Leveraging Human and Machine Preferences for Zero-shot Detection of AI-Generated Text

Lei Jiang[1], Desheng Wu[1*], Xiaolong Zheng[2], Cuicui Luo[1]

[1]University of Chinese Academy of Sciences, Beijing, China
[2]Institute of Automation, Chinese Academy of Sciences, Beijing, China
jianglei232@mails.ucas.ac.cn, dwu@ucas.ac.cn, xiaolong.zheng@ia.ac.cn,
luocuicui@ucas.ac.cn

Abstract

In recent years, the rapid advancement of large language models (LLMs) has enabled generated texts to closely mimic human writing, posing significant challenges to the detection of AI-generated content. Current mainstream zero-shot detection methods largely adopt a machine-centric perspective, relying on proxy models to compute token-level AI-likelihood scores and treating all tokens equally during overall detection. However, such approaches overlook the prediction discrepancies that arise when humans and large language models interpret the same text. We argue that tokens exhibiting greater divergence between human and machine predictions can provide stronger clues for determining the authorship of a text. To address this limitation, we propose **HAPDA**—a human-machine prediction discrepancy adapter for AI-generated text detection (AGTD). The framework consists of two core components: (1) a joint fine-tuning strategy for training paired human-preference and machine-preference models, and (2) a discrepancy-aware reweighting mechanism designed to calibrate token-level detection scores in downstream detectors. Extensive experiments demonstrate that HAPDA consistently and significantly enhances the detection performance of five representative baseline models under various evaluation scenarios.

1 Introduction

In recent years, LLMs have made remarkable strides, substantially enhancing the capabilities of natural language processing technologies. The text generated by these models now rivals human writing in fluency and coherence (Li, 2025). However, as LLMs see wider adoption in real-world applications, their associated risks have become increasingly evident, including challenges in verifying content authenticity (Zhang et al., 2025a), the spread of misinformation (Yin et al., 2024), and potential misuse (Abdali et al., 2024). Consequently, developing efficient and accurate methods for detecting AI-generated text has become critically important.

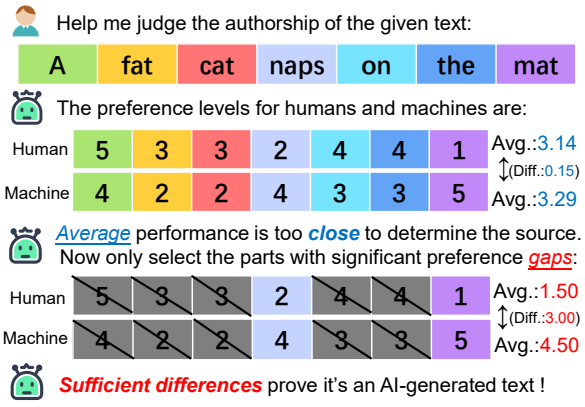

Figure 1: Illustration of human-machine predictive discrepancy in AGTD tasks. For the same piece of text, humans and machines may make similar predictions for some tokens (e.g., "A" and "the"). In such cases, tokens with significant predictive discrepancies (e.g., "naps" and "mat") often carry more information for authorship attribution.

A class of zero-shot AGTD methods computes token-level "AI-likeness" scores (e.g., entropy (Gehrmann et al., 2019), probability (Solaiman et al., 2019)) and averages them to obtain a document-level score, classifying the text as AI-generated based on a predefined threshold. In our work, we refer to this family of methods as **MeanZero**. They offer the advantages of being fast, convenient and requiring no labeled training data, but their accuracy degrades when dealing with high-quality text generated by LLMs.

Existing research has increasingly recognized that not all tokens in a text contribute equally to identifying the author's identity. The discriminative power of low-probability tokens is highlighted by POGER (Shi et al., 2024); necessary tokens are

Findings of the Association for Computational Linguistics: ACL 2026, pages 13732–13750

extracted by PECOLA (Liu et al., 2024b) using YAKE (Campos et al., 2020); textual coherence is modeled by CoCo (Liu et al., 2023) using a graph structure; and dynamic token weights based on semantics, context, and positional information are assigned by PAWN (Miralles-González et al., 2025). However, these methods are often based on supervised learning, which incurs high training costs and is prone to overfitting to specific training data. In addition, although MeanZero methods are enhanced using log-rank information by DetectLRR and DetectNPR (Su et al., 2023), they still operate solely from the LLMs' perspective and overlook the human generation mechanism.

Within the same passage, there exist significant differences in the predicted probability distributions of specific tokens between humans and machines (Ippolito et al., 2020); such differences can be regarded as the informational association between the token and the latent variable of "author identity." Besides, if a token's generation probability distribution differs greatly between human and machine models, it indicates higher **mutual information** for inferring author identity (Yoo et al., 2024; West et al., 2025; Zhang et al., 2025b). These tokens contribute more effectively to reducing uncertainty in the probabilistic space and are thus the most discriminative components. As suggested in Figure 1, our intuition is that *tokens exhibiting greater divergence in human and machine predictions offer stronger cues for authorship attribution*. We provide a mathematical analysis of this intuition from the perspective of mutual information in Appendix A. In this context, "human perspective" simulates an idealized human author's writing preferences by an optimized language model, favoring common human vocabulary and expressions. We model an idealized human author's aggregate word choice preference, not individual prediction stochasticity. This is analogous to LLMs learning text distributions, not specific personas. Similarly, the "machine perspective" mimics AI-generated text patterns through another approach.

Based on these two perspectives, we propose **HAPDA**, a framework designed to enhance the performance of MeanZero detectors. Motivated by Direct Preference Optimization (DPO) (Rafailov et al., 2023), we introduce a novel joint fine-tuning strategy called **HAPDA-Finetune**, which trains two auxiliary models based on an open-source language model. They are optimized to exhibit stronger preferences for human-written and AI-generated texts, allowing them to capture predictive discrepancies between human and machine perspectives. Based on these auxiliary models, we propose a token-level reweighting mechanism, termed **HAPDA-Calibration**, which leverages token-wise disagreement and uncertainty to assign higher weights to more discriminative tokens during the computation of the overall detection score. Our main contributions are summarized as follows:

(1) We are the first to reconsider the AGTD task from a joint human and machine predictive perspective.

(2) We design a novel joint fine-tuning strategy that obtains a pair of LLMs with stronger preferences for human-written and AI-generated texts, respectively.

(3) We model the predictive discrepancies between human and machine to provide token-level reweighting for detection scores in MeanZero detectors.

(4) Extensive experiments across multiple datasets demonstrate that HAPDA consistently and significantly improves the performance of baselines under diverse evaluation settings.

2 Related Work

AI-generated Text Detection. A common approach in AGTD exploits LLMs' tendency to generate tokens with higher conditional probabilities, reflecting greater "confidence." Metrics such as perplexity (Hans et al., 2024) and log probability (Xu et al., 2024a) have been used to identify AI-generated text. (Mitchell et al., 2023) show that perturbations reduce these probabilities, while other studies (Shi et al., 2024; Wang et al., 2023) treat token-level probability sequences as features for supervised detection. LLM text also often exhibits lower entropy (Gehrmann et al., 2019), which has been used in watermarking to select insertion points (Wu et al., 2025; Liu and Bu, 2024) and adjust watermark strength (Lu et al., 2024). Our work improves zero-shot AGTD by amplifying tokens with large prediction discrepancies between human and machine models.

Fine-tuning Strategies in AGTD Research. Fine-tuning has been widely used in AGTD. For example, Li et al. (2024) fine-tunes a rewriting model to amplify perturbations, Wang et al. (2025) uses reinforcement learning to humanize outputs of small LMs, Zeng et al. (2024) fine-tunes proxy models to better match source distributions, and Xu and

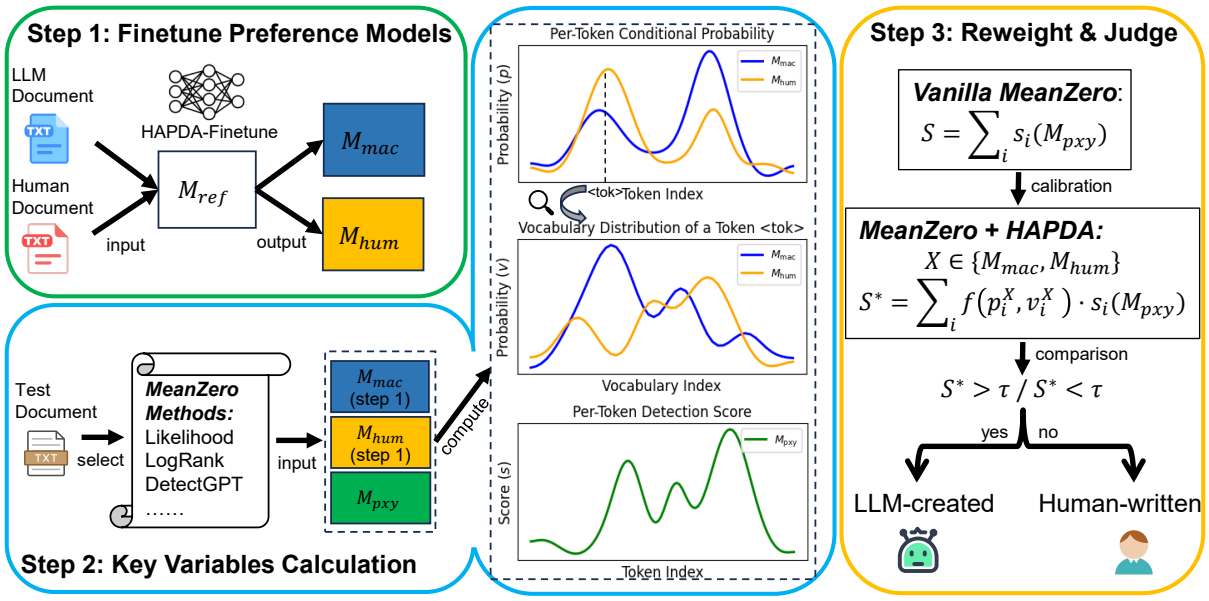

Figure 2: The workflow of HAPDA. Step 1: Fine-tune the machine-preference model $\mathcal{M}_{\text{mac}}$ and the human-preference model $\mathcal{M}_{\text{hum}}$ on a training corpus that contains both AI-generated and human-written texts. Step 2: Scan the input text to compute key variables. $\mathcal{M}_{\text{mac}}$ and $\mathcal{M}_{\text{hum}}$ are used to compute the per-token conditional probability p and vocabulary distribution v, while the proxy model $\mathcal{M}_{\text{pxy}}$ is used to compute the raw detection score of the selected MeanZero method. Step 3: Adjust the raw detection score using a reweighting coefficient derived from p and v, and compare the adjusted score against a predefined threshold τ to determine the authorship of the given text.

