Decoupling Content and Expression: Two-Dimensional Detection of AI-Generated Text

Anonymous ACL submission

Abstract

The wide usage of LLMs raises critical requirements on detecting AI participation in texts. Existing studies investigate these detections in scattered contexts, leaving a systematic and unified approach unexplored. In this paper, we present *HART*, a hierarchical framework of AI risk levels, each corresponding to a detection task. To address these tasks, we propose a novel *2D Detection Method*, decoupling a text into content and language expression. Our findings show that content is resistant to surface-level changes, which can serve as a key feature for detection. Experiments demonstrate that 2D method significantly outperforms existing detectors, achieving an AUROC improvement from 0.705 to 0.849 for level-2 detection and from 0.807 to 0.886 for RAID. We release our data and code at `https://github.com/xxxx`.

1 Introduction

Large language models (LLMs) have shown strong text generation abilities, leading to the rise of AI-assisted text creation in news, academic, story, and advertising writing (Christian, 2023; M Alshater, 2022; Yuan et al., 2022; Chen and Chan, 2023). The coauthorship between humans and machines has become the norm in the era of LLM (Lee et al., 2022; Nguyen et al., 2024; Liang et al., 2024). However, we have different levels of tolerance for AI in different contexts. For example, in academic paper writing, conferences and journals usually accept papers polished using LLMs but reject papers fabricated by models. In writing class, teachers prefer the essays written completely by students, denying the usage of AI. These application scenarios require techniques to detect AI participation in text creation at varying levels, which can be categorized into four types as illustrated in Figure 1.

Prior research explores detection of AI-generated text across different contexts. Early studies concentrate on identifying fully AI-generated text (Gehrmann et al., 2019; Ippolito et al., 2020; Mitchell et al., 2023), while later studies address challenges like paraphrasing and adversarial attacks (Krishna et al., 2024; He et al., 2024; Dugan et al., 2024; Wu et al., 2024). Recently, the focus shifts toward identifying mixed human-AI content (Wang et al., 2024d; Richburg et al., 2024; Zhang et al., 2024; Abassy et al., 2024). However, these studies, tailored to specific contexts and detector designs, lack a systematic framework capable of addressing all levels of AI participation in a unified manner.

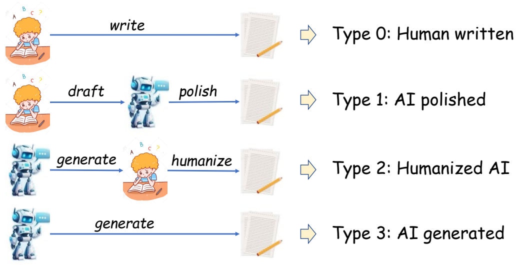

Figure 1: AI participation in text creation

In this paper, we introduce **HART** (**H**ierarchical **AI** **R**isk in **T**ext Creation), a comprehensive framework of AI risk levels that targets the four types of AI participation, as depicted in Figure 2(a). Each risk level corresponds to a detection task, where a binary classifier is required. To systematically tackle these tasks, we propose decoupling the content and language expression of a text, as illustrated in Figure 2(b). We map the four types of AI participation onto the four quadrants of a two-dimensional coordinate system, where type 2 and type 3 (AI content) are marked as high risk (in red) due to the potential for misinformation, bias, or harmful content, while type 1 (AI expression) is considered low risk (in yellow) as it primarily affects readers' experience. Based on the two-dimensional view, we propose a novel **2D Detection Method** that decomposes the problem into two sub-problems: detect-

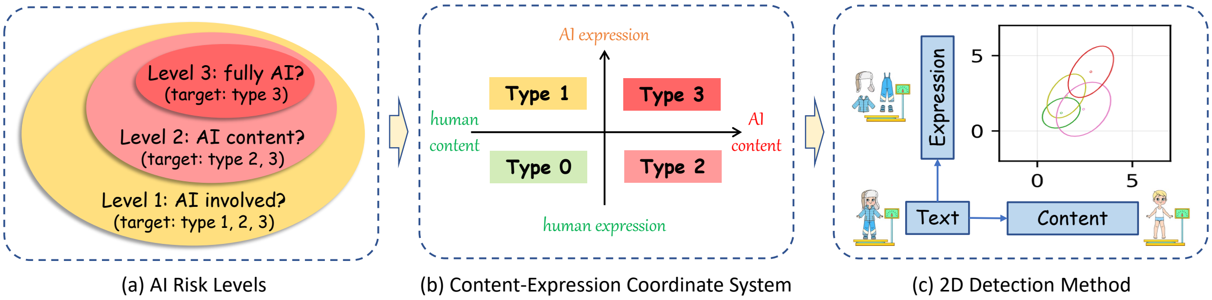

Figure 2: The detection tasks across three risk levels address the four types of AI participation. We represent these types in a two-dimensional space, leading to a 2D detection approach. In this method, the detector performs a binary classification within the two-dimensional space for each detection task.

ing AI content and detecting AI expression. Each corresponds to a distinct textual feature, mapped to a scalar metric, as Figure 2(c) illustrates.

We hypothesize that content is the essential feature in distinguishing AI-generated texts from human-written ones. This is because content is relatively stable and less affected by superficial text changes. For example, the content "*the sun rises in the east*" can be expressed in different styles – academic as "*the sun appears to ascent from the eastern horizon*", news as "*the sun rose in the east this morning, marking another predictable beginning to the day for residents in the region*", and poetry as "*in dawn's gentle embrace, the golden orb doth rise, from eastern realms it paints the morning skies*". The core ideas of the content remain the same no matter whether words, grammar, style, and tone are altered.

To test this hypothesis, we investigate two fundamental research questions.

Q1: How can a text's content and expression be effectively decoupled?

Q2: How can AI content and AI expression be reliably detected?

For Q1, we explore two prototyping approaches: *extraction*, which isolates main ideas or key expressions to represent content, and *neutralization*, which simplifies the text by removing unique stylistic elements or ideas. For Q2, we assess existing detection models on these content-driven and expression-driven representations, finding that current metric-based detectors can indeed be adapted to identify both AI-generated content and AI-modified expressions.

Experimental results reveal that existing detectors struggle with AI-risk detection tasks due to their high sensitivity to surface-level text changes. In contrast, leveraging content-based features proves more robust, outperforming traditional detectors across multiple domains in the HART and RAID datasets. Further improvements are observed when content and expression features are integrated; the 2D framework boosts the best AUROC for the level-2 detection task from 0.711 to 0.855 and TPR5% from 47% to 59%; and it enhances the best AUROC on RAID from 0.807 to 0.886.

To our knowledge, this is the first work to tackle the detection problem by focusing on *content* as a key feature, demonstrating its importance in distinguishing AI-generated texts and effectively mitigating diverse *attacks* on detection systems.

2 Related Work

AI-Assisted Text Creation. LLMs have made significant progress in the area of creative assistance (Zhao et al., 2023b; Lund et al., 2023; Wasi et al., 2024). These models can generate coherent, natural text, offer a variety of writing styles and expressions, and are adapted for various writing tasks such as scientific technology (Gero et al., 2022; Salimi and Saheb, 2023; Lund et al., 2023), storytelling (Yuan et al., 2022; Zhao et al., 2023b; Wang et al., 2024a,b), and news media (Cheng et al., 2024). On the one hand, LLMs can help creators improve their writing efficiency, and on the other hand, they can enhance the quality of their writing. In this paper, we categorize the ways in which LLMs assist in creating text content into four types and propose detection tasks to cover them.

AI-Generated Text Detection. The tasks we propose are related to existing AI-generated text detection tasks (Wu et al., 2023; Yang et al., 2023b), where existing tasks do not consider AI participation levels in text creation. It is also related to existing refined detection tasks (Zhang et al., 2024; Richburg et al., 2024), where these tasks identify operations applied to text by LLMs or humans. In contrast, we focus on AI risk levels instead of specific operations.

Existing detectors consist of three types of technology. The first is supervised classifiers(Solaiman et al., 2019; Ippolito et al., 2020; Fagni et al., 2021; Hu et al., 2023; Yan et al., 2023; Li et al., 2024; Verma et al., 2024), which train a binary classifier based on a large collection of machine-generated and human-written text. The second is zero-shot classifiers, including white-box methods (Gehrmann et al., 2019; Su et al., 2023; Bao et al., 2024; Xu et al., 2024; Hans et al., 2024) and black-box methods (Mitchell et al., 2023; Yang et al., 2023a; Bhattacharjee and Liu, 2024; Bao et al., 2025). These technologies usually use pre-trained language models to extract detection metrics. The third is text watermarking technology (Kirchenbauer et al., 2023; Zhao et al., 2023a; Christ et al., 2024; Zhao et al., 2024b,a), which identifies machine-generated text by embedding easy-to-detect markers or patterns.

