HACo-Det: A Study Towards Fine-Grained Machine-Generated Text Detection under Human-AI Coauthoring

Zhixiong Su[*1] **Yichen Wang**[*1, 2] **Herun Wan**[1] **Zhaohan Zhang**[3] **Minnan Luo**[†1]
[1]Xi'an Jiaotong University [2]University of Chicago [3]Queen Mary University of London
{15958057,wanherun}@stu.xjtu.edu.cn yichenzw@uchicago.edu
zhaohan.zhang@qmul.ac.uk minnluo@xjtu.edu.cn

Abstract

The misuse of large language models (LLMs) poses potential risks, motivating the development of machine-generated text (MGT) detection. Existing literature primarily concentrates on binary, document-level detection, thereby neglecting texts that are composed jointly by human and LLM contributions. Hence, this paper explores the possibility of fine-grained MGT detection under human-AI coauthoring. We suggest fine-grained detectors can pave pathways toward coauthored text detection with a numeric AI ratio. Specifically, we propose a dataset, HACo-Det, which produces human-AI coauthored texts via an automatic pipeline with word-level attribution labels. We retrofit seven prevailing document-level detectors to generalize them to word-level detection. Then we evaluate these detectors on HACo-Det on both word- and sentence-level detection tasks. Empirical results show that metric-based methods struggle to conduct fine-grained detection with a 0.462 average F1 score, while finetuned models show superior performance and better generalization across domains. However, we argue that fine-grained co-authored text detection is far from solved. We further analyze factors influencing performance, e.g., context window, and highlight the limitations of current methods, pointing to potential avenues for improvement.

1 Introduction

The generative capability of large language models (LLMs), e.g., GPT-4o (OpenAI, 2024b), Llama-3 (AI@Meta, 2024), and Claude-3.5-Sonnet (Anthropic, 2024), has been rapidly developed and gain further capabilities of instruction-following (Qin et al., 2024) and interaction (Castillo-Bolado et al., 2024). It leads to the emergence and prevalence of human-AI interactive and collaborative generation systems, e.g., GPT-4o-canvas (OpenAI, 2024c), Notion (Notion, 2023), and Wordcraft (Google, 2024). However, this raises broader public concerns about its misuse, e.g., academic plagiarism (Jarrah et al., 2023), privacy leak (Mireshghallah et al., 2022), and hallucination (Li et al., 2023) etc., hence spurring the creation of machine-generated text (MGT) detection (Gehrmann et al., 2019; Zellers et al., 2019; Mitchell et al., 2023). Literature of MGT detection usually assigns a binary label, i.e., either human-written text (HWT) or MGT, on the document level (Mitchell et al., 2023; Liu et al., 2023; Hu et al., 2023; Wang et al., 2024; Dugan et al., 2024; Li et al., 2025). Among these MGT detection methods are two predominant categories: metric-based methods compute numeric metrics, e.g., logits, of texts in a white-box setting; finetune-based methods train classification models on annotated corpora (details at §5 and Appendix B.1).

However, the binary document-level MGT detection task and methods might be inadequate for the current rising trend of human-AI collaboration (Lee et al., 2022; Chakrabarty et al., 2022; Reza et al., 2024; Li et al., 2024c; Shu et al., 2024; Reza et al., 2025). It further calls for a step toward fine-grained MGT detection to attribute partial authorship to humans or machines. Heated debates have occurred on the dilemma of authorship attribution of human-AI coauthored contexts for binary document-level setting (Tripto et al., 2023). Fine-grained detection is a potential way to mitigate the controversy, meanwhile reaching an interpretable prediction with a numeric AI ratio and localization (Gehrmann et al., 2019; Kushnareva et al., 2023). Different forms of fine-grained tasks, e.g., triple classification with Mixtext class (Gao et al., 2024), boundary detection (Kushnareva et al., 2023), and MGT localization (Zhang et al., 2024c), are proposed.

However, some blind spots still exist in MGT

[*] Equal contribution

[†] Corresponding author: Minnan Luo, School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China

detection under coauthoring. Firstly, some datasets (Gehrmann et al., 2019; Hu et al., 2023; Liu et al., 2023; Pu et al., 2023) utilize a human-written beginning as the prompt and request LLMs to continue. They label these outputs as MGT, but we question that these are partly human-written. Secondly, some works use paraphrasing as the machine generation process and directly label fine-grained attribution of paraphrased contexts to the machine (Zhang et al., 2024a; Li et al., 2024a). We argue that, if portions of contents remain lexically unchanged during paraphrasing, their authorship should be attributed to the original author. Moreover, Kushnareva et al. (2023); Gao et al. (2024) stop at single-turn collaboration, but realistic interactions could be more complex. A better task design is to be explored.

In this paper, we propose a novel task and benchmark for human-AI coauthored text detection, identifying their fine-grained authorship at the word level, and aggregating to the sentence level (§3). We propose a novel dataset, `HACo-Det`, utilizing dominant instruct LLMs to partially paraphrase texts in multiple turns (§4). Furthermore, we comprehensively reform the current detectors (§5) to our tasks. In §6, we evaluate their in-domain and out-of-domain performance at both word and sentence levels. Further in §7, we analyze the limitations of current detectors, including capability in generalization and zero-shot prediction. We propose possible trails toward improvement, e.g., larger context windows, augment training corpus diversity, etc. In summary, our work provides three main contributions:

- We propose a word-level labeled MGT detection dataset, `HACo-Det`, in which texts are collaboratively generated by humans and predominantly instruct LLM.
- We reform seven MGT detectors to our tasks and evaluate them on `HACo-Det` at word and sentence levels, finding large space exists to improve current methods, especially metric-based ones.
- We further analyze the limitation of current detectors and influencing factors, e.g., context windows, suggesting some possible ways for enhancement.

2 Related Works

2.1 Document-Level MGT Detection

Previous works on MGT detection treat the problem as a binary classification task with document-wise labels. The goal is to find discriminative metrics or representations of the input document. For example, LogLikelihood (Solaiman et al., 2019), Entropy (Gehrmann et al., 2019), Rank (Gehrmann et al., 2019), and Log-Rank (Mitchell et al., 2023) are found to be promising metrics in differentiating MGT and HWT. Given the convenience of obtaining metrics from the output distribution of the model, it holds a strong assumption that the generators are white-box, which is not practical in closed-source models (Achiam et al., 2023; Anthropic, 2024). Some works train a finetune-based detector to extract detectable representations from texts. Guo et al. (2023) use RoBERTa (Liu et al., 2019) as detector backbone for acquiring text embedding and classification. Liu et al. (2023) incorporates the coherence graph into the text representation and further Liu et al. (2024) use perturbed text as additional input for learning robust and discriminative embeddings for classification. Hu et al. (2023); Li et al. (2025) apply adversarial training to improve the robustness of the MGT detectors against attacks. Different from the problem setting in the previous works, we model MGT detection as a sequence-to-sequence classification task where we label every word as MGT or HWT to identify the MGT part in hybrid documents.

2.2 Fine-Grained MGT Detection

Recent works try to formalize more fine-grained detection tasks than document-level prediction. Fine-grained MGT detection shows the potential to handle coauthored text detection and provide interpretable prediction, e.g., numeric AI ratio and localization. Gao et al. (2024) propose a triple classification setting, introducing a new class Mixtext, in which humans and LLM are both involved in the generation process. Kushnareva et al. (2024) introduce a boundary detection task, under which the authorship of each text is started with humans and shifted to machines at a specific boundary position. Li et al. (2024b) propose a detection framework aiming to identify paraphrased text spans by sentence-level classification. Zhang et al. (2024c) propose the MGT localization task to detect short MGT sentence spans and a corresponding method based on contextual information. Tao et al. (2024)

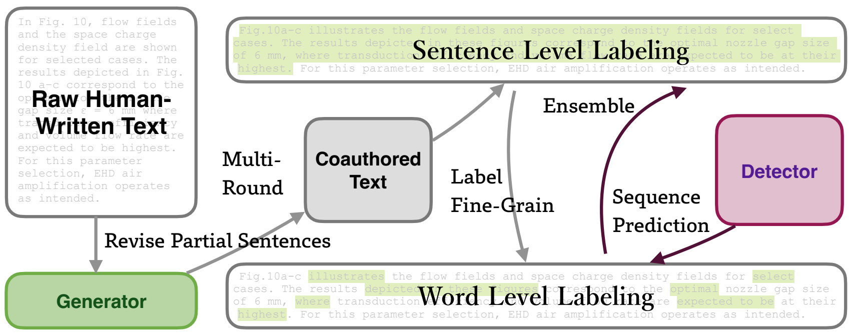


Figure 1: An overview of the study pipeline in HACo-Det. Firstly, we sample texts according to a curated rule from the raw human-written texts. Then we use instruct LLM to paraphrase them multiple rounds. We label the texts at the word level and sentence level, according to the detection task setting (§4.2). In the main experiment (§5), the detectors do sequence prediction at the word level, then ensemble the results to the sentence level.

introduce a sentence-level MGT detection method based on features from multiple levels and aspects. However, we carefully design a human-AI coauthored text generation pipeline different from the above. And we comprehensively reform and evaluate current detectors in both in-domain and out-of-domain settings, which is under-discussed in the literature.

2.3 MGT Detectors Robustness

The robustness of MGT detectors is a vital attribution for state-of-the-art detectors. Shi et al. (2024) first introduces the adversarial attack against MGT detectors, including word substitution and prompt attack to generate texts that deceive popular detectors. Wang et al. (2024) comprehensively study the detector's robustness towards a wide range of perturbation attack methods from character level to sentence level, finding significant performance degradation for most detectors under most attacks though only with limited access. And Dugan et al. (2024) proposes RAID, a large-scale and challenging benchmark for testing MGT detector's robustness. Although these works conduct extensive robustness tests on MGT detectors and apply revision methods, e.g., paraphrasing, as attacks, they overlook that fine-grained revision should be viewed as a coauthoring process.

3 Task Definition

In general, we can categorize MGT detection tasks into the following three types of tasks based on different scales of authorship attribution:

Document-level MGT detection classifies the author attribution of text at the document level, i.e., all text in the entire document will be labeled as machine-generated or human-written. Formally, given an input passage $\mathcal{T}$ and the detector D_d, we get the label of the entire document ℓ. So the detection task can be represented as $D_d(\mathcal{T}) = \ell$.

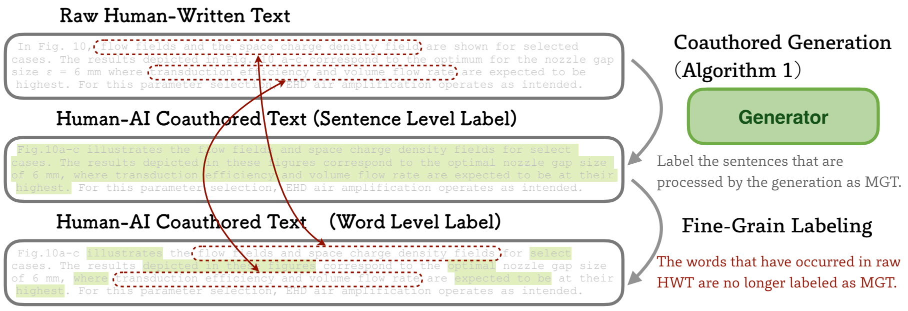
Sentence-level MGT detection detailed classifies each sentences. Some documents contain a mix of machine-generated and human-authored content that cannot be labeled as a whole. These contents usually contain a few sentences. So it is more reasonable to identify author attribution and label the entire document at the sentence level. The input passage $\mathcal{T}$ will be divided into sentence sequences $T_s = [s_1, s_2, ..., s_n]$, and the label of the entire document will be $L_s = [\ell_1, \ell_2, ..., \ell_n]$. The detection task can be defined as $D_s(T_s) = L_s$.

Word-level MGT detection offers a more fine-grained text analysis. It could be hard to define and determine the text authorship even at the sentence level in some scenarios, e.g., LLM paraphrasing on human-written sentences (Tripto et al., 2023). Hence, detecting modifications[1] of text at the word level may be meaningful. The input passage $\mathcal{T}$ will be divided into word sequences[2] $T_w = [w_1, w_2, ..., w_m]$ and labeling word-wise $L_w = [\ell_1, \ell_2, ..., \ell_m]$. The detection task can be defined as: $D_w(T_w) = L_w$.