Sheng (2024) adapts DetectGPT (Mitchell et al., 2023) to code generation. Our work aims to obtain auxiliary models that capture human/machine preferences through fine-tuning.

3 Proposed Method

3.1 Task Description

We consider the AGTD task from a document composed of tokens $\{x_1, x_2, \ldots, x_n\}$. A class of methods, referred to as MeanZero in this work, computes a detection score for each token $s(x_i)$, and aggregates them via simple averaging $S = \sum_{i=1}^{n} s(x_i)/n$, which is compared against a threshold to determine the text's authorship.

We generalize this framework by introducing a token-wise weighting mechanism. Rather than treating all tokens equally, we assign each token x_i an importance weight w_i, yielding a weighted detection score $S^* = \sum_{i=1}^{n} w_i \cdot s(x_i)$. Therefore, we propose HAPDA, consisting of two components: HAPDA-Finetune (step 1 in Figure 2) and HAPDA-Calibration (steps 2 and 3 Figure 2).

HAPDA is not a supervised detector, but a generalizable adapter for zero-shot detectors. Its advantage lies not in outperforming supervised methods on in-domain data, but in providing a robust, flexible, and interpretable solution for real-world zero-shot scenarios where labeled data from the target domain is unavailable.

3.2 HAPDA-Finetune

The goal of HAPDA-Finetune is to fine-tune a pair of models—a *Human-preference Model* ($\mathcal{M}_{\text{hum}}$) and a *Machine-preference Model* ($\mathcal{M}_{\text{mac}}$)—from a white-box *reference model* ($\mathcal{M}_{\text{ref}}$), in order to achieve: *Alignment*: $\mathcal{M}_{\text{hum}}$ should assign higher scores to human-written texts, while $\mathcal{M}_{\text{mac}}$ should favor AI-generated texts; *Distinctiveness*: Under the same input, the scoring behaviors of $\mathcal{M}_{\text{hum}}$ and $\mathcal{M}_{\text{mac}}$ should diverge on both human and machine responses. We use these two models as specialized probes. The difference in their "opinions" (predictive distributions) on a given token reveals whether that token is more characteristic of human-like or machine-like statistical regularities. This is precisely the "human-machine prediction discrepancy" we aim to capture.

Alignment: Guides the models to maintain consistent preferences for texts from specific sources, thereby enhancing the base discriminative power of detection. To incorporate human preferences into our framework, we draw inspira-

tion from the prior work (Wang et al., 2025) and adapt the DPO strategy to simultaneously fine-tune two models with opposing objectives: one favoring machine-generated text and the other favoring human-written text. Given a prompt P, a human-written response H, and a AI-generated response M, we define the alignment loss for models $\mathcal{M}_{\text{mac}}$ and $\mathcal{M}_{\text{hum}}$ as:

$$\mathcal{L}_{\text{a}} = -\log \sigma(\Delta_{\text{mac}}) - \log \sigma(\Delta_{\text{hum}}),$$

where $\sigma(\cdot)$ is the sigmoid function. Δ_{mac} and Δ_{hum} denote the preference margins of $\mathcal{M}_{\text{mac}}$ and $\mathcal{M}_{\text{hum}}$, which are expressed as:

$$\Delta_{\text{mac}} = \log \frac{\pi_{\text{mac}}(M \mid P)}{\pi_{\text{ref}}(M \mid P)} - \log \frac{\pi_{\text{mac}}(H \mid P)}{\pi_{\text{ref}}(H \mid P)},$$
$$\Delta_{\text{hum}} = \log \frac{\pi_{\text{hum}}(H \mid P)}{\pi_{\text{ref}}(H \mid P)} - \log \frac{\pi_{\text{hum}}(M \mid P)}{\pi_{\text{ref}}(M \mid P)},$$

where $\pi(\cdot \mid \cdot)$ denotes the conditional probability. For $\mathcal{M}_{\text{mac}}$, M is treated as the preferred sample and H as the less preferred one; the opposite holds for $\mathcal{M}_{\text{hum}}$. A lower $\mathcal{L}_{\text{a}}$ indicates that the preference models are effectively aligned with their intended preference directions.

Distinctiveness: Encourages the models to produce different scoring behaviors on the same input, enhancing the separability between human and machine texts. To achieve this while maintaining alignment, we introduce a Jensen-Shannon (JS) divergence-based loss (Lin, 2002) to enhance the token-level distributional distinctiveness between $\mathcal{M}_{\text{hum}}$ and $\mathcal{M}_{\text{mac}}$. Specifically, for a given prompt P and a response $Y \in \{M, H\}$, we define:

$$p^Y_{\text{mac}} = \text{softmax}\big(\log \pi_{\text{mac}}(Y \mid P)\big),$$
$$p^Y_{\text{hum}} = \text{softmax}\big(\log \pi_{\text{hum}}(Y \mid P)\big),$$

where p^Y_{mac} and p^Y_{hum} represent the token-level probability distributions from $\mathcal{M}_{\text{mac}}$ and $\mathcal{M}_{\text{hum}}$, respectively. Then we define the mixture distribution and KL divergence (Kullback and Leibler, 1951):

$$m^Y = \frac{1}{2}(p^Y_{\text{mac}} + p^Y_{\text{hum}}),$$
$$\text{KL}(p||m) = \sum_i p_i \log \frac{p_i}{m_i}.$$

The JS divergence is expressed as :

$$\text{JS}(p^Y_{\text{mac}}||p^Y_{\text{hum}}) = \frac{1}{2}\,\text{KL}(p^Y_{\text{mac}}||m^Y) + \frac{1}{2}\,\text{KL}(p^Y_{\text{hum}}||m^Y).$$

Finally, the distinctiveness loss is then defined as:

$$\mathcal{L}_{\text{d}} = 2\log 2 - \text{JS}(p^M_{\text{mac}}||p^M_{\text{hum}}) - \text{JS}(p^H_{\text{mac}}||p^H_{\text{hum}}),$$

which encourages the two models to produce more distinguishable token distributions under the same input. Note that $\mathcal{L}_{\text{d}} \in [0, 2\log 2]$, where a lower value indicates the preference models exhibit more distinct behaviors on the same input. Compared to adopting KL divergence as an alternative (Huang et al., 2024), JS divergence has the advantages of being symmetric, bounded, and more stable during optimization.

Loss Function. Combining *alignment* and *distinctiveness*, we propose a joint optimization objective. The total loss of HAPDA-Finetune is defined as:

$$\mathcal{L} = \mathcal{L}_{\text{a}} + \lambda \cdot \mathcal{L}_{\text{d}},$$

where λ is a hyperparameter controlling the trade-off between preference alignment and model distinctiveness.

3.3 HAPDA-Calibration

The goal of HAPDA-Calibration is to derive a detection score by reweighting each token in the input sequence according to the disagreement between the two specialized models $\mathcal{M}_{\text{hum}}$ and $\mathcal{M}_{\text{mac}}$ obtained from HAPDA-Finetune. It emphasizes tokens where human and machine preferences diverge and down-weights uncertain predictions.

Step 1: Measuring Token-wise Prediction Disagreement. For each token x_i in the input sequence $X = \{x_1, x_2, \ldots, x_n\}$, where $x_{<i}$ denotes the sequence of tokens preceding x_i, we first compute the absolute difference between the predicted probabilities from the two models:

$$\Gamma_i = \left|\pi_{\text{hum}}(x_i \mid x_{<i}) - \pi_{\text{mac}}(x_i \mid x_{<i})\right|,$$

which captures the degree of human-machine disagreement. A larger Γ_i indicates greater divergence and conveys stronger discriminative power.

Step 2: Adjusting for Model Uncertainty. Inspired by the prior works (Ye et al., 2025; Wood et al., 2024), we introduce an entropy-based regularization term to avoid placing high importance on predictions made under uncertainty. Specifically, we compute the entropy of the two predictive distributions at position x_i, for $d \in \{\text{hum}, \text{mac}\}$:

$$E_d(x_i) = -\sum_{x \in V} \pi_d(x \mid x_{<i}) \log \pi_d(x \mid x_{<i}).$$

Then the uncertainty is expressed as:

$$U_i = \frac{1}{2 \log V}(E_{\text{hum}}(x_i) + E_{\text{mac}}(x_i)),$$

where V is the vocabulary size.[1] The quantity $1 - U_i$, where $U_i \in [0, 1]$, serves as a confidence score for the disagreement.

Step 3: Computing Token Weights and Final Score. Combining the prediction disagreement and confidence adjustment, we first compute the unnormalized weight for each token:

$$\tilde{w}_i = \Gamma_i \cdot (1 - U_i).$$

Note that $\tilde{w}_i$ lies within the interval $[0, 1]$ and is positively correlated with disagreement while being inversely correlated with uncertainty. We then normalize the weights across the sequence to ensure interpretability and stability: $w_i = \tilde{w}_i / \sum_{j=1}^{n} \tilde{w}_j$.

