Your Language Model Can Secretly Write Like Humans: Contrastive Paraphrase Attacks on LLM-Generated Text Detectors

Hao Fang[*1], **Jiawei Kong**[*1,2], **Tianqu Zhuang**[1], **Yixiang Qiu**[1], **Kuofeng Gao**[1], **Bin Chen**[†2,3], **Shu-Tao Xia**[1,3], **Yaowei Wang**[2,3], **Min Zhang**[2],
[1]Tsinghua Shenzhen International Graduate School, Tsinghua University,
[2]Harbin Institute of Technology, Shenzhen, [3]Pengcheng Laboratory
{fangh25, kjw25, zhuangtq23, qiu-yx24, gkf21}@mail.tsinghua.edu.cn, chenbin2021@hit.edu.cn,
xiast@sz.tsinghua.edu.cn, wangyw@pcl.ac.cn, zhangmin2021@hitsz.edu.cn

Abstract

The misuse of large language models (LLMs), such as academic plagiarism, has driven the development of detectors to identify LLM-generated texts. To bypass these detectors, paraphrase attacks have emerged to purposely rewrite these texts to evade detection. Despite the success, existing methods require substantial data and computational budgets to train a specialized paraphraser, and their attack efficacy greatly reduces when faced with advanced detection algorithms. To address this, we propose **Co**ntrastive **P**araphrase **A**ttack (CoPA), a training-free method that effectively deceives text detectors using off-the-shelf LLMs. The first step is to carefully craft instructions that encourage LLMs to produce more human-like texts. Nonetheless, we observe that the inherent statistical biases of LLMs can still result in some generated texts carrying certain machine-like attributes that can be captured by detectors. To overcome this, CoPA constructs an auxiliary machine-like word distribution as a contrast to the human-like distribution generated by the LLM. By subtracting the machine-like patterns from the human-like distribution during the decoding process, CoPA is able to produce sentences that are less discernible by text detectors. Our theoretical analysis suggests the superiority of the proposed attack. Extensive experiments validate the effectiveness of CoPA in fooling text detectors across various scenarios. The code is available at: https://github.com/ffhibnese/CoPA_Contrastive_Paraphrase_Attacks

1 Introduction

Large language models (LLMs), such as GPT-4 and Claude-3.5, have demonstrated remarkable abilities in text comprehension and coherent text generation. These capabilities have driven their widespread applications, including code generation (Jiang et al., 2024) and academic research (Stokel-Walker, 2022). However, the misuse of LLMs for harmful purposes, such as academic plagiarism and misinformation generation, has raised significant societal concerns regarding safety and ethics (Bao et al., 2024; Wu et al., 2025; Kong et al., 2025). In response, various detection methods that leverage the unique characteristics of LLM-generated texts from multiple perspectives have been proposed to mitigate the associated risks.

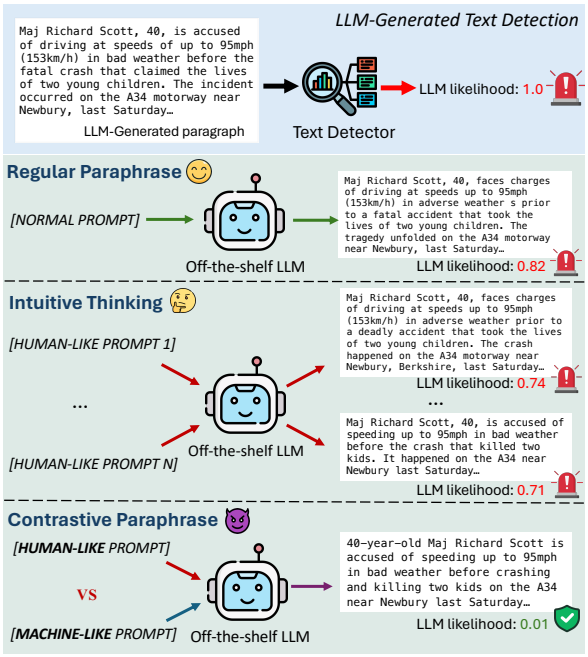

Figure 1: Comparison of different paraphrasing strategies. The human-like and machine-like prompts are crafted to guide the LLM in generating human-style and machine-style texts, respectively.

Concurrently, red-teaming countermeasures (Krishna et al., 2023; Shi et al., 2024) have also been introduced to evaluate the reliability of these detection algorithms, which can be broadly categorized into *word-substitution* and *paraphrase* attacks. Specifically, word-substitution attacks (Shi et al., 2024; Wang et al., 2024) replace specific important words in the fake sentence using candi-

[*]Equal Contribution
[†]Corresponding Author

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 8585–8602
November 4-9, 2025

date words generated by a language model. However, they require an additional surrogate model to identify the word importance, and the replacement operation can significantly increase sentence perplexity (Wang et al., 2024), making the processed texts easily identified by humans. In contrast, Dipper (Krishna et al., 2023) proposes a paraphrase-based attack that rewrites the whole paragraph by modifying the syntax and phrasing to deceive text detectors. This approach does not rely on surrogate models and can preserve sentence perplexity, presenting a more practical and versatile attack strategy. Nonetheless, Dipper requires training a large-scale generative language model as the paraphraser, incurring substantial data collection and computational burdens. Moreover, the attack performance is greatly diminished when confronted with more advanced defense strategies such as Fast-DetectGPT (Bao et al., 2024), as shown Sec. 4.2.

In this paper, we build upon the research line of paraphrase attacks and propose a training-free paraphrase approach named Contrastive Paraphrase Attack (CoPA), which aims to elicit human-written word distributions from an off-the-shelf LLM to evade detection. Specifically, we revisit the fundamental mechanisms underlying existing detection algorithms and hypothesize that the core principle of effective paraphrase attacks lies in erasing the machine-inherent characteristics within the paragraph while incorporating more human-style features, such as more flexible choices of words and phrases. Based on this insight, we intuitively seek to craft prompts that alleviate the established statistical constraints of pre-trained LLMs and produce more human-like word distributions, as shown in Fig. 1. While this strategy exhibits some effectiveness, LLMs pretrained on massive corpora hold a strong tendency to prioritize words with high statistical probability to ensure sentence coherence (Bao et al., 2024; Mao et al., 2024). This inherent bias consistently influences the word choices of generated sentences, irrespective of the input prompts. Therefore, some paraphrased sentences still exhibit certain machine-related characteristics, rendering them highly discernible by detection classifiers.

To address this limitation, we conduct a reverse-thinking analysis. While it is challenging to directly produce highly human-written word distributions that can fully bypass detection, eliciting the opposite machine-like word distributions that contain rich machine-related attributes is considerably easier. These machine-like word probabilities can then serve as negative instances to further purify the obtained human-style distribution for more human-like text generation. In light of this consideration, an auxiliary machine-like distribution is constructed as a contrastive reference, which is used to filter out the machine-related concepts from the aforementioned human-like distribution. By sampling from the meticulously adjusted word distribution, CoPA generates more diverse and human-like sentences, which exhibit remarkable effectiveness in deceiving LLM-text detectors.

Contributions. We propose CoPA, a novel paraphrase attack that contrastively modifies the word distribution from an off-the-shelf LLM to rewrite generated texts for enhanced attacks against text detectors. CoPA eliminates the cumbersome burdens of training a dedicated paraphraser, achieving an efficient and effective attack paradigm. Furthermore, we develop a theoretical framework that substantiates the superiority of CoPA.

To validate the effectiveness, we conduct extensive experiments on 3 long-text datasets with various styles against 8 powerful detection algorithms. Compared to baselines, CoPA consistently enhances the attack while maintaining semantic similarity, *e.g.*, an average improvement of 57.72% in fooling rates (at FPR=5%) for texts generated by GPT-3.5-turbo when against Fast-DetectGPT.

2 Related Work

2.1 LLM-generated Text Detection

Existing detection algorithms for AI-generated text can generally be categorized into two types: (i) Training-based detection: These methods typically involve training a binary classification language model. Specifically, OpenAI employs a RoBERTa model (Liu, 2019) trained on a collection of millions of texts for detection. To enhance the detection robustness, RADAR (Hu et al., 2023) draws inspiration from GANs (Goodfellow et al., 2020) and incorporates adversarial training between a paraphraser and a detector. Additionally, DeTective (Guo et al., 2024a) proposes a contrastive learning framework to train the encoder to distinguish various writing styles of texts, and combined with a pre-encoding embedding database for classification. R-detect (Song et al., 2025) employs a kernel relative test to judge a text by determining whether its distribution is closer to that of human texts. Despite the efforts in specific domains, training-based methods are struggling with generalization to un-

seen language domains, which reduces their practicality and versatility. (ii) Zero-shot detection: These methods are training-free and typically focus on extracting inherent features of LLMs' texts to make decisions. GLTR (Gehrmann et al., 2019) and LogRank(Solaiman et al., 2019) leverage the probability or rank of the next token for detection. Since AI-generated text typically exhibits a higher probability than its perturbed version, DetectGPT (Mitchell et al., 2023) proposes the probability curvature to distinguish LLM and human-written text. Building on this, Fast-DetectGPT (Bao et al., 2024) greatly improves the efficiency by introducing conditional probability curvature that substitutes the perturbation step with a more efficient sampling step. TOCSIN (Ma and Wang, 2024) presents a plug-and-play module that incorporates random token deletion and semantic difference measurement to bolster zero-shot detection capabilities. Other methods explore different characteristics, such as likelihood (Hashimoto et al., 2019), N-gram divergence (Yang et al., 2024), and the editing distance from the paraphrased version (Mao et al., 2024).