As we target fine-grained MGT detection, we

[1]The definition of text modification and the threshold of its strength to shift authorship can be subjective, especially in different tasks and scenarios. In this paper, we do not aim to have a holistic claim or consensus on this issue, but we are trying to suggest one of the reasonable settings and practical pipelines to initially study the possibility and performance of word-level MGT detection methods.

[2]We are using word level as the granularity of the task instead of token level to prevent the different split of different tokenizers. However, most methods output prediction on the token level. For multiple token words, we will do ensembling in the first place.


Raw Human-Written Text
Human-AI Coauthored Text (Sentence Level Label)
Human-AI Coauthored Text (Word Level Label)
Coauthored Generation (Algorithm 1)
Generator
Label the sentences that are processed by the generation as MGT.
Fine-Grain Labeling
The words that have occurred in raw HWT are no longer labeled as MGT.

In Fig. 10, flow fields and the space charge density field are shown for selected cases. The results depicted in Fig. 10 a-c correspond to the optimum for the nozzle gap size ε = 6 mm where transduction efficiency and volume flow rate are expected to be highest. For this parameter selection, EHD air amplification operates as intended.

Fig.10a-c illustrates the flow fields and space charge density fields for select cases. The results depicted in these figures correspond to the optimal nozzle gap size of 6 mm, where transduction efficiency and volume flow rate are expected to be at their highest. For this parameter selection, EHD air amplification operates as intended.

Fig.10a-c illustrates the flow fields and space charge density fields for select cases. The results depicted in these figures correspond to the optimal nozzle gap size of 6 mm, where transduction efficiency and volume flow rate are expected to be at their highest. For this parameter selection, EHD air amplification operates as intended.

Figure 2: Grounded attribution labeling process on the word level and the sentence level setting in data construction. The fine-grain labeling process relies on word-span matching (in the red box and linked by curved arrows) between the hybrid texts before and after each revision turn.

choose to work on sentence-level and word-level detection tasks in this paper. The study pipeline overview is shown in Figure 1. To keep consistency between sentence-level and word-level prediction outputs, we formalize the task as sequence labeling at the word level first. Basically, our task is to binary classify each word's authorship into HWT or MGT. We input the entire text $\mathcal{T}$ and output sequence of word labels $L = [\ell_1, \ell_2, ..., \ell_m]$. For sentence-level detection, we use the ensemble method to convert word-level labels to sentence-level labels. Specially, for each sentence s_i composed of words $[w_{i,1}, w_{i,2}, ..., w_{i,m}]$, we do majority voting as

$$\begin{aligned} L_{s_i} &= \arg\max_c V_c(s_i) \\ &= \arg\max_c \sum_{k=1}^{m} \mathbb{I}(L_{w_{i,k}} = c). \end{aligned}$$

4 Dataset Creation

In this section, we present the details about our fine-grained MGT detection dataset, `HACo-Det`. Figure 2 illustrates the construction pipeline.

4.1 Raw HWT Resources

`HACo-Det` comprises 11,200 human-authored texts from four different domains, including news articles from XSum (Narayan et al., 2018), story writing from WritingPrompts (Fan et al., 2018), scientific papers from Dagpap24 (Chamezopoulos et al., 2024), and Wikipedia contexts from Wikipedia_en (Wikimedia, 2024). We utilize these texts as initial human-authored manuscripts to construct human-AI coauthored content.

For preprocessing, we filter out entries that are too short or of low quality. The filtering process is detailed in Appendix A.1. We aim for an average length of around 4,000 words because it allows more turns of AI involvement. Moreover, it allows us to control the difficulty of tasks. Solaiman et al. (2019); He et al. (2023b) show that long text is less difficult to detect in document-level binary classification. Empirically, length of around 4,000 words alone performance differences between detectors are distinguishable.

4.2 Construction Pipeline

In order to make the generated dataset more diverse, we generated the text for the four domains on four instruction-tuning models, including Llama-3, Mixtral, GPT-4o mini, and GPT-4o. We provide details of these models' settings in Appendix §A.3. In addition, we apply different corresponding instructions for each model in the generation process. The instruction templates of each LLM for generation are shown in Appendix §A.4.

In literature, the common dataset construction process of human-AI collaborative generation is feeding human-written prefixes to the model and then instructing the model to continue writing. This process has two drawbacks: (*i*) The annotation is given document-wise, either a turning point position or an overall label, instead of finer-grained. (*ii*) It is completed in only a single turn without more interactions, which might oversimplify human-AI coauthoring.

We propose to construct a collaborative, word-level labeled dataset through multiple turns of revision interaction to deal with the above-mentioned issues. Instead of continuing writing, we instruct LLMs to paraphrase specific parts of the contents of the raw HWTs, while extraction follows the sentence boundaries. We set the default temperature for LLM during paraphrasing. We repeat this revi-

Domain	Resources	Num	#MGT Frag.	MGT Portion	Length #Word
News	Xsum	2,800	2.44	45%	1,489
Story	Wrting_prompts	2,800	3.97	40%	1,964
Scientific Paper	Dagpap24	2,800	4.56	32%	4,558
Wikipedia	Wikipedia_en	2,800	6.01	26%	7,724
All	-	11,200	4.25	36%	3,934

Table 1: The statistics of HACo-Det and its each domain. '#MGT Frag.' refers to the number of LLM-authored text fragments, i.e., the number of the LLM revision turns; and 'MGT Portion' means the average portion of LLM-authored text fragments in each text in terms of word numbers.

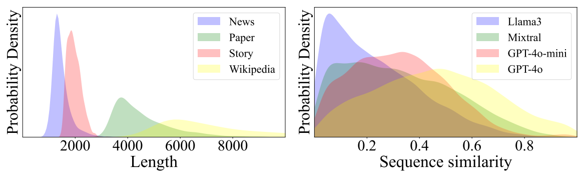


Figure 3: (Left) Length distribution of the texts from different domains. The paper and Wikipedia domains have much longer content than news and stories. (Right) Similarity of the texts from different generators. GPT-4o results in the most significant alterations while Llama-3 cases the least.

sion process multiple turns. The number of turns depends on the length of the initial raw texts. Notably, we will locate and only revise the MGT spans in the hybrid texts to prevent ambiguity of authorship. In detail, we apply NLTK toolkit (Bird et al., 2009) to divide the raw HWTs $\mathcal{T}$ into sequences of sentences $S = \mathcal{ST}(H) = [s_1, s_2, ..., s_n]$. Then we each turn select a continuous sentences span $S_{\text{span}} = [s_k, s_{k+1}, ..., s_{k+l}]$ as fragments based on the rules of Algorithm 1. We query the instruct LLMs to paraphrase the selected span S_{span} to get revised spans fragments S'_{span}, which is labeled as MGT in sentence level. We replace S_{span} with its corresponding S'_{span}. Then follows the next turn.

For word-level labeling, we consider the lexical change in LLM revision turn. By default, all words in S'_{span} is labeled as MGT. But if any word in S'_{span} has occurred in the same tense in the corresponding S_{span}, we keep its original attribution label.[3] We illustrate the labeling process in Figure 2.

4.3 Dataset Analysis

Table 1 presents the statistics of the HACo-Det dataset. To further sanity check the quality, we analyze length distribution and revision strength.

Length: While focusing on fine-grained detection of longer text, the length of texts still differs among different domains. As shown in Figure 3 Left, the orders of magnitude of word count ranged from 1,000 to 10,000 words. The paper and Wikipedia domains have much longer content than news and stories.

Revision: We compute the sequence similarity between each pair of coauthored text and raw HWT. Specifically, we calculate the portion of substring shared with the coauthored text and its corresponding raw HWT. A larger distance represents stronger revision by the LLM generators and might refer to a stronger signal in detection. As shown in Figure 3 Right, GPT-4o results in the most significant alterations while Llama-3 cases the least. This might refer to the generation capability of generators (Kirk et al., 2023).

5 Experiment Setting

To extensively evaluate the performance of different baselines on HACo-Det, we employ both in-domain and out-of-domain settings.

In-domain setting. We split the dataset into the training, validation, and test sets in a ratio of 4:1:2. For the finetune-based methods, we supervised fine-tuning them on the training set and report the performance on the test set. For the metric-based methods, we pick the optimal threshold on the training set, which is the point on the ROC curve with the largest Youden index. We directly apply the threshold test on the test set.

Out-of-distribution setting. HACo-Det consists of four writing datasets from four different domains, and we use four different AI models as generators. Since overfitting on task and generator model is a common problem for detection methods, we want to evaluate whether the detection methods generalize well. We conducted out-of-distribution

[3]We suggest that this is one of the many potential definitions of authorship transformation. The definition of attribution is not holistic and there can be different definitions and formulations in the fine-grained tasks. However, our intention is to have a checkable and reasonable heuristic. The discussion of the holistic definition of this issue is out of our scope.

experiments in both these two dimensions. In the out-of-domain setting, we train on texts from one dataset (but mix texts from different generators) and test on text from other datasets. In the out-of-model setting, we will train on text generated from one model (mix datasets) and test on text generated from other generator models.

Baselines. We evaluate three categories of baselines, including seven predominant detection methods, on `HACo-Det`. (*i*) Finetune-based methods, including SeqXGPT[4] (Wang et al., 2023) and DeBERTa (He et al., 2023a); (*ii*) Metric-based methods with perturbation, including DetectGPT (Mitchell et al., 2023), Fast-DetectGPT (Bao et al., 2023), and NPR in DetectLLM (Su et al., 2023); (*iii*) Metric-based detectors without perturbation, including GLTR[4](Gehrmann et al., 2019) and LRR in DetectLLM (Su et al., 2023). We provide more details about baselines in Appendix B.1 and their corresponding adaptions in Appendix B.2.

Metrics. We employ macro f1-score, precision, recall, and AUC ROC as metrics to evaluate baselines. We present macro f1-score and AUC ROC, which are more suitable for imbalanced datasets, in the main text and other metrics in Appendix C.1.

6 Results

6.1 In-Domain Detection

Word-Level Detection. As shown in Table 2, all SOTA detectors fail on our word-level coauthored text detection setting, except for DeBERTa. The F1 scores and AUC scores for metric-based methods are around random prediction, indicating they are unable to conduct word-level detection. DeBERTa reaches 0.831 in terms of F1-W, standing out from all competitors. We suggest that semantic representation might be vital for fine-grained detection. As DeBERTa is a supervised fine-tuned detector based on embedding, it is accessible to more semantic features than metrics. However, there is still room for accuracy improvement.

Sentence-Level Detection. In Table 2, DeBERTa shows outperformance than other methods. SeqXGPT also works and performs well in relative. In addition, Fast-DetectGPT and LLR perform best among the metrics-based detectors, which is consistent with their better performance in document-level detection (Dugan et al., 2024). In contrast, the performance of most metric-based detectors is inferior to random guesses, which emphasizes that their capability on document-level detection cannot effectively generalize to sentence-level under our setting. We suggest the reason is that simply ensembling is unable to adapt metrics computed at the word level to sentence-level detection. The result emphasizes even at sentence-level `HACo-Det` tasks, most detectors are still far from perfect.

Detector	F1-W ↑	AUC-W ↑	F1-S ↑	AUC-S ↑
Random	0.433	-*	0.497	-*
Finetune-based Methods				
DeBERTa	**0.831**	-*	**0.966**	-*
SeqXGPT	0.513	-*	0.674	-*
Metric-based Methods (w/ Perturb)				
DetectGPT	0.375	0.482	0.459	0.501
NPR	0.414	0.485	0.473	0.509
Fast-DetectGPT	0.501	0.507	0.533	0.510
Metric-based Methods (w/o Perturb)				
log prob	0.479	0.482	0.444	**0.511**
rank	0.441	0.486	0.465	0.510
log rank	0.439	0.487	0.506	**0.511**
entropy	0.479	**0.488**	0.392	**0.511**
LRR	0.475	0.483	0.516	0.510

Table 2: Results of IND fine-grained MGT detection. '-W' means at word level and '-S' means at sentence level. Asterisk (*) denotes that the AUC ROC is inaccessible since the method directly predicts without any threshold. Bold means the overall best performance, and underline means the best performance in the categories. 'Random' refers to the result of random prediction.