We take the Likelihood method (Solaiman et al., 2019) as an example to further illustrate the process of adapting HAPDA to downstream MeanZero-based methods. Suppose it employs a proxy model $\mathcal{M}_{\text{pxy}}$, the "AI-likeness" score for each token is computed as: $s_i = \log \pi_{\text{pxy}}(x_i \mid x_{<i})$. [2] Under the "Likelihood+HAPDA" setting, the final detection score of a given text X is calculated as:

$$S_X^* = \sum_{i=1}^{n} \frac{\tilde{w}_i}{\sum_{j=1}^{n} \tilde{w}_j} \cdot \log \pi_{\text{pxy}}(x_i \mid x_{<i}).$$

The complete deployment of HAPDA based on the above discussion can be found in Appendix H.

4 Experiments Settings

4.1 Source and Proxy Models

Following the setup in (Xu et al., 2024b), we select nine famous open-source models, including OPT (Zhang et al., 2022a) and Llama3 (Grattafiori et al., 2024), with parameter sizes ranging from 1.5B to 13B, as well as two of the latest closed-source models: ChatGPT and GPT-4 (Hurst et al., 2024), as the source models. In line with prior works such as (Mitchell et al., 2023; Xu et al., 2024b), we adopt GPT-J (Wang and Komatsuzaki, 2021) as the proxy model $\mathcal{M}_{\text{pxy}}$ for all methods unless otherwise specified. A more detailed introduction to the models can be found in Appendix B.

[1] We assume that $\mathcal{M}_{\text{hum}}$ and $\mathcal{M}_{\text{mac}}$ originate from the same base model to ensure that V is consistent.

[2] More MeanZero expressions are detailed in Appendix C.

4.2 Datasets

We conduct the evaluation experiments on four topic-specific subsets: **Books**, **Reviews**, **Wiki**, and **News**, from the AGTD benchmark RAID (Dugan et al., 2024). Each subset contains 150 human-written examples, each containing 150 to 300 words. For each human-written example H, we use the text from the `prompt` field P as input to the source models to generate an LLM-produced continuation M. To validate the generalizability of HAPDA across diverse text domains, we select four distinct subsets from the RAID dataset (**Abstract**, **Poetry**, **Recipes**, and **Reddit**) for fine-tuning, ensuring they are disjoint from the evaluation datasets. The generation models also differ from the source and proxy models, including eight LLMs such as MPT (Team, 2023) and Cohere (Alnumay et al., 2025). During the construction of each subset, for each LLM, we randomly select 160 unique human-written examples and pair them with their corresponding LLM-generated texts (all provided by RAID) to form human-machine text pairs. More details on the dataset can be found in Appendix B.

4.3 Fine-tuning Details

We also adopt GPT-J as the reference model $\mathcal{M}_{\text{ref}}$, and to reduce training costs, we use its 4-bit quantized version (Dettmers et al., 2022). Inspired by (Hao et al., 2025), we apply the LoRA technique (Hu et al., 2022), with a rank of 16 and a scaling factor of 32. The adaptation modules include `q_proj` and `v_proj`. The loss function uses a default hyperparameter $\lambda = 0.3$. Following the common configurations, we set the batch size to 16, the optimizer to `AdamW` (Loshchilov and Hutter, 2017), the learning rate to 5e-5, the number of epochs to 10, and apply early stopping with a patience of 2. All experiments are conducted on two NVIDIA A100 80GB GPUs. The experimental results of fine-tuning can be found in Appendix E.

4.4 Baselines and Metrics

We select five MeanZero-based methods that are compatible with our task definition as baselines: Entropy (Gehrmann et al., 2019), LogRank (Solaiman et al., 2019), Likelihood (Solaiman et al., 2019), DetectGPT (Mitchell et al., 2023), and Fast-DetectGPT (Bao et al., 2024). In addition, we compare our method with two closely related correction-based approaches: DetectLRR and DetectNPR (Su et al., 2023). Beyond the above dis-

cussed baselines, we further include four latest zero-shot detectors outside the MeanZero framework for comparison: DNA-GPT (Yang et al., 2024), Raidar (Mao et al., 2024), Binoculars (Hans et al., 2024) and Lastde (Xu et al., 2024b). Detailed descriptions of all methods can be found in Appendix C. Following prior works (Mitchell et al., 2023; Xu et al., 2024b), we adopt AUROC as the binary classification metric for the AGTD task.

5 Main Results

5.1 White-box Detection Results

In the white-box setting, the proxy model is identical to the source model. As shown in Table 1, applying HAPDA leads to performance improvements for most source models across the MeanZero baselines under the white-box scenario. Specifically, with the integration of HAPDA, the average AUROC scores of Entropy, LogRank, Likelihood, DetectGPT, and Fast-DetectGPT increase by 12.3%, 5.3%, 5.6%, 6.4%, and 0.3%, respectively. Moreover, compared to DetectLRR and DetectNPR, HAPDA achieves a higher average AUROC by 3.0% and 2.1%, respectively.

5.2 Black-box Detection Results

In the black-box setting, the proxy model differs from the source model. As shown in Table 2, applying HAPDA in the black-box scenario leads to significant performance improvements for most source models across the MeanZero baselines. Specifically, with the incorporation of HAPDA, the average AUROC scores of Entropy, LogRank, Likelihood, DetectGPT, and Fast-DetectGPT increase by 12.4%, 9.0%, 20.3%, 14.4%, and 5.0%, respectively. In addition, compared to the correction-based methods DetectLRR and DetectNPR, HAPDA achieves a higher average AUROC by 9.5% and 4.9%, respectively. As shown in Table 3, the DetectGPT detector incorporated with HAPDA achieves AUROC scores comparable to or even surpassing the best zero-shot detectors across multiple tasks. This demonstrates the potential of HAPDA to elevate relatively weaker classifiers to performance levels close to the latest detectors. Besides, the comparison with supervised method (see Appendix F) demonstrates HAPDA's superior generalization capability. The black-box setting represents a more realistic and widely applicable scenario than the white-box setting. Therefore, all subsequent experiments in this context are conducted by default under the black-box setting.

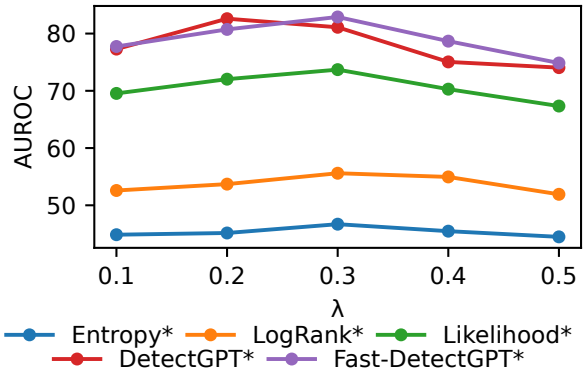

Figure 3: Hyperparameters sensitivity analysis results under five λ values: 0.1, 0.2, 0.3, 0.4, 0.5, with GPT-4 as the source model. "*" denotes "+HAPDA".

5.3 Hyperparameters Sensitivity Analysis

In our work, λ is the trade-off coefficient between the alignment loss and the distinctiveness loss. A larger value of λ indicates that the model focuses more on distinctiveness, while a smaller value emphasizes preference alignment. As demonstrated by the experimental results in Figure 3, when $\lambda = 0.3$, the HAPDA-enhanced methods—Entropy, Likelihood, LogRank, and FastDetectGPT—achieve the best performance. A larger λ leads to a degradation in detection performance. Therefore, we recommend setting $\lambda = 0.3$ as a reasonable hyperparameter configuration.

5.4 Ablation Study

The settings and results of our ablation experiments are presented in Table 4. Compared to the base variant (-D-U), adding *Distinctiveness* during fine-tuning (-U) improves AUROC by 1.4%–6.5%, while adding *Uncertainty* during calibration (-D) brings a gain of 1.1%–5.8%. The complete HAPDA strategy achieves the best performance, improving AUROC by 3.4%–9.4% across all five base methods. On this basis, when only the human perspective (-M) or the machine perspective (-H) is considered, the AUROC decreases by 2.5%–5.7% and 3.8%–10.8%, respectively.

5.5 Decoding Strategies

Following the setup of (Xu et al., 2024b), we re-implement our experiments under different decoding strategies. Specifically, top-p, top-k, and temperature control the generation quality of the source model from various perspectives. As shown in Figure 4, under all three decoding strategies, HAPDA

Methods/Models	GPT-2	Neo-2.7	OPT-2.7	GPT-J	BLOOM-7	Falcon-7	Llama3-8	OPT-13	Llama2-13	Average
Entropy	54.1	53.4	49.6	55.4	62.2	58.6	27.8	52.7	58.9	52.5
+HAPDA	**64.5**	**63.3**	**65.0**	**65.9**	**73.5**	**72.3**	**49.1**	**64.8**	**65.1**	**64.8**(+12.3)
LogRank	91.7	82.5	80.3	74.2	82.5	67.7	62.3	73.6	59.8	75.0
+HAPDA	**93.9**	**88.5**	**83.3**	**79.6**	**87.8**	**71.5**	**69.0**	**80.6**	**68.4**	**80.3**(+5.3)
Likelihood	94.5	87.5	85.8	80.1	87.2	73.0	93.8	78.9	61.7	82.5
DetectLRR	92.3	86.7	88.2	82.6	85.9	76.9	**99.0**	84.1	70.0	85.1
+HAPDA	**94.3**	**88.4**	**90.0**	**88.8**	**88.3**	**80.3**	98.7	**87.0**	**77.1**	**88.1**(+3.0)
DetectGPT	95.0	91.2	91.9	87.5	90.8	82.3	**99.4**	88.6	74.7	89.0
DetectNPR	97.0	95.7	94.8	93.1	95.6	89.6	99.2	92.4	82.6	93.3
+HAPDA	**97.7**	**96.9**	**95.6**	**95.7**	**96.8**	**93.2**	99.3	**94.3**	**88.7**	**95.4**(+2.1)
Fast-DetectGPT	99.7	99.0	98.6	98.3	**99.6**	97.9	**99.9**	98.3	94.9	98.5
+HAPDA	**99.8**	**99.2**	**99.0**	**98.7**	99.5	**98.6**	99.8	**98.7**	**95.8**	**98.8**(+0.3)

Table 1: Overall detection results under the white-box scenario. All methods use the source model for scoring. The AUROC values for each method are averaged across Books, Reviews, Wiki, and News (same for Tables 2 and 4, as well as Figures 3-7).

