Beemo: Benchmark of Expert-edited Machine-generated Outputs

Anonymous ACL submission

Abstract

The rapid proliferation of large language models (LLMs) has increased the volume of machine-generated texts (MGTs) and blurred text authorship in various domains. However, most existing MGT benchmarks include *single-author* texts (human-written & machine-generated). This conventional design fails to capture more practical *multi-author* scenarios, where the user refines the LLM response for natural flow, coherence, and factual correctness. Our paper introduces the **B**enchmark of **E**xpert-**e**dited **M**achine-generated **O**utputs (Beemo), which includes 6,585 texts written by humans, generated by ten instruction-finetuned LLMs and edited by experts for various use cases, ranging from creative writing to summarization. Beemo additionally comprises 13,170 machine-generated & LLM-edited texts, which allows for diverse MGT detection evaluation across various edit types. We document the Beemo's creation protocol and present the results of benchmarking 33 configurations of MGT detectors in different experimental setups. We find that expert-based editing evades MGT detection, while LLM-edited texts are unlikely to be recognized as human-written. Beemo and all materials will be released upon acceptance.

1 Introduction

The rapid advancement of large language models (LLMs) has significantly improved their ability to assist users with a wide range of writing tasks (Gero et al., 2022; Yang et al., 2022; Chakrabarty et al., 2024b). While the benefits of LLMs are commendable, their widespread adoption has raised concerns regarding authenticity of textual content and potential malicious uses within the domains of news, social media, science, and education (Lucas et al., 2023; Crothers et al., 2023; Gupta et al., 2024; Chamezopoulos et al., 2024; Tang et al., 2024).

Over the last few years, a broad range of machine-generated text (MGT) detection benchmarks have been created to facilitate the development of reliable detectors, aimed at mitigating the risks associated with the misuse of LLMs across different domains and languages (e.g., Macko et al., 2023, 2024; Pu et al., 2023; Dugan et al., 2024; Tripto et al., 2024; Wang et al., 2024c,b). However, most of them focus on a *single-author* scenario, comprising only machine-generated and human-written texts. This well-established design does not account for common applications of LLMs, where the user refines the LLM's response for natural flow, coherence, and factual correctness.

This paper introduces the **B**enchmark of **E**xpert-**e**dited **M**achine-generated **O**utputs (Beemo[1]), which consists of 6,585 texts written by humans, generated by ten open-source instruction-finetuned LLMs and edited by expert annotators, who are well-experienced in refining LLM-generated content. Furthermore, each MGT is edited by two state-of-the-art LLMs using several diverse editing prompts, which results in 13,170 machine-generated & LLM-edited texts. Beemo covers five use cases: open-ended generation, rewriting, summarization, and open & closed question answering. Our design enables various diagnostic evaluation scenarios, ranging from out-of-domain MGT detection to analyzing how an MTG detector's behavior changes after the LLM response is refined by experts and "humanized" by other LLMs.

The main contributions are summarized as follows: (i) we create Beemo, one of the first *multi-author* benchmarks of LLM-generated & expert-edited responses for fine-grained MGT detection, which counts 19,755 texts in total; (ii) we evaluate 33 configurations of zero-shot and pretrained MGT detectors; (iii) we release Beemo[2], our codebase, and all annotation materials.

[1] Our benchmark is named after BMO (abbreviated from "Be MOre", phonetically spelled "Beemo"), one of the main characters of Adventure Time. Logo: `slackmojis.com/bmo`.

[2] The links are removed for anonymity purposes.

Name	Size	# Models	Machine-generated & LLM-edited	Machine-generated & Human-edited
TuringBench (Uchendu et al., 2021)	200k	19	✗	✗
MULTITuDE (Macko et al., 2023)	75k	8	✗	✗
M4 (Wang et al., 2024c)	122k	6	✗	✗
M4GT-Bench (Wang et al., 2024b)	217k	8	✗	✗
HC3 Plus (Su et al., 2023b)	210k	1	✗	✗
MAGE (Li et al., 2024)	447k	27	✗	✗
RAID (Dugan et al., 2024)	6.2M	11	✗	✗
RoFT (Dugan et al., 2023)	21k	2	✗	✗
OpenLLMText (Chen et al., 2023)	340k	5	✗	✗
MultiSocial (Macko et al., 2024)	472k	7	✗	✗
BUST (Cornelius et al., 2024)	25k	7	✗	✗
MixSet (Zhang et al., 2024a)	3.6k	8	✓	✓
LLM-DetectAIve (Abassy et al., 2024)	303k	14	✓	✗
LAMP (Chakrabarty et al., 2024a)	1k	7	✓	✓
Beemo (ours)	19.7k	10	✓	✓

Table 1: Comparison of publicly available monolingual and cross-lingual MGT detection evaluation resources. Beemo is one of the first benchmarks that contains machine-generated & LLM/human-edited texts.

2 Related Work

Table 1 summarizes commonly used monolingual and cross-lingual MGT detection evaluation resources. To the best of our knowledge, only three of them comprise machine-generated & LLM-edited texts, and only two of them include machine-generated & human-edited texts. Below, we provide an overview of the resources by their type and task formulation with the main focus on English.

2.1 Standard MGT Detection Benchmarks

MGT detection features various task formulations and labelling schemes: binary classification, neural authorship attribution, boundary-detection, and multi-author classification.

Binary Classification Binary classification is a well-established design of MGT benchmarks (Radford et al., 2018; Fagni et al., 2021; Liyanage et al., 2022; Macko et al., 2023; Cui et al., 2023b). The task is to determine if a given text is machine-generated or not.

Authorship Attribution Authorship Attribution aims to identify the author of a given text (Uchendu et al., 2020, 2021, 2023), a multi-class classification task with humans and LLMs as the labels.

Boundary Detection Boundary detection is a less explored task formulation, which aligns with application of LLMs for text continuation tasks. Here, the goal is to detect a change point in the text, where a natural text transitions into a neural one (Dugan et al., 2023; Wang et al., 2024b).