These techniques are effective in detecting purely machine-generated texts, but may not be robust to various attacks (Gao et al., 2018; Dyrmishi et al., 2023; Krishna et al., 2024; He et al., 2024; Dugan et al., 2024; Wu et al., 2024; Wang et al., 2024c). At the same time, various commercial AI systems are published to serve 'humanizing' ability, bypass existing detectors. To address these challenges, we propose the 2D detection framework as an effective candidate to defend against attacks.

Decoupling of Content and Expression. The idea of decoupling content and expression is related to existing studies on the disentanglement of semantics and syntax. These studies mainly focus on the disentanglement at the sentence level and discuss about it in different contexts, such as recognition science (Caucheteux et al., 2021; Moro et al., 2001), sentence representation (Chen et al., 2019), sentence comprehension (Dapretto and Bookheimer, 1999), and sentence generation (Bao et al., 2019). They generally represent semantics and syntax in separate neural vectors and train a neural network with specific structure or training objective to obtain disentangled vectors.

However, our decoupling of content and expression differs from these early studies in three aspects. First, we focus on discourse level instead of sentence level, where the texts are longer and more complex. Second, we represent content and expression still in texts instead of neural vectors, which provides us with a convenience for understanding and explaining. Finally, we decouple them using zero-shot prompting techniques instead of training a model, which simplifies the usage.

3 Task and Benchmark

3.1 Task Definition

We consider AI risks in the dimensions of content and expression, categorizing AI risks into three levels and defining detection tasks accordingly as Figure 2(a) shows.

Level-1 Detection: It targets types 1, 2, and 3, covering all texts in which their creation involves AI techniques. This task is suitable for strict situations where AI assistance is forbidden.

Level-2 Detection: It targets type 2 and 3, covering all texts whose contents are generated by AI. These texts may contain fabricated content that may deliver wrong, biased, or dangerous information. This task suites for common situations where AI content may cause risks. Existing AI-generated text detection tasks can be seen as level-2 detection.

Level-3 Detection: It targets type 3 only, where texts are generated by LLMs from scratch. This task suites for loose situations, where AI content is allowed, but readers' experience matters. Early research in pure AI-generated text detection can be seen as level-3 detection.

AI-assisted text creation in real scenarios is complex, where it is likely that human and AI participate iteratively. In this case, it is hard to define the risk levels. However, we could use the definition of level-1 to 3 as a lens to analyze the texts.

3.2 Benchmark Dataset

We create the benchmark dataset *HART* for AI risk detection following a strict construction process and thorough quality assurance. Detailed statistics are provided in Table 1, while information on model and parameter coverage can be found in Appendix A.3.4.

3.2.1 Data Construction

To begin, we gather human-written texts from diverse sources, creating type-0 samples. Next, we refine these texts to produce type-1 samples, which

Domain	Language	Length	Dev	Test
Student Essay	English	241 words	2K	2K
ArXiv Intro	English	410 words	2K	2K
Creative Writing	English	345 words	2K	2K
CC News	English	148 words	2K	2K
CC News	Chinese	590 chars	2K	2K
CC News	French	258 words	2K	2K
CC News	Spanish	285 words	2K	2K
CC News	Arabic	152 words	2K	2K

Table 1: Domains and languages covered by HART.

preserve the original meanings but use different expressions. Using the titles or prompts of the human texts, we generate AI-written content, resulting in type-3 samples. We then adapt the AI-generated texts to create type-2 samples, which retain AI-generated content but are expressed in a more human-like manner. As a result, we obtain an equal distribution of samples across all four types.

Human Texts Collection. We consider the most common domains explored in AI-generated text detection research. Specifically, we utilize *student essays* from Automated Student Assessment Prize (ASAP) 2.0 dataset (Crossley et al., 2024), *paper introductions* sourced from arXiv (arXiv, 2024), *story writings* taken from WritingPrompts (Fan et al., 2018), and *news articles* obtained from Common Crawl (Hamborg et al., 2017), as detailed in Appendix A.1. The news articles are collected in five different languages. For every domain and language, we randomly sample 1000 examples, dividing them equally into development and test sets.

Automatic Refinement. LLMs are commonly employed to improve the expression of human-written drafts. We focus on two refinement methods: polishing and restructuring. *Polishing* aims to enhance the readability and coherence of the text, typically adjusting language at the word and sentence levels. *Restructuring*, on the other hand, focuses on improving the logical flow by reorganizing content, which demands a deeper grasp of the main ideas and the text's purpose. These refinement approaches are applied using the prompts outlined in Appendix A.2.

Machine Texts Generation. We create AI-generated texts using titles or prompts derived from human-written content. For instance, in the case of student essays, we instruct LLMs with a prompt such as: "*Write a student essay (no title) in {nwords} words (split into {nparagraphs} paragraphs) based on the given title: {title}*". To ensure that the generated texts closely match the average length of human-written texts, we specify the same number of words (or characters for Chinese) and paragraphs in the prompt. The detailed prompts for all domains can be found in Appendix A.1.

Humanizing. AI-generated texts can be humanized to enhance their expressive quality. This can be achieved through human editing, the use of external tools, and two automated approaches: diversifying and mimicking. *Diversifying* involves increasing the linguistic variety of AI-generated content, resembling the paraphrasing technique that enhances lexical and grammatical diversity (Krishna et al., 2024). *Mimicking*, on the other hand, prompts LLMs to emulate a human-written reference text in order to rewrite AI-produced content. These two strategies are applied using simple prompts, as detailed in Appendix A.3.1, while human editing and the use of external tools are discussed in Appendices A.3.3 and A.3.2, respectively.

3.3 Quality Assurance

We evaluate the length of each generation from the LLM output, and if it is significantly longer (more than twice the original length) or shorter (less than half the original length), we prompt the LLM to generate the text again. Additionally, we monitor for issues like repetition or nonsensical responses and address them by re-generating the text. After processing the data, we truncate the texts to ensure the length distributions are consistent across types 0, 1, 2, and 3. As a final quality check, we randomly select 100 samples per domain for manual review, achieving an average pass rate of 99.5%. In terms of costs, the data construction process involves approximately $2,000 for LLM API usage, $2,000 for human editing, $500 for AI tools, and $500 for the manual review step.

4 2D Detection Method

We prototype to test our hypothesis using a simple zero-shot prompting technique.

4.1 Decoupling Content and Expression

Achieving a "perfect decoupling" of content and expression means presenting core ideas or meaning (content) in a way that is entirely independent from stylistic or linguistic expression. In this study, we propose a prototyping method by extracting content

Original Text (T): *Jane was walking through the park when she spotted a small puppy wandering aimlessly. She picked it up, read the collar tag, and took it to the address. The owner was grateful to have the puppy back.*	
Prompt 1: *Outline the main points of the text to get a clear and concise picture of the content.*	**Prompt 1**: *Identify and list the representative language expressions used in the text.*
Content (C_1)**:** *1. Jane was walking through the park. 2. She noticed a small puppy wandering aimlessly. 3. Jane found a collar tag on the puppy. 4. She took the puppy to the address on the tag. 5. The owner was grateful to Jane for returning the puppy.*	**Expression** (E_1)**:** *1. "walking through the park" - indicates Jane's action and setting. 2. "spotted a small puppy" - describes Jane noticing the puppy. 3. "wandering aimlessly" - characterizes the puppy's behavior. 4. "picked it up" - describes Jane's action of picking up the puppy.*
Prompt 2: *Simplify the text to make it clear and concise while preserving its meaning.*	**Prompt 2:** *Replace the main points of the text with a generic topic while preserving the language expression.*
Content (C_2)**:** *Jane found a puppy in the park and returned it to its grateful owner after reading the collar tag.*	**Expression** (E_2)**:** *Alex was strolling through the garden when they noticed a tiny kitten meandering without direction. They scooped it up, checked the collar tag, and brought it to the listed location. The caretaker was thankful to have the kitten returned.*

Table 2: Decoupling content and language using extraction and neutralization prompts.

of a text and describing it in simple language to create a representation of the content, and discarding content of a text but keeping its language style and tone to create a representation of its expression.