Methods/Models	GPT-2	Neo-2.7	OPT-2.7	BLOOM-7	Falcon-7	Llama3-8	OPT-13	Llama2-13	ChatGPT	GPT-4	Average
Entropy	39.6	34.7	36.5	35.9	46.5	28.4	42.9	41.0	44.8	40.2	39.1
+HAPDA	**46.1**	**48.9**	**53.2**	**46.6**	**57.4**	**48.7**	**57.6**	**47.4**	**62.3**	**46.7**	**51.5**(+12.4)
LogRank	36.7	36.4	43.7	33.1	49.6	69.2	54.6	49.5	50.2	47.1	47.0
+HAPDA	**44.5**	**49.1**	**50.8**	**43.4**	**59.4**	**73.3**	**61.1**	**57.2**	**65.5**	**55.6**	**56.0**(+9.0)
Likelihood	42.9	43.8	50.4	39.3	53.7	90.6	59.8	54.3	55.1	52.6	54.3
DetectLRR	54.1	58.8	66.7	53.2	62.3	98.2	71.8	61.4	63.1	61.2	65.1
+HAPDA	**69.0**	**71.1**	**78.7**	**62.5**	**74.9**	**98.9**	**76.8**	**67.3**	**73.5**	**73.7**	**74.6**(+9.5)
DetectGPT	60.1	65.1	72.0	60.1	68.6	**98.8**	76.8	65.5	67.4	66.1	70.1
DetectNPR	73.2	78.1	81.3	76.5	80.7	97.8	84.2	73.9	76.4	74.1	79.6
+HAPDA	**83.6**	**81.6**	**85.1**	**83.1**	**87.0**	98.6	**88.1**	**77.5**	**79.7**	**81.1**	**84.5**(+4.9)
Fast-DetectGPT	83.1	83.7	83.6	77.4	79.7	99.2	89.1	72.1	76.9	75.2	82.0
+HAPDA	**88.5**	**85.6**	**86.9**	**85.9**	**87.1**	**99.4**	**92.4**	**83.0**	**85.5**	**82.9**	**87.7**(+5.7)

Table 2: Overall detection results under the black-box scenario.

Methods/Models	OPT-13	Llama2-13	ChatGPT	GPT-4
DetectGPT	75.9	67.8	65.2	69.4
DetectGPT*	<u>89.3</u>	**78.9**	78.8	**83.3**
DNA-GPT	85.5	73.1	76.3	75.1
Raidar	84.0	76.5	78.6	78.4
Binoculars	87.8	77.0	<u>79.5</u>	<u>81.9</u>
Lastde	**90.5**	<u>77.2</u>	**80.6**	81.5

Table 3: Additional detection results compared with advanced zero-shot detectors in the black-box scenario for Books dataset. "<u>value</u>" denotes the second-best AUROC. "*" denotes "+HAPDA".

Methods/Settings	Ours	-D	-U	-D-U	-M	-H
Entropy+HAPDA	**51.5**	47.9	48.6	42.1	45.8	40.7
LogRank+HAPDA	**56.0**	53.6	53.3	50.2	52.1	48.6
Likelihood+HAPDA	**74.6**	71.8	72.0	68.4	70.5	66.2
DetectGPT+HAPDA	**84.5**	82.2	82.5	81.1	81.9	80.7
Fast-DetectGPT+HAPDA	**87.7**	85.7	85.9	84.1	85.2	83.3

Table 4: Ablation study for HAPDA. The AUROC values for each method are averaged across all ten source models in the black-box setting. "-D" and "-U" denote removing *Distinctiveness* ($\lambda = 0$) and *Uncertainty* ($U_i = 0$), respectively. "-M" and "-H" denote removing the machine perspective (by replacing $\mathcal{M}_{\text{mac}}$ with $\mathcal{M}_{\text{ref}}$) and the human perspective (by replacing $\mathcal{M}_{\text{hum}}$ with $\mathcal{M}_{\text{ref}}$), respectively.

consistently and significantly improves the detection performance of the vanilla MeanZero method, and also outperforms DetectLRR and DetectNPR.

5.6 Proxy Models Selection

In addition to GPT-J, we investigate the detection performance under other proxy models. As demonstrated by the experimental results in Figure 5, HAPDA consistently and significantly improves the detection performance of MeanZero baselines across all proxy models. Specifically, compared to vanilla Likelihood and DetectLRR, HAPDA-enhanced Likelihood achieves AUROC gains ranging from 15.9% to 21.2% and 9.2% to 14.8%, respectively. Compared to vanilla DetectGPT and

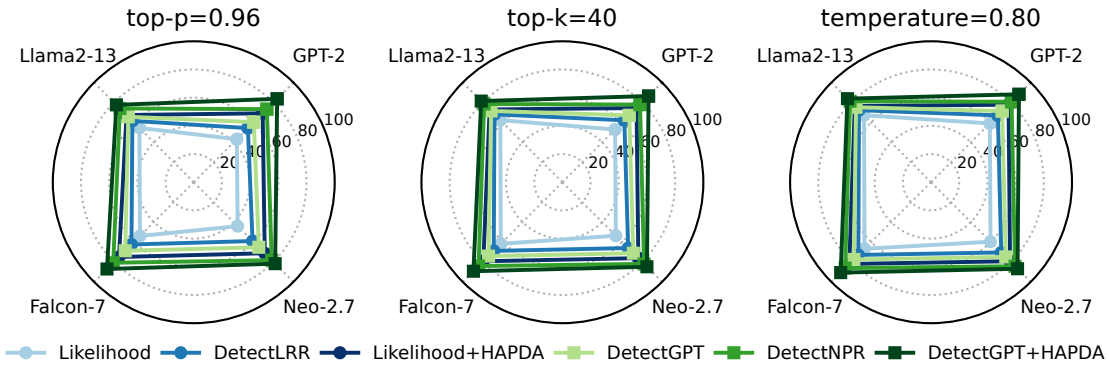

Figure 4: Detection results under different decoding strategies: "top-p=0.96", "tok-k=40" and "temperature=0.80". The source models are GPT-2, Neo-2.7, Falcon-7, and Llama2-13.

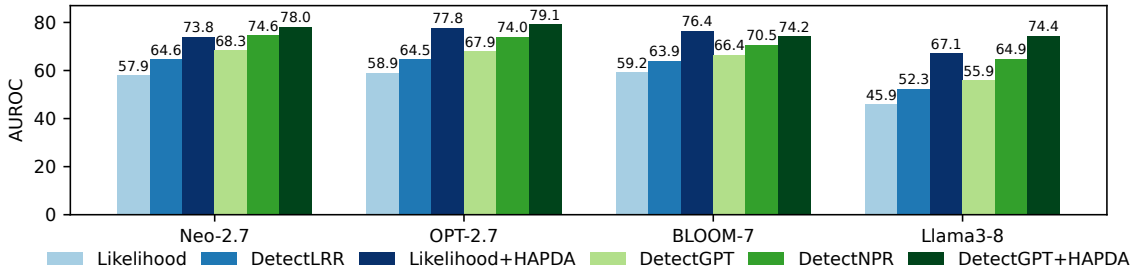

Figure 5: Detection results under four proxy models: Neo-2.7, OPT-2.7, BLOOM-7, Llama3-8, with GPT-4 as the source model.

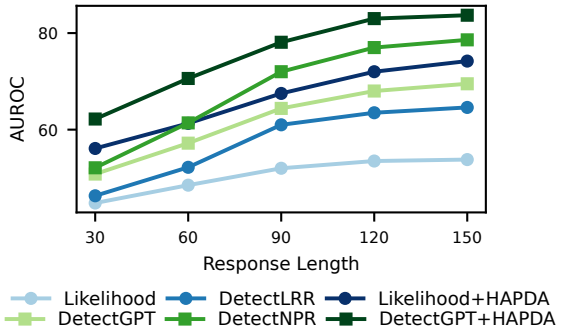

Figure 6: Average AUROC for all ten source models in black-box setting under five response lengths: 30, 60, 90, 120, 150.

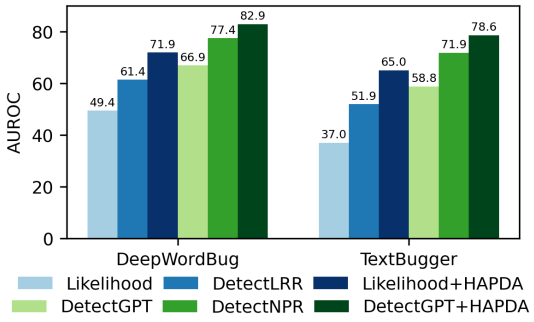

Figure 7: Average AUROC for all ten source models in black-box setting under two adversarial attacks: DeepWordBug and TextBugger.

