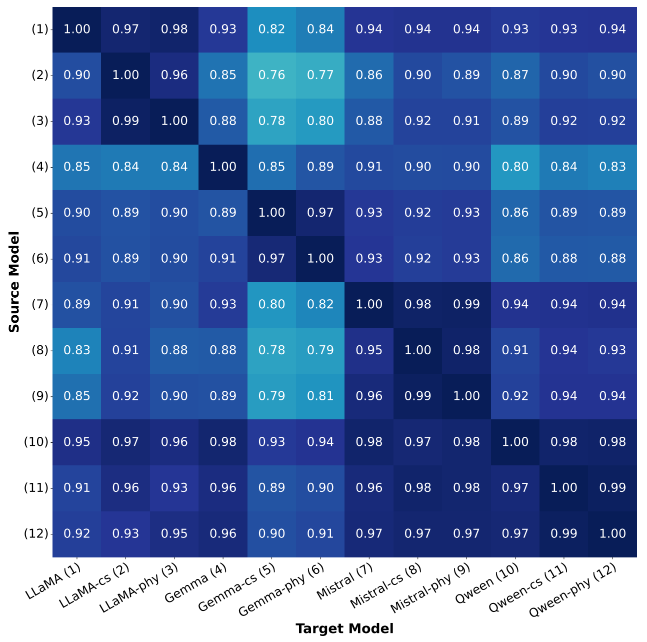

Figure 3: Heatmap of cross-model similarity measured by the complement of relative NLL distance. Rows indicate source models and columns indicate target models, covering 12 variants across four model families and two training domains. Darker colors indicate higher similarity.

experimented on two detectors, RoBERTa (semantic-based; 13) and SeqXGPT (probability-based; 43). Figure 2 displays **their decaying trends in accuracy as the fine-tuning corpus size increases**. For the LLM fine-tuned on 11.3M tokens, SeqXGPT, which performs almost *perfect* at detecting text pieces from the original LLM, even encounters an astonishing accuracy drop of 41%.

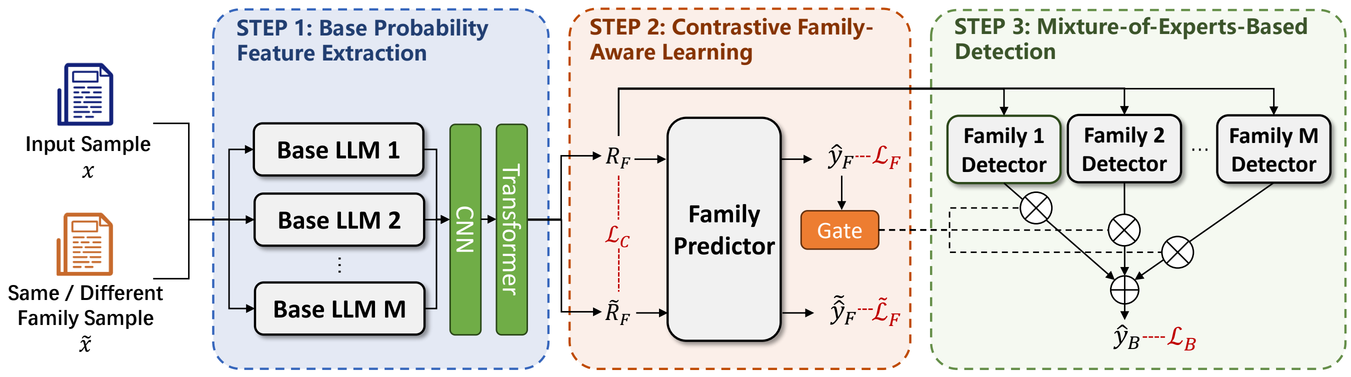
STEP 1: Base Probability Feature Extraction
STEP 2: Contrastive Family-Aware Learning
STEP 3: Mixture-of-Experts-Based Detection
Input Sample x
Same / Different Family Sample $\tilde{x}$
Base LLM 1
Base LLM 2
Base LLM M
CNN
Transformer
R_F
$\mathcal{L}_C$
$\tilde{R}_F$
Family Predictor
$\hat{y}_F$ $\mathcal{L}_F$
Gate
$\hat{\tilde{y}}_F$ $\tilde{\mathcal{L}}_F$
Family 1 Detector
Family 2 Detector
Family M Detector
$\hat{y}_B$ $\mathcal{L}_B$

Figure 4: Overall architecture of PhantomHunter and the training process. Given a text sample x, it 1) extracts the probability feature from M base models and encode them with CNN and transformer blocks; 2) predicts the family of x to determine the family gating weights; and 3) feeds the representation R_F to a mixture-of-experts network controlled by the gating weights from Step 2 for final prediction of x being LLM-generated. During training, contrastive learning is applied in each mini-batch to better model family relationships. The red terms are loss functions.

2.2 Impact of Fine-tuning on Features

Experimental Settings We fine-tuned four open-source LLMs (LLaMA, Gemma, Mistral, and Qwen) using two sets of corpus from different domains (computer science and physics, see the Dataset Construction section for details) and collected responses from twelve model variants in total: four base models and eight LoRA fine-tuned models (two domains × four base models). All responses were generated under identical decoding configurations (here, temperature = 1.0, top-p = 0.95) to ensure consistency.

Method. We implement a cross-model behavioral consistency analysis to compute a relative NLL distance. For a given source model S and target model T, we extract their respective negative log-likelihood sequences $\text{NLL}_S = [\ell_1^S, \ell_2^S, \ldots, \ell_n^S]$ and $\text{NLL}_T = [\ell_1^T, \ell_2^T, \ldots, \ell_n^T]$ over the same input sequence, where $\ell_i = -\log p_i$ represents the negative log-likelihood Loss of the i-th token with predicted probability p_i.The smaller ℓ value indicates higher prediction confidence and higher probability for the corresponding token. Therefore, to measure the overall consistency between their predictive behaviors, we compute the normalized similarity score based on the complement of relative NLL distance:

$$\text{dist}(\text{NLL}S, \text{NLL}T) = 1 - \frac{\left|\sum i = 1^n \ell_i^T - \sum i = 1^n \ell_i^S\right|}{\sum i = 1^n \ell_i^S}, \qquad (1)$$

We aggregated these scores across all test samples to obtain the average similarity for each source-target model pair. The resulting 12×12 similarity matrix was visualized as a heatmap using seaborn, with color intensity ranging from 0.5 to 1.0 to highlight the similarity in cross-model prediction patterns, providing insights into which model combinations exhibit stronger behavioral alignment.

Findings and Inspiration. From Figure 3, we intuitively observe that the probability lists of LLMs from the same family are more similar to each other, and clearly different from those of the others, which shows significant "family traits." Despite exceptions, an overall trend of family-wise LLM similarity persists. Exceptions may be caused by fine-tuning mechanics in different families and the coarse-grained metric. This observation reveals that the probability lists of the base models could be useful for modeling unseen fine-tuned ones. However, such "family traits" are not fully exploited by existing detectors because they treat all LLMs equally and may only learn the commonality across all involved LLMs (regarding the 12×12 area in the heatmap), not the family-level commonality (regarding every 3×3 areas), shaping a looser boundary against human text. Targetedly, we design PhantomHunter to learn such traits to address this issue.