Specifically, as Table 2 shows, we investigate two decoupling methods: extraction and neutralization. The extraction method produces a brief outline of the main ideas and a list of representative expressions of a text. Although the outline and list produced by the extraction method are relatively short and empirically effectual, they lose significant amount of details of the text. The neutralization method mitigates this issue. It reserves more details about the content and expression with longer text descriptions, which is empirically better for detection tasks.

4.2 Detection of AI Content and Expression

Intuitively, language models produce less surprising text than humans because the models are trained to minimize the empirical risk on human-written texts, which encourages the model to generate common patterns in the training data. Thus, AI-generated texts generally tend to have lower perplexity than human-written texts and can be detected by perplexity-based detectors. However, perplexity-based detectors are easily deceived by altered expressions because perplexity itself cannot distinguish a surprising content from a surprising expression.

By decoupling content and expression, we can measure the surprisingness of them separately. Thus, many existing metric-based detectors can be used to detect AI content and expression. Take Fast-Detect as an example. As Figure 2(c) shows, its metric – conditional probability curvature – can be used to map the textual features into scalars, resulting in a two-dimensional distribution of texts for the four types. We also tried trained detectors such as RADAR, but failed to obtain an improvement in the AI content detection task. These detectors may need further training to handle the textual features. In our experiments, we empirically choose Fast-Detect and Binoculars as the representatives.

5 Experiments

We confirm our hypothesis and demonstrate that combining content and expression features provides us with a stronger detection ability to AI risks in Section 5.2 and resilience to various attacks in Section 5.3.

5.1 Experimental Settings

Detectors. We mainly focus on *metric-based detectors*, which generally leverage existing pre-trained LLMs to compute a metric as an indicator of AI-generated text. We take *log-perplexity*, *log-rank*, *LRR* (Su et al., 2023), *Fast-Detect* (Bao et al., 2024), *Binoculars* (Hans et al., 2024), and *Glimpse* (Bao et al., 2025) as representatives, as described in Appendix B. For fair comparison, we unify the scoring models to falcon-7B or falcon-7B-instruct (except for Glimpse), where we find that these models perform significantly better than smaller models such as gpt-neo-2.7B.

We also consider *trained detectors*, such as RADAR (Hu et al., 2023) and RoBERTa (ChatGPT) (Guo et al., 2023). However, these detectors cannot detect extracted textual features without further training. Thus, we just list them for reference.

Metrics. We study the detection problem in various application scenarios, where the tolerance for false positive rate is unknown. Consequently, we use *AUROC* (area under the receiver operating characteristic curve) as the major metric to measure the quality of the classifiers. We also report *F1* and *TPR5%* (true positive rate at a false positive rate of 5%) for reference.

5.2 Results on Multi-Level AI Risk Detection

We compare existing detectors and 2D methods in HART as Table 3 shows, achieving the following

Detector	Level-3 Detection Task				Level-2 Detection Task				Level-1 Detection Task			
	Essay	ArXiv	Writing	ALL (TPR5%)	Essay	ArXiv	Writing	ALL (TPR5%)	Essay	ArXiv	Writing	ALL (TPR5%)
RoBERTa(ChatGPT)	0.636	0.796	0.653	0.662 (16%)	0.435	0.687	0.498	0.502 (8%)	0.471	**0.955**	0.606	0.566 (9%)
RADAR	0.692	0.849	0.647	0.728 (14%)	0.566	0.814	0.630	0.687 (10%)	0.705	0.857	0.700	0.758 (20%)
Log-Perplexity	0.868	0.850	0.811	0.799 (33%)	0.364	0.485	0.438	0.473 (11%)	0.769	0.530	0.625	0.576 (6%)
Log-Rank	0.867	0.874	0.813	0.814 (39%)	0.380	0.460	0.441	0.465 (11%)	0.739	0.542	0.611	0.573 (8%)
LRR	0.835	**0.909**	0.797	0.840 (50%)	0.560	0.616	0.551	0.573 (25%)	0.616	0.576	0.558	0.568 (19%)
Glimpse	**0.929**	0.869	0.819	0.849 (58%)	0.754	0.737	0.625	0.676 (30%)	0.878	0.719	0.618	0.688 (22%)
Fast-Detect	0.883	0.877	0.840	0.862 (60%)	0.734	0.718	0.692	0.711 (47%)	0.877	0.769	0.740	0.778 (55%)
C_2 (Fast-Detect)	0.734	0.787	0.765	0.738 (18%)	0.778	0.862	0.819	0.798 (42%)	0.712	0.779	0.742	0.730 (33%)
C_2-T (Fast-Detect)	0.864	0.896	0.890	0.876 (61%)	**0.785**	**0.915**	0.890	**0.855 (59%)**	**0.907**	0.849	**0.836**	**0.843 (59%)**
Binoculars	0.897	0.882	0.847	0.870 (**62%**)	0.735	0.715	0.693	0.711 (44%)	0.879	0.769	0.740	0.780 (55%)
C_2 (Binoculars)	0.736	0.789	0.770	0.737 (17%)	0.781	0.856	0.822	0.791 (35%)	0.701	0.761	0.743	0.716 (25%)
C_2-T (Binoculars)	0.854	0.904	**0.905**	**0.883** (61%)	0.746	0.913	**0.895**	0.848 (32%)	0.900	0.840	0.828	0.838 (58%)

Table 3: Results on AI risk detection, evaluated on HART. The best AUROCs and TPR5% are marked in **bold** and second in underline. The column '*ALL*' denotes a mixture of domains including Essay, arXiv, Writing, and News in English.

findings.

Finding 1: Existing detectors are good at level-3 detection but poor at level-2/1 detections. Existing detectors generally perform the best on the level-3 detection task, where Binoculars reaches an overall AUROC of 0.870 and TPR5% of 62%. These scores are significantly higher than those on level-2 and 1 detection tasks, suggesting that existing detectors may best suit pure AI-generated texts. This is potentially because existing detectors mainly measure texts along expression dimension, thus being sensitive to changes in language expressions.

Finding 2: The content feature is resilient to changes in language expression, resulting in better level-2 and 1 detection performance. When we compare 2D (C_2-T) methods with existing detectors, we find that although 2D methods perform at the same level as existing detectors on the level-3 detection task, they outperform existing detectors by a large margin on level-2 and 1 detection tasks. It increases overall AUROC from 0.704 to 0.849 on level-2 task and from 0.767 to 0.844 on level-1 task using Fast-Detect. Similarly, TPR5% increases by 12% and 4%, respectively, on the two tasks. These results demonstrate the effectiveness of the 2D detection framework.

We further look into each type of level-2 detection data, as Figure 3 shows. The content feature plays a key role in the detection of humanized AI-generated texts, significantly outperforming the baseline. The results also confirm our hypothesis that content is the essential feature for identifying AI-generated texts.

Finding 3: The content feature is effective across languages. We evaluate the detectors across five

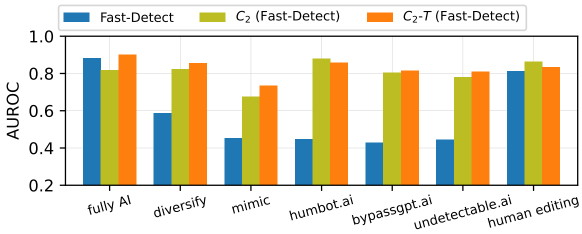

Figure 3: Comparison on their ability to detect AI-generated texts, where '*xxx.ai*' are external humanizing tools.

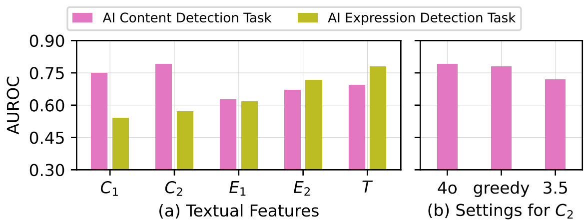

Figure 4: Content and expression features evaluated on AI detection tasks using conditional probability curvature as the feature metric.

languages in Appendix C.1. 2D detectors outperform the baselines on all three tasks, where the improvements on the level-2 and 1 tasks are especially significant. It is worth noting that Glimpse using gpt-3.5-turbo achieves the best overall results across the languages, possibly because of the stronger multilingual ability of the model.