DetectNPR, HAPDA-enhanced DetectGPT yields AUROC improvements of 7.8% to 18.5% and 3.4% to 9.5%, respectively.

5.7 Response Lengths

Previous research (Mao et al., 2024) has indicated that the input length can affect the AGTD methods' performance. We truncate human-written and corresponding AI-generated texts to different lengths to re-implement our experiments. As demonstrated by the experimental results in Figure 6, HAPDA consistently improves the detection performance of the MeanZero method even on short texts. Notably, when the text length is relatively short (e.g., response length of 30 or 60), the performance gain of HAPDA is more pronounced compared to DetectLRR and DetectNPR.

5.8 Adversarial Attacks

We adopt two challenging adversarial attack strategies from TextAttack (Morris et al., 2020): DeepWordBug (Gao et al., 2018) and TextBugger (Li et al., 2019) to evaluate the robustness of HAPDA under perturbations. Our implementation follows the same settings described in (Morris et al., 2020). We provide a detailed description of them in Appendix D. As demonstrated by the results in Fig-

ure 7. Although the detection performance of our proposed MeanZero method degrades slightly compared to the non-adversarial setting, it still achieves the best overall performance. Compared to DetectLRR and DetectNPR, our method outperforms them by 10.5% and 5.5% under DeepWordBug, and by 13.1% and 6.7% under TextBugger.

6 Conclusion

In our work, we propose a novel adapter named HAPDA from the perspective of human-machine prediction discrepancy. It is designed to enhance the detection performance of the MeanZero detectors. We introduce a new joint fine-tuning strategy to obtain high-quality human/machine preference models, which are then used to calibrate the downstream detection scores. Extensive experiments demonstrate that across various detection scenarios (white-box, black-box, adversarial attacks, and short texts) and experimental settings (proxy models and decoding strategies), the introduction of HAPDA consistently and significantly improves the detection performance of baselines.

Limitations

Although our proposed approach has been shown to significantly improve the MeanZero method across various application scenarios, HAPDA still has certain limitations: (1) HAPDA is not a complete detection system but is designed as an extension of the existing MeanZero detector, and it is not directly applicable to some zero-shot and supervised methods. In future work, we will explore its transferability to these settings. (2) The introduction of auxiliary preference models in HAPDA inevitably incurs additional runtime and memory overhead during training and inference. While we mitigate this issue in our work using techniques such as quantization and LoRA, we plan to further optimize the framework through lightweight and parallelized designs.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 72210107001, 72225011, 72434005 and L242400108, the Chinese Academy of Science President's International Fellowship Initiative (CAS PIFI) International Outstanding Team Project under Grant 2024PG0013, the National Social Science Fund of China (Major Program, Grant No. 25&ZD161), and the CAS Pilot Project for Basic and Interdisciplinary Scientific Research (Grant No. XDB1380303).

References

Sara Abdali, Richard Anarfi, CJ Barberan, and Jia He. 2024. Decoding the ai pen: Techniques and challenges in detecting ai-generated text. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6428–6436.

Yazeed Alnumay, Alexandre Barbet, Anna Bialas, William Darling, Shaan Desai, Joan Devassy, Kyle Duffy, Stephanie Howe, Olivia Lasche, Justin Lee, Anirudh Shrinivason, and Jennifer Tracey. 2025. Command r7b arabic: A small, enterprise focused, multilingual, and culturally aware arabic llm. *Preprint*, arXiv:2503.14603.

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*.

Amrita Bhattacharjee, Raha Moraffah, Joshua Garland, and Huan Liu. 2024. Eagle: A domain generalization framework for ai-generated text detection. *arXiv preprint arXiv:2403.15690*.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. Gpt-neo: Large scale autoregressive language modeling with mesh-tensorflow, march 2021. *URL https://doi. org/10.5281/zenodo*, 5297715(5):3.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289.

David L Davies and Donald W Bouldin. 1979. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2):224–227.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in neural information processing systems*, 35:30318–30332.

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)*, pages 12463–12492.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. 2019. Gltr: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 111–116.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Arthur Gretton, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex Smola. 2006. A kernel method for the two-sample-problem. *Advances in neural information processing systems*, 19.

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. *Preprint*, arXiv:2401.12070.

Wei Hao, Ran Li, Weiliang Zhao, Junfeng Yang, and Chengzhi Mao. 2025. 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)*, pages 6421–6434, Vienna, Austria. Association for Computational Linguistics.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Guanhua Huang, Yuchen Zhang, Zhe Li, Yongjian You, Mingze Wang, and Zhouwang Yang. 2024. Are ai-generated text detectors robust to adversarial perturbations? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6005–6024.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. Automatic detection of generated text is easiest when humans are fooled. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1808–1822.

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. *Preprint*, arXiv:2310.06825.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

J Li, S Ji, T Du, B Li, and T Wang. 2019. Textbugger: Generating adversarial text against real-world applications. In *26th Annual Network and Distributed System Security Symposium*.

Ran Li, Wei Hao, Weiliang Zhao, Junfeng Yang, and Chengzhi Mao. 2024. Learning to rewrite: Generalized llm-generated text detection. *arXiv preprint arXiv:2408.04237*.

Xinzhe Li. 2025. A review of prominent paradigms for llm-based agents: Tool use, planning (including rag), and feedback learning. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 9760–9779.

Jianhua Lin. 2002. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37(1):145–151.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Shengchao Liu, Xiaoming Liu, Yichen Wang, Zehua Cheng, Chengzhengxu Li, Zhaohan Zhang, Yu Lan, and Chao Shen. 2024b. Does detectgpt fully utilize perturbation? bridging selective perturbation to fine-tuned contrastive learning detector would be better. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1874–1889.

Xiaoming Liu, Zhaohan Zhang, Yichen Wang, Hang Pu, Yu Lan, and Chao Shen. 2023. Coco: Coherence-enhanced machine-generated text detection under low resource with contrastive learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 16167–16188.

Yepeng Liu and Yuheng Bu. 2024. Adaptive text watermark for large language models. In *International Conference on Machine Learning*, pages 30718–30737. PMLR.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Yijian Lu, Aiwei Liu, Dianzhi Yu, Jingjing Li, and Irwin King. 2024. An entropy-based text watermarking detection method. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11724–11735.

Chengzhi Mao, Carl Vondrick, Hao Wang, and Junfeng Yang. 2024. Raidar: generative AI detection via rewriting. In *The Twelfth International Conference on Learning Representations*.

Pablo Miralles-González, Javier Huertas-Tato, Alejandro Martín, and David Camacho. 2025. Not all tokens are created equal: Perplexity attention weighted networks for ai generated text detection. *arXiv preprint arXiv:2501.03940*.

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*, pages 24950–24962. PMLR.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.

Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

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. *arXiv preprint arXiv:2402.09199*.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, and 1 others. 2019. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*.

Jinyan Su, Terry 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*, pages 12395–12412.

MosaicML NLP Team. 2023. Introducing mpt-7b: A new standard for open-source, commercially usable llms. Accessed: 2023-03-28.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

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*, pages 1144–1156.

Tianchun Wang, Yuanzhou Chen, Zichuan Liu, Zhanwen Chen, Haifeng Chen, Xiang Zhang, and Wei Cheng. 2025. Humanizing the machine: Proxy attacks to mislead LLM detectors. In *The Thirteenth International Conference on Learning Representations*.

Alva West, Yixuan Weng, Minjun Zhu, Luodan Zhang, Zhen Lin, Guangsheng Bao, and Yue Zhang. 2025. Ai-generated text is non-stationary: Detection via temporal tomography. *arXiv preprint arXiv:2508.01754*.

Danny Wood, Theodore Papamarkou, Matt Benatan, and Richard Allmendinger. 2024. Model-agnostic variable importance for predictive uncertainty: an entropy-based approach. *Data Mining and Knowledge Discovery*, 38(6):4184–4216.

BigScience Workshop, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, and 1 others. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

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.

Yihuai Xu, Yongwei Wang, Yifei Bi, Huangsen Cao, Zhouhan Lin, Yu Zhao, and Fei Wu. 2024a. Training-free llm-generated text detection by mining token probability sequences. *arXiv preprint arXiv:2410.06072*.

Yihuai Xu, Yongwei Wang, Yifei Bi, Huangsen Cao, Zhouhan Lin, Yu Zhao, and Fei Wu. 2024b. Training-free llm-generated text detection by mining token probability sequences. *arXiv preprint arXiv:2410.06072*.

Zhenyu Xu and Victor S Sheng. 2024. Detecting ai-generated code assignments using perplexity of large language models. In *Proceedings of the aaai conference on artificial intelligence*, volume 38, pages 23155–23162.

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 *ICLR*.

Zihuiwen Ye, Luckeciano Carvalho Melo, Younesse Kaddar, Phil Blunsom, Sam Staton, and Yarin Gal. 2025. Uncertainty-aware step-wise verification with generative reward models. *arXiv preprint arXiv:2502.11250*.

Shu Yin, Peican Zhu, Lianwei Wu, Chao Gao, and Zhen Wang. 2024. Gamc: an unsupervised method for fake news detection using graph autoencoder with masking. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, pages 347–355.

Kiyoon Yoo, Wonhyuk Ahn, Yeji Song, and Nojun Kwak. 2024. Exploring causal mechanisms for machine text detection methods. In *Proceedings of the 4th Workshop on Trustworthy Natural Language Processing (TrustNLP 2024)*, pages 71–78, Mexico City, Mexico. Association for Computational Linguistics.

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*, 37:54947–54973.