Multi-author Classification Recent research has proposed several fine-grained MGT benchmarks, which help to explore how varying degrees of LLM intervention in writing tasks affect the behavior of MGT detectors. MixSet (Zhang et al., 2024a) comprises human-written, machine-generated, and human/LLM-refined MGTs and focuses on multi-author binary classification. LLM-DetectAIve (Abassy et al., 2024) offers a four-way classification task with two other classes (human-written/machine-generated & machine-polished), which reflects a common usage of LLMs for enhancing a human-written. LAMP (Chakrabarty et al., 2024a) consists of 1k non- and fiction MGTs edited by professionals and LLMs and explores automatic detection and rewriting of "problematic" spans in MGTs. Our work differs from these studies in the following aspects: (i) in line with Zhang et al.; Chakrabarty et al., we present one of the first attempts to create a benchmark of expert-edited MGTs; (ii) we use an instruction-tuning dataset as the source of prompts and human-written responses, which ensures coverage of various domains and use cases; (iii) our annotation protocol is not based on a predefined taxonomy of operations (MixSet) and edits (LAMP); instead, our annotators make the edits based on their expertise in refining LLM-generated content; (iv) Beemo includes the largest number of expert-edited MGTs compared to MixSet (1.2k) and LAMP (1k); (v) we analyze how the behavior of binary MGT detectors changes after edits are made w.r.t. edit ratio and use case.

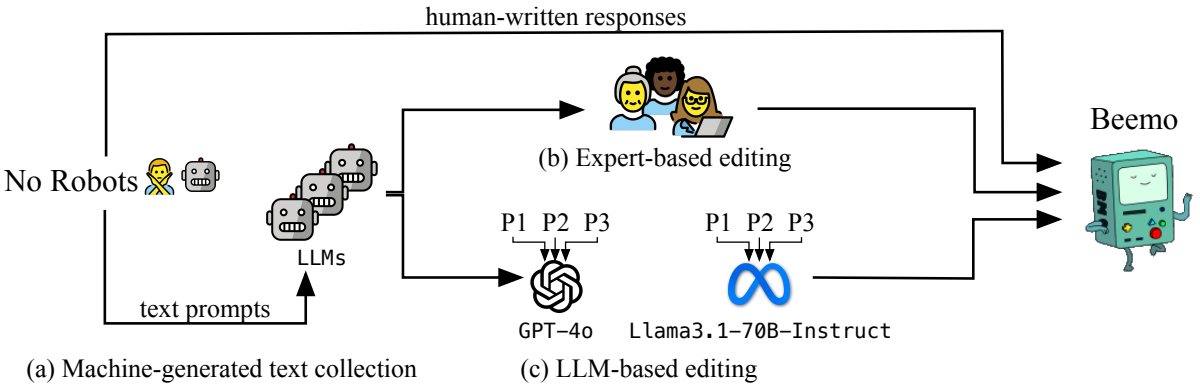

Figure 1: Overview of the Beemo's creation pipeline. (a) Use `No Robots` (Rajani et al., 2023) as the source of prompts and human-written responses across five categories. Generate responses from ten open-source instruction-finetuned LLMs. (b) Refine the LLMs' responses with a team of expert editors. (c) Refine the LLMs' responses using two state-of-the-art LLMs and editing prompts (P1-P3). Each of 2,195 instances includes nine text versions.

2.2 Editing MGTs

There are several approaches to editing generated content within MGT detection: operation-based, prompt-based, and expert-based editing.

Operation-based Editing Zhang et al. (2024a) propose five operations to refine an MGT at the token-, sentence-, and paragraph-level: polish, complete, rewrite, humanize, and adapt. This hybrid editing approach emulates real-world cases where humans aim to modify MGT to suit their preferences, improve quality, or align with the intended purpose.

Prompt-based Editing Emerging research leverages prompt-based LLM editing to mitigate the risks of "humanizing" LLM-generated content at scale. Hu et al. (2023a) employ few-shot in-context learning with human demonstrations to enhance MGT edits. Mitchell et al. (2023); Yang et al. (2023) use prompts to simulate humans limited editing behavior using T5 (Raffel et al., 2020). Abassy et al. (2024) utilize various prompts to refine MGTs and polish human-written texts, such as improving grammar and fluency. While prompt-based editing show benefits to assist humans, the output quality depends heavily on prompts and LLM editors, emphasizing the importance of careful prompt design and LLM considerations (Kamalloo et al., 2023; Zhang et al., 2023, 2024b).

Expert-based Editing Prior work employed human experts to edit and evaluate text across different domains (Nahar et al., 2024; Reid and Neubig, 2022; Du et al., 2022; Roberts and Moran, 1983; Lucas et al., 2023). However, limited research utilizes such experts for editing MGTs. The above mentioned works by Zhang et al. (2024a); Chakrabarty et al. (2024a) highlight human experts' unique value in refining machine-generated content, particularly in adapting to academic genres. Our work incorporates the prompt- and expert-based editing approaches, summarizes the experts' strategies to refine LLM content, and presents the results of analyzing the detectors' performance w.r.t. edit percentages

3 Beemo

Figure 1 outlines our high-level methodology for creating Beemo, which includes the following stages: generating instruction-finetuned LLMs' responses (§3.1), editing the responses by expert annotators (§3.2) and state-of-the-art LLMs (§3.3).

3.1 Machine-generated Data Collection

`No Robots` (Rajani et al., 2023) is a human-created instruction-tuning dataset, which is used as the source of prompts and corresponding human-written responses across the following categories: open-ended generation, rewriting, summarization, and open and closed question answering (QA)[3]. We randomly sample each prompt to generate a response with one of ten open-source instruction-finetuned LMs (see Table 2), which range in size from 2B to 70B. We use the default HuggingFace (Wolf et al., 2020) templates and inference hyperparameters.

[3]We aim to select more general and practical categories, where the user is likely to refine the model response. We leave extending Beemo with other user-oriented categories for future work (e.g., chat and brainstorming).

Model	Base	License	Source	SFT Corpus
Data generation				
zephyr-7b-beta	Mistral-7B-v0.1	MIT	Tunstall et al. (2023)	UltraChat, UltraFeedback
tulu2-2b tulu2-13b	Llama 2 7B	AI2 ImpACT	Ivison et al. (2023)	human-created, synthetic
gemma-2b-it gemma-7b-it	Gemma 2B Gemma 7B	Gemma	Gemma Team et al. (2024)	human-created, synthetic
Llama2-7b-chat-hf Llama2-13b-chat-hf Llama2-70b-chat-hf	Llama 2 7B Llama 2 13B Llama 2 70B	Llama	Touvron et al. (2023)	Misc.
Mistral-7B-Instruct-v0.1	Mistral-7B-v0.1	Apache-2.0	Jiang et al. (2023)	Misc.
Mixtral-8x7B-Instruct-v0.1	Mixtral 8x7B	Apache-2.0	Jiang et al. (2024)	Misc.
LLM-based editing				
Llama3.1-70B-Instruct	Llama 3.1 70B	Llama	Dubey et al. (2024)	Misc.
GPT-4o	GPT-4	OpenAI	OpenAI (2024)	Misc.