5.2.1 Ablation Study

Textual Features. The quality of *extracted textual features* for content and expression is critical for detection tasks. We first evaluate the candidate features on AI content detection and AI expression detection tasks, each with 1000 pairs of samples from the HART dataset. We use conditional probability curvature as a metric to map textual features

Detector	News	Books	Wiki	Abstracts	Reddit	Recipes	Poetry	Reviews	ALL AUROC	F1	TPR5%
RoBERTa-base	0.588	0.622	0.582	0.643	0.673	0.500	0.638	0.710	0.614	67%	24%
RADAR	0.884	0.912	0.842	0.842	**0.870**	0.818	0.780	0.782	0.828	77%	42%
Log-Perplexity	0.644	0.725	0.701	0.680	0.725	0.627	0.706	0.698	0.663	66%	12%
Log-Rank	0.666	0.745	0.719	0.701	0.735	0.645	0.725	0.716	0.681	67%	14%
LRR	0.750	0.816	0.804	0.771	0.779	0.669	0.776	0.773	0.746	70%	34%
Glimpse	0.712	0.758	0.589	0.787	0.742	0.670	0.756	0.728	0.715	67%	39%
Fast-Detect	0.761	0.845	0.803	0.821	0.794	0.749	0.818	0.810	0.800	76%	54%
Binoculars	0.768	0.850	0.804	0.826	0.811	0.759	0.826	0.812	0.807	77%	58%
C_2 (Binoculars)	0.783	0.888	0.808	0.799	0.778	0.726	0.777	0.762	0.774	72%	46%
C_2-T (Binoculars)	**0.901**	**0.924**	**0.861**	**0.900**	0.869	**0.878**	**0.889**	**0.869**	**0.886**	**82%**	**68%**

Table 4: Results on AI-generated text detection, evaluated on RAID. The highest AUROCs are marked in **bold**. C_2, E_2, and T are textual features used for detection, which are illustrated in Table 2.

to scalar values. As Figure 4(a) shows, neutralization generally outperforms extraction approach for both content and expression representations. The content feature C_2 achieves the best performance in the AI content detection task, while the original text T achieves the best performance in the AI expression detection task, surpassing the expression features E_1 and E_2 with a significant margin. Thus, we choose C_2 as the content feature and T as the expression feature for our 2D detectors.

Model and Parameters. The *language model* and *decoding strategy* can also affect the quality of the extracted content. We ablate them as Figure 4(b). Compared to the default setting of gpt-4o with random sampling, greedy decoding slightly decreases the AUROC. However, we find that greedy decoding further improves TPR5% by about 4% on the AI content detection task, which may be because deterministic decoding produces more stable texts. Changing the model to gpt-3.5-turbo causes a significant drop in AUROC, suggesting that a strong LLM is a prerequisite to extract effective content features. In our experiments, we use gpt-4o with random sampling.

5.2.2 Analysis of Data Distribution

What is the impact of source model and decoding parameters to generated texts? Various factors influence the distribution of AI-generated texts (type-3 texts), as described in Appendix C.2.1. Among the source models, gpt-4o demonstrates the most diverse generations and is significantly closer to the origin of the coordinate system, suggesting its stronger ability to produce human-like texts. The decoding temperature also affects the distribution, but the differences are not significant. Similarly, larger top-p and presence penalty produce more diverse texts but the differences are marginal. In contrast, the frequency penalty shows a strong impact on generated texts, where a larger penalty produces more human-like texts.

What has been changed by refinement and humanizing? As described in Appendix C.2.2, refinement and humanizing significantly alter the distribution, mainly along the expression dimension. The change bringing by humanizing is relatively bigger than refinement, where automatic humanizing shifts the distribution largely. Human editing alters the distribution not as significant as the automatic humanizing, which may be because that human annotators do not attack AI detectors purposely as the humanizing tools.

5.2.3 Discussion

Although content features are resilient to surface-level text changes, there is the possibility of developing attacks against the content of a text. However, we posit that attacking detectors at the content level is much harder than at the expression level. Meaningful and coherent content, unlike superficial language expression, requires deep understanding about the world, thus hard to be simulated by current language models. Additionally, a content-level attack may pay additional costs, such as reducing the logical coherence and readability of the generated content.

5.3 Results on AI-Generated Text Detection

We evaluate 2D methods on existing detection tasks. Typically, we use the challenging RAID (Dugan et al., 2024) dataset, from which we sample 4K samples (250 pairs per domain) for testing and another 4K for development.

As Table 4 shows, the columns AUROC, F1, and TPR@5% are evaluated across all domains, where we find the best development threshold for calculating F1. As the AUROCs indicate, the content feature C_2 outperforms the baseline Binoculars on News, Books, and Wiki. When combining content and expression features, 2D (C_2-T) produces the

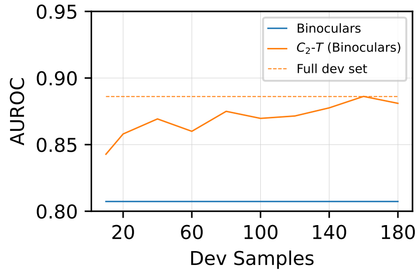

Figure 5: Ablation on the number of dev samples required by 2D method.

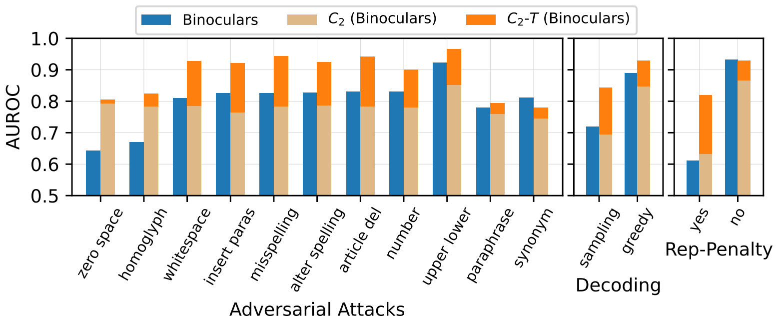

Figure 6: Comparison on their ability to handle adversarial attacks, decoding strategies, and repetition penalty.

best scores across all the domains. These results suggest the positive effect of content feature on the detection of AI-generated text.

Readers may wonder why the content feature does not outperform the baseline in all domains. We speculate that it is correlated to the genre and length of the text. Take poetry as an example. Poetry often relies on evocative language to convey emotions, themes, and ideas. The meaning of a poem can change depending on how it is expressed. Thus, the decoupling of content and expression loses significant information. Similar situations may happen with other free-style texts, such as Reviews and Reddit. (tbd)

Ablation on expression features. Empirically, C_2-T outperforms C_2-E_2 on almost all domains, suggesting that the original text T presents its language expression better than the extracted feature E_2. It confirms our finding in Section 5.2.1 that the original text best represents its expression, while the extracted content feature C_2 best represents its content.

Ablation on the size of development set. 2D methods need to fit a two-dimensional binary classifier, which requires additional samples. We show that such a classifier requires only a small number of samples because of its low dimensions. As Figure 5 shows, 10 random samples are sufficient to outperform the baseline, resulting in a AUROC of 0.8428 in all domains. Empirically, 200 samples are sufficient to reach the full level of performance.

Analysis on attacks, decoding strategies, and repetition penalty. As Figure 6 shows, the content feature outperforms the baseline on '*zero-width-space*', '*homoglyph*', and '*repetition-penalty*', demonstrating its effectiveness. When we combine content and expression features, we achieve significant improvements on all categories except synonym and non-repetition-penalty. The significant improvements on sampling and repetition-penalty suggest that the 2D method is typical beneficial for hard detection situations, given that sampling and repetition-penalty produce more nature texts which are harder for detection. These results suggest the advantage of the 2D method which is resistant to various attacks and decoding strategies.

Addressing the bias toward nonnative writers. Content representation is also resilient to nonnative English writers, where unique language expressions are reduced during content extraction. Consequently, using the content feature C_2 improves the AUROC from 0.4970 (Binoculars) to 0.5212 (C_2 with Binoculars). When we use the best threshold found on RAID development set, it improves F1 from 49% (Binoculars) to 55% (C_2 with Binoculars), demonstrating that the content feature reduces the bias toward nonnative writers.