Chaowei Zhang, Zongling Feng, Zewei Zhang, Jipeng Qiang, Guandong Xu, and Yun Li. 2025a. Is llms hallucination usable? llm-based negative reasoning for fake news detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 1031–1039.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, and 1 others. 2022a. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, and 1 others. 2022b. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Yuhan Zhang, Xingxiang Jiang, Hua Sun, Yao Zhang, and Deyu Tong. 2025b. Curvemark: Detecting ai-generated text via probabilistic curvature and dynamic semantic watermarking. *Entropy*, 27(8):784.

A Theoretical Justification via Mutual Information

Our key intuition is that tokens exhibiting larger divergence between human and machine prediction distributions carry higher mutual information regarding the author identity, thus being more discriminative than treating all tokens uniformly. In this section, we theoretically validate the feasibility of our intuition from the perspective of mutual information.

Formally, let the author identity A be a binary random variable where $A = 0$ represents a human author and $A = 1$ a machine author, with equal prior probabilities $P(A = 0) = P(A = 1) = \frac{1}{2}$.

For a token x_i at position i, we define its divergence δ_i as the total variation distance between the human prediction distribution $P_h(x_i \mid \text{context})$ and the machine prediction distribution $P_m(x_i \mid \text{context})$ over the vocabulary $\mathcal{V}$:

$$\delta_i = \frac{1}{2} \sum_{x \in \mathcal{V}} |P_h(x_i = x \mid \text{context}) - P_m(x_i = x \mid \text{context})|,$$

where by definition $\delta_i \in [0, 1]$.

The mutual information (MI) between token x_i and the author identity A quantifies the reduction in uncertainty about A given the token x_i, and is expressed as:

$$I(A; x_i) = H(A) - H(A \mid x_i),$$

where $H(A)$ is the entropy of the author identity, and $H(A \mid x_i)$ is the conditional entropy given the token.

Assuming the token distribution depends on the author's identity, we have:

$$P(x_i \mid A) = \begin{cases} P_h(x_i \mid \text{context}) & \text{if } A = 0, \\ P_m(x_i \mid \text{context}) & \text{if } A = 1. \end{cases}$$

Under this assumption, the mutual information $I(A; x_i)$ corresponds exactly to the Jensen-

Shannon divergence (JS) between the two distributions P_h and P_m:

$$I(A; x_i) = \mathrm{JS}(P_h \| P_m),$$

where the Jensen-Shannon divergence is defined as

$$\mathrm{JS}(P_h \| P_m) = \frac{1}{2}\left[\mathrm{KL}(P_h \| M) + \mathrm{KL}(P_m \| M)\right],$$

with $M = \frac{1}{2}(P_h + P_m)$ being the mixture distribution and $\mathrm{KL}(\cdot\|\cdot)$ the Kullback-Leibler divergence.

We now relate the total variation distance δ_i to the mutual information $I(A; x_i)$. Using the established relationship between JS and total variation distance, we have the following lower bound:

$$\mathrm{JS}(P_h \| P_m) \geq \frac{1}{2}\delta_i^2.$$

From the Pinsker's inequality applied to each KL-divergence term:

$$\mathrm{KL}(P_h \| M) \geq 2(\mathrm{TV}(P_h, M))^2,$$
$$\mathrm{KL}(P_m \| M) \geq 2(\mathrm{TV}(P_m, M))^2,$$

where $\mathrm{TV}(\cdot, \cdot)$ denotes total variation distance. Since $M = \frac{1}{2}(P_h + P_m)$, we have:

$$\mathrm{TV}(P_h, M) = \mathrm{TV}(P_m, M) = \frac{1}{2}\delta_i.$$

Substituting these into the JS definition:

$$\begin{aligned}\mathrm{JS}(P_h \| P_m) &= \frac{1}{2}\left[\mathrm{KL}(P_h \| M) + \mathrm{KL}(P_m \| M)\right] \\ &\geq \frac{1}{2}\left[2\left(\frac{1}{2}\delta_i\right)^2 + 2\left(\frac{1}{2}\delta_i\right)^2\right] = \frac{1}{2}\delta_i^2.\end{aligned}$$

Thus we obtain the fundamental relationship:

$$I(A; x_i) \geq \frac{1}{2}\delta_i^2.$$

This bound establishes that tokens with larger divergence δ_i necessarily have higher mutual information about author identity. Consider two tokens x_i and x_j such that $\delta_i > \delta_j$. It follows that:

$$I(A; x_i) \geq \frac{1}{2}\delta_i^2 > \frac{1}{2}\delta_j^2,$$

meaning x_i provides a strictly higher lower bound on mutual information than x_j. Moreover, since JS is a metric and increases with distribution divergence, $\delta_i > \delta_j$ typically implies $I(A; x_i) > I(A; x_j)$ in practice.

From a classification perspective, the minimal error probability P_e for author attribution is bounded by Fano's inequality:

$$H(A \mid \mathbf{x}) \leq H(P_e) + P_e \log_2(|\mathcal{A}| - 1),$$

where $\mathbf{x} = \{x_1, \ldots, x_n\}$ denotes the sequence of tokens. Maximizing the sum of mutual information of selected tokens: $\max_{\mathbf{x}} \sum_{i=1}^{n} I(A; x_i)$, corresponds to minimizing the conditional entropy $H(A \mid \mathbf{w})$ and thus the classification error P_e.

In summary, tokens with larger prediction divergence δ_i exhibit higher mutual information $I(A; x_i)$ about author identity, as both the lower bound $\frac{1}{2}\delta_i^2$ and the actual JS value increase with δ_i. Therefore, prioritizing tokens based on δ_i leads to maximal information gain and reduces uncertainty in authorship attribution more effectively than uniform weighting of all tokens.

B Details of Datasets

B.1 LLMs in Our Work

In our work, the evaluation dataset consists of texts sourced from: GPT-2 (Radford et al., 2019), Neo-2.7 (Black et al., 2021), OPT-2.7 (Zhang et al., 2022b), GPT-J (Wang and Komatsuzaki, 2021), BLOOM-7 (Workshop et al., 2022), Falcon-7 (Penedo et al., 2023), Llama3-8 (Grattafiori et al., 2024), OPT-13 (Zhang et al., 2022b), Llama2-13 (Touvron et al., 2023), ChatGPT (Hurst et al., 2024), GPT-4o (Hurst et al., 2024). The fine-tuning dataset consists of texts sourced from: Mistral (Jiang et al., 2023), Mistral-Chat (Jiang et al., 2023), MPT (Team, 2023), MPT-Chat (Team, 2023), LLaMA-Chat (Touvron et al., 2023), GPT-3 (Brown et al., 2020), Cohere (Alnumay et al., 2025), Cohere-Chat (Alnumay et al., 2025). The names, parameter sizes, and model versions of all LLMs involved in this paper are listed in Table 5.

B.2 Evalution Datasets

We select four independent and representative subsets from RAID—Books, Reviews, Wiki, and News—to evaluate both black-box and white-box scenarios. They respectively contain plot-centric summaries of books along with their titles, movie reviews from IMDb along with the names of the movies, introductions to various Wikipedia articles, and BBC news articles with associated titles. For each subset, we randomly sample 150 human-written texts. For "ChatGPT" and "GPT-4,"

Application	Model	Version	Parameters
Evaluation	GPT-2	openai-community/gpt2-xl	1.5B
	Neo-2.7	EleutherAI/gpt-neo-2.7B	2.7B
	OPT-2.7	facebook/opt-2.7b	2.7B
	GPT-J	EleutherAI/gpt-j-6B	6B
	BLOOM-7	bigscience/bloom-7b1	7B
	Falcon-7	tiiuae/falcon-7b	7B
	Llama3-8	meta-llama/Llama-3.1-8B	8B
	OPT-13	facebook/opt-13b	13B
	Llama2-13	TheBloke/Llama-2-13B-fp16	13B
	ChatGPT	gpt-3.5-turbo-0613	N.A.
	GPT-4	gpt-4-0613	N.A.
Fine-tune	Mistral	mistralai/Mistral-7B-v0.1	7B
	Mistral-Chat	mistralai/Mistral-7B-Instruct-v0.1	7B
	MPT	mosaicml/mpt-30b	30B
	MPT-Chat	mosaicml/mpt-30b-chat	30B
	LLaMA-Chat	meta-llama/Llama-2-70b-chat-hf	70B
	GPT-3	text-davinci-002	175B
	Cohere	command (co.generate())	N.A.
	Cohere-Chat	command (co.chat())	N.A.

Table 5: Details of the models involved in our work.

the corresponding machine-generated texts are directly provided by RAID. For the other models, we use the `prompt` field associated with each human-written sample from RAID to generate continuations as machine-generated texts. Consequently, each complete subset consists of 150 negative (human-written) and 150 positive (LLM-generated) samples.

B.3 Fine-tuning Datasets

For the selection of the fine-tuning training set, we deliberately chose texts that are as unrelated as possible to the topics of the evaluation datasets, in order to verify that HAPDA can achieve good generalization across different text domains. For fine-tuning, we select four additional topic-specific subsets from RAID: Abstract, Poetry, Recipes, and Reddit. They respectively contain abstracts scraped from ArXiv together with paper titles, poems collected from poemhunter.com along with their titles and genres, recipes with their dish names, and Reddit posts with their titles. The generation models used in this stage differ from the source and proxy models, including eight additional LLMs shown in Table 5. During the construction of each subset, for each LLM, we randomly select 160 unique human-written examples and pair them with their corresponding LLM-generated texts to form human-machine text pairs. Noted that all texts in the finetuning dataset are provided by RAID and do not require additional generation. Consequently, each complete subset consists of 1280 negative and 1280 positive samples. We randomly select 80% of the text pairs as the training set and 20% as the validation set.