Table 2: The LLMs used to generate (§3.1) and edit (§3.3) responses. The SFT corpus description is based on public information. We evaluate potential overlap between the SFT corpora and LLMs' outputs in §3.4. Corpora references: UltraChat (Ding et al., 2023); UltraFeedback (Cui et al., 2023a).

3.2 Expert-based Editing

We run an in-house annotation to create expert-edited versions of the machine-generated texts. Our team consists of two lead editors and 25 expert annotators, who are native English speakers and well-experienced in refining content produced by LLMs. The lead editors collaborate closely with the annotators throughout the annotation project, suggesting areas for improvement, exchanging feedback in a group chat, and manually validating each annotator's submission. We provide detailed annotation guidelines to the annotators before they start the editing phase (see Appendix A). Each annotator receives one example at a time (a category, a prompt, and an LLM's response) and is asked to (i) carefully read the prompt and the response, (ii) judge the response's relevance to the prompt and the category, (iii) fact check the response if required, (iv) edit the response by correcting factual inconsistency, removing hallucinations, and improving style, coherence, and fluency, and (v) proofread the edited version before submission. The recommended ratio of edits ranges between 20% and 40% of the response. The average pay rate is $20/hr. The annotator can skip an example (i) if it does not require any edits and aligns with the prompt intent or (ii) if it requires significant improvements or does not follow the prompt closely. We discard such examples and use only the refined and manually validated responses to create Beemo.

3.3 LLM-based Editing

We utilize open-source and proprietary LLMs for LLM-based editing: `Llama3.1-70B-Instruct` and `GPT-4o` (see Table 2). We automatically refine the machine-generated texts using three types of prompts based on distinct motivations (see Table 7 in Appendix B): (P1) Focuses on grammatical correctness and human-like qualities; (P2) Aims to remove artifacts in an LLM-generated text (e.g., unwanted text such as *"Sure, here is the summarized text"*); (P3) Emphasizes producing a more natural and native-sounding text. Our aim here is to address the diversity of LLM-based edits in real-world scenarios, which target various aspects of making a machine-generated text more human-like (Zhang et al., 2024a; Gero et al., 2022). We specify the recommended range of edits in (P1) and (P2) as 20%–40% and do not provide it in (P3) to ensure the LLM-editor is not restricted in refining the generated response.

3.4 Benchmark Analysis

General Statistics We summarize the Beemo's general statistics by category (see Table 3) and LLM (see Table 8 in Appendix C) based on count, stylometric, and edit distance metrics computed via `TextDescriptives` (Hansen et al., 2023) and `editdistance`[4]: the average text length in tokens, the average number of stopwords in a text, the

[4] github.com/roy-ht/editdistance

Category	# Examples	# Tokens (P)	# Tokens (H)	# Tokens (M)	# Tokens (E)	# Tokens (L)	# Tokens (G)
Closed QA	5283	266.0 ± 229.3	22.9 ± 14.4	70.9 ± 63.5	51.5 ± 53.6	73.9 ± 35.4	62.0 ± 37.4
Generation	4725	40.2 ± 34.2	192.5 ± 121.3	255.2 ± 147.8	235.2 ± 145.6	115.8 ± 23.1	116.0 ± 28.5
Open QA	3510	15.4 ± 44.8	76.6 ± 47.0	156.1 ± 121.2	109.8 ± 105.1	100.7 ± 30.3	91.6 ± 37.8
Rewrite	4365	296.7 ± 249.7	205.8 ± 171.5	211.8 ± 135.5	207.7 ± 145.9	115.7 ± 22.1	112.2 ± 28.2
Summarize	1872	274.5 ± 162.9	64.3 ± 39.2	123.7 ± 94.3	85.5 ± 51.0	102.5 ± 25.0	85.9 ± 30.8
Overall	19755	159.1 ± 204.0	131.2 ± 129.0	182.1 ± 137.8	156.94 ± 136.5	106.1 ± 29.3	99.2 ± 36.2

Table 3: General statistics by category. **P**=No Robots prompts; **H**=human-written (2195 texts); **M**=machine-generated (2195 texts); **E**=expert-edited (2195 texts); **L**=Llama3.1-70B-Instruct-edited (6585 texts); **G**=GPT-4o-edited (6585 texts). **L** and **G** are aggregated over three editing prompts (§3.3).

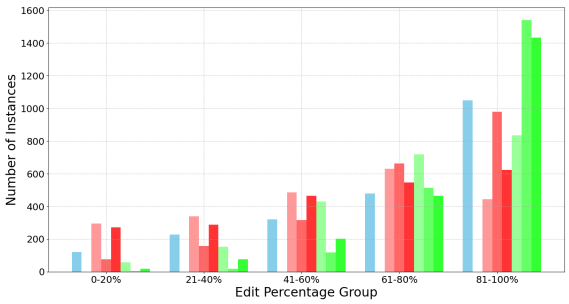

Figure 2: Distribution of edit percentages across five edit ranges for expert annotators, GPT-4o (P1, **P2**, **P3**), and Llama3.1-70B-Instruct (P1, **P2**, **P3**). The bars represent the number of instances falling within each edit percentage range for each editor type.

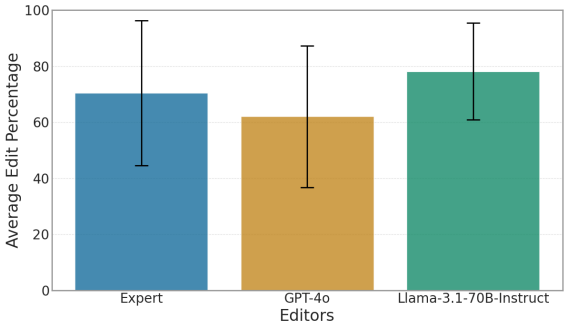

Figure 3: Comparison of average edit percentages among expert editors, GPT-4o, and Llama3.1-70B-Instruct. The bars represent the mean edit percentage for each editor type, with error bars indicating the standard deviation.