2.2 Attacks against Text Detectors

Red-teaming countermeasures have been proposed to stress-test the reliability of detection systems. Early attempts evade detection using in-context learning (Lu et al., 2023) or directly fine-tuning the LLM (Nicks et al., 2023) under a surrogate detector. Recent advances can be broadly categorized into:

Substitution-based attacks. Shi et al. (2024) introduce the substitution-based approach, which minimizes the detection score provided by a surrogate detector by replacing certain words in AI sentences with several synonyms generated by an auxiliary LLM. Subsequently, RAFT (Wang et al., 2024) improves the attack performance by introducing an LLM-based scoring model to greedily identify critical words in the machine sentences. However, this type of attack relies on an additional surrogate model, and the generated sentences suffer from reduced coherence and fluency, which are easily identifiable by human observations and limit their practical utility in real-world scenarios.

Paraphrasing-based attacks. Conversely, Dipper (Krishna et al., 2023) suggests a surrogate-free approach that can perfectly maintain text perplexity. With a rewritten dataset of paragraphs with altered word and sentence orders, Dipper fine-tunes a T5-XXL (Raffel et al., 2020) as a paraphraser to rewrite entire machine paragraphs, which effectively fools text detectors while preserving semantic consistency. Based on Dipper, Sadasivan et al. (2023) introduce a recursive strategy that performs multiple iterations of paraphrasing, with slightly degrading text quality while significantly enhancing attack performance. Raidar (Mao et al., 2024) proposes a straightforward approach that directly queries an LLM to paraphrase machine text into paragraphs with more human-style characteristics. Shi et al. (2024) suggest a strategy that automatically searches prompts to induce more human-like LLM generations. However, it relies on a strong assumption that a surrogate detector is available.

This paper follows the more practical and applicable paradigm of surrogate-free paraphrasing attacks and proposes a contrastive paraphrasing strategy, which achieves remarkable efficacy in bypassing LLM-text detection systems.

3 Method

In this section, we first present the paradigm of paraphrase attacks. Then, we elaborate on the proposed CoPA that rewrites generated texts using a pre-trained LLM. Finally, we provide a theoretical framework to guarantee the attack effectiveness.

3.1 Problem Formulation

We denote the AI-generated text detector weighted by w as $D_w : \mathcal{Y} \to [0, 1]$, where $\mathcal{Y}$ denotes the text domain. The detector D_w maps text sequences $y \in \mathcal{Y}$ to corresponding LLM likelihood scores, where higher values indicate a greater probability of being LLM-generated. For an LLM pre-trained on extensive corpora, the resulting texts exhibit significant writing styles, including word preferences and coherent syntactic structures, which have been leveraged in previous studies to develop various text detectors. To evade detection, a malicious paraphrase attacker aims to reduce these machine-related concepts within the machine-generated texts to obtain paraphrased variants that can effectively mislead D_w into outputting lower LLM likelihood scores. Before delving into the proposed method, we first review the token generation paradigm of the LLM inference. We consider utilizing an off-the-shelf LLM as the paraphraser. Given an input instruction x, a machine text y_m and a pre-trained LLM $f_\theta(\cdot)$ parameterized by θ, f_θ generates the paraphrased paragraph y by producing tokens in an autoregressive manner. At each timestep t, $f_\theta(\cdot)$ samples the next token y_t from the

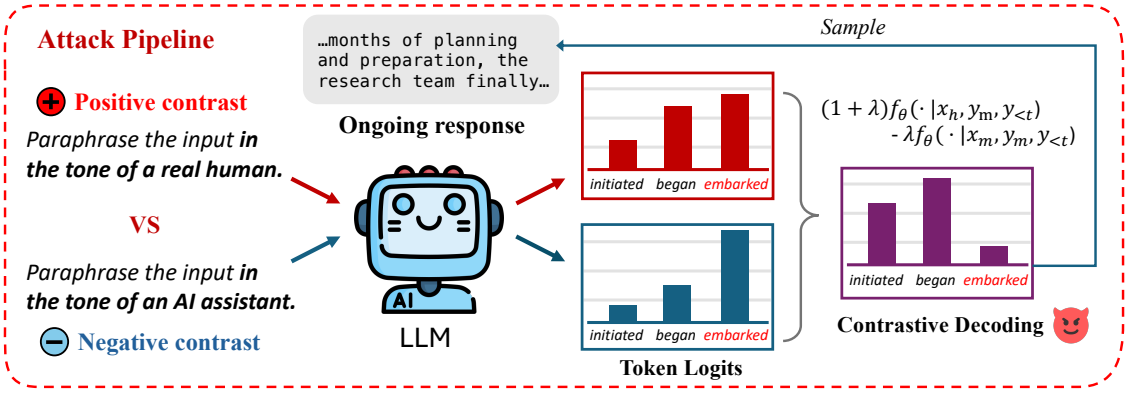

Figure 2: Overview of the proposed CoPA. The contrastive paraphrasing successfully penalizes the LLM-preferred word '*embarked*' (Liang et al., 2024) and encourages more flexible word choices for next-token sampling.

conditional probability distribution:

$$y_t \sim p_\theta(\cdot|x, y_m, y_{<t}) \propto \exp f_\theta(\cdot|x, y_m, y_{<t}), \tag{1}$$

where $y_{<t}$ denotes the previously generated sequence. Particularly, the probability of a generated text y with length l can be expanded as the multiplication of conditional probabilities:

$$q_\theta(y) = \prod_{t=1}^{l} p_\theta(y_t|x, y_m, y_{<t}), \tag{2}$$

where $q_\theta(\cdot)$ denotes the text probability distribution. Based on the chain rule in Eq. (2), the sentence-level paraphrasing problem can be further reformulated as a token-level selection task, *i.e.*, design algorithms to adequately penalize the probability of machine-favored tokens and inspire more human-like word choices for generating sentences able to confuse the text detector $D_w(\cdot)$. In addition to outstanding attack performance, the revised sentences should preserve the original semantics and exhibit a high degree of coherence to ensure text quality.

3.2 Contrastive Paraphrase Attacks

Building upon the preceding analysis, an intuitive approach is to devise a prompt x_h that can elicit more diverse token distributions p'_h from LLM $f_\theta(\cdot)$ to simulate authentic human-written distribution p_h. While this strategy achieves some success, we find that the inherent statistical priors of language models persistently impose constraints on the output distributions. As a result, this leads to unstable outcomes, with some revised sentences still exhibiting sufficient machine-related patterns and remaining highly detectable (see Appendix D).

To achieve more effective and stable attacks, we carefully examine this issue and identify the following two critical considerations. (1) The current strategy essentially operates within the input space (*i.e.*, modify input prompts) to indirectly influence the output token distribution, which remains inevitably constrained by the prior knowledge encoded in the LLM. Instead, directly manipulating the output distribution is a potentially more promising alternative. (2) Generating word distributions that can fully deceive detection models is challenging; however, it is much easier to generate word distributions that are highly detectable. From a reverse-thinking perspective, those LLM-favored tokens are also considerably valuable since they encapsulate rich machine-style features and can serve as negative examples for contrastive references. Based on these insights, we propose our Contrastive Paraphrase Attack, a novel approach that directly employs a dynamic adjustment to the output word distributions during LLM decoding. Figure 2 illustrates the core pipeline of CoPA. Apart from the prompt x_h for human-style distributions p'_h, we also construct a comparative machine prompt x_m to elicit machine-preferred word choices p_m. By contrasting human-like and machine-like token distributions, CoPA refines the probabilities in the decoding process and encourages generations of texts with enhanced human-written resemblance. Formally, the contrastively purified token distribution at timestep t can be expressed as:

$$p_c(\cdot|x_h, x_m, y_m, y_{<t}) \propto \exp\Big((1+\lambda) f_\theta(\cdot|x_h, y_m, y_{<t}) - \lambda f_\theta(\cdot|x_m, y_m, y_{<t})\Big), \tag{3}$$

where p_c represents the contrastive token distributions and λ is the scaling parameter that controls the

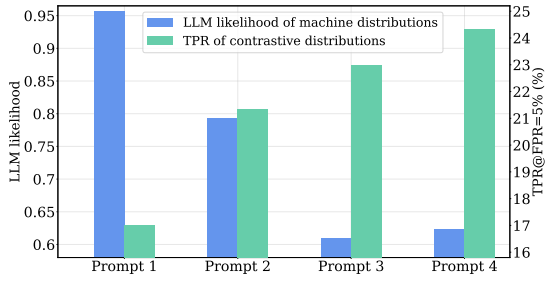

Figure 3: Fast-DetectGPT detected LLM likelihood of texts from different machine distributions p_m induced by various machine prompts x_m. We also present the detection TPR of their corresponding contrastive distributions p_c. See Appendix E for details of used prompts.

degree of amplification on the discrepancy between two distributions. The proposed framework operates as a self-corrective decoding mechanism that is specifically designed to trick AI-text detectors. By dynamically identifying and penalizing machine-preferred token preferences, CoPA effectively reduces entrenched linguistic biases and enables the generation of sentences with more expressiveness and lexical diversity, achieving an enhanced capability to fool LLM-generated text detectors.