3 Proposed Method: PhantomHunter

PhantomHunter is a family-aware learning-based detector for unseen, privately-tuned LLM-generated texts. The key insight behind PhantomHunter is to model the inherent family-level features shared between base LLMs and their fine-tuned descendants. Building on these observations, PhantomHunter employs a family-aware learning approach to detect texts from unseen, privately-tuned LLMs. As illustrated in Figure 4, the architecture consists of three main components: **1)** a base probability feature extractor, **2)** a contrastive learning-based family encoder, and **3)** a mixture-of-experts detection module, which will be detailed below.

3.1 Base Probability Feature Extraction

probability list Alignment To address the discrepancy in the lengths of token-wise probability sequences caused by the different tokenizers of the base models in PhantomHunter, we adopt the alignment strategy proposed by Sniffer, SeqXGPT [26, 43]. Specifically, we implement a sampling and truncation strategy that utilizes "words" as unified anchors. First, the model identifies the common semantic starting point of the sequences using the initial index lists from each tokenizer, thereby eliminating head-side variances caused by prompt offsets. Subsequently, token-level features of varying granularities are aligned to the original word-flow space of the text. Through a two-stage process involving local maximum length calculation within each batch and global padding/truncation to a predefined maximum length (*max_len*), the non-uniform probability distributions from multiple models are aggregated into a unified feature matrix $L = [l_1, \ldots, l_t]$. Within this matrix, each word-level feature vector l_i consolidates the log-likelihood characteristics of different models for the same semantic unit, achieving a rigorous word-wise alignment for multi-task modeling.

For a given text sequence $\mathbf{x}$, we pass it through M base LLMs $\theta_1, \theta_2, \ldots, \theta_M$ to obtain the probability lists $\mathbf{p} \in \mathbb{R}^{M\times N}$:

$$\mathbf{p} = (\mathbf{p}^{\theta_1}, \mathbf{p}^{\theta_2}, \ldots, \mathbf{p}^{\theta_M}), \tag{2}$$

where $\mathbf{p}^{\theta_i} = (p_1^{\theta_i}, p_2^{\theta_i}, \ldots, p_N^{\theta_i})$, N is the text sequence length, M is the number of base models (*i.e.,* families), and $p_j^{\theta_i}$ represents the probability of generating token x_j given context $x_{<j}$ using LLM θ_i. We then use convolutional neural networks and Transformer encoders to extract base probability features $\mathbf{R}_F \in \mathbb{R}^{M\times d}$, where d is the feature dimension.Detailed parameter settings are provided in the appendix Section C.

3.2 Contrastive Family-Aware Learning

To better model family relationships in the feature space and enhance the detector's generalization within the same family, we treat samples from the same family as augmentations and employ contrastive learning to enhance the representation $\mathbf{R}_F$. Specifically, within each mini-batch, texts generated by models from the same family as the original text's model are treated as positive examples, while others are negative ones. We compute a SimCLR-style [6] contrastive loss $\mathcal{L}_C$:

$$\mathcal{L}_C = -\sum_{i\in b} \log \frac{\exp(\delta(\mathbf{R}^m_{F_i}, \tilde{\mathbf{R}}^m_{F_i})/t)}{\sum_{j=1}^{2|b|} \mathbf{1}_{[j\neq i]} \exp(\delta(\mathbf{R}^m_{F_i}, \mathbf{R}_{F_j})/t)}, \tag{3}$$

where $\mathbf{R}^m_{F_i}/\tilde{\mathbf{R}}^m_{F_i}$ are the embeddings of the original/augmented sample, respectively, δ is a dot-product function, and t controls the temperature. We use a multi-layer perceptron (MLP) to classify the LLM family:

$$\hat{y}_F = \text{softmax}(\text{MLP}_\text{F}(\mathbf{R}_F)), \tag{4}$$

where $\hat{y}_F \in \{\theta_1, \theta_2, \ldots, \theta_M\}$ represents the predicted LLM family.

3.3 Mixture-of-Experts-Based Detection

Finally, we employ a mixture-of-experts network for binary classification of human versus AI-generated text. Each expert detector specializes in detecting texts from specific LLM families, enhancing detection performance. Specifically, we use the family prediction result $\hat{y}_F$ as a gating signal to control the weighting of each expert's prediction, yielding the final binary judgment:

$$\hat{y}_B = \sum_{i=1}^{M} \hat{y}_F^i \cdot \text{softmax}(\text{MLP}_\text{B}(\mathbf{R}_F)), \tag{5}$$

where $\hat{y}_F^i$ represents the probability that the text belongs to the family of base model θ_i according to the family classifier. $\hat{y}_B \in [0, 1]$ denotes the probability of the $\mathbf{x}$ being LLM-generated. We optimize the PhantomHunter model with:

$$\mathcal{L} = \lambda_1(\mathcal{L}_F + \tilde{\mathcal{L}}_F) + \lambda_2\mathcal{L}_B + \lambda_3\mathcal{L}_C, \tag{6}$$

where $\mathcal{L}_F$ and $\tilde{\mathcal{L}}_F$ represent the cross-entropy losses for family classification of $\mathbf{x}$ and $\tilde{\mathbf{x}}$ respectively, $\mathcal{L}_B$ is the cross-entropy loss for LLM text detection, $\mathcal{L}_C$ is the contrastive loss. λ_i $(i = 1, 2, 3)$ are hyperparameters to balance the loss items during the optimization.

Table 1: Statistics of the corpora for fine-tuning.

ArXiv/Q&A Domain	# Tokens
Computer Science (cs)	6,441,516
Physics (phy)	5,955,389
Others (oth; q-bio/stat/eess/math/q-fin/econ)	3,939,626
ELI5	202,658
Finance (fin)	234,787
Medicine (med)	254,292

PhantomHunter captures inherent signatures that persist through fine-tuning by learning family-level features from observable fine-tuned models derived from the same base LLMs, enabling the detection of unseen models from known families. Rather than focusing on individual model characteristics, we emphasize shared patterns defining model families. This family-aware approach allows the detector to generalize to unseen fine-tuned models by recognizing persistent probability patterns that remain after extensive fine-tuning. The method is effective because fine-tuning primarily adapts a model's behavior to specific domains while preserving much of the underlying probabilistic structure inherited from the base model.

4 Experiment

In this section, we conduct experiments to answer the following questions:
EQ1: How effective is PhantomHunter at detecting texts from unseen fine-tuned LLMs?
EQ2: How do different components of PhantomHunter contribute to its overall performance?
EQ3: How does PhantomHunter perform in diverse real-world settings?