Flesch-Kincaid grade of a text (FKG), and Levenshtein distance (LD) between different text versions. We observe that the number of stopwords as a distinctive measure of repetitiveness (Fröhling and Zubiaga, 2021) depends on the LLM rather than its size. However, larger LLMs generally produce text with higher readability scores (FKG), with the exception of the Llama2 LLMs. Analyzing the statistics by category, we find that the LLMs' responses and their expert-edited versions are generally longer than the human-written and LLM-edited texts. This implies that the LLM-editors tend to shorten the generated responses, which is indicated by higher LD values. The average LD between the machine- and human-written responses ranges from 153 to 256, which suggests a minimal overlap between No Robots and the LLMs' instruction-finetuning corpora.

Editing Analysis We compare the edit percentages between the machine-generated texts and their GPT-4o-, Llama3.1-70B-Instruct, and expert-edited versions using difflib[5]. Figure 2 and Figure 3 present a comparison of editing behaviors among the expert and LLM editors. The experts demonstrate an average edit percentage of 70%, falling between GPT-4o (60%) and Llama3.1-70B-Instruct (80%). The error bars indicate considerable variability in edit percentages across different instances. This variability is particularly pronounced for the LLM editors, indicating less consistency in their editing behavior than experts. Both LLMs make more extensive edits than specified in the prompts (§3.3), which suggests their potential limitation in controlled editing.

Effect of Prompt Figure 5 (see Appendix C) illustrates the LLMs' editing behavior the three prompts. Overall, Llama3.1-70B-Instruct makes the most extensive edits irrespective of the prompt. Both LLMs exhibit an increase in edit percentage from (P1) to (P2) of up to 15%, although (P2) aims to control the edit range. Using (P3) results in a slight decrease in edit percentages for both LLMs compared to (P2), but they remain higher than (P1). The results suggest that prompt engineering significantly influences the LLM's ability to follow specified editing ranges, which aligns with Wang et al. (2023).

[5] docs.python.org/library/difflib

Survey on Editing Strategies We send a voluntary survey to our editors to better understand their main editing strategies and identify common issues in the generated responses (see Appendix D). Our analysis of 14 respondents' answers reveals several common issues with LLMs' responses: (i) format inconsistencies (e.g., a partial or inconsistent ordering of items in a list), (ii) hallucinations and factual errors, (iii) struggling with creative writing (e.g., writing poems and haikus), (iv) complicated and repetitive vocabulary, and their responses can feel over-explained, and (v) responses often lack natural flow and exhibit a repetitive structure. Our expert editors follow a consistent approach based on their individual experience to refine the generated outputs: (i) carefully reading the prompt and LLM's response, (ii) fact-checking if required, (iii) focusing on complicated use of passive voice, repeated elements, and odd openings as common properties in our data, (iv) editing for natural flow, improving structure, and changing vocabulary for clarity and richness, (v) ensuring the response adheres to the required format, (vi) double-checking for grammar, phrasing and spelling errors, and (vii) reading through or aloud for a final quality check. Our experts report that it can be challenging to follow the recommended edit range of 20%–40%, and rewriting entire sections can be more effective than editing specific parts, particularly for summarization, rewriting, and open-ended generation. However, the core content of the LLM response remains consistent, even at higher edit ratios.

4 Experimental Setup

We use Beemo as an out-of-domain benchmark to evaluate generalization abilities of 33 zero-shot and pretrained MGT detectors' configurations in seven binary classification task formulations, which rely on single- and multi-author text versions. We list the detectors below and detail them in Appendix E.

Zero-shot MGT Detectors The zero-shot MGT detectors utilize log probabilities, entropy, and curvature-based properties to score a given text: (i) Binoculars (Hans et al., 2024); (ii) Log probability (Solaiman et al., 2019); (iii) Rank (Gehrmann et al., 2019); (iv) Log-Rank (Ippolito et al., 2020); (v) Entropy (Gehrmann et al., 2019; Mitchell et al., 2023); (vi) DetectLLM (Su et al., 2023a) Likelihood Log-Rank ratio (LRR) and Normalized Perturbed Log-Rank (NPR); and (vii) DetectGPT (Mitchell et al., 2023)[6]. We use the codebase by Hans et al. and Su et al. to run the detectors, which supports `GPT2-XL`[7], `OPT-1.3B`[8], `Falcon-7B`[9], and `Qwen2-7B`[10] as the default backbone models.

Pretrained MGT Detectors We consider the following pretrained detectors: (i) RADAR[11] (Hu et al., 2023b); (ii) AIGC MPU[12]; (iii) MAGE[13] (Li et al., 2024); and (iv) OpenAI RoBERTa-base[14]/large[15].

Task Formulations We explore the following binary classification task formulations by manipulating different text versions as the input:

- human-written (**H**; `label=0`) vs. machine-generated, expert-, `Llama3.1-70B-Instruct`-, and `GPT-4o`-edited (**M/E/L/G**; `label=1`). We treat a text as machine-generated even if the experts edit it. The goal is to analyze if detectors still identify the MGTs after it is refined.
- expert-edited, `Llama3.1-70B-Instruct`-, and `GPT-4o`-edited (**E/L/G**; `label=0`) vs. machine-generated (**M**; `label=1`). We treat a text as human-written even if the LLM-editors "humanize" it. Here, the goal is to analyze if detectors identify any edited MGT as human-written.

Performance Metric Following Verma et al. (2024); Mitchell et al. (2023), we evaluate the detector performance using Area Under the Receiver Operating Characteristic curve (AUROC). AUROC values represent the probability that a detector assigns higher scores to a randomly selected machine-generated text than a randomly selected human-written text. AUROC is commonly used in zero-shot scenarios – where the choice of the scoring threshold is crucial – as it considers the range of all possible thresholds (Krishna et al., 2024).

5 Results & Analysis

This section describes our empirical evaluation results on Beemo. We report the overall results in Table 4 and by category in Appendix F. Overall,

[6] DetectLLM NPR and DetectGPT rely on an additional LLM (`T5-3B`) to perturb the input text and score both original and perturbed texts. The perturbation number is set to 20.