C Details of Baselines

We select five MeanZero baselines: Entropy, LogRank, Likelihood, DetectGPT, and Fast-DetectGPT, as well as two methods closely related to our work, DetectNPR and DetectLRR. Unless otherwise specified, their configurations remain consistent with the original papers. We use the **boldface** text to highlight the "AI-likeness" scores s_i used by each method. Here, $\pi_{\text{pxy}}(x_i \mid x_{<i})$ denotes the probability assigned to token x_i by a proxy model $\mathcal{M}_{\text{pxy}}$ given the preceding context $x_{<i}$, and $\text{rank}(x_i \mid x_{<i})$ represents the descending order index of x_i in the predicted vocabulary distribution.

- **Entropy** (Gehrmann et al., 2019) assumes that AI-generated text exhibits lower "openness". The **average entropy** of each token is calculated based on the probability distribution over

the vocabulary $\mathcal{V}$:

$$s_i = -\sum_{w \in \mathcal{V}} \pi_{\text{pxy}}(w \mid x_{<i}) \log \pi_{\text{pxy}}(w \mid x_{<i}),$$

where $\mathcal{V}$ is the vocabulary space.

- **LogRank** (Solaiman et al., 2019) examines the logarithmic ranks of tokens generated by a language model. The **average log rank** of all tokens in the candidate text is used as the metric:

$$s_i = \log \text{rank}_{\text{pxy}}(x_i \mid x_{<i}),$$

where $\text{rank}_{\text{pxy}}(x_i \mid x_{<i})$ denotes the position of token x_i when sorting all vocabulary tokens in descending order of predicted probability.

- **Likelihood** (Solaiman et al., 2019) assumes that AI-generated text has higher "precision". The **average log probability** of all tokens is used as the metric:

$$s_i = \log \pi_{\text{pxy}}(x_i \mid x_{<i}).$$

- **DetectGPT** (Mitchell et al., 2023) assumes that generated text typically lies in regions of negative curvature in the model's log-probability space. The **perturbation discrepancy** is used as the metric:

$$s_i = \log \pi_{\text{pxy}}(x_i \mid x_{<i}) - \mathbb{E}_{\tilde{x}_i \sim K}\left[\log \pi_{\text{pxy}}(\tilde{x}_i \mid x_{<i})\right],$$

where $\tilde{x}_i \sim K(\cdot \mid x)$ denotes a perturbed context generated by applying small, meaning-preserving modifications to $x_{<i}$ using a perturbation function K.

- **Fast-DetectGPT** (Bao et al., 2024) improves upon DetectGPT by replacing the perturbation step with a more efficient sampling strategy. The **sampling discrepancy** is used as the metric:

$$s_i = \frac{\log \pi_{\text{pxy}}(x_i \mid x_{<i}) - \mu_\sim}{\sigma_\sim},$$

where:

$$\mu_\sim = \mathbb{E}_{x_i^{(j)} \sim \mathcal{Q}}[\log \pi_{\text{pxy}}(x_i^{(j)} \mid x_{<i})],$$
$$\sigma_\sim^2 = \mathbb{E}_{x_i^{(j)} \sim \mathcal{Q}}[(\log \pi_{\text{pxy}}(x_i^{(j)} \mid x_{<i}) - \mu_\sim)^2],$$

and $x_i^{(j)} \sim \mathcal{Q}$ denotes the j-th sampled context generated by the sampling model $\mathcal{Q}$.

- **DetectLRR** (Su et al., 2023) is a revised version of LogRank, designed to incorporate both the confidence and rarity of tokens. It uses the **ratio of log-likelihood to log-rank** as the score:

$$s_i = \frac{\log \pi_{\text{pxy}}(x_i \mid x_{<i})}{\log \text{rank}_{\text{pxy}}(x_i \mid x_{<i})}.$$

- **DetectNPR** (Su et al., 2023) is a revised version of DetectGPT, which measures the stability of token ranking under small perturbations. It uses the **normalized perturbed log-rank** as the metric:

$$s_i = \frac{1}{n}\sum_{p=1}^{n} \frac{\log \text{rank}_{\text{pxy}}(x_i \mid \tilde{x}_{<i}^{(p)})}{\log \text{rank}_{\text{pxy}}(x_i \mid x_{<i})},$$

where $\tilde{x}_{<i}^{(p)}$ is the p-th perturbed version of the context $x_{<i}$, and n is the total number of perturbations.

In Section 3.1, we defined our task as primarily aiming to enhance the existing MeanZero method. To enable a more comprehensive evaluation, beyond the above discussed baselines, we further include four latest zero-shot detectors outside the MeanZero framework for comparison: DNA-GPT (Yang et al., 2024), Raidar (Mao et al., 2024), Binoculars (Hans et al., 2024) and Lastde (Xu et al., 2024b). Specifically, for DNA-GPT, we use GPT-J to compute BScore, set the number of re-prompting iterations to 10, and fix the truncate rate to 0.5. For Raidar, we adopt the "Invariance" strategy, with GPT-4 as the generation model and the prompt template "Rewrite this for me:". For Binoculars, we compute PPL using gpt-j-6B and compute the denominator (i.e., X-PPL) using gpt-neo-2.7B and gpt-j-6B respectively. For Lastde, we set the sliding window size to 4 and the number of scales to 10. All other configurations follow the original implementations.

D Details of Adversarial Attacks

We adopt two challenging adversarial attack strategies from TextAttack (Morris et al., 2020): **DeepWordBug** (Gao et al., 2018) and **TextBugger** (Li et al., 2019) to evaluate the robustness of HAPDA under perturbations. These methods generate adversarial yet fluent texts by applying various transformations. Our implementations follow the configurations described in TextAttack (Morris et al., 2020).

- **DeepWordBug** (Gao et al., 2018) is an untargeted attack for classification tasks based on character-level Levenshtein edit distance constraints. It applies four types of character-level perturbations: *Character Insertion*, *Character Deletion*, *Neighboring Character Swap*, and *Character Substitution*. The search method used is a greedy approach called *Word Importance Ranking* (WIR). In our setup, we adopt the default TextAttack configuration.

- **TextBugger** (Li et al., 2019) is an untargeted black-box attack for classification, which maintains semantic similarity measured by cosine similarity on sentence embeddings. It utilizes similar character-level perturbations as DeepWordBug (*Insertion*, *Deletion*, *Swap*, *Substitution*). The search also uses a greedy Word Importance Ranking method. Our implementation follows the default TextAttack settings.

E HAPDA-Finetune Results

As shown in Figure 8(a), the probability distribution of AI-generated texts obtained from the proxy model is generally higher than that of human-written texts, which is consistent with findings in prior studies (Mitchell et al., 2023; Mao et al., 2024). As illustrated in Figure 8(b), the probability gap between AI-generated texts and human-written texts under $\mathcal{M}_{\text{mac}}$, obtained via HAPDA fine-tuning, is larger than that observed in Figure 8(a). This indicates that $\mathcal{M}_{\text{mac}}$ exhibits a stronger preference for AI-generated texts. Similarly, as shown in Figure 8(c), $\mathcal{M}_{\text{hum}}$ shows a stronger preference for human-written texts. Therefore, both $\mathcal{M}_{\text{mac}}$ and $\mathcal{M}_{\text{hum}}$ exhibit consistent preferences across texts from different origins, which provides a reliable foundation for the subsequent calibration process. This also aligns with our objective of alignment in Section 3.2.

Furthermore, we employ the Silhouette coefficient (Rousseeuw, 1987), DBI (Davies and Bouldin, 1979), and MMD (Gretton et al., 2006) to quantify the degree of preference divergence among these models towards human and machine text. As shown in Table 6, compared to the reference model $\mathcal{M}_{\text{ref}}$, the fine-tuned preference models obtained via HAPDA-Finetune demonstrate a significantly enhanced divergence in probability distributions across different text types. This finding aligns with the objective of distinctiveness in our training objective outlined in Section 3.2.

Metric	$\mathcal{M}_{\text{ref}}$	$\mathcal{M}_{\text{mac}}$	$\mathcal{M}_{\text{hum}}$
Silhouette (↑)	0.012	0.077	0.032
DBI (↓)	7.806	3.399	5.448
MMD (↑)	0.025	0.123	0.054

Table 6: The statistical results of the probability distributions of test documents under $\mathcal{M}_{\text{ref}}$, $\mathcal{M}_{\text{mac}}$, and $\mathcal{M}_{\text{hum}}$ in the Books/GPT-4 setting. The symbols ↑ and ↓ indicate the degree of separation between distributions (higher and lower).

Method/Settings	**In-Domain**	**Cross-Domain**
RoBERTa (Supervised)	**96.4**	80.7
DetectGPT (Zero-shot)	70.3	69.4
DetectGPT* (Zero-shot)	84.9	**83.3**

Table 7: Detection results (AUROC) under in-domain and cross-domain scenarios compared with supervised method. "*" denotes "+HAPDA"

F Comparison with Supervised Method

HAPDA differs fundamentally from supervised learning methods in its objectives, as it is designed to address a distinct problem setting. HAPDA is an addition to zero-shot frameworks, which require no labeled data from the target domain. Its strengths lie in its generalization capability, flexibility, and robustness to unknown domains or models. In contrast, supervised learning models are constrained by data dependency and the risk of overfitting. Their performance may significantly decline when test data originate from different domains or are generated by unknown models, and they often lack interpretability. HAPDA addresses this by providing interpretable decision justifications through token-level weighting and can flexibly enhance various MeanZero detectors as an adapter. To fairly demonstrate the trade-offs between the two paradigms in different scenarios, we supplement the comparison with RoBERTa (experimental configurations are consistent with the prior work(Bhattacharjee et al., 2024)) and conduct a new experiment in a black-box setting. For in-domain testing, we use GPT-4-generated samples from Reviews, Wiki, and News, splitting them into training/validation/test sets at an 8:1:1 ratio. For cross-domain testing, the trained supervised model from in-domain testing is evaluated on the unseen Books dataset.