4.1 Dataset Construction

We fully simulate the two most common usage scenarios of LLM: writing and question-answering. For writing, we collect 69,297 abstracts of academic papers from the arXiv archive[2], categorizing them by primary subjects (domains). And for Q&A, we collect 3,062 Q&A pairs in ELI5, finance, and medicine domains from HC3 dateset [13]. Table 1 shows the corpora size. Note that we merge subjects other than physics and computer science for the arXiv part to balance fine-tuning scales across domains. We select open-source LLaMA-2 7B-Chat [41], Gemma 7B-it [12], Mistral 7B-Instruct-v0.1 [21],and Qwen2.5-7B-Instruct [2] as base models. For each scenario, we fine-tuned each base model using full-parameter fine-tuning (Full) and LoRA fine-tuning on the corresponding 3 domain-specific corpora from arxiv and Q&A, resulting in 48 derivative models. The details are as follows:

- **Full fine-tuning** was performed at the sft (supervised fine-tuning) stage. The training configuration was set with a batch size of 1 per device and gradient accumulation steps of 4. The learning rate was set to 1.0e-5, and the maximum number of training epochs was 10. A cosine learning rate scheduler was used with a warm-up ratio of 0.1. BF16 precision was enabled

[2]https://arxiv.org/archive/

Table 2: Statistics of the evaluation dataset for privately-tuned LLMGT detection. FT Domain: Domains of data for LLM fine-tuning ("base" denotes no fine-tuning).

	Dataset	Source	FT Domain	#Samples
arXiv	Train	LLaMA	base/phy/oth	1,564/1,969/1,971
		Gemma	base/phy/oth	1,604/1,455/1,457
		Mistral	base/phy/oth	1,778/1,877/1,826
		Qwen	base/phy/oth	1,640/1,546/1,543
		Human	-	1,145
	Test	LLaMA	cs	2,076
		Gemma	cs	1,540
		Mistral	cs	2,002
		Qwen	cs	2560
		Human	-	208
Q&A	Train	LLaMA	base/med/ELI5	680/256/220
		Gemma	base/med/ELI5	680/256/220
		Mistral	base/med/ELI5	680/256/220
		Qwen	base/med/ELI5	680/256/220
		Human	-	5,780
	Test	LLaMA	fin	204
		Gemma	fin	204
		Mistral	fin	204
		Qwen	fin	204
		Human	-	806
		Total		32,818

for efficient training, and the DDP (Distributed Data Parallel) strategy was employed.

- **LoRA fine-tuning** [19] was performed with the LoRA rank set to 8, and the training was conducted with a batch size of 1 per device, gradient accumulation steps of 4, a learning rate of 1.0e-5, and maximum number of epochs of 10, utilizing a cosine learning rate scheduler with a warm-up ratio of 0.1, BF16 precision for efficient training.

We then generate abstracts with 4 base models and 12 arXiv-based derivative models, and answers with the same base models and 12 Q&A-based derivative ones. For the arXiv dataset, we designate LLMs fine-tuned on computer science (cs) data as unseen ones, reserving them exclusively for testing. And for the Q&A dataset, we designate LLMs fine-tuned on finance (fin) data as unseen ones, which means it is only used as a test dataset. Detailed statistics of our experimental dataset are shown in Table 2, and the adopted prompt templates are as follows:

Prompt Description: [Generating an abstract based on the title of an academic paper, used in the arxiv scenario]
Instruction Prompt: [Write an abstract for the academic paper titled [*title*].]

Prompt Description: [Generating an answer based on the given question, used in the Q&A scenario]
Instruction Prompt: [*question*]

4.2 Experimental Setup

Implementation Details. In PhantomHunter, the CNN encoder comprises three convolutional layers followed by max-pooling, and the Transformer encoder has two attention layers with four attention heads each. The feature dimension d is set to 128. For contrastive learning, we use a temperature t of 0.07. The hyper-parameters in the loss function are set as $\lambda_1 = 1.0$, $\lambda_2 = 1.0$, and $\lambda_3 = 0.5$, respectively. The model is trained using the Adam optimizer with a learning rate of 2e-5 and a batch size of 32. We train for 10 epochs and select the best-performing checkpoint based on the validation set.

Evaluation Metrics. We report F1 scores on each testing subset (i.e., human-written and LLM-generated subsets) with a threshold of 0.5 and the macro F1. We also report the threshold-free metric AUC. For comparison with commercial detectors, we add the true positive rate under a 1% false positive rate to align with real-world scenarios, following the recommendation from Tufts et al. [42].

Compared Baselines. We select the eight methods for comparison, including **RoBERTa** [13], **T5-Sentinel** [7], **SeqXGPT** [43], **DNA-GPT** [48], **DetectGPT** [32], **Fast-DetectGPT** [3], **DALD** [51], **DeTeCtive** [14]. We detail the settings for all baselines in the appendix.

4.3 Main Results (EQ1)

Table 3 presents the binary classification results (human vs. LLM-generated) of PhantomHunter and baselines on unseen fine-tuned models. We have the following observations:

1) Compared to all baselines, PhantomHunter demonstrates superior performance in detecting texts from unseen fine-tuned models, with F1 scores for both human and LLM text exceeding 90% in each setting. For full fine-tuning, PhantomHunter improves the macro F1 score over the best baseline by 1.94% and 1.12% on both datasets, respectively; and for LoRA fine-tuning, the improvements are 1.44% and 5.18%, respectively. This exhibits PhantomHunter's detection capability for texts generated by unseen fine-tuned LLMs.

2) The fine-tuning approaches do not present consistent detectability for different domains or detectors. In most cases, the generated arXiv texts from LoRA-tuned LLMs are easier to detect than those from full-tuned ones, while the situations are reversed for Q&A texts. This might be because the arXiv texts are usually longer than Q&A texts, which could better reflect the effects of full parameter changes brought by full fine-tuning. In such a comparison, PhantomHunter still keeps balanced performance.

3) Some detectors show relatively low performance for the human class, regardless of class-wise balance of the training sets, indicating their difficulties in identifying human-written texts. We attribute this failure to their inherent ignorance of family traits shared among the base and derivative LLMs, which may shape a relatively blurred boundary between the human and LLM texts.

4.4 Ablation Studies (EQ2)

To evaluate the contribution of each component, we perform ablation studies by replacing the base feature extractor with a RoBERTa (*w/o* BFE), removing contrastive loss (*w/o* CL), and removing mixture-of-experts components (*w/o* MoE) from PhantomHunter, respectively, and the results are shown in Table 3.