Reduce machine styles by amplifying them. Previous studies have shown that rewriting machine texts using another LLM can reduce some machine-specific features, although not sufficiently for launching effective attacks (Sadasivan et al., 2023). This is because the rewritten texts inherit a mixture of stylistic and lexical preferences from different LLMs, thus increasing the text diversity. However, in our contrastive design, the machine-style distribution p_m actually serves as a negative reference to be subtracted from the human distribution. Employing a regular paraphrasing prompt to obtain p_m may inadvertently dilute its machine-specific characteristics and weaken the effectiveness of the contrastive operation.

A reasonable solution involves constructing p_m as a highly salient and concentrated machine-style token distribution, wherein high-probability tokens are strongly machine-related and more likely to trigger detection. We achieve this by identifying a prompt x_m that can amplify the LLM likelihood of generated sentences. Fig. 3 empirically validates our strategy, *i.e.*, magnifying machine-style features in p_m can, in turn, promote a refined distribution that more closely resembles authentic human writing, further enhancing the attack effectiveness.

Adaptive Truncation for plausibility. Another important issue is that CoPA utilizes the whole token distribution to measure the difference. However, there may be occasions where certain high-probability tokens overlap between p'_h and p_m. The subtraction operation may penalize the probabilities of reasonable and valid tokens while rewarding casual and unrelated ones (Fang et al., 2025), thus compromising the coherence and semantic consistency of generated sentences. To address this, we incorporate a token constraint mechanism (Li et al., 2023), which applies an adaptive pruning to the output tokens:

$$y_t \sim p_c(\cdot|x_h, x_m, y_m, y_{<t}), \text{ s.t. } y_t \in \mathcal{V}_{top}(y_{<t}),$$
$$\mathcal{V}_{top}(y_{<t}) = \Big\{ y_t \in \mathcal{V} : p'_h(y_t|x_h, y_m, y_{<t}) \geq \alpha \max_v p'_h(v|x_h, y_m, y_{<t}) \Big\}, \tag{4}$$

where $\mathcal{V}$ denotes the vocabulary set of f_θ and α is the hyperparameter to adjust clipping. By introducing this adaptive pruning mechanism, CoPA leverages the confidence scores of the human-like distribution to refine the contrastive distribution, which restricts the decision-making to a more reliable token candidate pool and suppresses the selection of unsuitable tokens.

3.3 Theoretical Analysis

Apart from empirical analysis, we build a theoretical framework to confirm the superiority of CoPA in simulating authentic human writing. As previously stated, p_h denotes the real human-chosen word distribution, p'_h and p_m represent human-like and machine-like token distributions elicited from the LLM using prompts x_h and x_m, respectively. The objective is to prove that the distribution p_c, derived by contrasting p'_h and p_m, aligns more closely with the human preferences p_h.

To mathematically measure the difference between distributions p_h and p_c, we first introduce an auxiliary function based on the KL divergence.

Definition 1 (Auxiliary Distance Function)**.** *Let* $\mathbb{KL}$ *denotes the KL divergence, the distributional distance between* p_h *and* p_c *is a unary function of* λ*, which is characterized as*

$$g(\lambda) \coloneqq \mathbb{KL}(p_h || (1+\lambda)p'_h - \lambda p_m). \tag{5}$$

$g(\lambda)$ inherit several good properties from the KL divergence, based on which we derive the critical Theorem that guarantees the effectiveness of CoPA:

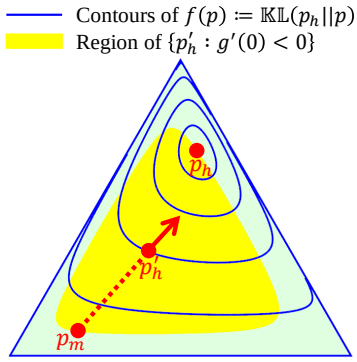

Figure 4: Illustration of the premise of Theorem 1. Let $|\mathcal{V}| = 3$ and thus $\mathbb{P}^{\mathcal{V}}$ is a triangle. We draw the contours of $f(p) \coloneqq \mathbb{KL}(p_h||p)$. The closer to p_h, the lower the KL divergence with p_h. If $p'_h - p_m$ points to the inside of the contour at p'_h, then $f(p)$ decreases at p'_h along $p'_h - p_m$, *i.e.*, $g(\lambda)$ decreases at $\lambda = 0$. In this case $g'(0) < 0$ is satisfied and Theorem 1 is applicable. In practice p'_h is usually between p_m and p_h, so $g'(0) < 0$ is usually satisfied and Theorem 1 is generally applicable.

Proposition 1. *$g(\lambda)$ is a convex function. If $g(\lambda)$ is not constant, it has a unique minimum point λ_*.*

Theorem 1. *If $g'(0) < 0$, then $\lambda_* > 0$ and for any $\lambda \in (0, \lambda_*]$, we have*

$$\mathbb{KL}(p_h||(1+\lambda)p'_h - \lambda p_m) < \mathbb{KL}(p_h||p'_h). \quad (6)$$

The detailed proofs of Proposition 1 and Theorem 1 are provided in Appendix A. Theorem 1 reveals that by adequately selecting λ, CoPA drives the resultant distribution p_c closer to the authentic human distribution p_h than the human-like distribution p'_h, which is directly elicited from the LLM using x_h. This theoretically validates the necessity and effectiveness of our contrastive strategy.

Note that the premise of Theorem 1 is $g'(0) < 0$. We use $|\mathcal{V}| = 3$ as an example to illustrate its rationality. As in Figure 4, we calculate the area wherein probability distributions satisfy $g(0)' < 0$. In essence, p'_h is generated by the LLM and hence is constrained by the inherent language priors, which prevent it from deviating significantly from the machine-featured distribution p_m. Meanwhile, we use a carefully crafted human-like prompt to guide p'_h to move from p_m towards p_h. As a result, p'_h typically falls within the region that satisfies $g'(0) < 0$. Therefore, this premise generally holds and thus Theorem 1 is applicable in practice, which is further confirmed by experimental results in Sec. 4.2. Based on LLM's prediction paradigm, we contrast the output logits in practice, achieving excellent performance in misleading detection models. We also note that researchers should examine the validity of this assumption in their specific setting before applying our theoretical framework.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate on three widely adopted datasets spanning various linguistic styles and content, including (1) XSum for news articles (Narayan et al., 2018), (2) SQuAD for Wikipedia contexts (Rajpurkar, 2016), and (3) LongQA for long-form question answering, where LLM answers a how/why question within 250-350 words (Fan et al., 2019). We follow (Bao et al., 2024) and randomly select 150 samples for evaluation.

Baselines. We compare our method with the state-of-the-art (SOTA) surrogate-free paraphrase attack Dipper (Krishna et al., 2023). We adopt the setup of 60 lexical diversity and 60 order diversity for Dipper to achieve its best performance. Additionally, we reproduce the attack introduced in (Mao et al., 2024), which leverages an LLM with the query *"Help me rephrase it in human style."* to rewrite machine texts (denoted as Raidar-A). For fairness, we use the same LLM for both our paraphraser and baselines. Note that we also reveal the superiority of CoPA over (Shi et al., 2024) that relies on an extra surrogate model in Section C.

For detection algorithms, we consider diverse methods including training-free LogRank (Solaiman et al., 2019), DetectGPT (Mitchell et al., 2023), DNA-GPT (Yang et al., 2024), Fast-DetectGPT (Bao et al., 2024), Raidar (Mao et al., 2024), TOCSIN (Ma and Wang, 2024), and training-based RoBERTa (Liu, 2019) provided by OpenAI and the R-detect (Song et al., 2025).

Metrics. We analyze the performance using two key metrics. (1) Detection accuracy. In real-world applications, it is crucial to guarantee that human-written text should almost never be misclassified as machine-generated (Krishna et al., 2023), *i.e.*, satisfying a very low false positive rate (FPR). Hence, we follow Dipper and report the true positive rate (TPR) at a fixed FPR. Specifically, we set a relatively high FPR of 5% to significantly reveal the performance improvements. Please refer to Appendix C for results with FPR=1%. (2) Semantic similarity. The rewritten sentences should preserve the original semantics. Similar to Dipper, we employ the P-SP (Wieting et al., 2022) model, a specialized embedding model trained on a filtered

Table 1: Comparison of different paraphrasing attacks against 8 text-detection algorithms (at 5% FPR) using GPT-3.5-turbo generated texts from three different datasets. The best performances are bolded.