[7] `hf.co/openai-community/gpt2-xl`

[8] `hf.co/facebook/opt-1.3b`

[9] `hf.co/tiiuae/falcon-7b`

[10] `hf.co/Qwen/Qwen2-7B`

[11] `hf.co/TrustSafeAI/RADAR-Vicuna-7B`

[12] `hf.co/yuchuantian/AIGC_detector_env1`

[13] `hf.co/yaful/MAGE`

[14] `hf.co/openai/roberta-base-openai-detector`

[15] `hf.co/openai/roberta-large-openai-detector`

Detector		H vs. M	E vs. M	H vs. E	L vs. M	H vs. L	G vs. M	H vs. G
Zero-shot MGT Detectors								
Binoculars		83.90	76.79	61.24	59.90	79.90	57.93	78.15
Log Probability	GPT2-XL	69.72	64.73	56.56	63.70	58.88	60.32	61.23
	OPT-1.3B	74.82	66.52	60.86	66.58	62.78	61.12	66.95
	Falcon-7B	86.07	65.48	78.41	65.36	80.78	59.65	82.44
	Qwen2-7B	87.77	68.68	78.02	56.23	84.52	62.45	81.27
Rank	GPT2-XL	60.36	56.70	54.50	59.85	50.65	56.79	53.64
	OPT-1.3B	65.95	57.44	59.94	63.56	53.48	57.10	59.86
	Falcon-7B	74.52	56.56	70.66	62.95	63.64	55.57	70.61
	Qwen2-7B	74.22	57.65	69.11	52.55	72.52	55.10	70.81
Log-Rank	GPT2-XL	70.26	64.95	56.78	65.25	57.42	61.20	60.68
	OPT-1.3B	73.03	66.11	59.03	67.21	59.30	61.83	63.90
	Falcon-7B	84.81	65.07	77.02	66.12	78.26	60.26	80.32
	Qwen2-7B	86.91	68.13	77.04	56.04	83.62	62.09	80.33
Entropy	GPT2-XL	38.93	42.42	46.06	36.74	51.72	41.95	46.64
	OPT-1.3B	37.63	42.74	44.18	35.41	51.33	41.07	45.72
	Falcon-7B	20.92	41.65	24.42	36.02	29.24	40.96	25.66
	Qwen2-7B	10.36	32.02	17.74	44.01	12.82	38.01	15.28
DetectLLM Likelihood Log-Rank Ratio	GPT2-XL	68.73	63.72	56.18	67.45	51.86	62.27	57.07
	OPT-1.3B	64.37	63.09	51.90	67.21	47.17	62.49	52.10
	Falcon-7B	76.73	62.16	68.41	66.77	64.68	61.28	68.31
	Qwen2-7B	79.68	64.12	69.40	54.71	76.25	59.41	72.83
DetectLLM Normalized Perturbed Log-Rank	GPT2-XL	64.06	63.07	62.00	54.08	63.34	51.07	63.93
	OPT-1.3B	65.29	63.86	62.86	56.98	64.04	51.36	65.12
	Falcon-7B	70.14	63.18	66.71	56.65	68.40	51.37	69.84
	Qwen2-7B	74.12	68.30	68.11	54.52	72.49	50.68	73.84
DetectGPT	GPT2-XL	64.91	63.71	62.75	59.15	63.53	58.97	63.63
	OPT-1.3B	67.27	66.87	64.22	64.39	64.68	61.65	65.32
	Falcon-7B	70.69	65.35	67.47	64.15	67.73	62.67	68.18
	Qwen2-7B	72.76	69.37	68.21	61.36	70.14	63.05	69.90
Pretrained MGT Detectors								
RADAR		51.41	50.93	50.80	34.51	62.65	37.96	60.16
MAGE		73.72	60.88	64.09	57.95	67.79	60.59	65.37
AIGC MPU		70.52	70.35	50.66	65.71	57.17	59.66	62.69
OpenAI RoBERTa-base		66.95	66.83	49.13	66.95	50.00	66.95	50.00
OpenAI RoBERTa-large		62.93	63.96	48.76	64.71	48.57	59.67	53.74

Table 4: The AUROC scores (%) of the 33 MGT detectors' configurations on Beemo. A random classifier has an AUROC of 50%. **H**=human-written; **M**=machine-generated; **E**=expert-edited; **L**=`Llama3.1-70B-Instruct`-edited; **G**=`GPT-4o`-edited. **L** and **G** are aggregated over three prompts (§3.3).

we observe that Binoculars is the best-performing zero-shot detector, while MAGE is the strongest pretrained detector in most scenarios. Entropy performs worse or on par with a random guessing classifier. We summarize our findings below w.r.t. task formulation, editing type, MGT detector type, category, and edit percentages.

Expert-based Editing Evades Detection Analyzing the results in the **H vs. M/E/L/G** scenarios, we find that all detectors often fail to recognize an MGT after it is refined by experts, with the AUROC scores decreasing by up to 22% for zero-shot (e.g., Binoculars, Log Probability, and Log-Rank) and pretrained (e.g., AIGC MPU, OpenAI RoBERTa-base/large) detectors. In contrast, many zero-shot detectors identify LLM-edited texts as MGTs, with Binoculars, Rank, Log-Rank, and Log Probability demonstrating stronger generalization abilities. However, RADAR, AIGC MPU, and OpenAI RoBERTa-base/large exhibit a random guessing performance across various edit types, which suggests a strong distribution shift.

LLM-edited Texts are Less Likely to Be Recognized as Human-authored A comparison of the results in the **E/L/G vs. M** scenarios confirms our previous finding that the expert-edited texts are more likely to be classified as human-written compared to the LLM-edited ones. No-

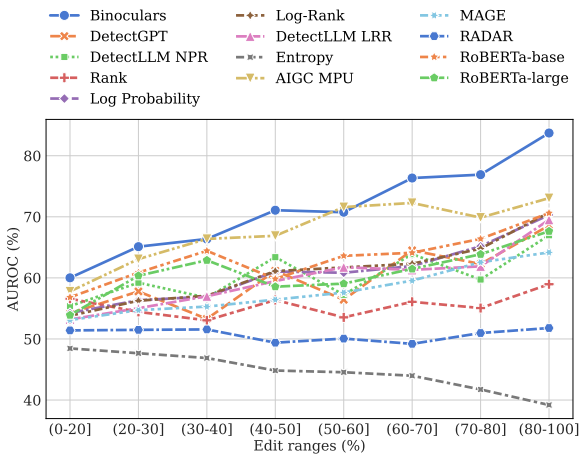

Figure 4: Results in the "Expert-edited" (`label=0`) vs. "machine-generated" (`label=1`) scenario divided into seven groups by the edit range.

tably, `GPT-4o`-edited texts are generally easier to identify as MGTs compared to expert- and `Llama3.1-70B-Instruct`-edited.