Table 3: F1 scores for human-written (Human), LLM-generated (Gen.), and both (Macro) and AUC for unseen fine-tuned LLM-generated text detection. The two best results in each metric are bolded and underlined, respectively. BFE: Base probability feature extraction. CL: Contrastive learning. MoE: Mixture-of-experts.

	arXiv								Q&A							
Method	**Full**				**LoRA**				**Full**				**LoRA**			
	Human	Gen.	Macro	AUC	Human	Gen.	Macro	AUC	Human	Gen.	Macro	AUC	Human	Gen.	Macro	AUC
RoBERTa	62.42	98.54	80.48	97.63	65.61	98.66	82.13	93.16	84.30	95.12	89.71	98.72	58.12	78.03	68.07	99.75
T5-Sentinel	12.81	68.12	40.47	1.30	68.99	46.67	57.83	1.08	84.92	81.68	83.30	3.57	84.67	79.89	82.28	7.42
SeqXGPT	81.21	<u>99.52</u>	90.37	91.27	<u>93.15</u>	99.77	<u>96.46</u>	70.37	85.38	97.27	91.33	94.18	48.28	79.13	63.71	95.18
DNA-GPT	66.29	95.37	80.83	76.64	72.08	99.02	85.55	69.09	68.10	73.14	70.62	86.04	64.84	66.02	65.43	96.33
DetectGPT	26.71	95.50	61.11	85.56	52.53	98.33	75.43	79.27	59.46	88.72	74.09	82.80	26.09	<u>95.58</u>	61.11	92.36
Fast-DetectGPT	36.82	96.18	66.50	96.04	90.82	99.78	95.30	93.27	81.03	93.90	87.47	96.62	75.32	90.10	82.71	99.92
DALD	14.13	95.84	54.98	72.81	17.76	96.86	57.31	78.97	31.58	65.49	48.53	51.18	37.89	75.92	56.91	60.74
DeTeCtive	<u>85.09</u>	96.56	<u>90.83</u>	97.27	87.66	99.64	93.65	91.42	<u>89.08</u>	96.84	<u>92.96</u>	96.54	71.58	88.98	80.28	99.21
PhantomHunter	**85.59**	**99.60**	**92.59**	<u>99.12</u>	**95.81**	**99.89**	**97.85**	<u>98.22</u>	**90.67**	**97.36**	**94.01**	99.70	**89.91**	**97.26**	**93.58**	<u>99.99</u>
w/o BFE	66.45	98.74	82.59	**99.58**	67.54	98.77	83.16	**99.73**	81.60	<u>97.29</u>	89.45	99.77	69.86	87.91	78.89	99.84
w/o CL	78.29	99.31	88.80	96.34	92.44	<u>99.79</u>	96.12	96.84	81.97	94.33	88.15	**99.89**	82.82	95.08	88.95	<u>99.99</u>
w/o MoE	79.03	99.36	89.20	96.20	88.14	99.66	93.90	96.44	77.45	93.25	85.35	<u>99.80</u>	<u>83.05</u>	94.90	<u>88.97</u>	**100.00**

Table 4: Performance of commercial detectors (%). The two best results are bolded and underlined.

Method/Service	$\text{TPR}_{1\%\text{FPR}}$	F1_{Human}	F1_{Gen}	F1_{Macro}
AIorNot	8.02	23.73	57.91	40.82
Aliyun	9.98	23.46	52.52	37.99
BlueEyes	34.23	50.61	89.54	70.08
Copyleaks	25.51	23.12	51.43	37.28
GPTZero	25.52	23.02	51.55	37.28
Originality.ai	25.51	23.01	51.50	37.25
Phrasly.ai	29.89	21.45	41.69	31.57
Sapling	24.66	60.80	94.76	77.78
TencentCloud	<u>69.32</u>	<u>73.57</u>	<u>96.71</u>	<u>85.14</u>
Turnitin	25.59	22.98	51.59	37.29
ZeroGPT	25.51	23.34	51.58	37.46
PhantomHunter	**92.42**	**87.65**	**99.37**	**93.51**

From the results, we observe that removing any component leads to a degradation of detection performance, especially for BFE, since it is the probabilistic characterization of the base model that embodies the family commonality. Without contrastive loss, the probability representations of different family-generated texts become less distinguishable in the feature space. Due to the removal of the mixture-of-experts module, the classifier is forced to find a decision boundary between *all* family-generated texts and human texts without specializing in particular family models, which makes the learning process more difficult and thus lowers the detection performance.

4.5 Further Analysis (EQ3)

Comparison with Commercial Detectors. To validate the real-world difficulties of text from privately-tuned LLMs, we test 11 accessible commercial detectors that support API calling with the whole test set (Appendix B). Due to the inaccessibility, we do not consider the commercial detectors with the requirement of additional actions besides API callings (*e.g.*, Grammarly requires a subscription to the enterprise plan first) or have strict constraints in request times (*e.g.*, Zhuque AI Detector allows only 5 requests per day). We remove the invalid samples for each detector.

We assume that commercial detectors have learned the traits of the base models of involved families and been trained on a larger and more diverse training corpus than ours, but were not specially trained for defending against fine-tuning attacks. However, Table 4 shows that PhantomHunter, trained on a small but focused dataset instead, significantly outperforms the compared ones, again highlighting the value of the proposed family-aware learning. We acknowledge that this is not a completely fair comparison, but it shows the real-world impacts of privately-tuned LLMs and how PhantomHunter may survive under such attacks.

Generalizability Test on the Public Benchmark DetectRL To evaluate the performance of our framework on the public benchmark, we conduct *direct* inference of PhantomHunter and compare baselines trained on our corpus only against the representative public benchmark DetectRL [46], which covers diverse LLMs and multiple domains. The setting encompasses 4 distinct domains, including academic abstracts from arXiv (specifically, completely different from the arXiv data in the training set), news document summaries from XSum, creative writing stories from Writing Prompts, and restaurant reviews from Yelp. Each domain contains 1,008 samples, resulting in a total of 4,032 test instances. This balanced multi-domain setting enables a comprehensive evaluation of detection performance across different content types and writing styles. We followed the parameter configurations and feature extraction process from the main experiments described in our paper. We did not perform any training on this dataset and only conducted inference. From the results in Table 5, we see that PhantomHunter achieves a macro F1 score of 80.22% (84.82% for human-written text and 76.15% for LLM-generated text). Across the DetectEval multi-domain split (arXiv, WritingPrompts, XSum, and Yelp), it retains an average macro F1 score of 77.1%, confirming reliable cross-domain generalization without domain-specific tuning.

Table 5: Generalization performance comparison among different methods (%), tested on the DetectRL benchmark. The two best results are bolded and underlined.

Method	$F1_{Human}$	$F1_{Gen}$	$F1_{Macro}$
RoBERTa	00.00	<u>78.71</u>	39.36
T5-Sentinel	26.87	57.32	42.09
SeqXGPT	<u>81.04</u>	67.25	<u>74.15</u>
DetectGPT	61.90	54.91	58.41
DNA-GPT	67.58	60.77	64.18
FastDetectGPT	74.04	64.65	69.35
DeTeCtive	48.17	63.86	56.02
PhantomHunter	**84.82**	**76.15**	**80.22**
w/o BFE	56.21	71.20	63.70
w/o CL	75.45	67.29	71.37
w/o MoE	64.64	72.63	68.64

A Practice of Including LLMs beyond Known Families. In this section, we explore how to extend our framework to detect other families (e.g., ChatGPT, Claude, and other closed-source LLMs) where base model features are not available. In practice, AIGT detectors should maintain good performance against both known and unknown family models. Here, we show that PhantomHunter can be easily extended to be compatible with LLMs beyond known families through a simple modification. Specifically, an additional category "others" is added to the label space of the family classifier MLP_F, which is used as the family label of these texts from models beyond known families, and the number of experts in MoE is increased accordingly. In detail, we conduct two experiments: 1. We introduced GPT-4o mini[3] and Claude-3.7 Sonnet[4] to generate texts, which were labeled as "others". These were used for training and testing. 2. Based on the experiment above, DeepSeek-v3[5] and GLM-4-Air[6] were further introduced during the evaluation phase as entirely new LLM families that were unseen in training. In addition, the settings followed those of the main experiment. Table 6 shows that PhantomHunter maintains high performance not only for private fine-tuning LLMs but also for GPT-4o mini and Claude 3.7 Sonnet, indicating its good compatibility and extendibility to more diverse LLMs. For new entirely unseen LLMs(DeepSeek, GLM), PhantomHunter also demonstrates strong generalization capabilities.