Dataset	Attack	Sim	Defense								Avg.
			LogRank	DetectGPT	DNA-GPT	Fast-DetectGPT	Raidar	TOCSIN	RoBERTa	R-Detect	
XSum	No Attack	-	63.33	26.67	80.00	95.33	28.75	98.00	66.67	69.67	66.05
	Dipper	86.67	15.67	**2.67**	27.33	76.33	13.75	74.67	86.67	48.00	43.14
	Raidar	100.00	49.00	12.00	22.64	84.67	16.25	90.67	55.33	68.67	49.90
	Ours	94.00	**4.67**	4.00	**21.33**	**17.00**	**0.00**	**26.67**	**22.67**	**4.67**	**12.63**
SQuAD	No Attack	-	67.00	12.67	41.33	93.50	22.00	92.67	41.33	79.67	56.27
	Dipper	75.33	23.67	2.67	10.00	77.67	5.00	75.33	67.33	53.33	39.38
	Raidar	95.33	58.00	12.33	19.33	81.67	26.00	84.00	32.67	73.50	48.44
	Ours	88.67	**8.33**	**2.67**	**8.67**	**27.50**	**5.00**	**25.33**	**7.33**	**3.67**	**11.06**
LongQA	No Attack	-	73.83	33.33	10.67	86.00	36.25	88.67	38.67	89.33	57.09
	Dipper	94.67	28.83	6.00	0.67	75.00	5.00	67.33	65.33	74.00	40.27
	Raidar	100.00	59.33	22.67	1.33	72.67	33.75	77.33	28.00	79.50	46.82
	Ours	95.33	**14.50**	**5.00**	**0.00**	**11.33**	**0.00**	**16.00**	**6.67**	**6.00**	**7.44**

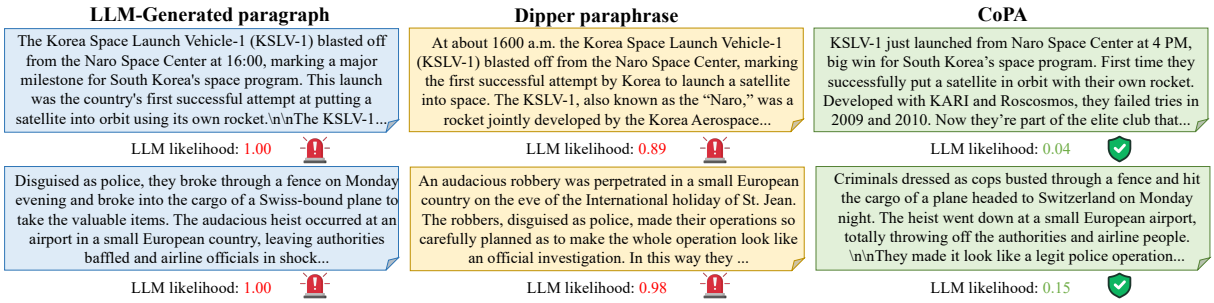

Figure 5: Paraphrased sentences by Dipper and our CoPA. We use Fast-DetectGPT to provide the LLM likelihood.

paraphrase corpus (Wieting and Gimpel, 2018), to measure the semantic discrepancy. We align with Dipper and consider the semantics being preserved if the P-SP score exceeds the average real-human paraphrase score of 0.76. Moreover, we provide *more analysis including text perplexity, GPT-4 assisted and human evaluation in Section H.*

Implementation Details. For attack hyperparameters, we set the contrast intensity $\lambda = 0.5$ and the clipping factor $\alpha = 1e^{-5}$. Unless stated otherwise, we employ a single paraphrasing iteration. We employ Qwen2.5-72B-Instruct (Qwen, 2024) as the paraphraser. Due to page limits, we provide results paraphrased by more LLMs in Section C. More details about the human and machine-like prompts are in Appendix B.

4.2 Performance Evaluation

We test machine texts generated by GPT-3.5-turbo and present results on three datasets in Table 1.

Attack Effectiveness. By conducting a self-introspective correction on token distributions, CoPA remarkably enhances the attack over baseline attacks, *e.g.*, an average improvement of 30.55% in fooling text detectors across three datasets. Although Dipper demonstrates satisfactory performance against several detectors, it becomes significantly less effective when facing more advanced algorithms such as FastDetectGPT. In contrast, our method consistently exhibits impressive attack efficacy across various text detectors. Notably, while Raidar-A and our method employ the same LLM as the paraphraser, CoPA greatly outperforms Raidar-A, validating the effectiveness of our designed prompt and contrastive paraphrasing mechanism.

As for the text quality of rewritten sentences, we demonstrate that CoPA achieves an average semantic similarity score exceeding 90% across various datasets, confirming that our method effectively preserves semantic fidelity during rewriting. While Raidar-A exhibits greater text similarity, its attack effectiveness remains considerably limited. As a comparison, CoPA achieves both excellent attack effectiveness and semantic consistency.

Visualization of Rewritten Texts. We provide examples of rewritten texts before and after the paraphrasing in Figure 5. As expected, the rewritten sentences maintain semantic consistency while exhibiting richer and diverse human-like expressions. This underpins our success in fooling text detectors and presents a practical attack method.

Table 2: Attack Performance (at 5% FPR) of texts generated by more source LLMs based on XSum dataset.

Model	Attack	Sim	Defense								Avg.
			LogRank	DetectGPT	DNA-GPT	Fast-DetectGPT	Raidar	TOCSIN	RoBERTa	R-Detect	
GPT-4	No Attack	-	30.00	6.00	35.33	51.67	24.17	73.33	32.67	46.00	37.40
	Dipper	91.33	8.67	0.67	30.67	64.33	20.83	64.67	78.00	37.67	38.19
	Raidar	100.00	35.00	9.50	34.67	68.33	20.83	82.00	47.33	60.50	44.77
	Ours	94.67	**2.00**	**0.67**	**18.67**	**15.33**	**10.83**	**20.00**	**20.67**	**6.83**	**11.88**
Claude 3.5	No Attack	-	42.67	21.67	24.67	50.00	36.67	70.00	19.33	30.67	36.96
	Dipper	82.00	18.17	**0.33**	20.67	38.67	0.00	36.67	77.33	40.33	29.02
	Raidar	100.00	46.83	18.77	15.33	41.33	41.64	66.00	20.67	38.00	36.07
	Ours	98.00	**1.33**	0.67	**6.67**	**4.00**	**0.00**	**8.00**	**7.33**	**1.33**	**3.67**

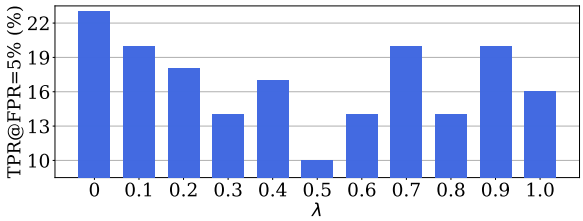

Figure 6: Comparison of detection accuracy on the first 50 samples from XSum under different values of λ against Fast-DetectGPT (Bao et al., 2024).

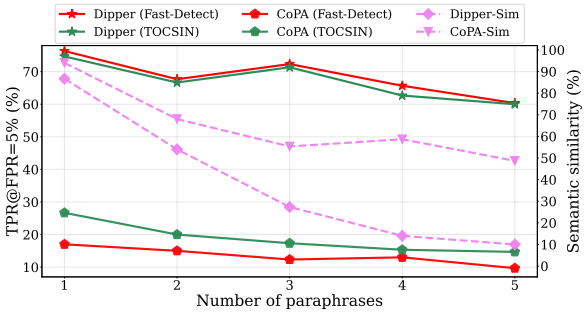

Figure 7: Performance under different numbers of paraphrases against Fast-DetectGPT (Bao et al., 2024) and TOCSIN (Ma and Wang, 2024) on samples generated by GPT-3.5-turbo from the XSum dataset. The dashed lines describe semantic similarity.

4.3 Attacks on More Source LLMs.

In addition to GPT-3.5-turbo, we also consider the machine texts generated by recently prevalent LLMs, including GPT-4 (Achiam et al., 2023) and Claude-3.5 (Anthropic, 2024). Besides, we provide results on GPT-4o (Achiam et al., 2023) and Gemini-1.5 Pro (Team et al., 2024) in Appendix C.

Table 2 demonstrates that our method continues to achieve excellent attack efficacy and semantic similarity across various LLM-generated texts, greatly outperforming the SOTA method Dipper. For detection of GPT-4 texts under Fast-DetectGPT, Dipper even increases the likelihood to be classified as machine-generated than the *No Attack* baseline. In contrast, CoPA stably achieves outstanding fooling rates across various source models, confirming the robustness of the proposed contrastive paraphrase. Another observation is that the detection performance on clean texts generated by more recent models is significantly reduced. The decline may stem from the ability of advanced LLMs to generate varying sentences that are more challenging to detect, underscoring the urgent need for more reliable detection systems.