Analysis by Detector Type Here, we compare the results between the zero-shot and pretrained detectors in more detail. The zero-shot detectors (i) better distinguish between human-written and machine-generated texts (**H vs. M**); (ii) are more generalizable to expert- and LLM-edited texts, which is indicated by lower AUROC δ-scores (e.g., up to 10% for Log-Rank, DetectLLM LLR/NPR, DetectGPT and up to 20% for AIGC MPU, MAGE, and OpenAI RoBERTa-base/large); and (iii) perform better with a larger backbone model in the **H vs. M/E/L/G** scenarios, however, this effect is less pronounced in the **E/L/G vs. M** scenarios.

Analysis by Category The key finding here is that the detectors' behavior w.r.t. different edit types remains the same, but the AUROC scores across most task formulations depend on category (see Appendix F). The detectors perform consistently better on open-ended generation and open QA (e.g., Binoculars=up to 86%; Log Probability=up to 93%; DetectLLM LLR=up to 86%; MAGE=up to 76% in **H vs. M/E/L/G**). The rewriting, summarization, and closed QA are more challenging, where the detectors exhibit significant performance drops after expert edits (e.g., Log Probability=up to 20%; Rank=up to 17%; Log-Rank=up to 23%; DetectLLM=up to 18%) or struggle with distinguishing between different MGT versions (e.g., RADAR and MAGE often perform worse than a random classifier). We attribute these trends to category-specific text generation issues identified by our experts in §3 and shorter text length (see Table 3), which remains an unresolved challenge in MGT detection.

Effect of Edit Percentage We analyze the effect of the edit percentage in the **E vs. M** scenario to assess the impact of expert edits (see Figure 4). The overall trend is that the detectors' performance remains moderate and stable across the experts' edit ratios, showing general improvement as the ratio increases. Most detectors achieve similar AUROC scores for both moderately (20%–40%) and significantly (60%–80%) edited texts. Lower AUROC scores at the lower edit ranges indicate that moderate editing can confuse the detectors.

6 Conclusion & Future Work

This work introduces Beemo, one of the first multi-author benchmarks of expert-edited and LLM-refined MGTs for English. Beemo covers five general expert and non-expert use cases of instruction-finetuned LLMs, ranging from creative writing to summarization. We document the Beemo creation approach and describe common issues in LLM responses and strategies for mitigating them based on the feedback from our expert editors. We conduct an extensive out-of-domain evaluation of 33 MGT detectors' configurations and analyze their performance in seven signle- and multi-author binary classification tasks. Our key empirical results demonstrate that detectors can be confused by moderate expert edits, while editing with state-of-the-art LLMs does not significantly influences the detection behavior. Furthermore, we find that zero-shot detectors are more generalizable to both expert- and LLM-edited MGTs than pretrained detectors. Our *future* work includes (i) exploration of other MGT detection task formulations, including authorship attribution, k-way classification (human-written, machine-generated, machine-generated & expert-/LLM-edited), and extraction of spans, (ii) establishment of the human baseline across various editing types; this has not been done due to limited resources; (iii) ablation studies on exploring the supervised detectors' robustness towards unseen instruction-finetuned LLM and writing task, and (iv) ablation studies on the effect of the LLM-edited MGTs in the training data on the detectors' generalization.

Limitations

Evaluation Design While we present the results of extensive empirical evaluation of a broad range of MGT detectors' configurations, we acknowledge certain limitations in our evaluation design. First, the LLM-based editing inherently depends on the prompt, and the best prompt configuration is LLM-specific (Voronov et al., 2024). Although we address this sensitivity by employing three distinct types of editing prompts, our results may still be affected by the effect of prompt engineering. Second, our MGT detectors represent commonly used open-source approaches, which does not address the analysis of commercial and API-based MGT detectors often used to check if the LLM's response refinement passes them. Furthermore, ensembling approaches can be considered, which are more generalizable to the out-of-domain data (Wang et al., 2024a).

Need for Continuous Data Collection The design of MGT benchmarks is fragile due to the rapid development of novel LLMs and their instruction-finetuned versions. This raises the need to continuously update Beemo to keep it with the current state of the field; however, it is expensive to collect expert-edited texts at scale. We encourage the NLP researchers and practitioners to contribute machine-generated and machine-generated & LLM-refined texts to account for the diversity of LLMs and editing prompts.

Lack of Human Baseline Similar to closely related studies on multi-author MGT detection (Zhang et al., 2024a; Chakrabarty et al., 2024a; Abassy et al., 2024), our work does not present the human baseline results due to limited resources. We aim to establish the human baseline across various editing types in our future work.

Ethics Statement

Expert-based Editing Our team of expert editors is based in the United States and Canada, and their pay rate exceeds the corresponding hourly minimum wage. The experts are warned about potentially sensitive and harmful content in the prompts and LLMs' responses related to various topics, including but not limited to politics, culture, sexual orientation, and religion.

Use of AI-assistants We use Grammarly[16] to correct grammar, spelling, phrasing, and style errors in our paper. Therefore, specific text segments can be detected as machine-generated, machine-edited, or human-generated & machine-edited.

Computational Costs Evaluating an MGT detector on Beemo does not require any finetuning. To reduce the evaluation costs in §4, we pre-compute and save the detectors' predictions for each text version. This allows us to estimate the performance efficiently by manipulating these pre-computed results based on the task formulation. Further inference costs can be reduced with the help of distributed inference libraries (e.g., `accelerate`[17]).

Potential Misuse We acknowledge that Beemo can be misused for malicious purposes, including but not limited to training multi-author MGT detectors to evade further detection at scale. We release Beemo for research and development purposes and encourage responsible use of our benchmark.

Transparency We release Beemo and all annotation materials following the standard open-source research practises. Our GitHub repository and HuggingFace dataset card[18] provide comprehensive documentation on our codebase, benchmark creation process, and data annotation guidelines.

Licensing Information The prompts and human-written responses from `No Robots` are under the original dataset's license (CC-BY-NC-4.0). The MGTs are subject to the underlying instruction-finetuned LLMs' licensing terms (see Table 2). The expert-edited MGTs are available under the MIT license, unless otherwise specified in the underlying instruction-finetuned LLMs' licensing terms.

References

Mervat Abassy, Kareem Elozeiri, Alexander Aziz, Minh Ngoc Ta, Raj Vardhan Tomar, Bimarsha Adhikari, Saad El Dine Ahmed, Yuxia Wang, Osama Mohammed Afzal, Zhuohan Xie, et al. 2024. LLM-DetectAIve: a Tool for Fine-Grained Machine-Generated Text Detection. *arXiv preprint arXiv:2408.04284*.