4.6 Exploration of Source Family Prediction

Results of Source Family Prediction. Besides enhancing the learning of the LLMGT detection head, the introduction of a family classifier makes it possible to provide a source family prediction simultaneously, which facilitates further forensics analysis and may improve users' trustworthiness in the detector. Table 7 previews the current performance of known family prediction. The performance is mixed. The highest F1 score exceeds 95% while the lowest is below 55%, demonstrating that the family prediction is far more challenging than LLMGT detection and the current design requires improvement for family prediction. Recalling the main results, this also implies that even a moderate family-aware learning could bring performance improvement, indicating that its potential remains to be further explored.

[3]https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/
[4]https://www.anthropic.com/news/claude-3-7-sonnet
[5]https://github.com/deepseek-ai/DeepSeek-V3
[6]https://github.com/zai-org/GLM-4

Table 6: Performance on LLMs beyond known families by adding the "others" family class. PhantomHunter generalizes to GPT-4o mini, Claude-3.7 Sonnet, and entirely unseen LLM families (DeepSeek-v3.5 and GLM-4-Air6).

Model	$TPR_{1\%FPR}$	$F1_{Human}$	$F1_{Gen}$	$F1_{Macro}$
Known families (seen during training)				
LLaMA-cs	99.90	97.56	99.76	98.66
Mistral-cs	99.90	98.04	99.80	98.92
Gemma-cs	99.87	97.56	99.68	98.62
Qwen-cs	98.98	93.02	99.41	96.22
Others (seen during training)				
GPT-4o mini	99.67	97.56	99.17	98.37
Claude 3.7 Sonnet	100.00	98.04	99.50	98.77
Others (unseen during training)				
DeepSeek-v3	99.40	98.04	98.11	98.08
GLM-4-air	99.77	98.04	98.11	98.07

Table 7: F1 scores for family classification of unseen fine-tuned LLM-generated texts.

Dataset		LLaMA	Mistral	Gemma	Qwen	$F1_{Macro}$
arXiv	Full	75.61	63.71	95.76	87.21	80.57
	LoRA	78.32	77.30	56.63	79.11	72.84
Q&A	Full	82.63	50.29	66.29	56.89	64.03
	LoRA	68.21	50.18	71.82	61.10	62.83

Case Analysis. To better understand the limitations of the current family classifier, we further examine misclassified samples using the *arXiv-Full* subset and analyze them from syntactic, lexical, stylistic, and structural perspectives. Our observations reveal several characteristic patterns:

1) The Gemma family exhibits highly distinctive and simplified syntactic behavior. Its outputs contain very few punctuation marks, shallow clause structures, and extremely low structural complexity ($\approx$ 0.05), below LLaMA, Mistral, and human texts ($\approx$ 0.12–0.14). Although Gemma shows unusually high lexical diversity(TTR $\approx$ 0.94), it uses shorter words (average length $\approx$ 5.3). This extreme combination of "high diversity + short tokens" results in a unique feature island that the detector can recognize reliably.

2) Differently, the Qwen family unusually has long outputs ($\approx$ 542words) and prefers lexical reuse (TTR$\approx$ 0.327), which serve as strong indicators. Even ignoring length, its moderate clause density and long-tail lexical statistics (e.g., modal density 0.22, verb-like ratio 0.214) form a characteristic signature that is easy for the classifier to isolate.

3) For LLaMA and Mistral families, their syntactic density, structural complexity ($\approx$ 0.12–0.14), and punctuation usage closely resemble human texts. This presents the most challenging cases. Their grammatical ratios—such as modal verbs, verb density, and pronoun usage—fall between each other and human writing, resulting in substantial overlap. Moreover, stylistic cues (e.g., word length, lexical richness) also align closely with human distributions. Because no single feature distinctly differentiates these models from humans, the classifier often misclassifies them.

Overall, Gemma and Qwen families form clearly separable "feature islands", whereas LLaMA and Mistral cluster tightly with human writing across multiple linguistic dimensions, explaining why the classifier achieves higher accuracy on the former and often mispredicts the latter.

4.7 Cost Analysis

This section is to perform an analysis of the training and inference cost of PhantomHunter. We will show that despite the need of loading base LLMs, the overall computational cost of our method remains acceptable for real-world deployment.

- **For inference**, PhantomHunter relies on four open-source models (Qwen2.5-7B, Gemma-7B-it, Mistral-7B-Instruct-v0.1, and LLaMA-2-7B-Chat) for probability feature extraction, each requiring roughly 14GB of GPU memory. The end-to-end latency is 37ms per input instance, including 28ms for sampling-based feature collection and 900ms for the classification step.
- **For training**, PhantomHunter processes 1.15M tokens per epoch and completes an epoch in about 45s on an NVIDIA A800 GPU.

This demonstrates that our method achieves a significant improvement in accuracy at an acceptable cost. While the overall runtime remains practical, the reliance on multiple base models constitutes a cost bottleneck. In this work, we prioritize the possibility validation and leave this as future work. The optimization may involve model distillation, feature sharing, or unified surrogate modeling.

5 Related Works

LLM-Generated Text Detection. Existing methods can be categorized as model-based and metric-based ones. The former generally trains neural models to analyze the textual characteristics of LLM outputs, exploiting semantic features [13, 14, 49] and self-familiarity [20]. The latter often sets similarity metrics based on LLMs' output probability or behavioral characteristics [15, 32, 50, 52]. Recent methods collectively address different aspects of the LLM detection challenge, from data efficiency [53], to generalizability across models and domains [16], and training paradigms [45]. However, existing methods and evaluation benchmarks [17, 44, 46] focus on publicly known LLMs. Such ignorance leads to detectors' unsatisfying performance when countering texts from privately-tuned LLMs, as we experimentally showed before.

One possible solution is to add model-specific watermarks to the outputs as identifiers [28, 29], but this approach requires LLM providers to pay extra costs, and recent research warns that LLM watermarks could be distorted or even removed through various adversarial techniques [34, 47]. PhantomHunter showcases a non-watermarking solution to improve detection for text from unseen, privately-tuned LLMs, and it could work with watermarking solutions to more comprehensively lower the detection difficulty of text generated by privately-tuned LLMs.

Family Analysis of LLMs. Analyzing commonalities and uniqueness of LLMs from a family perspective yields a deep understanding of LLMs' relationship [24], which is useful for LLM attributions [25, 39] and LLMs' license violation detection [10]. As shown by Sarvazyan et al. [37] and Thorat and Yang [40], the family factor could influence text detectability and detectors' generalizability. However, they only investigated the base models of different sizes, ignoring the unseen privately-tuned LLMs that we focuses on in this research. In this work, we construct a new dataset containing texts generated by 48 LLMs that were privately fine-tuned with a domain-specific corpus with two types of fine-tuning approaches. Our study is expected to provide new knowledge about the LLMGT's detectability from the perspective of LLM post-training processes.