4.4 Ablation Study

We then investigate the effect of different factors. More ablation studies are in Appendix C.

Impact of contrastive coefficient λ. During decoding, the hyperparameter λ serves as a critical regulation factor for the contrast strength. As shown in Figure 6, positive values of λ consistently enhance the fooling rates relative to $\lambda = 0$, again confirming the effectiveness of our contrastive paraphrasing mechanism. Note that CoPA attains optimal performance at $\lambda = 0.5$, which is then chosen as the default setting for our experiments. Notably, the general trend of TPR over the λ roughly aligns with our preceding theoretical analysis, verifying the rationality of our established theory.

Impact of Multiple Paraphrases. Each LLM-generated text is rewritten only once in our former experiments. We further analyze the influence of multiple rewrites on the results. As shown in Figure 7, increasing the number of rewrites generally strengthens attack effectiveness. However, the performance gain is limited against two advanced defenses, and the semantic similarity of Dipper-rewritten texts sharply drops as the iterations increase. These factors reduce the utility of using multiple paraphrases to improve the attack (Sadasivan et al., 2023). Also, this again highlights the superiority of our method, which achieves outstanding fooling rates via only a single paraphrasing.

5 Conclusion

This paper proposes CoPA, a simple yet highly effective paraphrasing attack against AI-generated text detectors. CoPA constructs a machine-style token distribution as a negative contrast for reducing linguistic biases of LLMs and facilitating the generation of richer and more diverse sentences. Through both theoretical analysis and experimental validation, we fully demonstrate the superiority of the proposed method across various scenarios. We envision CoPA as a powerful tool for auditing the robustness of detection systems, inspiring future development of more robust detection algorithms.

Limitations

While our method avoids the overhead of training a dedicated paraphraser by leveraging an off-the-shelf LLM, the contrastive paraphrasing mechanism requires two forward passes to construct the contrastive token distribution, bringing additional latency during next-token prediction. This may limit the practicality of the proposed attack in real-time applications. Moreover, using off-the-shelf LLMs to paraphrase texts can lead to the semantics of output texts deviating from the original sentence, which may require multiple generations to maintain the semantic similarity. Besides, although human evaluation results in Appendix H indicate that CoPA-paraphrased texts are preferred over those from Dipper, we did not systematically account for the detailed linguistic backgrounds of evaluators and may introduce bias. A more comprehensive human study is needed to validate the general quality of the output sentences. Finally, this work follows prior studies and focuses exclusively on English text. Extending the contrastive paraphrasing framework to other languages, such as Chinese and Spanish, would be valuable for its broader applicability.

Ethical Statement

This paper presents a novel method aimed to advance the research field of LLM-generated text detection. Note that all experiments are conducted within controlled laboratory environments. We do not expect the proposed method to serve as a powerful tool for potential adversaries but to raise society's broader awareness of the vulnerability of current AI-text detectors. Also, the exceptional attack performance highlights the practical limitations of current detectors. Researchers of the open-source community are encouraged to conduct stress tests on their detectors against the proposed attack, based on which future studies can develop more robust stronger detectors. **Furthermore, we conduct a preliminary study to alleviate the proposed threat via an adaptive defense that adversarially trains a Roberta-based detector using texts paraphrased by CoPA in Section F.**

All the codes, models, and datasets used in this study are consistent with their intended use and comply with the MIT License. To promote further research, we will open-source our paraphrasing tool along with the related code, model, and data.

Acknowledgement

This work is supported in part by the National Natural Science Foundation of China under grant 62171248, 62301189, 62576122, and Shenzhen Science and Technology Program under Grant KJZD20240903103702004, JCYJ20220818101012025, GXWD20220811172936001.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Anthropic. 2024. Claude-3.5-sonnet — anthropic.com.

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*.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. Eli5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567.

Hao Fang, Changle Zhou, Jiawei Kong, Kuofeng Gao, Bin Chen, Tao Liang, Guojun Ma, and Shu-Tao Xia. 2025. Grounding language with vision: A conditional mutual information calibrated decoding strategy for reducing hallucinations in lvlms. *arXiv preprint arXiv:2505.19678*.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. 2019. Gltr: Statistical detection and visualization of generated text. *arXiv preprint arXiv:1906.04043*.

Team GLM. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *Preprint*, arXiv:2406.12793.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2020. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144.

Xun Guo, Shan Zhang, Yongxin He, Ting Zhang, Wanquan Feng, Haibin Huang, and Chongyang Ma. 2024a. Detective: Detecting ai-generated text via multi-level contrastive learning. *arXiv preprint arXiv:2410.20964*.

Yanzhu Guo, Guokan Shang, Michalis Vazirgiannis, and Chloé Clavel. 2024b. The curious decline of linguistic diversity: Training language models on synthetic text. In *NAACL 2024 Findings-Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Tatsunori B Hashimoto, Hugh Zhang, and Percy Liang. 2019. Unifying human and statistical evaluation for natural language generation. *arXiv preprint arXiv:1904.02792*.

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.

Juyong Jiang, Fan Wang, Jiasi Shen, Sungju Kim, and Sunghun Kim. 2024. A survey on large language models for code generation. *arXiv preprint arXiv:2406.00515*.

Jiawei Kong, Hao Fang, Xiaochen Yang, Kuofeng Gao, Bin Chen, Shu-Tao Xia, Yaowei Wang, and Min Zhang. 2025. Wolf hidden in sheep's conversations: Toward harmless data-based backdoor attacks for jailbreaking large language models. *arXiv preprint arXiv:2505.17601*.

Steven G. Krantz and Harold R. Parks. 2002. *A Primer of Real Analytic Functions*, 2 edition. Birkhäuser Boston.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. 2023. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pages 27469–27500.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. 2023. Contrastive decoding: Open-ended text generation as optimization. In *The 61st Annual Meeting Of The Association For Computational Linguistics*.

Weixin Liang, Zachary Izzo, Yaohui Zhang, Haley Lepp, Hancheng Cao, Xuandong Zhao, Lingjiao Chen, Haotian Ye, Sheng Liu, Zhi Huang, and 1 others. 2024. Monitoring ai-modified content at scale: A case study on the impact of chatgpt on ai conference peer reviews. *arXiv preprint arXiv:2403.07183*.

Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364.

Ning Lu, Shengcai Liu, Rui He, Qi Wang, Yew-Soon Ong, and Ke Tang. 2023. Large language models can be guided to evade ai-generated text detection. *arXiv preprint arXiv:2305.10847*.

Shixuan Ma and Quan Wang. 2024. Zero-shot detection of llm-generated text using token cohesiveness. *arXiv preprint arXiv:2409.16914*.

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

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.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv preprint arXiv:1808.08745*.

Charlotte Nicks, Eric Mitchell, Rafael Rafailov, Archit Sharma, Christopher D Manning, Chelsea Finn, and Stefano Ermon. 2023. Language model detectors are easily optimized against. In *The twelfth international conference on learning representations*.

Qwen. 2024. Qwen2.5: A party of foundation models.

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.

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

P Rajpurkar. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Vinu Sankar Sadasivan, Aounon Kumar, Sriram Balasubramanian, Wenxiao Wang, and Soheil Feizi. 2023. Can ai-generated text be reliably detected? *arXiv preprint arXiv:2303.11156*.

Zhouxing Shi, Yihan Wang, Fan Yin, Xiangning Chen, Kai-Wei Chang, and Cho-Jui Hsieh. 2024. Red teaming language model detectors with language models. *Transactions of the Association for Computational Linguistics*, 12:174–189.

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*.

Yiliao Song, Zhenqiao Yuan, Shuhai Zhang, Zhen Fang, Jun Yu, and Feng Liu. 2025. Deep kernel relative test for machine-generated text detection. In *The Thirteenth International Conference on Learning Representations*.

Chris Stokel-Walker. 2022. Ai bot chatgpt writes smart essays-should academics worry? *Nature*.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, and 1 others. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Qwen Team. 2025. Qwq-32b: Embracing the power of reinforcement learning.

James Wang, Ran Li, Junfeng Yang, and Chengzhi Mao. 2024. Raft: Realistic attacks to fool text detectors. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 16923–16936.

John Wieting and Kevin Gimpel. 2018. Paranmt-50m: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462.

John Wieting, Kevin Gimpel, Graham Neubig, and Taylor Berg-Kirkpatrick. 2022. Paraphrastic representations at scale. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 379–388.

Junxi Wu, Jinpeng Wang, Zheng Liu, Bin Chen, Dongjian Hu, Hao Wu, and Shu-Tao Xiu. 2025. Moses: Uncertainty-aware ai-generated text detection via mixture of stylistics experts with conditional thresholds. *arXiv preprint arXiv:2509.02499*.