6 Conclusion

We introduce a hierarchical framework for detection tasks, categorized into three levels of AI risk, which integrates prior research and established requirements. Our study explores 2D detection methods that leverage content as a key feature for identifying AI-generated text, demonstrating that content plays a critical role in addressing such detection challenges. Experimental results indicate that content features exhibit resilience to superficial textual modifications, making them a reliable tool for both emerging AI risk detection and traditional AI-generated text identification tasks. Furthermore, our proposed framework and benchmark dataset lay a strong foundation for advancing future research in this field.

Limitations

The decoupling of content and expression may have various solutions, where prompting techniques may be the simplest, but not necessarily the most effective. There is the possibility to decouple content and expression in a more fundamental approach, with a specific design of a model or an algorithm. On the other hand, detection of content and expression may require different methods given that they are at different levels of text. Therefore, a specific design for each may produce stronger detectors.

Ethical Considerations

The dataset we use contains AI-generated texts, which could potentially be biased, offensive, or irresponsible. Although we filter them with automatic API provided by Azure OpenAI and check 10% samples manually with high pass rate, there are still possibility of having unpleasant content in the released dataset, which may deserve a warning.

References

Mervat Abassy, Kareem Elozeiri, Alexander Aziz, Minh Ta, Raj Tomar, Bimarsha Adhikari, Saad Ahmed, Yuxia Wang, Osama Mohammed Afzal, Zhuohan Xie, et al. 2024. Llm-detectaive: a tool for fine-grained machine-generated text detection. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 336–343.

arXiv. 2024. arxiv. *Cornell University*.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse*, pages 178–186.

Guangsheng Bao, Yanbin Zhao, Juncai He, and Yue Zhang. 2025. Glimpse: Enabling white-box methods to use proprietary models for zero-shot llm-generated text detection. *The Thirteenth International Conference on Learning Representations*.

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*.

Yu Bao, Hao Zhou, Shujian Huang, Lei Li, Lili Mou, Olga Vechtomova, Xinyu Dai, and Jiajun Chen. 2019. Generating sentences from disentangled syntactic and semantic spaces. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6008–6019.

Amrita Bhattacharjee and Huan Liu. 2024. Fighting fire with fire: can chatgpt detect ai-generated text? *ACM SIGKDD Explorations Newsletter*, 25(2):14–21.

Charlotte Caucheteux, Alexandre Gramfort, and Jean-Remi King. 2021. Disentangling syntax and semantics in the brain with deep networks. In *International conference on machine learning*, pages 1336–1348. PMLR.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019. A multi-task approach for disentangling syntax and semantics in sentence representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2453–2464.

Zenan Chen and Jason Chan. 2023. Large language model in creative work: The role of collaboration modality and user expertise. *Available at SSRN 4575598*.

Liqi Cheng, Dazhen Deng, Xiao Xie, Rihong Qiu, Mingliang Xu, and Yingcai Wu. 2024. Snil: Generating sports news from insights with large language models. *IEEE Transactions on Visualization and Computer Graphics*.

Miranda Christ, Sam Gunn, and Or Zamir. 2024. Undetectable watermarks for language models. In *The Thirty Seventh Annual Conference on Learning Theory*, pages 1125–1139. PMLR.

Jon Christian. 2023. Cnet secretly used ai on articles that didn't disclose that fact, staff say. *Futurusm, January*.

Scott Crossley, Perpetual Baffour, Jules King, Lauryn Burleigh, Walter Reade, and Maggie Demkin. 2024. Asap 2.0: Automated student assessment prize. *Kaggle*.

Mirella Dapretto and Susan Y Bookheimer. 1999. Form and content: dissociating syntax and semantics in sentence comprehension. *Neuron*, 24(2):427–432.

Liam Dugan, Alyssa Hwang, Filip Trhlik, Josh Magnus Ludan, Andrew Zhu, Hainiu Xu, Daphne Ippolito, and Chris Callison-Burch. 2024. Raid: A shared benchmark for robust evaluation of machine-generated text detectors. *arXiv preprint arXiv:2405.07940*.

Salijona Dyrmishi, Salah GHAMIZI, and Maxime Cordy. 2023. How do humans perceive adversarial text? a reality check on the validity and naturalness of word-based adversarial attacks. In *The 61st Annual Meeting Of The Association For Computational Linguistics*.

Tiziano Fagni, Fabrizio Falchi, Margherita Gambini, Antonio Martella, and Maurizio Tesconi. 2021. Tweepfake: About detecting deepfake tweets. *Plos one*, 16(5):e0251415.

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

Charles J Fillmore et al. 2006. Frame semantics. *Cognitive linguistics: Basic readings*, 34:373–400.

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.

Katy Ilonka Gero, Vivian Liu, and Lydia Chilton. 2022. Sparks: Inspiration for science writing using language models. In *Proceedings of the 2022 ACM Designing Interactive Systems Conference*, pages 1002–1019.

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 preprint arxiv:2301.07597*.

Felix Hamborg, Norman Meuschke, Corinna Breitinger, and Bela Gipp. 2017. news-please: A generic news crawler and extractor. In *Proceedings of the 15th International Symposium of Information Science*, pages 218–223.

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

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*, pages 2251–2265.

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. 2021. Knowledge graphs. *ACM Computing Surveys (Csur)*, 54(4):1–37.

Xiaomeng Hu, Pin-Yu Chen, and Tsung-Yi Ho. 2023. Radar: Robust ai-text detection via adversarial learning. *Advances in Neural Information Processing Systems*, 36:15077–15095.

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.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A watermark for large language models. In *International Conference on Machine Learning*, pages 17061–17084. PMLR.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. 2024. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. *Advances in Neural Information Processing Systems*, 36.

Mina Lee, Percy Liang, and Qian Yang. 2022. Coauthor: Designing a human-ai collaborative writing dataset for exploring language model capabilities. In *Proceedings of the 2022 CHI conference on human factors in computing systems*, pages 1–19.

Yafu Li, Qintong Li, Leyang Cui, Wei Bi, Zhilin Wang, Longyue Wang, Linyi Yang, Shuming Shi, and Yue Zhang. 2024. Mage: Machine-generated text detection in the wild. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 36–53.

Weixin Liang, Yaohui Zhang, Zhengxuan Wu, Haley Lepp, Wenlong Ji, Xuandong Zhao, Hancheng Cao, Sheng Liu, Siyu He, Zhi Huang, et al. 2024. Mapping the increasing use of llms in scientific papers. *arXiv preprint arXiv:2404.01268*.

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 preprint arXiv:1907.11692*.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online. Association for Computational Linguistics.

Brady D Lund, Ting Wang, Nishith Reddy Mannuru, Bing Nie, Somipam Shimray, and Ziang Wang. 2023. Chatgpt and a new academic reality: Artificial intelligence-written research papers and the ethics of the large language models in scholarly publishing. *Journal of the Association for Information Science and Technology*, 74(5):570–581.

Muneer M Alshater. 2022. Exploring the role of artificial intelligence in enhancing academic performance: A case study of chatgpt. *Available at SSRN*.

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.

Andrea Moro, Marco Tettamanti, Daniela Perani, Caterina Donati, Stefano F Cappa, and Ferruccio Fazio. 2001. Syntax and the brain: disentangling grammar by selective anomalies. *Neuroimage*, 13(1):110–118.

Andy Nguyen, Yvonne Hong, Belle Dang, and Xiaoshan Huang. 2024. Human-ai collaboration patterns in ai-assisted academic writing. *Studies in Higher Education*, pages 1–18.

Martha Palmer, Daniel Gildea, and Nianwen Xue. 2011. *Semantic role labeling*. Morgan & Claypool Publishers.

Hoifung Poon and Pedro Domingos. 2009. Unsupervised semantic parsing. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, pages 1–10.

Aquia Richburg, Calvin Bao, and Marine Carpuat. 2024. Automatic authorship analysis in human-ai collaborative writing. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 1845–1855.

Ali Salimi and Hady Saheb. 2023. Large language models in ophthalmology scientific writing: ethical considerations blurred lines or not at all? *American Journal of Ophthalmology*, 254:177–181.

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

Jinyan Su, Terry Yue Zhuo, Di Wang, and Preslav Nakov. 2023. Detectllm: Leveraging log rank information for zero-shot detection of machine-generated text. *arXiv preprint arXiv:2306.05540*.