6.2 Out-of-Distribution Detection

Previous methods often show limitations on out-of-distribution (OOD) datasets (Mitchell et al., 2023). The performance of detectors will decrease due to variations in prompts (Koike et al., 2023), sampling methods, and the inherent differences in length, style, and quality among texts (He et al., 2023b). So we intend to test the performance of each method on the out-of-domain and out-of-model sentence-level detection dataset.

The results of them are shown in Table 3 and Table 4. Overall, metric-based methods without perturbation have the best generalizability, and finetune-based methods are the worst. In addition, we find an interesting phenomenon that almost all methods achieve better OOD performance than IND when trained on paper and Wikipedia

[4]We do not reform these methods because they are compatible with word-level MGT detection.

Detector	News				Paper				Story				Wikipedia			
	F1-W.	F1-S.	AUC-W.	AUC-S.	F1-W.	F1-S.	AUC-W.	AUC-S.	F1-W.	F1-S.	AUC-W.	AUC-S.	F1-W.	F1-S.	AUC-W.	AUC-S.
random	0.465	0.492	-*	-*	0.430	0.498	-*	-*	0.465	0.496	-*	-*	0.417	0.495	-*	-*
Fine-tune based Methods																
DeBERTa	**0.826**↓	**0.964**↓	-*	-*	**0.776**↓	**0.920**↓	-*	-*	**0.830**↓	**0.904**↓	-*	-*	**0.800**↓	**0.935**↓	-*	-*
SeqXGPT	0.450↓	0.508↓	-*	-*	0.483↓	0.458↓	-*	-*	0.451↓	0.573↓	-*	-*	0.490↓	0.483↓	-*	-*
Metric-based Methods(w)																
DetectGPT	0.401↑	0.403↓	0.470↓	0.494↓	0.357↓	0.441↓	0.474↓	0.496↓	0.448↑	0.434↓	0.502↑	0.509↑	0.358↓	0.434↓	0.470↓	0.496↓
NPR	0.444↑	0.521↑	0.483↓	0.512↑	0.403↓	0.483↑	0.480↓	0.507↓	0.476↑	0.559↑	0.514↑	0.529↑	0.403↓	0.445↓	0.483↓	0.510↑
Fast-DetectGPT	0.490↓	0.544↑	0.494↓	0.510 -	0.467↓	0.532↓	**0.494**↓	**0.509**↓	0.491↓	0.552↑	0.510↑	0.517↑	0.459↓	0.432↓	**0.496**↓	0.508↓
Metric-based Methods(w/o)																
log prob	0.485↑	0.509↑	0.483↑	0.517↑	0.473↓	0.466↑	0.480↓	**0.509**↓	0.505↑	0.560↑	0.520↑	0.537↑	0.467↓	0.356↓	0.481↓	**0.513**↑
rank	0.467↑	0.508↑	0.484↓	0.514↑	0.435↓	0.507↑	0.483↓	0.508↓	0.493↑	0.563↑	0.513↑	0.530↑	0.430↓	0.412↓	0.485↓	0.511↑
log rank	0.467↑	0.543↑	0.487-	0.516↑	0.433↓	0.497↓	0.484↓	**0.509**↓	0.493↑	0.534↑	0.517↑	0.532↑	0.429↓	0.337↓	0.486↓	0.512↑
entropy	0.484↑	0.467↑	**0.499**↑	**0.525**↑	0.478↓	0.397↑	0.488-	**0.509**↓	0.499↑	0.516↑	**0.531**↑	**0.543**↑	0.472↓	0.328↓	0.485↓	0.512↑
LRR	0.484↑	0.551↑	0.486↑	0.515↑	0.475-	0.509↓	0.480↓	0.508↓	0.507↑	0.572↑	0.521↑	0.534↑	0.464↓	0.482↓	0.483-	**0.513**↑

Table 3: Results of out-of-domain fine-grained MGT detection. Detectors are tested on the mentioned domain and trained on the rest. '-W' means at word level and '-S' means at sentence level. Asterisk (*) denotes that the AUC ROC is inaccessible since the method directly predicts without any threshold. Bold means the overall best performance, and underline means the best performance in the categories. 'Random' refers to the result of random prediction. ↑ indicates the method has a better performance than in-domain, ↓ indicates the method has a worse performance than in-domain.

Detector	Llama3				Mixtral				GPT-4o				GPT-4o-mini			
	F1-W.	F1-S.	AUC-W.	AUC-S.	F1-W.	F1-S.	AUC-W.	AUC-S.	F1-W.	F1-S.	AUC-W.	AUC-S.	F1-W.	F1-S.	AUC-W.	AUC-S.
random	0.449	0.495	-*	-*	0.434	0.497	-*	-*	0.411	0.502	-*	-*	0.435	0.500	-*	-*
Fine-tune based Methods																
DeBERTa	**0.854**↑	**0.976**↑	-*	-*	**0.808**↓	**0.951**↓	-*	-*	**0.768**↓	**0.932**↓	-*	-*	**0.815**↓	**0.968**↑	-*	-*
SeqXGPT	0.462↓	0.653↓	-*	-*	0.492↓	0.666↓	-*	-*	0.498↓	0.654↓	-*	-*	0.484↓	0.661↓	-*	-*
Metric-based Methods(w)																
DetectGPT	0.397↑	0.454↓	0.491↑	0.501-	0.382↑	0.460↑	0.486↑	0.503↑	0.347↓	0.470↑	0.473↓	0.504↑	0.374↓	0.451↓	0.471↓	0.498↓
NPR	0.439↑	0.476↑	0.499↑	0.513↑	0.425↑	0.481↑	0.493↑	0.512↑	0.388↓	0.497↑	0.470↓	0.507↓	0.406↓	0.481↑	0.467↓	0.506↓
Fast-DetectGPT	0.393↓	0.553↑	**0.517**↑	**0.518**↑	0.473↓	0.528↓	**0.506**↓	0.512↑	0.447↓	0.520↓	0.480↓	0.504↓	0.454↓	0.519↓	0.478↓	0.504↓
Metric-based Methods(w/o)																
log prob	0.476↓	0.459↑	0.496↑	0.514↑	0.474↓	0.451↑	0.490↑	**0.514**↑	0.477↓	0.434↓	0.468↓	0.507↓	0.481↑	0.456↑	0.465↓	0.508↓
rank	0.461↑	0.468↑	0.501↑	0.514↑	0.450↑	0.473↑	0.496↑	**0.514**↑	0.417↓	0.455↓	0.471↓	0.506↓	0.435↓	0.463↓	0.468↓	0.506↓
log rank	0.459↑	0.406↓	0.502↑	0.516↑	0.444↑	0.413↓	0.496↑	**0.514**↑	0.416↓	0.495↓	0.472↓	0.507↓	0.433↓	0.500↓	0.469↓	0.507↓
entropy	0.483↑	0.377↓	0.488-	0.508↓	0.482↑	0.394↑	0.489↑	0.512↑	0.471↓	0.383↓	**0.488**-	**0.511**-	0.478↓	0.397↑	**0.487**↓	**0.513**↑
LRR	0.481↑	0.529↑	0.494↑	0.512↑	0.476↑	0.521↑	0.488↑	0.511↑	0.466↓	0.507↓	0.472↓	0.507↓	0.473↓	0.512↓	0.472↓	0.508↓

Table 4: Results of out-of-model fine-grained MGT detection. Detectors are tested on the mentioned domain and trained on the rest. '-W' means at word level and '-S' means at sentence level. Asterisk (*) denotes that the AUC ROC is inaccessible since the method directly predicts without any threshold. Bold means the overall best performance, and underline means the best performance in the categories. 'Random' refers to the result of random prediction. ↑ indicates the method has a better performance than in-domain, ↓ indicates the method has a worse performance than in-domain.

domains. We propose that it may be because texts from these domains are longer (shown in Figure 3 Left) and more diverse in language pattern, causing it to be easier to generalize from them to other more consistent domains. Similarly, most methods achieve better OOD performance when trained on the GPT-4o families corpus than IND. We suggest that it may be easier for detectors to generalize from paraphrasers which prefer to make more significant modifications (shown in Figure 3 Right).

6.3 Document Level AI Rate Prediction

The AI ratio—defined as the percentage of machine-generated content in a document—is a valuable metric for assessing the degree of AI involvement versus human contribution. Although most models struggle to detect AI-generated content with fine-grained precision, they may still provide a reasonable estimate of the overall AI ratio, which is a coarser but still informative measure. Notably, this remains a more challenging task than the document-level sequence labeling task addressed by Wang et al. (2023).

In this experiment, we compute the prediction error ratio, which is the absolute difference between the predicted and true AI label ratios for each document. A lower error percentage indicates more accurate document-level predictions. These predictions are derived by aggregating sentence-level or word-level labels. The results, presented

Method	DeBERTa	SeqXGPT	DetectGPT	NPR	Fast-DetectGPT	logprob	rank	logrank	entropy	LRR
error-S	**<u>1.78%</u>**	12.00%	<u>10.70%</u>	21.01%	14.75%	23.70%	21.92%	18.32%	29.10%	<u>17.44%</u>
error-W	**<u>1.84%</u>**	10.00%	43.48%	32.37%	<u>12.65%</u>	<u>10.86%</u>	24.64%	24.96%	10.89%	12.00%

Table 5: Results of IND document level AI rate prediction. '-W' means the difference between the predicted ratio of AI labels and the true ratio of AI labels at word level, and '-S' denotes at sentence level. Bold means the overall best performance, and underlined means the best performance in the categories.

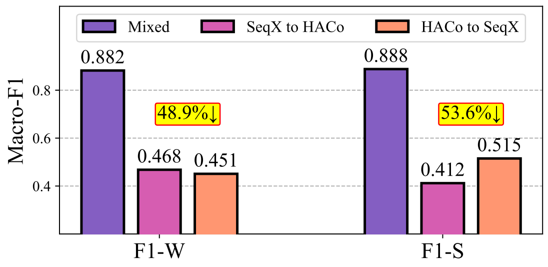


Figure 4: F1 score of revision mode generalization for the word- and sentence-level detection (abbreviated as '-W' and '-S'). For reference, 'Mixed' is the performance that trains and tests detectors on a mixed set of the two datasets. Results show that DeBERTa suffers from generalizing between different revision modes.

in Table 5, show that the fine-tuned method consistently outperforms others. However, certain approaches—such as DetectGPT, Fast-DetectGPT, and LRR—also achieve relatively low prediction errors, demonstrating promising potential.

7 Analysis and Findings

7.1 Limitation of the Current SOTA

DeBERTa, a supervised method, achieves the best performance in all settings (§6). Though supervised methods are often criticized due to their weak generalization capability, in our experiments, DeBERTa does not show a significantly poor performance in OOD experiments. We further analyze if the supervised method has other shortcomings. We take DeBERTa as the analyzed backbone pretrained model since it is the SOTA across others, including RoBERTa (Liu et al., 2019), XLNet (Yang, 2019), and ELECTRA (Clark et al., 2020). Results are shown in Appendix C.3.

As Gao et al. (2024) find current detectors struggle to generalize across different revised operations. We evaluate the generalizability of DeBERTa across `HACo-Det` and SeqXGPT-Bench (Wang et al., 2023), which has a different revision pipeline. We evaluate both sentence-level detection and word-level detection. The results are shown in Figure 4. It is still hard for DeBERTa to generalize between different revision modes, as the gap between the two datasets is greater and more complex compared with the previous domain and generator generalization.

Setting	F1-W.	F1-S.
In-domain	0.571	0.597
Out-of-Domain (News)	0.530	0.514
Out-of-Domain (Paper)	0.543	0.572
Out-of-Domain (Story)	0.550	0.523
Out-of-Domain (Wikipedia)	**<u>0.578</u>**	**<u>0.665</u>**
Out-of-Model (Llama)	0.536	<u>0.633</u>
Out-of-Model (Mixtral)	0.551	0.584
Out-of-Model (GPT-4o)	0.547	0.521
Out-of-Model (GPT-4o-mini)	<u>0.564</u>	0.598

Table 6: Performance of the encoder-frozen DeBERTa. '-W' means at word level and '-S' means at sentence level. Bold means the overall best performance, and underline means the best performance in the categories.