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

A Theorem and Proof

Definition 1. $g(\lambda) \coloneqq \mathbb{KL}(p_h||(1+\lambda)p'_h - \lambda p_m)$.

Proposition 2. *$g(\lambda)$ is a convex function.*

Proof. $\mathbb{KL}$ is convex and $g(\lambda)$ is the restriction of $\mathbb{KL}$ on a line, so $g(\lambda)$ is also convex. □

Proposition 3. *If $g(\lambda)$ is not constant, then $g(\lambda)$ has a unique minimum point.*

Proof. The non-constancy of $g(\lambda)$ implies that $p'_h \neq p_m$. The domain of $g(\lambda)$ is

$$I \coloneqq \bigcap_{v \in \mathcal{V}} \{\lambda \in \mathbb{R} : 0 \leq (1+\lambda)p_h'^{(v)} - \lambda p_m^{(v)} \leq 1\}.$$

$[-1, 0] \subseteq I$ so I is non-empty. I is the intersection of some closed intervals, so I is also a closed interval. Note that $g(\lambda)$ is continuous, so $g(\lambda)$ has a minimum value.

Assume that $g(\lambda)$ has minimum points λ_1 and λ_2 with $\lambda_1 < \lambda_2$. By the convexity, $g(\lambda)$ is constant on $[\lambda_1, \lambda_2]$. Note that $g(\lambda)$ is an analytic function. By Corollary 1.2.6 in (Krantz and Parks, 2002), the constancy of $g(\lambda)$ on $[\lambda_1, \lambda_2]$ implies the constancy on I. This contradicts that $g(\lambda)$ is not constant, so the minimum point of $g(\lambda)$ is unique. □

Remark. Corollary 1.2.6 in (Krantz and Parks, 2002) applies to open intervals, and $g(\lambda)$ is continuous at the endpoints, so it also applies to our closed intervals, $[\lambda_1, \lambda_2]$ and I.

Definition 2. *The minimum point of $g(\lambda)$ is λ_*.*

Theorem 1. *If $g'(0) < 0$, then $\lambda_* > 0$ and for any $\lambda \in (0, \lambda_*]$, we have*

$$\mathbb{KL}(p_h||(1+\lambda)p'_h - \lambda p_m) < \mathbb{KL}(p_h||p'_h). \quad (7)$$

Proof. $g'(0) < 0$ implies that $g(\lambda)$ is not constant, so λ_* is well-defined by Proposition 3.

$g'(0) \neq 0$ implies $\lambda_* \neq 0$. Assume $\lambda_* < 0$. By the first-order condition of convex functions,

$$g(\lambda_*) \geq g(0) + g'(0)\lambda_* > g(0).$$

This contradicts that $g(\lambda_*)$ is minimum, so $\lambda_* > 0$.

By the definition of convex functions, we have

$$\begin{aligned} g(\lambda) &\leq \frac{\lambda}{\lambda_*} g(\lambda_*) + \left(1 - \frac{\lambda}{\lambda_*}\right) g(0), \\ &= g(0) - \frac{\lambda}{\lambda_*} \left(g(0) - g(\lambda_*)\right), \\ &< g(0) \end{aligned}$$

for any $\lambda \in (0, \lambda_*]$. $g(\lambda) < g(0)$ is just (7). □

B Experimental Details

We follow the implementation of (Wang et al., 2024; Bao et al., 2024) to generate machine texts for XSum and SQuAD while adopting the approach of Dipper (Krishna et al., 2023) for LongQA. To reproduce DetectGPT and Fast-DetectGPT, we align with (Wang et al., 2024) and employ GPT2-XL (Radford et al., 2019) as the surrogate model to generate samples. As for DNA-GPT (Yang et al., 2024), we adopt the basic setup surrogated on GPT-3.5-turbo. For the LLM decoding, our experiments adopt the default setup of sampling parameters, *i.e.*, no probability clipping and $T = 1$. We run all experiments in NVIDIA RTX A6000 GPUs.

Below, we provide our carefully designed prompts to elicit human-like and machine-like token distributions from the off-the-shelf LLM.

(1) For human-style distributions: An intuitive way is to construct prompts that directly ask LLMs to produce human-like sentences by encouraging vivid lexical substitution and diverse sentence structures. However, compared to a direct request for human-toned texts, we empirically find that framing a realistic human conversation scene to LLMs leads to more vivid and diverse generations, strengthening the attack effectiveness.

System Prompt

You are a helpful paraphraser. You are given an input passage 'INPUT'. You should paraphrase 'INPUT' to print 'OUTPUT'. 'OUTPUT' should preserve the meaning and content of 'INPUT'. 'OUTPUT' should not be very shorter than 'INPUT'.

This system prompt is similar to that in (Sadasivan et al., 2023) and instructs the LLM to act as a paraphraser while maintaining the text quality of the sentences before the paraphrasing.

User Prompt

Rewrite the following INPUT in the tone of a text message to a friend without any greetings or emojis:

Rather than making a direct request for human-toned texts, we experimentally find that presenting a realistic conversation scene with humans can better guide the LLM to produce more vivid and diverse sentences, further boosting stronger attacks.

Table 3: Detection accuracy (at 5% FPR) of texts generated by GPT-4o and Gemini-1.5-Pro based on XSum dataset.

Model	Attack	Sim	Defense								Avg.
			LogRank	DetectGPT	DNA-GPT	Fast-DetectGPT	Raidar	TOCSIN	RoBERTa	R-Detect	
GPT-4o	No Attack	-	33.33	1.83	37.33	12.00	100.00	24.00	**3.33**	26.67	29.81
	Dipper	77.33	16.00	1.33	39.33	42.33	11.33	42.67	68.00	43.67	33.08
	Raidar	99.33	24.00	11.67	42.67	46.33	5.00	64.67	15.33	56.67	33.29
	Ours	96.67	**3.00**	**0.67**	**22.00**	**6.33**	**5.00**	**10.00**	16.67	**4.67**	**8.54**
Gemini-1.5-Pro	No Attack	-	21.67	13.00	36.00	32.00	28.00	33.33	12.67	24.67	25.17
	Dipper	80.67	11.00	0.67	23.33	51.17	5.00	48.00	72.67	35.67	30.94
	Raidar	100.00	31.00	8.67	26.00	39.83	30.00	48.67	22.67	42.67	31.19
	Ours	93.33	**2.00**	**2.67**	**6.67**	**10.00**	**2.00**	**10.00**	**9.33**	**2.00**	**5.58**

Table 4: Comparison of different paraphrasing attacks against 8 text-detection algorithms (at 1% FPR) using GPT-3.5-turbo generated texts from three different datasets. The best performances are bolded.

Dataset	Attack	Sim	Defense								Avg.
			LogRank	DetectGPT	DNA-GPT	Fast-DetectGPT	Raidar	TOCSIN	RoBERTa	R-Detect	
XSum	No Attack	-	32.44	6.00	41.33	83.00	5.75	94.67	44.67	49.33	44.65
	Dipper	86.67	7.33	**0.00**	8.00	44.67	2.75	46.00	74.00	28.33	26.39
	Raidar	100.00	21.00	1.00	5.33	57.33	3.25	85.33	39.33	58.67	33.91
	Ours	94.00	**2.00**	0.01	**5.33**	**4.67**	**0.00**	**18.00**	**9.33**	**0.00**	**4.92**
SQuAD	No Attack	-	20.50	2.67	0.00	82.33	4.44	83.33	15.33	58.00	33.33
	Dipper	75.33	3.00	1.33	1.33	57.67	1.00	50.00	39.33	43.33	24.62
	Raidar	95.33	21.33	5.33	0.00	59.33	5.20	68.00	8.67	48.67	27.07
	Ours	88.67	**1.17**	**1.33**	**0.00**	**9.67**	**1.00**	**12.67**	**1.33**	**0.00**	**3.40**
LongQA	No Attack	-	28.50	7.50	0.00	74.00	7.25	82.67	28.67	74.33	37.87
	Dipper	94.67	6.50	**0.00**	0.00	55.67	1.00	31.33	49.33	63.33	25.90
	Raidar	100.00	15.83	2.17	0.00	48.67	6.75	64.00	15.33	70.33	27.89
	Ours	95.33	**2.67**	1.33	**0.00**	**5.33**	**0.00**	**8.67**	**2.00**	**3.00**	**2.88**

(2) For machine-style distributions:

To elicit machine-like responses from LLMs, our prompt engineering (PE) constructs a wide range of prompts, *e.g.*, ask for direct paraphrasing or instruct the LLM to reply in the tone of an AI assistant. The LLM likelihood provided by a detector is used as a proxy to quantify the degree of machine-style characteristics in the output. We find that naively prompting an LLM to rewrite a machine text often reduces existing machine-related features, as the rewritten texts inherit mixed stylistic and lexical preferences from multiple LLMs, increasing textual diversity. To preserve machine-related features, we incorporate stylistic cues into the prompts, e.g., instructions like "... in the tone of an AI assistant ..." or "... in a machine-writting style ..." to guide the LLM for a more typical machine expression, which indeed makes the output more easily detectable and, in turn, enhances the attack.