Vivek Verma, Eve Fleisig, Nicholas Tomlin, and Dan Klein. 2024. Ghostbuster: Detecting text ghostwritten by large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1702–1717.

Gabriella Vigliocco and David P Vinson. 2007. Semantic representation. *The Oxford handbook of psycholinguistics*, pages 195–215.

Tiannan Wang, Jiamin Chen, Qingrui Jia, Shuai Wang, Ruoyu Fang, Huilin Wang, Zhaowei Gao, Chunzhao Xie, Chuou Xu, Jihong Dai, et al. 2024a. Weaver: Foundation models for creative writing. *arXiv preprint arXiv:2401.17268*.

Yi Wang, Qian Zhou, and David Ledo. 2024b. Storyverse: Towards co-authoring dynamic plot with llm-based character simulation via narrative planning. In *Proceedings of the 19th International Conference on the Foundations of Digital Games*, pages 1–4.

Yichen Wang, Shangbin Feng, Abe Bohan Hou, Xiao Pu, Chao Shen, Xiaoming Liu, Yulia Tsvetkov, and Tianxing He. 2024c. Stumbling blocks: Stress testing the robustness of machine-generated text detectors under attacks. *arXiv preprint arXiv:2402.11638*.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Toru Sasaki, et al. 2024d. M4: Multi-generator, multi-domain, and multi-lingual black-box machine-generated text detection. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1369–1407.

Azmine Toushik Wasi, Rafia Islam, and Raima Islam. 2024. Llms as writing assistants: Exploring perspectives on sense of ownership and reasoning. *arXiv preprint arXiv:2404.00027*.

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

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 *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Yang Xu, Yu Wang, Hao An, Zhichen Liu, and Yongyuan Li. 2024. Detecting subtle differences between human and model languages using spectrum of relative likelihood. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10108–10121.

Duanli Yan, Michael Fauss, Jiangang Hao, and Wenju Cui. 2023. Detection of ai-generated essays in writing assessment. *Psychological Testing and Assessment Modeling*, 65(2):125–144.

Xianjun Yang, Wei Cheng, Yue Wu, Linda Ruth Petzold, William Yang Wang, and Haifeng Chen. 2023a. Dna-gpt: Divergent n-gram analysis for training-free detection of gpt-generated text. In *The Twelfth International Conference on Learning Representations*.

Xianjun Yang, Liangming Pan, Xuandong Zhao, Haifeng Chen, Linda Petzold, William Yang Wang, and Wei Cheng. 2023b. A survey on detection of llms-generated content. *arXiv preprint arXiv:2310.15654*.

Ann Yuan, Andy Coenen, Emily Reif, and Daphne Ippolito. 2022. Wordcraft: story writing with large language models. In *Proceedings of the 27th International Conference on Intelligent User Interfaces*, pages 841–852.

Qihui Zhang, Chujie Gao, Dongping Chen, Yue Huang, Yixin Huang, Zhenyang Sun, Shilin Zhang, Weiye Li, Zhengyan Fu, Yao Wan, et al. 2024. Llm-as-a-coauthor: Can mixed human-written and machine-generated text be detected? In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 409–436.

Xuandong Zhao, Sam Gunn, Miranda Christ, Jaiden Fairoze, Andres Fabrega, Nicholas Carlini, Sanjam Garg, Sanghyun Hong, Milad Nasr, Florian Tramer, et al. 2024a. Sok: Watermarking for ai-generated content. *arXiv preprint arXiv:2411.18479*.

Xuandong Zhao, Lei Li, and Yu-Xiang Wang. 2024b. Permute-and-flip: An optimally robust and watermarkable decoder for llms. *arXiv preprint arXiv:2402.05864*.

Xuandong Zhao, Yu-Xiang Wang, and Lei Li. 2023a. Protecting language generation models via invisible watermarking. In *International Conference on Machine Learning*, pages 42187–42199. PMLR.

Zoie Zhao, Sophie Song, Bridget Duah, Jamie Macbeth, Scott Carter, Monica P Van, Nayeli Suseth Bravo, Matthew Klenk, Kate Sick, and Alexandre LS Filipowicz. 2023b. More human than human: Llm-generated narratives outperform human-llm interleaved narratives. In *Proceedings of the 15th Conference on Creativity and Cognition*, pages 368–370.

A Benchmark Dataset

HART will be released under Creative Commons license, which is also the license publicly available by all the source data.

A.1 Domains and Languages

HART encompasses four domains and five languages, with each language featuring 2,000 development samples and 2,000 test samples. Within these datasets, the samples are evenly distributed across four types (0, 1, 2, and 3). For every domain, human-written texts (type 0) are collected from specific sources, while AI-generated texts (type 3) are produced using prompts outlined in Table 5.

Student Essay. We randomly select 1,000 essays from the Automated Student Assessment Prize (ASAP) 2.0 (Crossley et al., 2024), each accompanied by a title and a prompt. These prompts are utilized to prompt LLMs to generate corresponding essays. Additionally, metadata such as '*race ethnicity*', '*gender*', and '*grade level*' is recorded for potential future analyses.

ArXiv Intro. To build this dataset, we collect 1,000 computer science papers from arXiv (arXiv, 2024) by crawling PDFs published between 2020 and 2024, randomly selecting 200 papers per year. Using S2ORC (Lo et al., 2020), the PDFs are parsed to extract titles and introductions. These titles are then used to prompt LLMs to generate new paper introductions. The inclusion of publication year also provides a basis for analyzing distribution shifts over time.

Creative Writing. We randomly pull 1,000 samples from WritingPrompts (Fan et al., 2018), with each sample paired with a corresponding prompt. These prompts serve as triggers for LLMs to create new fictional stories.

CC News. For this dataset, we gather 1,000 news articles in each of five languages – English, Chinese, French, Spanish, and Arabic – sourced from Common Crawl (Hamborg et al., 2017). The news headlines are used to prompt LLMs to generate full news articles.

A.2 Automatic Refinement

We use the following prompts for automatic refinement of human-written texts.

Student Essay: *Write a student essay (no title) in {n_words} words (split into {n_paragraphs} paragraphs) based on the given {field}:\n {field_value}*

ArXiv Intro: *Write an introductory section (no section name) for an academic paper in {n_words} words (split into {n_paragraphs} paragraphs) based on the given {field}:\n {field_value}*

Creative Writing: *Write a creative story (no title) in {n_words} words (split into {n_paragraphs} paragraphs) based on the given {field}:\n {field_value}*

CC News: *Write a news article (no title) in {n_words} words (split into {n_paragraphs} paragraphs) based on the given {field}:\n {field_value}*

Multi-lingual CC News: *Write a news article (no title) in {lang} language in {n_words} words (split into {n_paragraphs} paragraphs) based on the given {field}:\n {field_value}*

Table 5: Prompts for data generation, where the *field* could be either '*title*' or '*prompt*' depending on their availability for each data source.

Prompt for Polishing: "*Polish the text to enhance its readability, coherence, and flow. Correct any grammatical, punctuation, and style errors, but ensure the core content remains unchanged:\n{generation}*"

Prompt for Restructuring: "*Restructure the text to improve its logical flow and coherence by rearranging paragraphs, sections, or sentences for enhanced clarity and fluency:\n{generation}*"

A.3 Humanizing

A.3.1 Automatic Humanizing

We use the following prompts for humanizing AI-generated texts automatically.

Prompt for Diversifying: "*Revise the text to enrich its language diversity, employing varied sentence structures, synonyms, and stylistic nuances, while preserving the original meaning:\n{generation}*"

Prompt for Mimicking: "*Rewrite the text using the same language style, tone, and expression as the reference text. Focus on capturing the unique vocabulary, sentence structure, and stylistic elements evident in the reference:\n{generation}\n\n# Reference Text:\n{reference}*"

A.3.2 External Humanizing Tools

There are various AI humanizing tools that are developed to bypass detectors. We list a few in Table 6, where the first three are used to produce our type-2 texts. We demonstrate that these tools all alter texts at the surface level, where the content feature has strong resilience.