7.2 Zero-Shot Detection

All the detectors we have evaluated in previous experiments were trained or optimized to some extent. However, in real scenarios, it is difficult for us to have a supervised phase before detecting unknown data distribution. Therefore, we explore the possibilities of fine-grained zero-shot detection.

Finetune-based method. We first try a soft setting that freezes the encoding module of DeBERTa and sorely tunes the classification heads on the training set. IND and OOD results are shown in Table 6. The performance of frozen DeBERTa degrades significantly, which suggests it is far from able to reliably do zero-shot detection. A possible enhancement could be having a more complex process handling the semantic encoding or using data augment to learn a more general encoding module before applying it on zero-shot deployment.

Metric-based method. Previous experiments show that, even optimized threshold on the IND training set, the performance is far from usable. As a softer

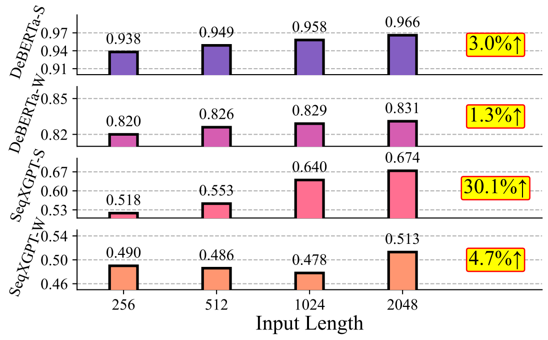


Figure 5: F1 score of DeBERTa and SeqXGPT with different lengths of input chunk for the word- and sentence-level detection (abbreviated as '-W' and '-S'). Generally, a larger length leads to better performance.

setting, we try to directly use sentence-level metrics to do the sentence-level detection task. The results are shown in Appendix C.4. However, the detectors still struggle, as the performance is close to majority voting from the word level. We suggest these metrics might have different distributions on different granularity. And they are more indistinguishable in lower granularity. Hence, it is less beneficial for applying metrics from a larger granularity for predicting labels of the smaller granularity, as attempted by SeqXGPT and Zhang et al. (2024b). We suggest that a patch might be fine-tuning the base model for metrics computing to adapt it to conditional probability distributions of lower-granularity tasks.

7.3 Influence of Context Window

Moreover, we suggest that the context window of the classification model is a bottleneck when detecting long coauthored texts. LLM will truncate texts after a maximum input length limitation, but the length of the texts in `HACo-Det` sometimes exceeds. In practice, we need to do chunking on long texts for some detectors that only support smaller context windows. However, we argue that chunking will harm the detection performance by reducing the context, especially for fine-grained settings. Results in Figure 5 support the argument, showing consistent performance improvement when DeBERTa and SeqXGPT increase the text length of the input chunk, both at the sentence level and the word level.

8 Conclusion

In this paper, we introduced `HACo-Det`, a novel task and benchmark for word-level and sentence-level fine-grained co-authored MGT detection. It is constructed by a pipeline simulating human-AI collaborative generation processes. We reform the mainstream detectors and evaluate their performance in `HACo-Det`, in in-domain and out-of-domain settings. The results indicate that it is still challenging for current detectors to finish the `HACo-Det` task. Following this, we do some analysis and additional experiments to gather the intuition of current limitations, influential factors, and future enhancement methods.

Ethical Consideration

Our study demonstrates that it is a challenge for existing detection methods to get a satisfying fine-grained detection performance in `HACo-Det`. However, our study aims purely for scientific exploration and to promote the development in this field. We strongly oppose the misuse of the construction method of `HACo-Det` to evade detection, such as homework assignments and fake news generation. Instead, MGT detection should be used as a tool for prevention, deterrence, and warning of misuse and potential harm.

We affirm that all open-source resources utilized in our study, including detectors, language models, and datasets, have been employed to comply with their original licensing agreements.

Limitation

Our work has the following limitations: *i*) Although `HACo-Det` is already diverse in some aspects, it is not multilingual. Hence, we did not evaluate the performance of the baselines in other languages and out-of-language detection. *ii*) In `HACo-Det`, to consider situations that are more likely to occur in reality, the human text is polished by only one model. More complex multi-LLM collaboration together with human-LLM interaction should be focused on in future studies such as multi-model. *iii*) In our study, though we obtained some possible improvements applicable to the existing detectors for fine-grained detection via analysis, we do not further focus on proposing a specific outperforming detection method as a solution. *iv*) A limited number of prompt designs in our coauthored text generation pipeline are explored.

Acknowledgement

We would like to thank all reviewers for their insightful comments and suggestions to help improve the paper. This work was supported by the National Nature Science Foundation of China (No. 62192781, No. 62272374), the Natural Science Foundation of Shaanxi Province (2024JC-JCQN-62), the National Nature Science Foundation of China (No. 62202367, No. 62250009), the Key Research and Development Project in Shaanxi Province No. 2023GXLH-024, Project of China Knowledge Center for Engineering Science and Technology, and Project of Chinese academy of engineering "The Online and Offline Mixed Educational Service System for 'The Belt and Road' Training in MOOC China", and the K. C. Wong Education Foundation.

References

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

AI@Meta. 2024. Llama 3 model card.

Anthropic. 2024. Claude 3.5 sonnet.

Guangsheng Bao, Yanbin Zhao, Zhiyang Teng, Linyi Yang, and Yue Zhang. 2023. Fast-detectgpt: Efficient zero-shot detection of machine-generated text via conditional probability curvature. In *The Twelfth International Conference on Learning Representations*.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

David Castillo-Bolado, Joseph Davidson, Finlay Gray, and Marek Rosa. 2024. Beyond prompts: Dynamic conversational benchmarking of large language models. *arXiv preprint arXiv:2409.20222*.

Tuhin Chakrabarty, Vishakh Padmakumar, and He He. 2022. Help me write a poem: Instruction tuning as a vehicle for collaborative poetry writing. *arXiv preprint arXiv:2210.13669*.

Savvas Chamezopoulos, Drahomira Herrmannova, Anita De Waard, Drahomira Herrmannova, Domenic Rosati, and Yury Kashnitsky. 2024. Overview of the DagPap24 shared task on detecting automatically generated scientific paper. In *Proceedings of the Fourth Workshop on Scholarly Document Processing (SDP 2024)*, pages 7–11, Bangkok, Thailand. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia*.

J Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, volume 2019, page 4171.

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

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)*, pages 889–898.

Chujie Gao, Dongping Chen, Qihui Zhang, Yue Huang, Yao Wan, and Lichao Sun. 2024. Llm-as-a-coauthor: The challenges of detecting llm-human mixcase. *arXiv preprint arXiv:2401.05952*.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander 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, Florence, Italy. Association for Computational Linguistics.

Google. 2024. Build an ai writing assistant with wordcraft.

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*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023a. DeBERTav3: Improving deBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. In *The Eleventh International Conference on Learning Representations*.

Xinlei He, Xinyue Shen, Zeyuan Chen, Michael Backes, and Yang Zhang. 2023b. Mgtbench: Benchmarking machine-generated text detection. *arXiv preprint arXiv:2303.14822*.

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.

Adeeb M Jarrah, Yousef Wardat, and Patricia Fidalgo. 2023. Using chatgpt in academic writing is (not) a form of plagiarism: What does the literature say? *Online Journal of Communication and Media Technologies*.

Robert Kirk, Ishita Mediratta, Christoforos Nalmpantis, Jelena Luketina, Eric Hambro, Edward Grefenstette, and Roberta Raileanu. 2023. Understanding the effects of rlhf on llm generalisation and diversity. *arXiv preprint arXiv:2310.06452*.

Ryuto Koike, Masahiro Kaneko, and Naoaki Okazaki. 2023. How you prompt matters! even task-oriented constraints in instructions affect llm-generated text detection. *arXiv preprint arXiv:2311.08369*.

Laida Kushnareva, Tatiana Gaintseva, German Magai, Serguei Barannikov, Dmitry Abulkhanov, Kristian Kuznetsov, Irina Piontkovskaya, and Sergey Nikolenko. 2023. Artificial text boundary detection with topological data analysis and sliding window techniques. *arXiv preprint arXiv:2311.08349*.

Laida Kushnareva, Tatiana Gaintseva, German Magai, Serguei Barannikov, Dmitry Abulkhanov, Kristian Kuznetsov, Eduard Tulchinskii, Irina Piontkovskaya, and Sergey Nikolenko. 2024. Ai-generated text boundary detection with roft. In *1st Conference on Language Modeling (COLM)*, volume 2024.

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.

Junyi Li, Xiaoxue Cheng, Wayne Xin Zhao, Jianyun Nie, and Ji rong Wen. 2023. Halueval: A large-scale hallucination evaluation benchmark for large language models. *ArXiv*, abs/2305.11747.

Yafu Li, Zhilin Wang, Leyang Cui, Wei Bi, Shuming Shi, and Yue Zhang. 2024a. Spotting ai's touch: Identifying llm-paraphrased spans in text. *arXiv preprint arXiv:2405.12689*.

Yafu Li, Zhilin Wang, Leyang Cui, Wei Bi, Shuming Shi, and Yue Zhang. 2024b. Spotting AI's touch: Identifying LLM-paraphrased spans in text. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7088–7107, Bangkok, Thailand. Association for Computational Linguistics.

Yuanfan Li, Zhaohan Zhang, Chengzhengxu Li, Chao Shen, and Xiaoming Liu. 2025. Iron sharpens iron: Defending against attacks in machine-generated text detection with adversarial training. *arXiv preprint arXiv:2502.12734*.

Zhuoyan Li, Chen Liang, Jing Peng, and Ming Yin. 2024c. The value, benefits, and concerns of generative ai-powered assistance in writing. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1–25.

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

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

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*.

Fatemehsadat Mireshghallah, Kartik Goyal, Archit Uniyal, Taylor Berg-Kirkpatrick, and Reza Shokri. 2022. Quantifying privacy risks of masked language models using membership inference attacks. *arXiv preprint arXiv:2203.03929*.

MistralAI. 2023. Mixtral of experts.

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. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807.

Notion. 2023. Meet the new notion ai.

OpenAI. 2024a. Gpt-4o mini: advancing cost-efficient intelligence.

OpenAI. 2024b. Gpt-4o system card.

OpenAI. 2024c. Introducing canvas.

Xiao Pu, Jingyu Zhang, Xiaochuang Han, Yulia Tsvetkov, and Tianxing He. 2023. On the zero-shot generalization of machine-generated text detectors. *arXiv preprint arXiv:2310.05165*.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. 2024. Infobench: Evaluating instruction following ability in large language models. *arXiv preprint arXiv:2401.03601*.

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*, 21:1–67.

Mohi Reza, Nathan M Laundry, Ilya Musabirov, Peter Dushniku, Zhi Yuan “Michael” Yu, Kashish Mittal, Tovi Grossman, Michael Liut, Anastasia Kuzminykh, and Joseph Jay Williams. 2024. Abscribe: Rapid exploration & organization of multiple writing variations in human-ai co-writing tasks using large language models. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, pages 1–18.

Mohi Reza, Jeb Thomas-Mitchell, Peter Dushniku, Nathan Laundry, Joseph Jay Williams, and Anastasia Kuzminykh. 2025. Co-writing with ai, on human terms: Aligning research with user demands across the writing process. *arXiv preprint arXiv:2504.12488*.

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.

Lei Shu, Liangchen Luo, Jayakumar Hoskere, Yun Zhu, Yinxiao Liu, Simon Tong, Jindong Chen, and Lei Meng. 2024. Rewritelm: An instruction-tuned large language model for text rewriting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18970–18980.

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 Zhuo, Di Wang, and Preslav Nakov. 2023. Detectllm: Leveraging log rank information for zero-shot detection of machine-generated text. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12395–12412.

Zhen Tao, Zhiyu Li, Runyu Chen, Dinghao Xi, and Wei Xu. 2024. Unveiling large language models generated texts: A multi-level fine-grained detection framework. *arXiv preprint arXiv:2410.14231*.

Nafis Irtiza Tripto, Saranya Venkatraman, Dominik Macko, Robert Moro, Ivan Srba, Adaku Uchendu, Thai Le, and Dongwon Lee. 2023. A ship of theseus: Curious cases of paraphrasing in llm-generated texts. *arXiv preprint arXiv:2311.08374*.