Tuhin Chakrabarty, Philippe Laban, and Chien-Sheng Wu. 2024a. Can AI writing be salvaged? Mitigating Idiosyncrasies and Improving Human-AI Alignment

[16] `grammarly.com`
[17] `github.com/huggingface/accelerate`
[18] The links will be provided upon acceptance.

in the Writing Process through Edits. *arXiv preprint arXiv:2409.14509*.

Tuhin Chakrabarty, Vishakh Padmakumar, Faeze Brahman, and Smaranda Muresan. 2024b. Creativity Support in the Age of Large Language Models: An Empirical Study Involving Professional Writers. In *Proceedings of the 16th Conference on Creativity & Cognition*, pages 132–155.

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.

Yutian Chen, Hao Kang, Vivian Zhai, Liangze Li, Rita Singh, and Bhiksha Raj. 2023. Token prediction as implicit classification to identify llm-generated text. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13112–13120.

Joseph Cornelius, Oscar Lithgow-Serrano, Sandra Mitrović, Ljiljana Dolamic, and Fabio Rinaldi. 2024. Bust: Benchmark for the evaluation of detectors of llm-generated text. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8022–8050.

Evan N Crothers, Nathalie Japkowicz, and Herna L Viktor. 2023. Machine-generated Text: A Comprehensive Survey of Threat Models and Detection Methods. *IEEE Access*, 11:70977–71002.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2023a. Ultrafeedback: Boosting language models with high-quality feedback. *Preprint*, arXiv:2310.01377.

Wanyun Cui, Linqiu Zhang, Qianle Wang, and Shuyang Cai. 2023b. Who said that? benchmarking social media ai detection. *arXiv preprint arXiv:2310.08240*.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*.

Wanyu Du, Vipul Raheja, Dhruv Kumar, Zae Myung Kim, Melissa Lopez, and Dongyeop Kang. 2022. Understanding iterative revision from human-written text. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3573–3590.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Liam Dugan, Alyssa Hwang, Filip Trhlik, Josh Magnus Ludan, Andrew Zhu, Hainiu Xu, Daphne Ippolito, and Chris Callison-Burch. 2024. RAID: A Shared Benchmark for Robust Evaluation of Machine-Generated Text Detectors. *arXiv preprint arXiv:2405.07940*.

Liam Dugan, Daphne Ippolito, Arun Kirubarajan, Sherry Shi, and Chris Callison-Burch. 2023. Real or fake text?: Investigating human ability to detect boundaries between human-written and machine-generated text. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 12763–12771.

Tiziano Fagni, Fabrizio Falchi, Margherita Gambini, Antonio Martella, and Maurizio Tesconi. 2021. TweepFake: About Detecting Deepfake Tweets. *Plos one*, 16(5):e0251415.

Leon Fröhling and Arkaitz Zubiaga. 2021. Feature-based Detection of Automated Language Models: Tackling GPT-2, GPT-3 and Grover. *PeerJ Computer Science*, 7:e443.

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.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

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

Vipul Gupta, Pranav Narayanan Venkit, Shomir Wilson, and Rebecca J Passonneau. 2024. Sociodemographic bias in language models: A survey and forward path. In *Proceedings of the 5th Workshop on Gender Bias in Natural Language Processing (GeBNLP)*, pages 295–322.

Abhimanyu Hans, Avi Schwarzschild, Valeriia Cherepanova, Hamid Kazemi, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2024. Spotting LLMs With Binoculars: Zero-Shot Detection of Machine-Generated Text. In *Forty-first International Conference on Machine Learning*.

Lasse Hansen, Ludvig Renbo Olsen, and Kenneth Enevoldsen. 2023. TextDescriptives: A Python package for calculating a large variety of metrics from text. *arXiv preprint arXiv:2301.02057*.

Xiaomeng Hu, Pin-Yu Chen, and Tsung-yi Ho. 2023a. Radar: Robust ai-text detection via adversarial learning. In *Annual Conference on Neural Information Processing Systems*.

Xiaomeng Hu, Pin-Yu Chen, and Tsung-Yi Ho. 2023b. RADAR: robust ai-text detection via adversarial learning. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

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

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A Smith, Iz Beltagy, et al. 2023. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *arXiv preprint arXiv:2311.10702*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Ehsan Kamalloo, Nouha Dziri, Charles Clarke, and Davood Rafiei. 2023. Evaluating open-domain question answering in the era of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5591–5606, Toronto, Canada. Association for Computational Linguistics.

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

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

Vijini Liyanage, Davide Buscaldi, and Adeline Nazarenko. 2022. A benchmark corpus for the detection of automatically generated text in academic publications. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4692–4700.

Jason Lucas, Adaku Uchendu, Michiharu Yamashita, Jooyoung Lee, Shaurya Rohatgi, and Dongwon Lee. 2023. Fighting fire with fire: The dual role of llms in crafting and detecting elusive disinformation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14279–14305.

Dominik Macko, Jakub Kopal, Robert Moro, and Ivan Srba. 2024. MultiSocial: Multilingual Benchmark of Machine-Generated Text Detection of Social-Media Texts. *arXiv preprint arXiv:2406.12549*.

Dominik Macko, Robert Moro, Adaku Uchendu, Jason Lucas, Michiharu Yamashita, Matúš Pikuliak, Ivan Srba, Thai Le, Dongwon Lee, Jakub Simko, and Maria Bielikova. 2023. MULTITuDE: Large-scale multilingual machine-generated text detection benchmark. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9960–9987, Singapore. Association for Computational Linguistics.

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.

Mahjabin Nahar, Haeseung Seo, Eun-Ju Lee, Aiping Xiong, and Dongwon Lee. 2024. Fakes of varying shades: How warning affects human perception and engagement regarding llm hallucinations. *arXiv preprint arXiv:2404.03745*.

OpenAI. 2024. Gpt-4o mini: Advancing cost-efficient intelligence. Accessed: 2024-09-30.

Jiameng Pu, Zain Sarwar, Sifat Muhammad Abdullah, Abdullah Rehman, Yoonjin Kim, Parantapa Bhattacharya, Mobin Javed, and Bimal Viswanath. 2023. Deepfake text detection: Limitations and opportunities. In *2023 IEEE symposium on security and privacy (SP)*, pages 1613–1630. IEEE.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving Language Understanding by Generative Pre-training.