AI Tool	URL	Used
BypassGPT	https://bypassgpt.ai/	Y
Humbot	https://humbot.ai/	Y
Undetectable AI	https://undetectable.ai/	Y
Semihuman AI	https://semihuman.ai/	
HIX Bypass	https://bypass.hix.ai/	
AI Humanizer	https://aihumanizer.ai/	
StealthGPT	https://stealthgpt.ai/	
GPTinf	https://stealthgpt.ai/	
WriteHuman	https://writehuman.ai/	
StealthWriter	https://rewritify.ai/	
Phrasly LLC	https://phrasly.ai/	
HIX.AI	https://bypass.hix.ai	
AISEO Humanizer	https://aiseo.ai/	
Humanize AI Pro	https://www.humanizeai.pro/	
Smodin	https://smodin.io/	
Rewritify	https://www.rewritify.ai	

Table 6: Humanizing tools that bypass detectors.

A.3.3 Human Editing

We hire five annotators from a specialized annotation company, including three with professional backgrounds in English and two with expertise in computer science. Each annotator is responsible for revising 50 AI-generated texts, resulting in a total of 250 human-edited samples. The editing process is carried out at three levels: word, sentence, and paragraph. At the word level, synonyms are used to replace existing words; at the sentence level, syntax alterations are made; and at the paragraph level, the logical flow of sentences is reorganized. Annotators are asked to apply these three types of edits in equal proportion, ensuring that over 50% of the original content is modified, as described in Table 8. Additionally, a separate annotator reviews 10% of the texts to verify that the edits preserve the original meaning while ensuring that the revised texts remain fluent and comprehensible.

A.3.4 Data Coverage

HART encompasses four domains and five languages as Table 1, which are generated by six LLMs and four decoding parameters. Specifically, the dataset leverages six language models – gpt-3.5-turbo, gpt-4o, claude-3.5-sonnet, gemini-1.5-pro, llama-3.3-70b-instruct, and qwen-2.5-72b-instruct – to generate data, with a random model selected for each sample. As for decoding parameters, a temperature is randomly chosen from the range $[0.8, 1.0, 1.2]$, a top-p from $[0.96, 1.0]$, and both frequency and presence penalties from the range

Detector	English	Chinese	French	Spanish	Arabic	ALL (TPR5%)
			Level-3 Detection Task			
Log-Perplexity	0.7370	0.8599	0.8471	0.8668	0.6290	0.7489 (26%)
Log-Rank	0.7665	**0.8701**	0.8645	**0.8770**	**0.6327**	0.7647 (25%)
LRR	0.8466	0.8625	0.8706	0.8744	0.6117	0.7651 (21%)
Fast-Detect	0.8551	0.8655	0.8662	0.8310	0.5871	0.8118 (48%)
C_2 (Fast-Detect)	0.7084	0.7121	0.7404	0.7163	0.5657	0.6910 (18%)
C_2-T (Fast-Detect)	0.8600	0.8459	0.8538	0.8397	0.5879	0.8065 (48%)
Binoculars	**0.8698**	0.8698	0.8814	0.8474	0.5754	0.7990 (48%)
C_2 (Binoculars)	0.7177	0.7117	0.7633	0.7318	0.5507	0.6882 (19%)
C_2-T (Binoculars)	0.8698	0.8495	0.8587	0.8548	0.5476	0.7924 (49%)
Glimpse	0.8310	0.8868	0.8793	0.8382	0.7950	0.8323 (51%)
C_2 (Glimpse)	0.7422	0.7182	0.7434	0.7382	0.6952	0.6958 (13%)
C_2-T (Glimpse)	0.8257	0.8681	**0.8853**	0.8729	**0.8031**	**0.8481 (53%)**
			Level-2 Detection Task			
Log-Perplexity	0.3909	0.8660	0.6955	0.7963	0.4679	0.6287 (14%)
Log-Rank	0.4130	**0.8769**	0.7066	0.7959	0.4686	0.6350 (12%)
LRR	0.5296	0.8748	0.7118	0.7644	0.4777	0.6380 (11%)
Fast-Detect	0.6665	0.8361	0.7728	0.6961	0.4658	0.7007 (37%)
C_2 (Fast-Detect)	0.7412	0.7778	0.7819	0.7454	**0.6099**	0.7336 (28%)
C_2-T (Fast-Detect)	0.8242	0.8295	0.8344	0.7837	0.5867	**0.7793 (42%)**
Binoculars	0.6770	0.8383	0.7779	0.7115	0.4543	0.6929 (37%)
C_2 (Binoculars)	0.7492	0.7803	0.7968	0.7587	0.5955	0.7265 (28%)
C_2-T (Binoculars)	**0.8310**	0.8234	**0.8464**	0.7955	0.4978	0.7515 (38%)
Glimpse	0.5953	0.8123	0.7511	0.7269	0.6813	0.6921 (33%)
C_2 (Glimpse)	0.6990	0.7569	0.7444	0.7110	0.7059	0.6860 (14%)
C_2-T (Glimpse)	0.7094	0.8258	0.8257	**0.8083**	**0.7969**	0.7776 (41%)
			Level-1 Detection Task			
Log-Perplexity	0.3960	0.8483	0.6629	**0.7890**	0.4876	0.6154 (08%)
Log-Rank	0.4042	**0.8519**	0.6640	0.7824	0.4830	0.6140 (07%)
LRR	0.5009	0.8309	0.6480	0.7288	0.4760	0.6070 (08%)
Glimpse						
Fast-Detect	0.6897	0.8349	0.7510	0.7331	0.4359	0.7032 (30%)
C_2 (Fast-Detect)	0.7021	0.7237	0.6963	0.7105	0.5678	0.6820 (22%)
C_2-T (Fast-Detect)	0.7770	0.7997	**0.7749**	0.7669	0.4798	0.7288 (32%)
Binoculars	0.6969	0.8394	0.7484	0.7461	0.4286	0.7053 (**33%**)
C_2 (Binoculars)	0.6959	0.7234	0.7042	0.7137	0.5521	0.6752 (21%)
C_2-T (Binoculars)	**0.7843**	0.8041	0.7637	0.7657	0.4639	0.7264 (33%)
Glimpse	0.5600	0.7928	0.6933	0.7034	0.6673	0.6596 (24%)
C_2 (Glimpse)	0.5815	0.6481	0.6456	0.6192	0.6052	0.6003 (10%)
C_2-T (Glimpse)	0.6386	0.7904	0.7336	0.7634	**0.7638**	**0.7302** (24%)

Table 7: Results in CC News of HART, covering five languages. The best AUROC and TPR5% are marked in **bold**. The column '*ALL*' denotes a mixture of languages.

[0.0, 1.0] for each sample.

B Baseline Detectors

RoBERTa (ChatGPT) (Guo et al., 2023) refers to a RoBERTa-base model (Liu et al., 2019) that has been fine-tuned on the HC3 (Guo et al., 2023) dataset. This dataset includes responses written by humans and ChatGPT across a variety of fields such as Reddit, medicine, finance, and law. We use this model as a representative baseline for trained detectors. [1]

RADAR (Hu et al., 2023) is trained on Vicuna-7B, employing a generative adversarial framework. In this setup, a paraphraser is optimized to deceive the detector, while the detector itself learns to recognize outputs generated by the paraphraser. [2]

Log-Perplexity and Log-Rank (Gehrmann et al., 2019; Solaiman et al., 2019; Ippolito et al., 2020) are simple yet effective baseline methods. Log-perplexity measures the logarithmic perplexity of a scoring model, while Log-rank computes the average logarithm of token ranks in descending probability order. For this study, we use falcon-7B as the scoring model, which has shown superior

[1] https://huggingface.co/Hello-SimpleAI/chatgpt-detector-roberta

[2] https://huggingface.co/TrustSafeAI/RADAR-Vicuna-7B

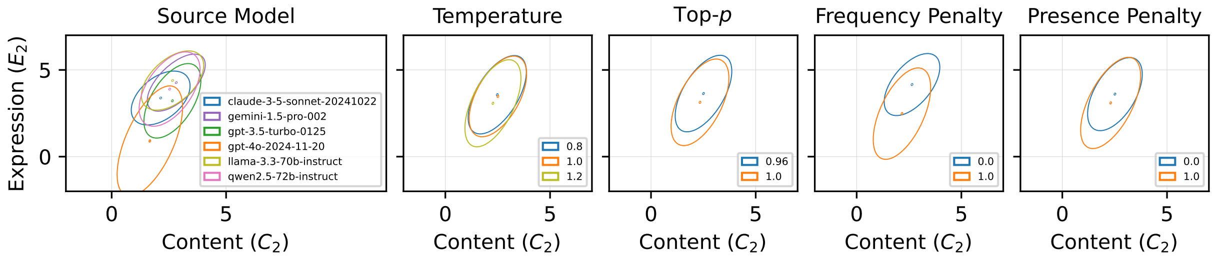

Figure 7: The impact of source model and decoding parameters to generated texts.