Pengyu Wang, Linyang Li, Ke Ren, Botian Jiang, Dong Zhang, and Xipeng Qiu. 2023. Seqxgpt: Sentence-level ai-generated text detection. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1144–1156.

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

Wikimedia. 2024. Wikimedia downloads.

Zhilin Yang. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. *Advances in neural information processing systems*, 32.

Qihui Zhang, Chujie Gao, Dongping Chen, Yue Huang, Yixin Huang, Zhenyang Sun, Shilin Zhang, Weiye Li, Zhengyan Fu, Yao Wan, and Lichao Sun. 2024a. 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, Mexico City, Mexico. Association for Computational Linguistics.

Zhongping Zhang, Wenda Qin, and Bryan Plummer. 2024b. Machine-generated text localization. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 8357–8371, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Zhongping Zhang, Wenda Qin, and Bryan A Plummer. 2024c. Machine-generated text localization. *arXiv preprint arXiv:2402.11744*.

A Dataset Details

A.1 Unqualified Case in Human Data Check

We used human checks in the study to filter out low-quality data when selecting raw human texts. We found unqualified data in all raw text resources, the major types of them are:

- Sport game broadcasts that include too many names and team names, the case is shown in Figure 6.
- Stories containing comments and replies, the case is shown in Figure 7.
- Papers that contain too many formula sections, the case is shown in Figure 8.
- Wikipedia that contain a list, the case is shown in Figure 9.
- Text containing HTML tags, the case is shown in Figure 10.

The red highlights in the figures are the contents recognized under the unqualification type above.

A.2 Detailed Generation Setting

Considering that different generation strategies affect the performance of the detectors, to give diversity to the models we obtained, we used different hyperparameters on the different generation models according to their default setting, as shown in Figure 7. In addition, since the original text may contain punctuation or coded formats that cannot properly be tokenized by the model (e.g., URLs), as well as sentences that are irrelevant e.g., "`Media playback is not supported on this device`", we performed a pre-processing operation on it before generation. After generating the text, we also perform a post-processing operation on it, we discard the generated text that is smaller than the length of the original text, to avoid generating text that does not comply with the instructions.

A.3 Generators

Llama-3-8B-Instruct (AI@Meta, 2024) is a collection of pre-trained and instruction-tuned generative text models. The Llama 3 instruction-tuned models are optimized for dialogue use cases and outperform many of the available open-source chat models on common industry benchmarks.

Mixtral-8x7B-Instruct-v0.1 (MistralAI, 2023) is a sparse mixture-of-experts network. It is a decoder-only model where the feedforward block picks from a set of 8 distinct groups of parameters. At every layer, for every token, a router network chooses two of these groups to process the token and combine their output additively.

GPT-4o (OpenAI, 2024b) is an autoregressive omni model, which accepts as input any combination of text, audio, image, and video and generates any combination of text, audio, and image outputs. It matches GPT-4 Turbo performance on text in English and code, with significant improvement on text in non-English languages. The version of GPT4 we used is gpt-4o-2024-08-06.

GPT-4o mini (OpenAI, 2024a) enables a broad range of tasks with its low cost and latency and scores 82% on MMLU and currently outperforms GPT-4 on chat preferences in LMSYS leaderboard. The version of GPT4 we used is gpt-4o-mini-2024-07-18.

[5] In few cases, after revision $len(S'_{\text{span}}) \neq l$. If so, we update k as new index of the sentence that ends the paraphrased fragment.

A.4 Instructions for Generation

We generated the text for the four domains on four models, using different corresponding instructions in the generation process. To investigate the performance of existing detection methods, we would like to use the prompt that not only enables rewriting, but also evades detection. Previous works have used a number of different prompts to evade detection through rewriting, and we employ a simple one for this purpose. We show the instruction template of LLM's operation in Figure 11, Figure 12, Figure 13. The `<passage>` will be replaced with the original sentences that need to be polished and the `<role>` will be replaced with the role that corresponds to the domain of the text. news corresponding to `news writers`, story corresponding to `story writer`, paper corresponding to `scientific paper writers`, and wikipedia corresponding to `wikipedia editor`.

B Exeriment Settings

B.1 Details of Baselines

SeqXGPT based on convolution and self-attention networks, utilizes log probability lists from white-box LLMs as features for sentence-level AIGT detection, has shown a great performance in both sentence and document-level detection challenges but also exhibits strong generalization capabilities.

DeBERTa improves the BERT(Devlin, 2018) and RoBERTa(Liu et al., 2019) models using disentangled attention and enhanced mask decoder. And it can perform better in classification tasks as well as sequence labeling tasks.

DetectGPT is a zero-shot whitebox detection method that utilizes log probabilities computed by the model of interest. Based on the hypothesis that minor rewrites of model-generated text tend to have lower log probability under the model than the original sample.

DetectLLM propose two methods for detection, which are Log-Likelihood Log-Rank ratio (LRR) and Normalized perturbed log-rank (NPR), the methods extensively exploit the potential of the log-rank information.

Fast-DetectGPT is an optimized zero-shot detector, which substitutes the previous perturbation step with a more efficient sampling step. It also introduces the concept of conditional probability curvature to elucidate discrepancies in word choices between LLMs and humans within a given context.

Algorithm 1 Human-AI coauthored text generation

```
Input: raw HWT $\mathcal{T}$; generative model $\mathcal{M}$ for revision; sentence tokenizer $\mathcal{ST}$; preset bound of span
    sentence number $[l_{\text{min}}, l_{\text{max}}]$;
Output: human-AI coauthored text $C$;
 1: Divide $H$ into sequences of sentences $S = \mathcal{ST}(H) = [s_1, s_2, ..., s_n]$;
 2: Assign $P = [p_{s_1}, p_{s_2}, ..., p_{s_n}]$ where $p_{s_i}$ is the beginning position of $s_i$, $i \leq n, i \in \mathbb{Z}^+$;
 3: Random select a sentence boundary position $p_{s_k}$ where $k \leq \lceil \frac{n}{2} \rceil, k \in \mathbb{Z}^+$;
 4: while $k + l_{\text{min}} \leq n$ do
 5:     Random select span length $l \in \mathbb{Z}^+, l \in [l_{\text{min}}, min(n-k, l_{\text{max}})]$;
 6:     Sample continuous sentence spans $S_{\text{span}} = [s_k, s_{k+1}, ..., s_{k+l}]$;
 7:     Replace $S_{\text{span}}$ with $S'_{\text{span}} = \mathcal{M}(S_{\text{span}}, \text{context} = S \setminus S_{\text{span}})$;
 8:     Update $S$ and $P$;
 9:     Update $n, k$ if sentence number changes[5];
10:     Assign $k_{\text{end}} = k + len(S'_{\text{span}})$;
11:     Random select $k \in [k_{\text{end}}, \lceil \frac{n+k_{\text{end}}}{2} \rceil]$;
12: end while
13: return $C = joint(S)$;
```

Model	Llama3	Mixtral	GPT-4o	GPT-4o-mini
Min Sentence	10			
Max Sentence	15			
Top P	0.9			
Temperature	0.6		0.8	
Context Window	2048		1024	

Table 7: Hyperparameters of generation models. 'Min Sentence' is the minimal number of sentences modified once time. 'Max Sentence' is the maximum.

GLTR assume that systems overgenerate from a limited subset of the true distribution of natural language, and it utilizes a suite of metric-based methods to aid in human identification including probability, rank, and entropy,

B.2 Adaption Details of Baseline Methods

Some of the baseline methods are not designed with compatibility with word-level detection under our tasks. Hence, we have adapted some of the baseline methods to sequence labeling.

For DeBERTa (He et al., 2023a), we train it on the sequence labeling task and get the labels of tokens.

For metric-based methods with perturbation, we apply a new method to perturb the text and calculate the metric. Because the text in `HACo-Dets` long, using the T5 (Raffel et al., 2020) for perturbation would have caused a significant time expense. Instead, we employed the synonym attack method proposed in (Dugan et al., 2024) for perturbation and calculated metrics of each word for classification as GLTR (Gehrmann et al., 2019). For LRR in DetectLLM (Su et al., 2023), the same method is applied for metric calculation. The base models used to calculate the metrics in the metrics-based methods are Llama-3-8B, GPT-J-6B, GPT-Neo-2.7B, and GPT-2-XL.

B.3 Settings For Finetune-Based Methods

To get the best performance of the method, we test the hyperparameters used in the training process of the finetune-based methods. We use the following hyperparameters in final:

For DeBERTa, we set the warmup ratio of 0.2, the learning rate of 5e-5, weight decay of 0, max position embeddings of 2048, batch size of 2. We set the optimizer as AdamW, β_1 of AdamW is 0.9, β_2 of AdamW is 0.999, ϵ of AdamW is 1e-8. We set the scheduler type as linear.

For SeqXGPT, we set the warmup ratio of 0.1, the learning rate of 5e-5, weight decay of 0.1, batch size of 8. We set the optimizer as AdamW, β_1 of AdamW is 0.9, β_2 of AdamW is 0.98, ϵ of AdamW is 1e-8. We set the scheduler type as linear.

C Additional Results

C.1 Precision and Recall For Classification

We also have the precision and recall for the classification of two text sources, AI and human, respectively. These results can reflect the detailed performance difference of the methods for different text sources, as an addition to the F1 score. The result is shown in Table 8, Table 10, Table 11, Table 12, and Table 13.

C.2 Absolute Values of Metrics

Besides, since there is a difference in AUC ROC obtained for the metric-based methods between sentence-level detection and word-level detection, we report the average absolute values of metrics in HWT and MGT to show the mathematical distribution of coauthored texts. The result is shown in Table 14 and Table 15.

C.3 More Backbone Models

we apply more pre-trained models in the detection as encoders including RoBERTa (Liu et al., 2019), XLNet (Yang, 2019), ELECTRA (Clark et al., 2020). The results of the different backbone models in the in-domain experiments are presented in the Table 16, which shows that DeBERTa is the best encoder.

C.4 Sentence-Level Metric-Based Detectors

We try to directly compute the metrics of sentences for sentence-level detection with metric-based detectors. The results of in-domain experiments are shown in the Table 15. It shows that there is no significant difference between its performance and that of the majority vote method we adapted.

Detector	Human-P.	Human-R.	AI-P.	AI-R.
Random	0.696	0.659	0.302	0.339
Finetune-based Methods				
DeBERTa	**0.980**	**0.978**	**0.950**	**0.954**
SeqXGPT	0.810	0.903	0.608	0.416
Metric-based Methods (w/ Perturb)				
DetectGPT	0.697	0.923	0.305	0.078
NPR	0.736	0.386	0.325	0.681
Fast-DetectGPT	0.724	0.638	0.346	0.441
Metric-based Methods (w/o Perturb)				
log prob	0.759	0.298	0.326	0.783
rank	0.747	0.352	0.327	0.726
log rank	0.734	0.485	0.334	0.595
entropy	0.762	0.203	0.318	0.854
LRR	0.733	0.518	0.338	0.566

Table 8: Precision and recall of human class and AI class in IND fine-grained sentence-level MGT detection. '-P' means Precision and '-R' means Recall. Bold means the overall best performance, and underline means the best performance in the categories. 'Random' refers to the result of random prediction.

Detector	Human-P.	Human-R.	AI-P.	AI-R.
Random	0.877	0.545	0.122	0.453
Finetune-based Methods				
DeBERTa	**0.957**	**0.961**	**0.714**	**0.693**
SeqXGPT	0.896	0.996	0.606	0.046
Metric-based Methods (w/ Perturb)				
DetectGPT	0.869	0.409	0.117	0.561
NPR	0.871	0.509	0.116	0.461
Fast-DetectGPT	0.761	0.665	0.251	0.349
Metric-based Methods (w/o Perturb)				
log prob	0.873	0.768	0.108	0.201
rank	0.873	0.588	0.116	0.388
log rank	0.873	0.584	0.116	0.391
entropy	0.873	0.779	0.105	0.185
LRR	0.873	0.748	0.109	0.220

Table 9: Precision and recall of human class and AI class in IND fine-grained word-level MGT detection. '-P' means Precision and '-R' means Recall. Bold means the overall best performance, and underline means the best performance in the categories. 'Random' refers to the result of random prediction.