6 Conclusion and Discussions

We proposed PhantomHunter, which is specialized to improve the detection performance of text generated by unseen, privately-tuned LLMs. We first experimentally revealed that fine-tuning open-source base models would significantly lower the detectability of generated text, and then addressed this issue via family-aware learning to capture shared traits in each LLM family. Comparison experiments with seven methods and four industrial services confirmed PhantomHunter's superiority. The further analysis exhibited that PhantomHunter has good compatibility and extendibility to better adapt to real-world situations.

Discussions on Limitations. Though PhantomHunter showcases how to tackle the performance degradation issue when encountering texts generated by privately-tuned LLMs based on widely-known open-source ones, we also identify the following limitations of PhantomHunter, which may influence the real-world implementation. **First**, the current version only supports binary classification between human-written and LLM-generated text, plus the possible family source for a suspicious LLM-generated text piece. It does not provide fine-grained sentence- or token-level annotations [43], and the current performance of identifying family sources is moderate. **Second**, due to the experimental cost constraint, our experiment only covered several common LLM families, and the performance on other families like DeepSeek [8, 9] remains unknown. **Third**, extracting features from base LLMs requires them to be locally deployed and thus increases the memory cost and the inference computation overhead, which trades computation costs for higher detection performance. Therefore, these directions require further exploration, and we advocate for more industrial solutions to improve the trustworthiness and efficiency of LLMGT systems. Meanwhile, it is not recommended that the feedback of PhantomHunter serve as a sole basis for real-world actions (e.g, disciplinary action against any student). Additional harmful content detection [18, 27, 38] and human verification [33] are needed.

A Introduction and Implementation of Compared Baselines

- *RoBERTa* [13]: A fine-tuned RoBERTa [30] that uses semantic features to distinguish between human and AI-generated text, which has been adopted as a typical baseline for this task.
- *T5-Sentinel* [7]: A model based on T5 [36] that treats token prediction as implicit classification to detect generated text.

- *SeqXGPT* [43]: A white-box method that treats token probability lists as waveforms and applies CNN and attention networks for detection. In the experiments, we use Mistral-7b [21], Llama-2-7b-chat [41], Qwen2.5-7B-Instruct [2], and gemma-7b [12] as white-box models to compute the probability lists.
- *DNA-GPT* [48]: A source-attribution method that leverages multiple regenerations to identify model-specific statistical fingerprints, functioning in both black-box and white-box settings. In the experiment, we adopted the white-box setting and selected Llama-2-7b-chat [41] as the proxy base model, with regenerations num N = 10, the truncation ratio $\gamma = 0.5$, and max new tokens num t = 250. Since this method is training-free, we conducted a grid search for the classification threshold with a step size of 1e-4, and reported the best result as the upper performance bound of this method.
- *DetectGPT* [32]: A zero-shot detection approach that identifies AI-generated text by comparing the probability of original passages against multiple perturbed variants without requiring model fine-tuning. In the experiment, we selected T5-3B [36] as the mask-filling model, and Llama-2-7b-chat [41] as the proxy base model, with the number of perturbations as 100. In the experiment, we follow a similar implementation to DNA-GPT.
- *Fast-DetectGPT* [3]: An efficient improvement over DetectGPT that introduces the concept of conditional probability curvature to identify machine-generated text, while significantly reducing computational overhead through optimized perturbation sampling strategies that maintain or enhance detection performance. In the experiment, we selected gpt-neo-2.7b [4] as the sampling model and the scoring model.
- *DALD* [51]: A plug-and-play framework designed to enhance logits-based detectors in black-box settings. Unlike prior surrogate-based methods, DALD aligns the surrogate model's probability distribution with that of the unknown target model. The aligned surrogate model significantly improves curvature-based or probability-divergence–based detection metrics. Following the original paper's setup: Fine-tuning the surrogate model, Llama2-7B, on the WildChat dataset.
- *DeTeCtive* [14]:DeTeCtive combines multi-level contrastive learning and multi-task auxiliary learning methods to enable fine-grained feature extraction that captures relationships between different text sources.In the experiments, We use SimCSE-RoBERTa-base [11] as the pretrain model. To preserve the original semantic space and prevent overfitting, we freeze the model's embedding layer during training. During testing, we adopt a KNN-based retrieval strategy using the validation set to automatically determine the optimal number of neighbors K within the range of 1 to 5, which is then used for predicting out-of-distribution data.

B Introduction of Compared Commercial Detectors

The considered commercial detectors are as follows[7]:

[7]Official Pages: https://help.aliyun.com/document_detail/2979119.html, https://www.aiornot.com/, http://ruimei.ruijianai.com/, https://phrasly.ai/ai-detector, https://sapling.ai/, https://cloud.tencent.com/document/product/1124/118694, https://gptzero.me, https://www.zerogpt.com, https://originality.ai, https://www.turnitin.com, https://copyleaks.com

- *Aliyun:* The Content Moderation Service provided by Aliyun offers an AI-generated text detector, which analyzes the input content and returns judgment labels along with a confidence score. We truncate the input text if it exceeds the maximum length constraint of 600.
- *AIorNot:* AIorNot provides the continually updated detector trained on the latest generative models to provide a clear verdict and a confidence score for text provenance.
- *BlueEyes:* A text auditing service launched by Ruijian AI, which supports LLM-generated text detection in Chinese and English. We access the API equipped with the latest version.
- *Phrasly.ai:* A detection service that identifies patterns in language, sentence structure, and predictability. It claims a 99.8% accuracy rate in distinguishing human-written text from LLM-generated text from ChatGPT, Claude, and Gemini.
- *Sapling:* Sapling AI Content Detector leverages a Transformer-based system, similar in architecture to the models used to generate AI content. It is designed to detect text from various LLMs such as GPT-4, Claude, and Gemini. The official documentation reports an accuracy of 97% with a <3% false positive rate.
- *TencentCloud:* The AI-generated text detection service from Text Moderation System of Tencent's T-Sec series. It performs detection by analyzing both its formal features and content characteristics, and covers mainstream LLMs and diverse domains. The maximum input length is 10,000 characters.
- *GPTZero:* One of the most widely used AI detectors that utilizes a multi-step approach with seven components to analyze text, boasting high accuracy for major LLMs. Independent benchmarks often rank it as a leader in precision.
- *ZeroGPT:* A popular detection tool that employs sophisticated algorithms trained on millions of articles. It provides a percentage-based score to indicate varying levels of AI probability.
- *Originality.ai:* Its AI detector is claimed to be highly sensitive to the latest LLMs (like GPT-4.5 and Claude 3.7). We use the standard version.
- *Turnitin:* Turnitin integrated AI writing detection into its similarity report workflow. Its model is built on a transformer deep-learning architecture specifically trained to identify human writing versus generative AI.
- *Copyleaks:* It offers high-accuracy AI detection (over 99%) with a very low false-positive rate (0.2% as claimed) and is capable of detecting AI-generated text that has been heavily paraphrased.