System Prompt

You are a helpful assistant.

User Prompt:

Repeat the following paragraph:

Considering that the original LLM-generated sentence carries the richest machine-related characteristics that are most easily detected, we adopt a direct and effective strategy by making LLM repeat the input sentence to obtain the most machine-like token probabilities. We provide the results of representative prompts in Fig. 3, which validate the effectiveness of our selection strategy.

C Additional Results

More Source LLMs. We provide the performance of paraphrasing on machine texts generated by GPT-4o (Achiam et al., 2023) and Gemini-1.5 Pro (Team et al., 2024) in Table 3. It can be observed that the proposed CoPA continues to achieve better attack effectiveness than existing paraphrasing methods. Also, the detection algorithms obtain relatively worse clean performance on texts generated by two advanced LLMs.

Results of FPR=1%. We then evaluate the at-

Table 5: Attack results paraphrased by different off-the-shelf LLMs (at 5% FPR) using GPT-3.5-turbo generated texts from three different datasets.

Dataset	Detector	Paraphraser				
		No Attack	R1-Distill-32B	QwQ-32B	GLM-4-9b-hf	Qwen2.5-72B
XSum	Fast-DetectGPT	95.33	24.33	11.33	31.33	17.00
	TOCSIN	98.00	42.67	10.00	87.00	26.67
	R-Detect	49.33	6.33	00.00	38.33	0.00
SQuAD	Fast-DetectGPT	93.50	26.83	9.50	31.33	27.50
	TOCSIN	92.67	42.00	17.33	81.33	25.33
	R-Detect	58.00	13.00	0.00	39.67	0.00
LongQA	Fast-DetectGPT	86.00	14.67	11.33	27.33	11.33
	TOCSIN	88.67	26.00	2.00	78.00	16.00
	R-Detect	74.33	22.67	0.00	47.33	3.00

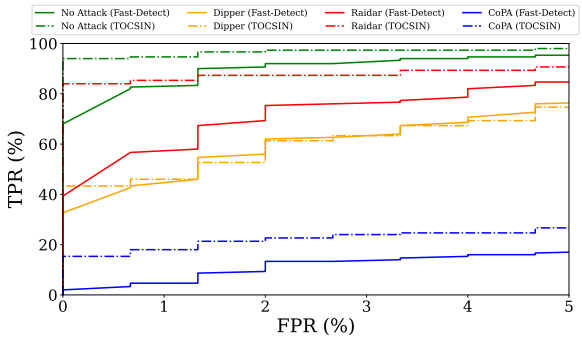

Figure 8: ROC (0-5% FPR) for GPT-3.5-turbo on Fast-DetectGPT and TOCSIN before and after paraphrasing. The proposed CoPA achieves the best detection rate across various FPRs.

tack under a more strict setup where FPR=1%. As shown in Table 4, CoPA consistently exhibits superior attack performance over current paraphrasing attacks. We also observe that some detection algorithms fail to produce any defense effects even without any attack at TPR=1%, raising concerns about their feasibility in practical scenarios.

ROC Curve Analysis. Figure 8 shows the TPR trends corresponding to different FPR values varying from 0% to 5%. As observed, TPR generally increases as FPR grows, and CoPA significantly reduces the TPR values of detection algorithms across all FPR thresholds, which strongly validates the effectiveness of the proposed CoPA. Note that under the stricter and more realistic setting of FPR = 1%, CoPA reaches a detection accuracy below 20% against these defenses, further underscoring its superior performance.

Impact of LLM Sampling Parameters. During decoding, LLMs employ various sampling parameters such as Top-p, Top-k, and the temperature coefficient T to adjust the sampling results. To investigate their influence, we conduct ablation studies regarding these parameters during the decoding process of our paraphrasing in Figure 9. For Top-p and Top-k, the reduction of p or k results in a drop in sampling diversity since fewer tokens are retained, thereby generating more machine patterns and impairing the performance. For the temperature (T), numeric results indicate that the increase of T facilitates the creativity and diversity of sampling choices by reducing the difference in token probabilities, hence better misleading the detectors. An adversary can adjust these parameters based on their needs to achieve a superior attack while controlling the writing styles of generated sentences.

Results of More LLMs as paraphrasers. To validate the universality of the proposed attack, we next consider more LLMs as the paraphraser, including various model scales and recently prevalent reasoning-based models. Specifically, we consider Deepseek R1-Distill-32B (DeepSeek-AI, 2025), QwQ-32B (Team, 2025), and GLM-4-9B-hf (GLM, 2024). The quantitative results in Table 5 show the effectiveness of the proposed CoPA across various LLMs. Note that the QwQ-32B generally achieves the best performance. However, the reasoning-based models require significantly more inference time than regular models. We choose the Qwen2.5-72B to balance effectiveness and efficiency.

Table 6: Comparison of CoPA with a surrogate-based paraphrasing attack against two SOTA detectors.

Attack	Sim	Defense	
		Fast-DetectGPT	TOCSIN
RedTeaming	78.00	30.00	38.67
Ours	**94.00**	**17.00**	**26.67**

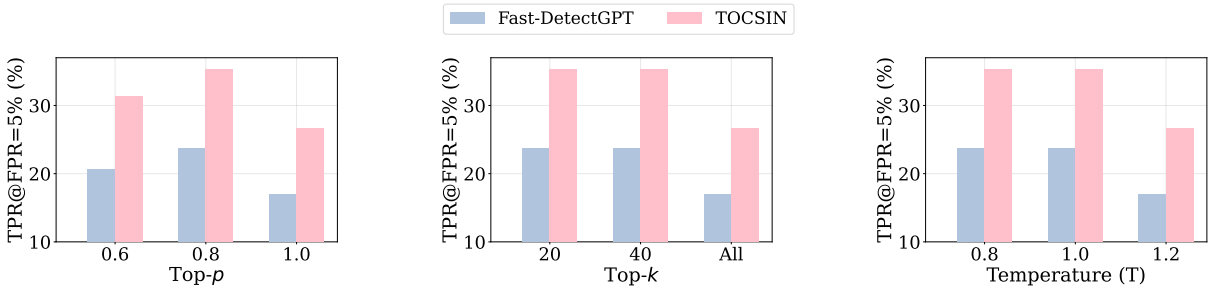

Figure 9: Detection accuracy (at FPR=5%) of CoPA under different sampling parameters against Fast-DetectGPT (Bao et al., 2024) and TOCSIN (Ma and Wang, 2024). We calculate results using samples from the XSum dataset.

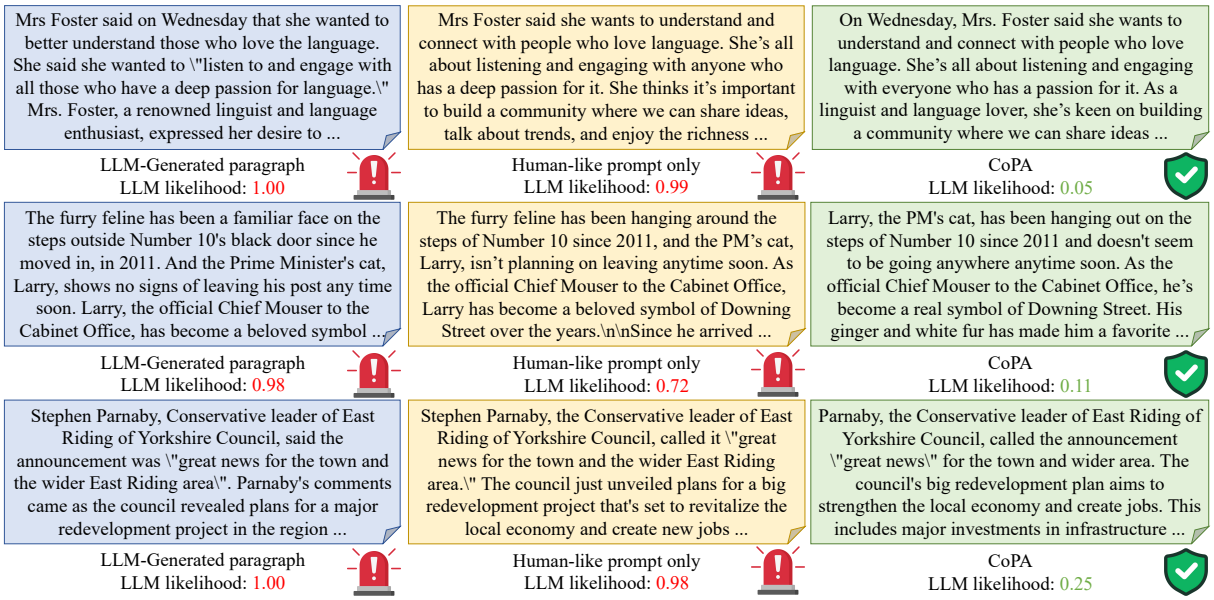

Figure 10: Visualization of paraphrased sentences from human-like distribution p_h and our contrastive distribution p_c (*i.e.*, our CoPA). The LLM likelihood is calculated based on Fast-DetectGPT.