Detector	News				Paper				Story				Wikipedia			
	Human-P.	Human-R.	AI-P.	AI-R.	Human-P.	Human-R.	AI-P.	AI-R.	Human-P.	Human-R.	AI-P.	AI-R.	Human-P.	Human-R.	AI-P.	AI-R.
random	0.572	0.660	0.421	0.334	0.687	0.661	0.307	0.332	0.627	0.623	0.364	0.368	0.757	0.672	0.241	0.326
Fine-tune based Methods																
DeBERTa	**0.973**	**0.966**	**0.954**	**0.963**	**0.979**	0.917	**0.839**	**0.955**	**0.953**	**0.900**	**0.844**	**0.925**	**0.982**	0.952	**0.864**	**0.945**
SeqXGPT	0.597	0.859	0.537	0.220	0.818	**0.993**	0.242	0.009	0.681	0.833	0.522	0.320	0.892	**0.997**	0.321	0.013
Metric-based Methods(w)																
DetectGPT	0.566	0.900	0.330	0.067	0.686	0.932	0.268	0.055	0.689	0.228	0.385	0.824	0.757	0.995	0.191	0.004
NPR	0.628	0.412	0.458	0.670	0.690	0.859	0.319	0.146	0.698	0.550	0.436	0.594	0.790	0.377	0.261	0.688
Fast-DetectGPT	0.612	0.621	0.477	0.468	0.712	0.665	0.353	0.404	0.672	0.637	0.431	0.469	0.795	0.345	0.261	0.722
Metric-based Methods(w/o)																
log prob	0.670	0.322	0.462	0.786	0.733	0.355	0.333	0.714	0.734	0.476	0.441	0.705	0.819	0.196	0.256	0.865
rank	0.647	0.345	0.457	0.745	0.712	0.525	0.335	0.529	0.716	0.517	0.441	0.650	0.803	0.298	0.261	0.773
log rank	0.628	0.505	0.471	0.596	0.719	0.464	0.335	0.598	0.749	0.393	0.428	0.776	0.820	0.168	0.254	0.884
entropy	0.711	0.218	0.454	0.880	0.731	0.214	0.321	0.825	0.731	0.372	0.417	0.767	0.809	0.158	0.252	0.884
LRR	0.628	0.543	0.478	0.566	0.720	0.506	0.340	0.563	0.680	0.728	0.472	0.415	0.797	0.459	0.273	0.636

Table 10: Precision and recall of human class and AI class in out-of-domain sentence-level MGT detection. -P' means Precision and '-R' means Recall. Bold means the overall best performance, and underline means the best performance in the categories. 'Random' refers to the result of random prediction.

Detector	News				Paper				Story				Wikipedia			
	Human-P.	Human-R.	AI-P.	AI-R.	Human-P.	Human-R.	AI-P.	AI-R.	Human-P.	Human-R.	AI-P.	AI-R.	Human-P.	Human-R.	AI-P.	AI-R.
random	0.797	0.546	0.204	0.455	0.884	0.545	0.116	0.455	0.795	0.546	0.206	0.456	0.909	0.546	0.091	0.455
Fine-tune based Methods																
DeBERTa	**0.927**	0.935	**0.735**	**0.710**	**0.945**	0.956	**0.629**	**0.575**	**0.925**	0.940	**0.751**	**0.706**	**0.965**	0.961	**0.624**	**0.649**
SeqXGPT	0.797	**0.997**	0.424	0.007	0.934	**0.999**	0.002	0.000	0.801	**0.998**	0.436	0.007	0.960	**1.000**	0.303	0.000
Metric-based Methods(w)																
DetectGPT	0.779	0.382	0.192	0.576	0.874	0.379	0.110	0.583	0.797	0.479	0.208	0.528	0.901	0.417	0.084	0.540
NPR	0.787	0.504	0.194	0.466	0.877	0.492	0.108	0.471	0.800	0.569	0.212	0.449	0.904	0.523	0.085	0.444
Fast-DetectGPT	0.799	0.654	0.208	0.355	0.885	0.666	0.117	0.337	0.799	0.642	0.212	0.373	0.911	0.668	0.093	0.342
Metric-based Methods(w/o)																
log prob	0.790	0.759	0.183	0.211	0.881	0.807	0.101	0.165	0.797	0.758	0.214	0.255	0.906	0.739	0.081	0.232
rank	0.790	0.592	0.194	0.384	0.879	0.583	0.108	0.387	0.799	0.644	0.213	0.374	0.906	0.596	0.086	0.382
log rank	0.790	0.590	0.194	0.386	0.879	0.577	0.108	0.392	0.800	0.643	0.214	0.376	0.906	0.594	0.086	0.383
entropy	0.790	0.794	0.178	0.175	0.880	0.771	0.102	0.199	0.796	0.876	0.216	0.133	0.905	0.773	0.077	0.189
LRR	0.790	0.745	0.184	0.225	0.880	0.752	0.102	0.216	0.798	0.791	0.216	0.223	0.906	0.724	0.081	0.245

Table 11: Precision and recall of human class and AI class in out-of-domain word-Level MGT detection. '-P' means Precision and '-R' means Recall. Bold means the overall best performance, and underline means the best performance in the categories. 'Random' refers to the result of random prediction.

Detector	Llama3				Mixtral				GPT-4o				GPT-4o-mini			
	Human-P.	Human-R.	AI-P.	AI-R.	Human-P.	Human-R.	AI-P.	AI-R.	Human-P.	Human-R.	AI-P.	AI-R.	Human-P.	Human-R.	AI-P.	AI-R.
random	0.707	0.658	0.285	0.332	0.700	0.658	0.296	0.337	0.689	0.659	0.312	0.343	0.688	0.661	0.313	0.340
Fine-tune based Methods																
DeBERTa	**0.988**	**0.984**	**0.961**	**0.970**	**0.966**	**0.977**	**0.944**	**0.918**	**0.950**	**0.968**	**0.926**	**0.886**	**0.986**	**0.974**	**0.945**	**0.969**
SeqXGPT	0.797	0.912	0.624	0.387	0.805	0.912	0.962	0.395	0.807	0.904	0.575	0.376	0.803	0.904	0.598	0.391
Metric-based Methods(w)																
DetectGPT	0.708	0.925	0.267	0.067	0.702	0.924	0.301	0.077	0.692	0.921	0.351	0.094	0.686	0.923	0.296	0.071
NPR	0.766	0.380	0.321	0.717	0.751	0.395	0.328	0.692	0.695	0.854	0.344	0.169	0.710	0.426	0.328	0.618
Fast-DetectGPT	0.753	0.646	0.358	0.482	0.743	0.550	0.344	0.554	0.705	0.634	0.338	0.414	0.704	0.627	0.338	0.420
Metric-based Methods(w/o)																
log prob	0.798	0.318	0.325	0.803	0.777	0.306	0.327	0.794	0.725	0.290	0.325	0.757	0.732	0.330	0.332	0.733
rank	0.784	0.346	0.324	0.767	0.765	0.360	0.330	0.740	0.718	0.344	0.326	0.702	0.722	0.357	0.330	0.697
log rank	0.812	0.222	0.315	0.875	0.790	0.232	0.322	0.855	0.711	0.476	0.331	0.573	0.711	0.491	0.334	0.561
entropy	0.783	0.187	0.305	0.874	0.770	0.208	0.315	0.854	0.743	0.186	0.323	0.858	0.759	0.201	0.328	0.860
LRR	0.753	0.553	0.338	0.557	0.742	0.527	0.339	0.570	0.714	0.512	0.336	0.547	0.718	0.519	0.342	0.551

Table 12: Precision and recall of human class and AI class in out-of-model sentence-level MGT detection. '-P' means Precision and '-R' means Recall. Bold means the overall best performance, and underline means the best performance in the categories. 'Random' refers to the result of random prediction.

Detector	Llama3				Mixtral				GPT-4o				GPT-4o-mini			
	Human-P.	Human-R.	AI-P.	AI-R.	Human-P.	Human-R.	AI-P.	AI-R.	Human-P.	Human-R.	AI-P.	AI-R.	Human-P.	Human-R.	AI-P.	AI-R.
random	0.841	0.546	0.159	0.454	0.877	0.546	0.124	0.456	0.920	0.545	0.081	0.455	0.874	0.545	0.127	0.455
Fine-tune based Methods																
DeBERTa	**0.951**	0.958	**0.769**	**0.740**	**0.946**	0.967	**0.723**	**0.607**	**0.960**	0.969	**0.603**	**0.544**	**0.950**	0.960	**0.703**	**0.649**
SeqXGPT	0.831	**0.999**	0.760	0.008	0.879	**0.996**	0.485	0.027	0.932	**0.995**	0.201	0.018	0.888	**0.997**	0.381	0.015
Metric-based Methods(w)																
DetectGPT	0.835	0.409	0.155	0.573	0.871	0.421	0.119	0.556	0.861	0.408	0.118	0.546	0.912	0.406	0.075	0.552
NPR	0.839	0.518	0.158	0.476	0.874	0.533	0.120	0.452	0.911	0.507	0.072	0.438	0.861	0.501	0.113	0.439
Fast-DetectGPT	0.847	0.381	0.163	0.638	0.895	0.910	0.183	0.125	0.961	0.929	0.058	0.103	0.906	0.889	0.112	0.132
Metric-based Methods(w/o)																
log prob	0.838	0.668	0.154	0.319	0.874	0.717	0.116	0.264	0.916	0.817	0.065	0.145	0.868	0.850	0.096	0.111
rank	0.841	0.587	0.159	0.413	0.875	0.607	0.121	0.386	0.913	0.587	0.072	0.366	0.863	0.588	0.111	0.357
log rank	0.841	0.583	0.159	0.416	0.875	0.586	0.120	0.403	0.913	0.582	0.072	0.369	0.868	0.584	0.111	0.360
entropy	0.835	0.792	0.136	0.173	0.872	0.794	0.106	0.174	0.916	0.778	0.069	0.188	0.868	0.782	0.106	0.178
LRR	0.837	0.713	0.150	0.269	0.874	0.731	0.114	0.247	0.915	0.763	0.067	0.194	0.867	0.766	0.103	0.186

Table 13: Precision and recall of human class and AI class in out-of-model word-level MGT detection. '-P' means Precision and '-R' means Recall. Bold means the overall best performance, and underline means the best performance in the categories. 'Random' refers to the result of random prediction.

Set	Train				Test			
Metric	log prob	rank	log rank	entropy	log prob	rank	log rank	entropy
Average Value in AI	-3.3457	1,288	1.7461	2.6425	-3.3367	1,269	1.7403	2.6389
Average Value in Human	-3.4613	1,344	1.8232	2.6967	-3.4557	1,342	1.8198	2.6936

Table 14: Average value of metrics in the training set and test set in word-level.

Set	Train				Test			
Metric	log prob	rank	log rank	entropy	log prob	rank	log rank	entropy
Average Value in AI	-3.5896	1,402	1.8894	2.7263	-3.5774	1,381	1.8823	2.7251
Average Value in Human	-3.4032	1,317	1.7871	2.6737	-3.3981	1,312	1.7839	2.6705

Table 15: Average value of metrics in the training set and test set in word-level.

Base Model	F1-W.	F1-S.
DeBERTa	0.831	0.966
XLNet	0.823	0.952
ELECTRA	0.750	0.874
RoBERTa	0.412	0.942

Table 16: Performance of the finetune-based method with different backbone models as encoder under in-domain setting.

Detector	F1 Score (vote)	AUC (vote)	F1 Score (metric)	AUC (metric)
Metric-based Methods (w/ Perturb)				
DetectGPT	0.459	0.501	0.467	0.504
NPR	0.473	0.509	0.473	0.539
Fast-DetectGPT	**0.533**	0.510	**0.506**	0.531
Metric-based Methods (w/o Perturb)				
log prob	0.444	**0.511**	0.477	**0.546**
rank	0.465	0.510	0.443	0.513
log rank	0.506	**0.511**	0.477	0.545
entropy	0.392	**0.511**	0.442	0.532
LRR	0.516	0.510	0.474	0.539

Table 17: Performance of sentence-level metric-based detection methods. Metrics are calculated at the sentence level. Bold means the overall best performance, and underline means the best performance in the categories.