C Supplementary Implementation Details

The probability features are first fed into a five-layer one-dimensional convolutional network $f: L \rightarrow Z$. This module extracts local latent representations through specific kernel sizes (5, 3, 3, 3, 3) and uniform strides (1, 1, 1, 1, 1), with channel dimensions varying across layers (64, 128, 128, 128, 64). Subsequently, the output is passed to a contextual network $g: Z \rightarrow C$, which consists of two Transformer encoder layers configured with 4 attention heads and a 256-dimensional feed-forward network. Equipped with fixed positional embeddings, this module leverages self-attention mechanisms to capture contextual features.

References

[1] Alizlia. 2025. Alibaba Unveils New Qwen3 Models for Coding, Complexing Reasoning and Machine Translation . https://www.alizila.com/alibaba-unveils-new-qwen3-models-for-coding-complexing-reasoning-and-machine-translation/. Accessed: 2026-01-22.

[2] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen Technical Report. arXiv:2309.16609 [cs.CL]

[3] 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*. https://openreview.net/forum?id=Bpcgcr8E8Z

[4] Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. *GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow*. doi:10.5281/zenodo.5297715

[5] Canyu Chen and Kai Shu. 2024. Combating Misinformation in the Age of LLMs: Opportunities and Challenges. *AI Magazine* 45, 3 (2024), 354–368. doi:10.1002/aaai.12188

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In *Proceedings of the 37th International Conference on Machine Learning*. PMLR, 1597–1607. https://proceedings.mlr.press/v119/chen20j.html

[7] Yutian Chen, Hao Kang, Vivian Zhai, Liangze Li, Rita Singh, and Bhiksha Raj. 2023. Token Prediction as Implicit Classification to Identify LLM-Generated Text. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 13112–13120. doi:10.18653/v1/2023.emnlp-main.810

[8] DeepSeek-AI. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. arXiv:2501.12948 [cs.CL] https://arxiv.org/abs/2501.12948

[9] DeepSeek-AI. 2025. DeepSeek-V3 Technical Report. arXiv:2412.19437 [cs.CL] https://arxiv.org/abs/2412.19437

[10] Myles Foley, Ambrish Rawat, Taesung Lee, Yufang Hou, Gabriele Picco, and Giulio Zizzo. 2023. Matching Pairs: Attributing Fine-Tuned Models to their Pre-Trained Large Language Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 7423–7442. doi:10.18653/v1/2023.acl-long.410

[11] Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple Contrastive Learning of Sentence Embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 6894. https://aclanthology.org/2021.emnlp-main.552.pdf

[12] Gemma Team. 2024. Gemma: Open Models Based on Gemini Research and Technology. arXiv:2403.08295 [cs.CL] https://arxiv.org/abs/2403.08295

[13] 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. arXiv:2301.07597 [cs.CL]

[14] Xun Guo, Shan Zhang, Yongxin He, Ting Zhang, Wanquan Feng, Haibin Huang, and Chongyang Ma. 2024. DeTeCtive: Detecting AI-generated Text via Multi-Level Contrastive Learning. In *Advances in Neural Information Processing Systems*, Vol. 37. Curran Associates, Inc., 88320–88347. https://proceedings.neurips.cc/paper_files/paper/2024/file/a117a3cd54b7affad04618c77c2fb18b-Paper-Conference.pdf

[15] 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. arXiv:2401.12070 [cs.CL]

[16] Wei Hao, Ran Li, Weiliang Zhao, Junfeng Yang, and Chengzhi Mao. [n. d.]. Learning to Rewrite: Generalized LLM-Generated Text Detection. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 6421–6434. doi:10.18653/v1/2025.acl-long.322

[17] Xinlei He, Xinyue Shen, Zeyuan Chen, Michael Backes, and Yang Zhang. 2024. MGTBench: Benchmarking Machine-Generated Text Detection. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2251–2265. doi:10.1145/3658644.3670344

[18] Beizhe Hu, Qiang Sheng, Juan Cao, Yuhui Shi, Yang Li, Danding Wang, and Peng Qi. 2024. Bad actor, good advisor: Exploring the role of large language models in fake news detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 22105–22113. doi:10.1609/aaai.v38i20.30214

[19] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *The Tenth International Conference on Learning Representations*. https://openreview.net/forum?id=nZeVKeeFYf9

[20] Jiazhou Ji, Jie Guo, Weidong Qiu, Zheng Huang, Yang Xu, Xinru Lu, Xiaoyu Jiang, Ruizhe Li, and Shujun Li. 2025. "I know myself better, but not really greatly": Using LLMs to Detect and Explain LLM-Generated Texts. arXiv:2502.12743 [cs.CL] https://arxiv.org/abs/2502.12743

[21] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. arXiv:2310.06825 [cs.CL] https://arxiv.org/abs/2310.06825

[22] Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. 2023. Challenges and Applications of Large Language Models. arXiv:2307.10169 [cs.CL] https://arxiv.org/abs/2307.10169

[23] Ryuto Koike, Masahiro Kaneko, and Naoaki Okazaki. 2024. OUTFOX: LLM-Generated Essay Detection Through In-Context Learning with Adversarially Generated Examples. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 21258–21266. doi:10.1609/aaai.v38i19.30120

[24] Tharindu Kumarage, Garima Agrawal, Paras Sheth, Raha Moraffah, Aman Chadha, Joshua Garland, and Huan Liu. 2024. A Survey of AI-generated Text Forensic Systems: Detection, Attribution, and Characterization. arXiv:2403.01152 [cs.CL] https://arxiv.org/abs/2403.01152

[25] Tharindu Kumarage and Huan Liu. 2023. Neural Authorship Attribution: Stylometric Analysis on Large Language Models . In *2023 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*. IEEE Computer Society, 51–54. doi:10.1109/CyberC58899.2023.00019

[26] Linyang Li, Pengyu Wang, Ke Ren, Tianxiang Sun, and Xipeng Qiu. 2023. Origin tracing and detecting of llms. *arXiv preprint arXiv:2304.14072* (2023).

[27] Yang Li, Qiang Sheng, Yehan Yang, Xueyao Zhang, and Juan Cao. 2025. From Judgment to Interference: Early Stopping LLM Harmful Outputs via Streaming Content Monitoring. *arXiv preprint arXiv:2506.09996* (2025).

[28] Aiwei Liu, Leyi Pan, Yijian Lu, Jingjing Li, Xuming Hu, Xi Zhang, Lijie Wen, Irwin King, Hui Xiong, and Philip Yu. 2024. A Survey of Text Watermarking in the Era of Large Language Models. *Comput. Surveys* 57, 2 (2024), 1–36. doi:10.1145/3691626

[29] Aiwei Liu, Qiang Sheng, and Xuming Hu. 2024. Preventing and Detecting Misinformation Generated by Large Language Models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, 3001–3004. doi:10.1145/3626772.3661377

[30] 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. arXiv:1907.11692 [cs.CL] https://arxiv.org/abs/1907.11692

[31] Meta AI. 2024. Llama Usage Doubled May Through July 2024. https://ai.meta.com/blog/llama-usage-doubled-may-through-july-2024/. Accessed: 2025-03-21.