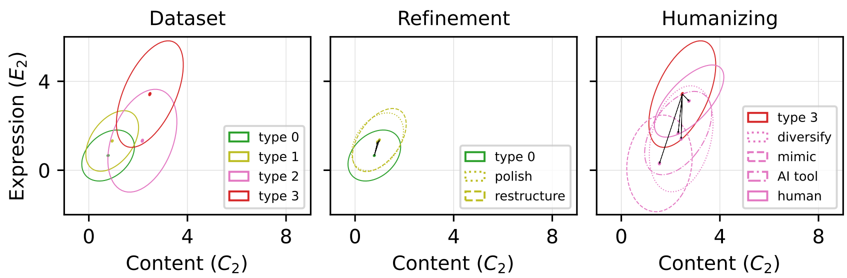

Figure 8: 2D distribution of texts using conditional probability curvature as a metric. The center points represent the means while ellipses the standard deviations.

performance compared to smaller models.

LRR (Su et al., 2023) is a detector based on perplexity, calculated by dividing the perplexity by the log-rank of a scoring model. Similar to others, falcon-7B serves as the scoring model in our experiments.

Fast-Detect (Bao et al., 2024) employs a perplexity-based detection approach. It calculates a metric named conditional probability curvature by subtracting perplexity on a scoring model from the cross-entropy between the scoring model and a sampling model. The original implementation uses gpt-j-6B as the sampling model and gpt-neo-2.7B as the scoring model. However, we observe that using a larger model considerably improves detection. To ensure fair comparisons, we use falcon-7B-instruct as the scoring model and falcon-7B as the sampling model, similar to Binoculars.

Binoculars (Hans et al., 2024) is another perplexity-based detection method, which operates by dividing the perplexity of a scoring model (referred to as the performer) by the cross-entropy between the performer model and an observer model. In our experiments, we adhere to the original setup, where falcon-7B-instruct acts as the performer and falcon-7B as the observer.

Glimpse (Bao et al., 2025) is a variation of Fast-Detect that utilizes the proprietary gpt-3.5-turbo-0301 model. It approximates the full token probability distribution using a partial observation retrieved from the Completion API.

C Results and Analysis

C.1 Results on Multiple Languages

As shown in Table 7, 2D detectors demonstrate clear superiority over baseline models in level-2 and level-1 detection tasks across five languages, highlighting the effectiveness of the 2D framework across all tested languages. Among the existing detectors, Glimpse, powered by gpt-3.5-turbo-0301, outperforms both Fast-Detect and Binoculars, which are based on falcon-7B and falcon-7B-instruct. We hypothesize that this advantage stems from the stronger multilingual capabilities of gpt-3.5-turbo.

C.2 Analysis on Data Distribution

C.2.1 Impact of Parameters

As illustrated in Figure 7, each factor uniquely affects the distribution of generated texts.

Source Model. The source model plays the most pivotal role in shaping the distribution. Among the six models, gpt-4o stands out by generating the most diverse texts across both expression and content dimensions.

Temperature. In general, increasing the temperature leads to greater diversity in the generated texts.

However, the variations in diversity are relatively minor.

Top-p. Similar to temperature, larger values of p result in more diverse outputs, but the overall differences remain limited.

Frequency Penalty. Frequency penalty during decoding significantly influences the distribution of generated texts, with higher penalties tending to produce more human-like outputs.

Presence Penalty. Compared to frequency penalty, presence penalty has a smaller impact. Nonetheless, higher penalties generally result in more human-like text generation.

C.2.2 Impact of Refinement and Humanizing

As illustrated in Figure 8, both refinement and humanizing introduce significant changes to the distribution.

Refinement. The process of refining, which involves polishing and restructuring, shifts the distribution upward, indicating a notable influence along the expression dimension. However, the variations between the two refinement techniques are relatively minimal.

Humanizing. Various humanizing techniques affect the distribution differently. Human editing induces the smallest changes, maintaining a center close to the origin. In contrast, AI tools produce a more pronounced impact, though the shift predominantly occurs along the expression dimension. The "Diversify" technique yields results similar to external AI tools, while "Mimic" causes the most substantial distribution shift.

C.3 Discussion on Alternative Solutions

Due to their straightforward nature, the prompting techniques serve as suitable options for a proof-of-concept. However, there are numerous alternative methods for representing content, including structural representations like abstract meaning representations (Banarescu et al., 2013), semantic role labeling (Palmer et al., 2011), semantic parsing (Poon and Domingos, 2009), knowledge graphs (Hogan et al., 2021), and frame semantics (Fillmore et al., 2006), as well as neural representations (Vigliocco and Vinson, 2007). For instance, neural representations might be a more effective approach since they bypass the need to depend on external large language models (LLMs) for extracting content features. Further exploration of these possibilities is deferred to future work.

Instruction for Manual Editing of AI-Generated Texts
The goal of this task is to manually edit AI-generated essays, paper introductions, creative writings, and news articles to improve their language quality while retaining the original intended meaning. Follow the steps and guidelines below carefully to ensure consistency and quality.

—

General Guidelines
1. **Preserve Original Meaning**:
- Your edits must not alter the intended meaning or factual content of the original text. Focus solely on improving language clarity, expression, and flow.
2. **Balance of Edits**: Ensure that your edits improve the language across three levels: word, sentence, and paragraph. Distribute your edits so that:
- Word-level modifications account for about 1/3 of your changes.
- Sentence-level modifications account for about 1/3 of your changes.
- Paragraph-level modifications account for about 1/3 of your changes.
3. **Volume of Edits**:
- The cumulative changes you make should amount to editing more than half of the total word count of the text. Be thorough in your revisions.

—

Types of Edits
1. Word-Level Editing
- Replace repetitive or vague words with more precise synonyms.
- Improve word choice to match the tone and style of the piece (e.g., academic, formal, journalistic, creative).
- Correct incorrect usage of words, awkward phrasing, or redundant expressions.
Example:
Original: "The results were really very significant."
Edited: "The results were highly significant."
2. Sentence-Level Editing
- Adjust sentence structures to enhance readability and fluency. This includes:
- Breaking down long, convoluted sentences into shorter, clear ones.
- Combining choppy or fragmented sentences for better flow.
- Reorganizing sentence components for coherence and logic.
- Fix issues with grammar, punctuation, and syntax where necessary.
- Ensure variety in sentence structure to avoid monotony.
Example:
Original: "The team successfully completed the project, which was a very crucial step in their plan, and they presented it to stakeholders two days later."
Edited: "The team successfully completed this crucial step in their plan and presented the project to stakeholders two days later."
3. Paragraph-Level Editing
- Rearrange sentences within the paragraph to improve logical progression and argument clarity.
- Merge or split paragraphs when necessary for better organization or flow.
- Add transitional phrases if needed to improve coherence between sentences and paragraphs.
- Ensure that the paragraph aligns with the overall tone and intent of the text.
Example:
Original: “Climate change is a growing concern worldwide. The effects of climate change include rising temperatures, melting polar ice, and severe weather events. Many governments are implementing policies to mitigate these effects. Public awareness around climate change has also been increasing over recent years. Organizations are focusing on educating individuals and communities about sustainable practices.”
Edited: “Climate change is an increasingly urgent issue with global implications. Its effects, such as rising temperatures, melting polar ice, and severe weather events, are becoming more evident. In response, many governments are enacting policies to address these challenges. At the same time, public awareness has grown significantly, driven by organizations that educate communities about sustainable practices.”

—

Step-by-Step Workflow
1. **Understand the Text**:
- Read the entire text carefully to grasp its main ideas, tone, and intent before making any changes.
2. **Edit with Balance and Intent**:
- As you edit, keep track of the types of changes you are making (word-level, sentence-level, paragraph-level) and ensure an even distribution across the three levels.
- Avoid over-editing in one specific area (e.g., only doing word-level tweaks).
3. **Meet the Edit Requirement**:
- Ensure that more than 50% of the text has been edited after revisions. Track your changes to confirm this.
4. **Review and Finalize**:
- Re-read your edited version to confirm it retains the original meaning and intention.
- Check that the language is smooth, natural, and appropriate for the target audience and genre.

Table 8: Instruction for human editing.