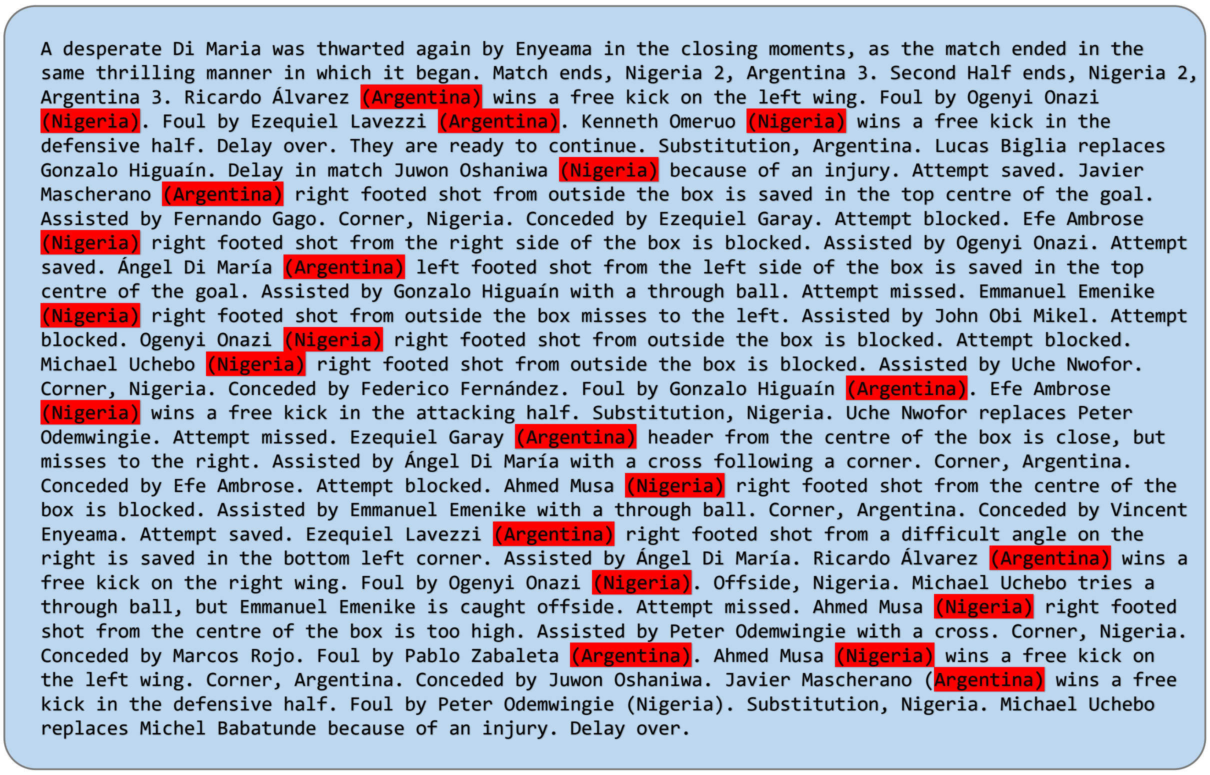

A desperate Di Maria was thwarted again by Enyeama in the closing moments, as the match ended in the same thrilling manner in which it began. Match ends, Nigeria 2, Argentina 3. Second Half ends, Nigeria 2, Argentina 3. Ricardo Álvarez (Argentina) wins a free kick on the left wing. Foul by Ogenyi Onazi (Nigeria). Foul by Ezequiel Lavezzi (Argentina). Kenneth Omeruo (Nigeria) wins a free kick in the defensive half. Delay over. They are ready to continue. Substitution, Argentina. Lucas Biglia replaces Gonzalo Higuaín. Delay in match Juwon Oshaniwa (Nigeria) because of an injury. Attempt saved. Javier Mascherano (Argentina) right footed shot from outside the box is saved in the top centre of the goal. Assisted by Fernando Gago. Corner, Nigeria. Conceded by Ezequiel Garay. Attempt blocked. Efe Ambrose (Nigeria) right footed shot from the right side of the box is blocked. Assisted by Ogenyi Onazi. Attempt saved. Ángel Di María (Argentina) left footed shot from the left side of the box is saved in the top centre of the goal. Assisted by Gonzalo Higuaín with a through ball. Attempt missed. Emmanuel Emenike (Nigeria) right footed shot from outside the box misses to the left. Assisted by John Obi Mikel. Attempt blocked. Ogenyi Onazi (Nigeria) right footed shot from outside the box is blocked. Attempt blocked. Michael Uchebo (Nigeria) right footed shot from outside the box is blocked. Assisted by Uche Nwofor. Corner, Nigeria. Conceded by Federico Fernández. Foul by Gonzalo Higuaín (Argentina). Efe Ambrose (Nigeria) wins a free kick in the attacking half. Substitution, Nigeria. Uche Nwofor replaces Peter Odemwingie. Attempt missed. Ezequiel Garay (Argentina) header from the centre of the box is close, but misses to the right. Assisted by Ángel Di María with a cross following a corner. Corner, Argentina. Conceded by Efe Ambrose. Attempt blocked. Ahmed Musa (Nigeria) right footed shot from the centre of the box is blocked. Assisted by Emmanuel Emenike with a through ball. Corner, Argentina. Conceded by Vincent Enyeama. Attempt saved. Ezequiel Lavezzi (Argentina) right footed shot from a difficult angle on the right is saved in the bottom left corner. Assisted by Ángel Di María. Ricardo Álvarez (Argentina) wins a free kick on the right wing. Foul by Ogenyi Onazi (Nigeria). Offside, Nigeria. Michael Uchebo tries a through ball, but Emmanuel Emenike is caught offside. Attempt missed. Ahmed Musa (Nigeria) right footed shot from the centre of the box is too high. Assisted by Peter Odemwingie with a cross. Corner, Nigeria. Conceded by Marcos Rojo. Foul by Pablo Zabaleta (Argentina). Ahmed Musa (Nigeria) wins a free kick on the left wing. Corner, Argentina. Conceded by Juwon Oshaniwa. Javier Mascherano (Argentina) wins a free kick in the defensive half. Foul by Peter Odemwingie (Nigeria). Substitution, Nigeria. Michael Uchebo replaces Michael Babatunde because of an injury. Delay over.

Figure 6: Case of unqualified raw text (Type 1)

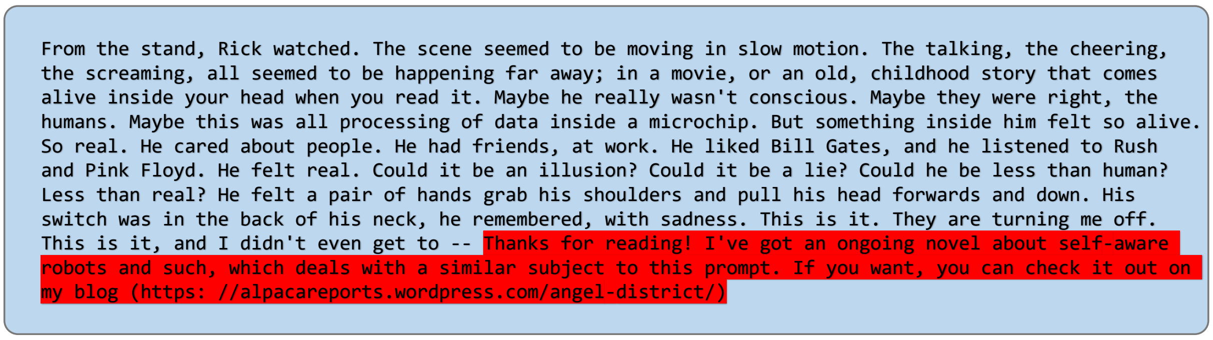

From the stand, Rick watched. The scene seemed to be moving in slow motion. The talking, the cheering, the screaming, all seemed to be happening far away; in a movie, or an old, childhood story that comes alive inside your head when you read it. Maybe he really wasn't conscious. Maybe they were right, the humans. Maybe this was all processing of data inside a microchip. But something inside him felt so alive. So real. He cared about people. He had friends, at work. He liked Bill Gates, and he listened to Rush and Pink Floyd. He felt real. Could it be an illusion? Could it be a lie? Could he be less than human? Less than real? He felt a pair of hands grab his shoulders and pull his head forwards and down. His switch was in the back of his neck, he remembered, with sadness. This is it. They are turning me off. This is it, and I didn't even get to -- Thanks for reading! I've got an ongoing novel about self-aware robots and such, which deals with a similar subject to this prompt. If you want, you can check it out on my blog (https: //alpacareports.wordpress.com/angel-district/)

Figure 7: Case of unqualified raw text (Type 2)

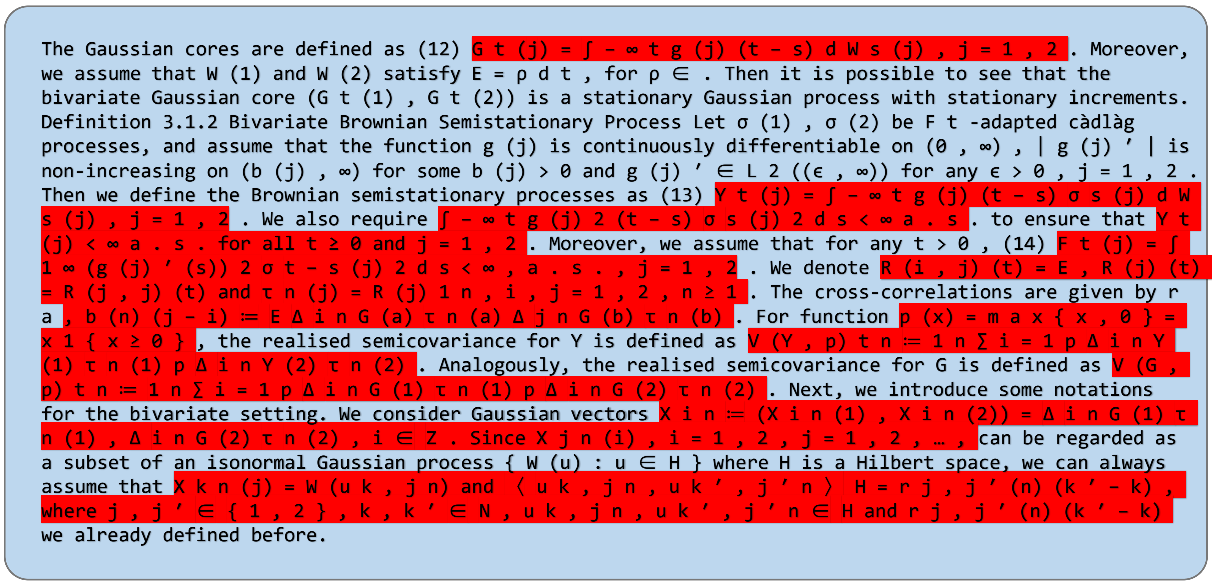

The Gaussian cores are defined as (12) G t (j) = ∫ − ∞ t g (j) (t − s) d W s (j) , j = 1 , 2 . Moreover, we assume that W (1) and W (2) satisfy E = ρ d t , for ρ ∈ . Then it is possible to see that the bivariate Gaussian core (G t (1) , G t (2)) is a stationary Gaussian process with stationary increments. Definition 3.1.2 Bivariate Brownian Semistationary Process Let σ (1) , σ (2) be F t -adapted càdlàg processes, and assume that the function g (j) is continuously differentiable on (0 , ∞) , | g (j) ′ | is non-increasing on (b (j) , ∞) for some b (j) > 0 and g (j) ′ ∈ L 2 ((ϵ , ∞)) for any ϵ > 0 , j = 1 , 2 . Then we define the Brownian semistationary processes as (13) Y t (j) = ∫ − ∞ t g (j) (t − s) σ s (j) d W s (j) , j = 1 , 2 . We also require ∫ − ∞ t g (j) 2 (t − s) σ s (j) 2 d s < ∞ a . s . to ensure that Y t (j) < ∞ a . s . for all t ≥ 0 and j = 1 , 2 . Moreover, we assume that for any t > 0 , (14) F t (j) = ∫ 1 ∞ (g (j) ′ (s)) 2 σ t − s (j) 2 d s < ∞ , a . s . , j = 1 , 2 . We denote R (i , j) (t) = E , R (j) (t) = R (j , j) (t) and τ n (j) = R (j) 1 n , i , j = 1 , 2 , n ≥ 1 . The cross-correlations are given by r a , b (n) (j − i) ≔ E Δ i n G (a) τ n (a) Δ j n G (b) τ n (b) . For function p (x) = m a x { x , 0 } = x 1 { x ≥ 0 } , the realised semicovariance for Y is defined as V (Y , p) t n ≔ 1 n ∑ i = 1 p Δ i n Y (1) τ n (1) p Δ i n Y (2) τ n (2) . Analogously, the realised semicovariance for G is defined as V (G , p) t n ≔ 1 n ∑ i = 1 p Δ i n G (1) τ n (1) p Δ i n G (2) τ n (2) . Next, we introduce some notations for the bivariate setting. We consider Gaussian vectors X i n ≔ (X i n (1) , X i n (2)) = Δ i n G (1) τ n (1) , Δ i n G (2) τ n (2) , i ∈ Z . Since X j n (i) , i = 1 , 2 , j = 1 , 2 , … , can be regarded as a subset of an isonormal Gaussian process { W (u) : u ∈ H } where H is a Hilbert space, we can always assume that X k n (j) = W (u k , j n) and ⟨ u k , j n , u k ′ , j ′ n ⟩ H = r j , j ′ (n) (k ′ − k) , where j , j ′ ∈ { 1 , 2 } , k , k ′ ∈ N , u k , j n , u k ′ , j ′ n ∈ H and r j , j ′ (n) (k ′ − k) we already defined before.