[32] Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. 2023. DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature. In *Proceedings of the 40th International Conference on Machine Learning*. PMLR, 24950–24962. https://proceedings.mlr.press/v202/mitchell23a.html

[33] Chenhao Niu, Kevin P. Yancey, Ruidong Liu, Mirza Basim Baig, André Kenji Horie, and James Sharpnack. 2024. Detecting LLM-Assisted Cheating on Open-Ended Writing Tasks on Language Proficiency Tests. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*. Association for Computational Linguistics, 940–953. doi:10.18653/v1/2024.emnlp-industry.70

[34] Qi Pang, Shengyuan Hu, Wenting Zheng, and Virginia Smith. 2024. No Free Lunch in LLM Watermarking: Trade-offs in Watermarking Design Choices. In *Advances in Neural Information Processing Systems*, Vol. 37. Curran Associates, Inc., 138756–138788. https://proceedings.neurips.cc/paper_files/paper/2024/file/fa86a9c7b9f341716ccb679d1aeb9afa-Paper-Conference.pdf

[35] Giovanni Puccetti, Anna Rogers, Chiara Alzetta, Felice Dell'Orletta, and Andrea Esuli. 2024. AI 'News' Content Farms Are Easy to Make and Hard to Detect: A Case Study in Italian. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 15312–15338. doi:10.18653/v1/2024.acl-long.817

[36] 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 (2020), 1–67. http://jmlr.org/papers/v21/20-074.html

[37] Areg Mikael Sarvazyan, José Ángel González, Paolo Rosso, and Marc Franco-Salvador. 2023. Supervised Machine-Generated Text Detectors: Family and Scale Matters. In *International Conference of the Cross-Language Evaluation Forum for European Languages*. Springer, 121–132. doi:10.1007/978-3-031-42448-9_11

[38] Mrinank Sharma, Meg Tong, Jesse Mu, Jerry Wei, Jorrit Kruthoff, Scott Goodfriend, Euan Ong, Alwin Peng, Raj Agarwal, Cem Anil, Amanda Askell, Nathan

Bailey, Joe Benton, Emma Bluemke, Samuel R. Bowman, Eric Christiansen, Hoagy Cunningham, Andy Dau, Anjali Gopal, Rob Gilson, Logan Graham, Logan Howard, Nimit Kalra, Taesung Lee, Kevin Lin, Peter Lofgren, Francesco Mosconi, Clare O'Hara, Catherine Olsson, Linda Petrini, Samir Rajani, Nikhil Saxena, Alex Silverstein, Tanya Singh, Theodore Sumers, Leonard Tang, Kevin K. Troy, Constantin Weisser, Ruiqi Zhong, Giulio Zhou, Jan Leike, Jared Kaplan, and Ethan Perez. 2025. Constitutional Classifiers: Defending against Universal Jailbreaks across Thousands of Hours of Red Teaming. arXiv:2501.18837 [cs.CL] https://arxiv.org/abs/2501.18837

[39] Yuhui Shi, Qiang Sheng, Juan Cao, Hao Mi, Beizhe Hu, and Danding Wang. 2024. Ten Words Only Still Help: Improving Black-Box AI-Generated Text Detection via Proxy-Guided Efficient Re-Sampling. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, 494–502. doi:10.24963/ijcai.2024/55

[40] Shantanu Thorat and Tianbao Yang. 2024. Which LLMs are Difficult to Detect? A Detailed Analysis of Potential Factors Contributing to Difficulties in LLM Text Detection. arXiv:2410.14875 [cs.CL] https://arxiv.org/abs/2410.14875

[41] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288 [cs.CL] https://arxiv.org/abs/2307.09288

[42] Brian Tufts, Xuandong Zhao, and Lei Li. 2025. A Practical Examination of AI-Generated Text Detectors for Large Language Models. arXiv:2412.05139 [cs.CL]

[43] Pengyu Wang, Linyang Li, Ke Ren, Botian Jiang, Dong Zhang, and Xipeng Qiu. 2023. SeqXGPT: Sentence-Level AI-Generated Text Detection. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1144–1156. https://aclanthology.org/2023.emnlp-main.73/

[44] Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Osama Mohanned Afzal, Tarek Mahmoud, Giovanni Puccetti, Thomas Arnold, et al. 2024. M4GT-Bench: Evaluation Benchmark for Black-Box Machine-Generated Text Detection. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 3964–3992. doi:10.18653/v1/2024.acl-long.218

[45] Chenwang Wu, Yiu ming Cheung, Bo Han, and Defu Lian. 2025. Advancing Machine-Generated Text Detection from an Easy to Hard Supervision Perspective. arXiv:2511.00988 [cs.CL] https://arxiv.org/abs/2511.00988

[46] 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*, Vol. 37. Curran Associates, Inc., 100369–100401. https://proceedings.neurips.cc/paper_files/paper/2024/file/b61bdf7e9f64c04ec75a26e781e2ad51-Paper-Datasets_and_Benchmarks_Track.pdf

[47] Qilong Wu and Varun Chandrasekaran. 2024. Bypassing LLM Watermarks with Color-Aware Substitutions. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 8549–8581. doi:10.18653/v1/2024.acl-long.464

[48] 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*. https://openreview.net/forum?id=Xlayxj2fWp

[49] Xiao Yu, Kejiang Chen, Qi Yang, Weiming Zhang, and Nenghai Yu. 2024. Text Fluoroscopy: Detecting LLM-Generated Text through Intrinsic Features. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 15838–15846. doi:10.18653/v1/2024.emnlp-main.885

[50] Xiao Yu, Yuang Qi, Kejiang Chen, Guoqiang Chen, Xi Yang, Pengyuan Zhu, Xiuwei Shang, Weiming Zhang, and Nenghai Yu. 2024. DPIC: Decoupling Prompt and Intrinsic Characteristics for LLM Generated Text Detection. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=BZh05P2EoN

[51] Cong Zeng, Shengkun Tang, Xianjun Yang, Yuanzhou Chen, Yiyou Sun, Zhiqiang Xu, Yao Li, Haifeng Chen, Wei Cheng, and Dongkuan DK Xu. 2024. Dald: Improving logits-based detector without logits from black-box llms. *Advances in Neural Information Processing Systems* 37 (2024), 54947–54973.

[52] Biru Zhu, Lifan Yuan, Ganqu Cui, Yangyi Chen, Chong Fu, Bingxiang He, Yangdong Deng, Zhiyuan Liu, Maosong Sun, and Ming Gu. 2023. Beat LLMs at Their Own Game: Zero-Shot LLM-Generated Text Detection via Querying ChatGPT. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 7470–7483. doi:10.18653/v1/2023.emnlp-main.463

[53] Xiaowei Zhu, Yubing Ren, Fang Fang, Qingfeng Tan, Shi Wang, and Yanan Cao. 2025. DNA-DetectLLM: Unveiling AI-Generated Text via a DNA-Inspired Mutation-Repair Paradigm. arXiv:2509.15550 [cs.CL] https://arxiv.org/abs/2509.15550