As shown in Table 7, the supervised method achieves optimal performance in the in-domain sce-

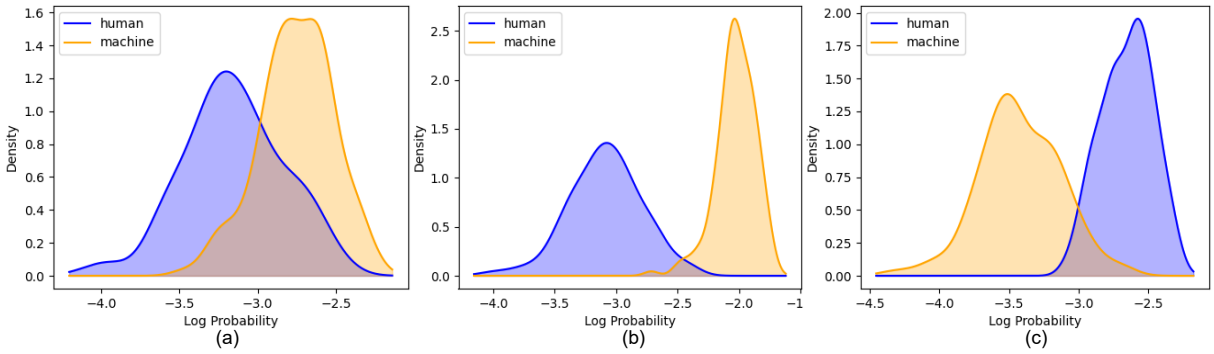

Figure 8: Probability distributions of test documents under $\mathcal{M}_{\text{ref}}$ (a), $\mathcal{M}_{\text{mac}}$ (b), and $\mathcal{M}_{\text{hum}}$ (c) in the Books/GPT-4 setting.

nario where labeled data is available; however, in the cross-domain setting, the supervised model experiences a sharp performance decline due to overfitting to the training domain, whereas HAPDA demonstrates significantly stronger generalization capability. Thus, rather than aiming to replace supervised learning, HAPDA provides a robust zero-shot solution for real-world scenarios that lack labeled data yet demand high generalization performance.

G Supplementary Experiments

G.1 Evaluation Datasets Extension

To further support our conclusions, we conduct additional comparisons on other datasets from the RAID benchmark. In Table 8, we present the results on the **Reviews** and **Wiki** datasets (using the same settings described in Section B.2) with GPT-4 as the source model.

The original Fast-DetectGPT (with AUROC scores of 75.4 and 73.8, respectively) underperforms DNA-GPT (with AUROC scores of 76.8 and 74.2, respectively) and other methods. However, when augmented with HAPDA, its performance significantly surpasses all existing detectors, achieving AUROC scores of 86.4 and 84.7, respectively. This demonstrates that HAPDA can consistently and significantly improve the baseline method across different datasets.

G.2 Evaluation LLMs Extension

We also conducted new experiments using two of the latest and most powerful large language models, GPT-5.2 (Hurst et al., 2024) and DeepSeek-V3.2 (Liu et al., 2024a), under their default API settings. The results on the **Books** dataset under the black-box setting are presented in Table 9. Even on these newest models, HAPDA continues to deliver significant performance gains, demonstrating its robustness and practical value for the latest AI-generated text.

Methods/Datasets	**Reviews**	**Wiki**
DNA-GPT	76.8	74.2
Raidar	79.5	77.8
Binoculars	82.7	80.9
Lastde	83.1	81.5
Fast-DetectGPT	75.4	73.8
Fast-DetectGPT*	**86.4**	**84.7**

Table 8: Comparison with zero-shot detectors on Reviews and Wiki datasets (GPT-4 as source model). "*" denotes "+HAPDA".

Methods/Models	**GPT-5.2**	**DeepSeek-V3.2**
Likelihood	48.1	52.4
Likelihood*	60.5	61.7
DNA-DetectGPT	63.2	66.9
DetectGPT*	77.1	84.0
Fast-DetectGPT	74.6	76.3
Fast-DetectGPT*	87.0	90.8

Table 9: Detection results on two recent LLMs. "*" denotes "+HAPDA".

H Pseudocode of HAPDA

In Section 3, we have provided a detailed description of HAPDA's workflow. In this section, we summarize the procedures of HAPDA-Finetune and HAPDA-Calibration in the pseudocode shown in Algorithms 1 and 2.

Algorithm 1 HAPDA-Finetune

Input: Reference model $\mathcal{M}_{\text{ref}}$, initial weights for $\mathcal{M}_{\text{hum}}, \mathcal{M}_{\text{mac}}$ (copied from $\mathcal{M}_{\text{ref}}$), dataset $\mathcal{D} = \{(P, H, M)\}$ of (prompt, human, machine) triples, trade-off λ, learning rate η, batch size B, number of epochs T.
Output: Fine-tuned models $\mathcal{M}_{\text{hum}}, \mathcal{M}_{\text{mac}}$.
for $epoch = 1 \dots T$ **do**
  for each mini-batch $\mathcal{B} \subset \mathcal{D}$ of size B **do**
    for each $(P, H, M) \in \mathcal{B}$ **in parallel do**
      compute token-level (or sequence-level) log-probabilities under $\mathcal{M}_{\text{hum}}, \mathcal{M}_{\text{mac}}, \mathcal{M}_{\text{ref}}$:

$$\log \pi_{\text{hum}}(H \mid P),\ \log \pi_{\text{hum}}(M \mid P),\ \log \pi_{\text{mac}}(H \mid P),\ \log \pi_{\text{mac}}(M \mid P),$$

      and similarly for $\mathcal{M}_{\text{ref}}$.
      compute preference margins (per triple):

$$\Delta_{\text{mac}} \leftarrow \log \frac{\pi_{\text{mac}}(M \mid P)}{\pi_{\text{ref}}(M \mid P)} - \log \frac{\pi_{\text{mac}}(H \mid P)}{\pi_{\text{ref}}(H \mid P)},$$

$$\Delta_{\text{hum}} \leftarrow \log \frac{\pi_{\text{hum}}(H \mid P)}{\pi_{\text{ref}}(H \mid P)} - \log \frac{\pi_{\text{hum}}(M \mid P)}{\pi_{\text{ref}}(M \mid P)}.$$

    end for
    compute alignment loss over batch:

$$\mathcal{L}_{\text{a}} \leftarrow -\frac{1}{|\mathcal{B}|} \sum_{(P,H,M)\in\mathcal{B}} \big(\log \sigma(\Delta_{\text{mac}}) + \log \sigma(\Delta_{\text{hum}}) \big).$$

    compute token-level distributions $p^Y_{\text{mac}}, p^Y_{\text{hum}}$ for $Y \in \{H, M\}$ and the JS divergence terms,

$$\mathcal{L}_{\text{d}} \leftarrow \frac{1}{|\mathcal{B}|} \sum_{(P,H,M)\in\mathcal{B}} \Big(2\log 2 - \text{JS}(p^M_{\text{mac}} || p^M_{\text{hum}}) - \text{JS}(p^H_{\text{mac}} || p^H_{\text{hum}}) \Big).$$

    total loss: $\mathcal{L} \leftarrow \mathcal{L}_{\text{a}} + \lambda \cdot \mathcal{L}_{\text{d}}$.
    update $\mathcal{M}_{\text{hum}}, \mathcal{M}_{\text{mac}}$ by gradient descent on $\mathcal{L}$ with lr η (optionally apply LoRA/quantization-aware steps).
  end for
end for
return $\mathcal{M}_{\text{hum}}, \mathcal{M}_{\text{mac}}$

Algorithm 2 HAPDA-Calibration

Input: Fine-tuned preference models $\mathcal{M}_{\text{hum}}, \mathcal{M}_{\text{mac}}$, proxy detector model $\mathcal{M}_{\text{pxy}}$, input token sequence $X = (x_1, \ldots, x_n)$, vocabulary size V.
Output: Calibrated detection score S_X^*.
for $i = 1$ to n **do**
    compute conditional probabilities for token x_i:

$$p_{\text{hum}}(\cdot \mid x_{<i}) \leftarrow \pi_{\text{hum}}(\cdot \mid x_{<i}), \quad p_{\text{mac}}(\cdot \mid x_{<i}) \leftarrow \pi_{\text{mac}}(\cdot \mid x_{<i}).$$

    compute disagreement magnitude:

$$\Gamma_i \leftarrow \left|\pi_{\text{hum}}(x_i \mid x_{<i}) - \pi_{\text{mac}}(x_i \mid x_{<i})\right|.$$

    compute entropies:

$$E_{\text{hum}}(x_i) \leftarrow -\sum_{v \in V} p_{\text{hum}}(v \mid x_{<i}) \log p_{\text{hum}}(v \mid x_{<i}),$$

$$E_{\text{mac}}(x_i) \leftarrow -\sum_{v \in V} p_{\text{mac}}(v \mid x_{<i}) \log p_{\text{mac}}(v \mid x_{<i}).$$

    compute normalized uncertainty:

$$U_i \leftarrow \frac{1}{2 \log V}\big(E_{\text{hum}}(x_i) + E_{\text{mac}}(x_i)\big).$$

    unnormalized weight:

$$\tilde{w}_i \leftarrow \Gamma_i \cdot (1 - U_i).$$

    compute per-token AI-likeness score (example: log-prob from proxy model):

$$s_i \leftarrow \log \pi_{\text{pxy}}(x_i \mid x_{<i}).$$

end for
normalize weights: $w_i \leftarrow \tilde{w}_i / \sum_{j=1}^{n} \tilde{w}_j$ (if denominator is 0, fall back to uniform weights).
compute final calibrated detection score:

$$S_X^* \leftarrow \sum_{i=1}^{n} w_i \cdot s_i.$$

return S_X^*