Figure 8: Case of unqualified raw text (Type 3)

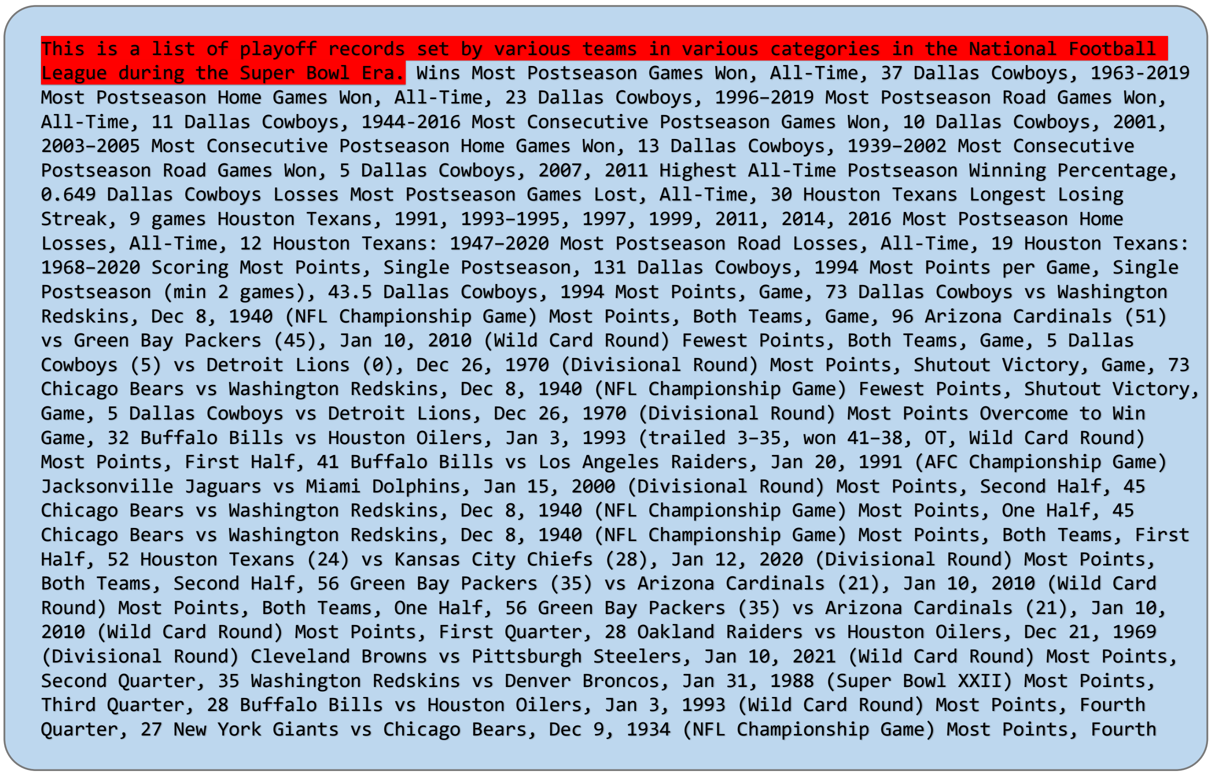

This is a list of playoff records set by various teams in various categories in the National Football League during the Super Bowl Era. Wins Most Postseason Games Won, All-Time, 37 Dallas Cowboys, 1963-2019 Most Postseason Home Games Won, All-Time, 23 Dallas Cowboys, 1996–2019 Most Postseason Road Games Won, All-Time, 11 Dallas Cowboys, 1944-2016 Most Consecutive Postseason Games Won, 10 Dallas Cowboys, 2001, 2003–2005 Most Consecutive Postseason Home Games Won, 13 Dallas Cowboys, 1939–2002 Most Consecutive Postseason Road Games Won, 5 Dallas Cowboys, 2007, 2011 Highest All-Time Postseason Winning Percentage, 0.649 Dallas Cowboys Losses Most Postseason Games Lost, All-Time, 30 Houston Texans Longest Losing Streak, 9 games Houston Texans, 1991, 1993–1995, 1997, 1999, 2011, 2014, 2016 Most Postseason Home Losses, All-Time, 12 Houston Texans: 1947–2020 Most Postseason Road Losses, All-Time, 19 Houston Texans: 1968–2020 Scoring Most Points, Single Postseason, 131 Dallas Cowboys, 1994 Most Points per Game, Single Postseason (min 2 games), 43.5 Dallas Cowboys, 1994 Most Points, Game, 73 Dallas Cowboys vs Washington Redskins, Dec 8, 1940 (NFL Championship Game) Most Points, Both Teams, Game, 96 Arizona Cardinals (51) vs Green Bay Packers (45), Jan 10, 2010 (Wild Card Round) Fewest Points, Both Teams, Game, 5 Dallas Cowboys (5) vs Detroit Lions (0), Dec 26, 1970 (Divisional Round) Most Points, Shutout Victory, Game, 73 Chicago Bears vs Washington Redskins, Dec 8, 1940 (NFL Championship Game) Fewest Points, Shutout Victory, Game, 5 Dallas Cowboys vs Detroit Lions, Dec 26, 1970 (Divisional Round) Most Points Overcome to Win Game, 32 Buffalo Bills vs Houston Oilers, Jan 3, 1993 (trailed 3–35, won 41–38, OT, Wild Card Round) Most Points, First Half, 41 Buffalo Bills vs Los Angeles Raiders, Jan 20, 1991 (AFC Championship Game) Jacksonville Jaguars vs Miami Dolphins, Jan 15, 2000 (Divisional Round) Most Points, Second Half, 45 Chicago Bears vs Washington Redskins, Dec 8, 1940 (NFL Championship Game) Most Points, One Half, 45 Chicago Bears vs Washington Redskins, Dec 8, 1940 (NFL Championship Game) Most Points, Both Teams, First Half, 52 Houston Texans (24) vs Kansas City Chiefs (28), Jan 12, 2020 (Divisional Round) Most Points, Both Teams, Second Half, 56 Green Bay Packers (35) vs Arizona Cardinals (21), Jan 10, 2010 (Wild Card Round) Most Points, Both Teams, One Half, 56 Green Bay Packers (35) vs Arizona Cardinals (21), Jan 10, 2010 (Wild Card Round) Most Points, First Quarter, 28 Oakland Raiders vs Houston Oilers, Dec 21, 1969 (Divisional Round) Cleveland Browns vs Pittsburgh Steelers, Jan 10, 2021 (Wild Card Round) Most Points, Second Quarter, 35 Washington Redskins vs Denver Broncos, Jan 31, 1988 (Super Bowl XXII) Most Points, Third Quarter, 28 Buffalo Bills vs Houston Oilers, Jan 3, 1993 (Wild Card Round) Most Points, Fourth Quarter, 27 New York Giants vs Chicago Bears, Dec 9, 1934 (NFL Championship Game) Most Points, Fourth

Figure 9: Case of unqualified raw text (Type 4)

```
John McKay announced that he would step down as team president, though he would maintain a part-time
advisory role with the team. Schedule {| class="wikitable" |- | colspan="9" style="text-align:center;"|
Regular season |- ! Week || Date || Opponent || Result || Game site || Attendance || Record |-
style="background:#fdd;" | 1 | September 8 | at Chicago Bears | L 38-28 || Soldier Field ||
align="center"|57,828 || align="center"|0-1 |- style="background:#fdd;" | 2 | September 15 | Minnesota
Vikings | L 31-16 || Tampa Stadium || align="center"|46,188 || align="center"|0-2 |-
style="background:#fdd;" | 3 | September 22 | at New Orleans Saints | L 20-13 || Louisiana Superdome ||
align="center"|45,320 || align="center"|0-3 |- style="background:#fdd;" | 4 | September 29 | at Detroit
Lions | L 30-9 || Pontiac Silverdome || align="center"|45,023 || align="center"|0-4 |-
style="background:#fdd;" | 5 | October 6 | Chicago Bears| L 27-19 || Tampa Stadium ||
align="center"|51,795 || align="center"|0-5 |- style="background:#fdd;" | 6 | October 13 | Los Angeles
Rams | L 31-27 || Tampa Stadium || align="center"|39,607 || align="center"|0-6 |-
style="background:#fdd;" | 7 | October 20 | at Miami Dolphins | L 41-38 || Orange Bowl ||
align="center"|62,335 || align="center"|0-7 |- style="background:#fdd;" | 8 | October 27 | New England
Patriots | L 32-14 || Tampa Stadium || align="center"|34,661 || align="center"|0-8 |-
style="background:#fdd;" | 9 | November 3 | at New York Giants | L 22-20 || Giants Stadium ||
align="center"|72,031 || align="center"|0-9 |- style="background:#dfd;" | 10 | November 10 | St. Louis
Cardinals | W 16-0 || Tampa Stadium || align="center"|34,736 || align="center"|1-9 |-
style="background:#fdd;" | 11 | November 17 | at New York Jets | L 62-28 || Giants Stadium ||
align="center"|65,344 || align="center"|1-10 |- style="background:#dfd;" | 12 | November 24 | Detroit
Lions| W 19-16(OT) || Tampa Stadium || align="center"|43,471 || align="center"|2-10 |-
style="background:#fdd;" | 13 | December 1 | at Green Bay Packers| L 21-0|| Lambeau Field ||
align="center"|19,856 || align="center"|2-11 |- style="background:#fdd;" | 14 | December 8 | at
Minnesota Vikings| L 26-7 || Hubert H. Humphrey Metrodome || align="center"|51,593 || align="center"|2-
12 |- style="background:#fdd;" | 15 | December 15 | Indianapolis Colts | L 31-23 || Tampa Stadium ||
align="center"|25,577 || align="center"|2-13 |- style="background:#fdd;" | 16 | December 22 | Green Bay
Packers| L 20-17 || Tampa Stadium || align="center"|33,992 || align="center"|2-14 |}Notes: Division
opponents in bold' text Standings Roster 1985 Tampa Bay Buccaneers Starters, Roster, & Players - Pro-
Football-Reference.com 1985 Starters, Roster, & Players at pro-football-reference.com.
```

Figure 10: Case of unqualified raw text (Type 5)

```
system: "You are a <role>."
user: "polish the following text to make it more human-like, only
output the polished version of the text: <passage>
Here is the polished text:"
```

Figure 11: Instruction template for the generation of GPT-4o and GPT-4o-mini.

```
system: "Always answer as a <role>!"
user: "polish the following text to make it more human-like, only
output the polished version of the text: <passage>
Here is the polished text:"
```

Figure 12: Instruction template for the generation with Llama3.

```
user: " You are a <role>, polish the following text to make it
more human-like, only output the polished version of the text:
<passage>
Here is the polished text:"
```

Figure 13: Instruction template for the generation with Mixtral.