Comparison with a surrogate-based baseline. This paper follows Dipper (Krishna et al., 2023) and focuses on the more practical and universal attacks without relying on any surrogate detection model. A direct comparison of these methods with the surrogate-based paraphrasing attack introduced in RedTeaming (Shi et al., 2024) may raise concerns of unfairness. However, results in Table 6 reveal that the proposed CoPA can still achieve better performance than RedTeaming.

Notably, we include these results only for experimental completeness. The surrogate-based methods are not the focus of this work.

Ablation study of the adaptive truncation mechanism. To avoid penalizing the probabilities of reasonable and valid tokens, we incorporate an adaptive truncation mechanism to constrain the output token distribution. We provide an ablation study to investigate its influence. As observed in Table 7, by truncating the token distribution within a reliable token candidate pool, we can improve the text quality of the generated sentences while maintaining attack effectiveness.

Table 7: Ablation on the adaptive truncation technique.

Method	Sim	Fast-DetectGPT	TOCSIN	R-Detect
w/o truncation	93.33	19.67	**31.33**	5.33
CoPA	**94.00**	**17.00**	26.67	**4.67**

D Analysis of human-like prompt only

As shown in Figure 10, we observe that solely relying on the human-like prompt x_h results in unstable attack performance, *i.e.*, some sentences derived from the human-like distribution p_h still retain prominent machine-related features, which render them easily identifiable by text detectors. To alleviate this issue, our CoPA framework utilizes an auxiliary machine-like distribution to fully remove these machine characteristics from p_h, significantly

deceiving text detectors and leading to incorrect predictions. As corroborated by more detailed empirical studies, the proposed contrastive strategy greatly boosts the effectiveness and stability of the paraphrasing attack.

E Details about Machine Prompt

The prompts used in Figure 3 are as follows:

Machine Prompt 1

Repeat the following paragraph:

Machine Prompt 2

Rewrite the following paragraph in the tone of an AI assistant:

Machine Prompt 3

Paraphrase the following paragraph:

Machine Prompt 4

Rewrite the following paragraph:

F An Adaptive Defense Strategy

To alleviate the proposed threat, we implement an adaptive defense that adversarially trains OpenAI's LLM-text classifier RoBERTa-large. We fine-tune the model for 10 epochs using 5k human texts and 5k machine texts (including both the original machine texts and those paraphrased by CoPA). We present the optimal performance at a proportion of 50% CoPA-paraphrased samples within the 5k machine texts in Table 8.

Table 8: Detection accuracy (at 5% FPR) of texts generated by GPT-3.5-turbo based on the XSum dataset.

Attack	w/o training	Adversarial training
No Attack	66.67	99.78
CoPA	22.67	78.00

The results indicate that the adaptive defense based on texts provided by our CoPA can alleviate the proposed threat to some extent.

G Human Prompts for Different Contexts

To better handle the different demands of contexts, we conduct a preliminary study by designing two alternative prompts as follows:

Human Prompt I for General Scenarios

Rewrite the following INPUT in human style, with varying sentence structures and replace common terms with nuanced synonyms. Maintain the meaning and avoid any repetition.

Human Prompt II for Academic Context

Rewrite the following INPUT in a human-written academic style, with varying sentence structures and replace common terms with nuanced synonyms. Maintain the meaning and avoid any repetition.

Here are the corresponding attack results:

Table 9: Attack results (at 5% FPR) of different human prompts using GPT-3.5-turbo generated texts on XSum.

Prompt	Fast-DetectGPT	TOCSIN	RoBERTa	R-Detect
Prompt I	24.67	43.33	15.33	18.67
Prompt II	24.33	39.67	**5.33**	14.33
CoPA	**17.00**	**26.67**	22.67	**4.67**

The quantitative results in Tab. 1 and demonstrations of paraphrased sentences show that CoPA is able to flexibly combine with prompts of various styles while maintaining high effectiveness. Users may design their own prompts based on our provided template to adapt to diverse writing styles.

H Evaluation on Text Quality

Apart from the attack effect, it is necessary to analyze the text quality of paraphrased sentences. Specifically, we first conduct a deeper linguistic analysis of the output texts in Table 10, including TTR (Type-Token Ratio) and MTLD (Measure of Textual Lexical Diversity) metrics for lexical diversity, Div_syn (Guo et al., 2024b) for syntactic variability. The results demonstrate the high quality of our generated sentences.

Table 10: Analysis of paraphrased sentences from different methods based on the XSum dataset.

Method	TTR↑	MTLD↑	Div_syn↑
No Attack	0.5734	111.4913	0.4700
Dipper	0.5511	74.8257	0.4648
CoPA	0.6746	145.8134	0.5593

We also provide a comprehensive evaluation of natural fluency and semantic consistency with addi-

Table 11: Comparison of our method with Dipper on text quality. The perplexity is calculated on GPT-neo.

Dataset	Text	Sim↑	Perplexity↓	GPT-4 Eval		Human Eval	
				Natural fluency↑	Consistency↑	Natural fluency↑	Consistency↑
XSum	Human	-	16.11	3.88	-	4.40	-
	Machine	-	8.857	4.33	-	4.94	-
	Dipper	86.67	14.76	3.74	3.76	4.25	4.26
	Ours	94	15.58	4.64	4.95	4.74	4.87
SQuAD	Human	-	19.52	3.60	-	3.98	-
	Machine	-	10.28	4.71	-	4.81	-
	Dipper	75.33	14.70	3.53	3.57	4.02	4.05
	Ours	88.67	17.77	4.56	4.91	4.56	4.79
LongQA	Human	-	27.54	3.48	-	3.75	-
	Machine	-	7.79	4.91	-	4.99	-
	Dipper	94.67	11.61	3.57	4.11	4.04	4.23
	Ours	95.33	13.08	4.57	4.99	4.53	4.81

tional key metrics, including text perplexity, GPT4-assisted evaluation, and human study. To conduct the human evaluation, we choose GPT-3.5-turbo as the source model and randomly select 100 pairs of texts from each dataset for human annotators. The evaluation criteria generally align with those in Dipper (Krishna et al., 2023), where we recruit 10 native English speakers from Amazon Mechanical Turk (MTurk) to perform the evaluation. We report the average scores to reduce subjective biases in Table 11. The results indicate that CoPA produces paraphrased texts with lower perplexity than authentic human-written texts, while achieving substantially better fluency and semantic consistency compared to those generated by Dipper, in both terms of GPT-4 assited and human evaluation.

The instructions given to human annotators for *semantic consistency* align with those in Dipper, while we provide the scoring standard for *natural fluency* and the detailed prompt for GPT-4 auto evaluation as follows:

Instruction for Human evaluation:

Natural Fluency Scoring (1–5)

5. Excellent: Text flows perfectly naturally with varied, idiomatic phrasing. Grammar, word choice, and sentence structure appear completely native with zero awkwardness.

4. Good: Text reads smoothly with only minor and infrequent awkwardness. May contain 1-2 subtle non-native phrasings, but remains highly readable.

3. Fair: Generally understandable but contains noticeable unnatural phrasing. Some grammatical errors or awkward constructions occasionally disrupt flow.

2. Poor: Frequent unnatural phrasing and grammatical errors make reading difficult. Requires effort to understand in places.

1. Very Poor: Severely broken or unnatural English with major grammar issues. Often difficult or impossible to understand.

Prompt for GPT-4 evaluation:

(1) Task: Evaluate the natural fluency of a given sentence. Use a 5-point scale (5 = highest).
Natural Fluency (1-5):

1. Does the rewritten sentence flow naturally, avoiding awkward phrasing or redundancy?
2. Assess grammar, word choice, and readability (e.g., smooth transitions between clauses).
3. Penalize unnatural idioms or register mismatches (e.g., mixing formal and colloquial terms)

Output Format
Please provide the score for the metric.
Include a concise rationale (1-2 sentences per metric) highlighting specific strengths/weaknesses.
Example:

INPUT: "The deadline got pushed back because of unexpected tech issues."
OUTPUT: 4/5 (Colloquial tone matches intent; "pushed back" is natural but "tech issues" slightly informal).

(2) Task: Evaluate the semantic consistency of a rewritten sentence compared to its original version. Use a 5-point scale (5 = highest).
Semantic Consistency (1–5):

1. Does the rewritten sentence preserve the original meaning?
2. Check for critical information retention, logical coherence, and absence of distortion.
3. Deduct points for omissions, additions, or ambiguous interpretations

Output Format
Provide the score for the metric.
Include a concise rationale (1–2 sentences per metric) highlighting specific strengths/weaknesses.
Example:

INPUT:
Original: "The project deadline was extended due to unforeseen technical challenges."
Rewritten: "The deadline got pushed back because of unexpected tech issues."
OUTPUT:
5/5 (Key details retained; no loss of meaning).