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.

Nazneen Rajani, Lewis Tunstall, Edward Beeching, Nathan Lambert, Alexander M. Rush, and Thomas Wolf. 2023. No robots. https://huggingface.co/datasets/HuggingFaceH4/no_robots.

Machel Reid and Graham Neubig. 2022. Learning to model editing processes. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3822–3832.

Teresa L Roberts and Thomas P Moran. 1983. The evaluation of text editors: methodology and empirical results. *Communications of the ACM*, 26(4):265–283.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, and Jasmine Wang. 2019. Release strategies and the social impacts of language models.

Jinyan Su, Terry Zhuo, Di Wang, and Preslav Nakov. 2023a. 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, Singapore. Association for Computational Linguistics.

Zhenpeng Su, Xing Wu, Wei Zhou, Guangyuan Ma, and Songlin Hu. 2023b. Hc3 plus: A semantic-invariant human chatgpt comparison corpus. *arXiv preprint arXiv:2309.02731*.

Ruixiang Tang, Yu-Neng Chuang, and Xia Hu. 2024. The Science of Detecting LLM-generated Text. *Communications of the ACM*, 67(4):50–59.

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

Nafis Irtiza Tripto, Saranya Venkatraman, Dominik Macko, Róbert Móro, Ivan Srba, Adaku Uchendu, Thai Le, and Dongwon Lee. 2024. A ship of theseus: Curious cases of paraphrasing in llm-generated texts. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 6608–6625. Association for Computational Linguistics.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. 2023. Zephyr: Direct distillation of LM alignment. *arXiv preprint arXiv:2310.16944*.

Adaku Uchendu, Thai Le, and Dongwon Lee. 2023. Attribution and obfuscation of neural text authorship: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, 25(1):1–18.

Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship attribution for neural text generation. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pages 8384–8395.

Adaku Uchendu, Zeyu Ma, Thai Le, Rui Zhang, and Dongwon Lee. 2021. Turingbench: A benchmark environment for turing test in the age of neural text generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2001–2016.

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

Anton Voronov, Lena Wolf, and Max Ryabinin. 2024. Mind Your Format: Towards Consistent Evaluation of In-context Learning Improvements. *arXiv preprint arXiv:2401.06766*.

Yiming Wang, Zhuosheng Zhang, and Rui Wang. 2023. Element-aware summarization with large language models: Expert-aligned evaluation and chain-of-thought method. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8640–8665.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Osama Mohammed Afzal, Tarek Mahmoud, Giovanni Puccetti, and Thomas Arnold. 2024a. SemEval-2024 task 8: Multidomain, multimodel and multilingual machine-generated text detection. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 2057–2079, Mexico City, Mexico. Association for Computational Linguistics.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Osama Mohammed Afzal, Tarek Mahmoud, Giovanni Puccetti, Thomas Arnold, Alham Aji, Nizar Habash, Iryna Gurevych, and Preslav Nakov. 2024b. M4GT-bench: Evaluation benchmark for black-box machine-generated text detection. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3964–3992, Bangkok, Thailand. Association for Computational Linguistics.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Toru Sasaki, Thomas Arnold, Alham Fikri Aji, Nizar Habash, Iryna Gurevych, and Preslav Nakov. 2024c. M4: Multi-generator, multi-domain, and multi-lingual black-box machine-generated text detection. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1369–1407, St. Julian's, Malta. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pier-

ric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Daijin Yang, Yanpeng Zhou, Zhiyuan Zhang, Toby Jia-Jun Li, and Ray LC. 2022. AI as an Active Writer: Interaction strategies with generated text in human-AI collaborative fiction writing. In *Joint Proceedings of the ACM IUI Workshops*, volume 10, pages 1–11. CEUR-WS Team.

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

Biao Zhang, Barry Haddow, and Alexandra Birch. 2023. Prompting large language model for machine translation: A case study. In *International Conference on Machine Learning*, pages 41092–41110. PMLR.

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.

Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B. Hashimoto. 2024b. Benchmarking large language models for news summarization. *Transactions of the Association for Computational Linguistics*, 12:39–57.

A Annotation guidelines

Objective

The goal of this task is to refine machine-generated texts to make them more **human-like** in terms of fluency, factual correctness, coherence, and style.

Task

You are provided with an instruction pair, comprising two components: (1) an instruction given to an AI assistant, such as ChatGPT, and (2) the assistant's response. Your objective is to refine the response, making it sound more human-like and ensuring it is free of errors.

How many edits should be done?

Attempt to edit 20% to 40% of the provided text. Reject the text if it requires more significant improvements or does not follow the instructions completely. Aim for real-life settings: if a text misses the mark or requires a lot of refinement, most likely, the user would ask the AI assistant for a better response.

To **reject** the text, hit the "**Too bad**" button. If there is **nothing to edit** in the text, hit the "**Already perfect**" button and proceed to the next example. No need to fill in the "**Edit the AI Response**" field in either of these two cases.

1. **Read:** Carefully read the provided text, paying attention to the instruction and to how well the AI assistant followed it.
2. **Check:** Check that the prompt matches the category. See the fields **Category** and **Prompt** at the top part of the interface. If the prompt doesn't fall into the specified category, reject the text by hitting the "**Too bad**" button.
3. **Assess:** Evaluate the AI assistant's response for factual correctness, coherence, grammar, style, and overall human-like quality. Identify any errors or areas that need improvement.
4. **Edit:** Make necessary revisions to the AI assistant's response to enhance its human-like quality. This may include rephrasing sentences, correcting grammar mistakes, ensuring coherence, and adding a personal touch to it. See the list of the issues you may want to correct below. We provide the recommended range of edits 20%–40% but it is not strict; make the edits based on your experience in editing and working with data generated by language models.
5. **Proofread:** Review the edited response to ensure it is free of errors and flows smoothly. Check for any remaining issues or inconsistencies.
6. **Submit:** Once you are satisfied with the edits and the AI response meets the criteria of being human-like and error-free, submit your final version.

Table 5: A shortened version of the annotation guidelines for expert-based editing in §3.2 (Part 1). The full version with the editing examples will be provided in our GitHub repository upon acceptance.

Issues of AI-generated text that you may want to correct

This list is not exhaustive. Follow your intuition and edit the text where you feel it's reasonable to do so.

1. **Repetitions.** The same word, phrase, or sentence is repeated multiple times.
   Do: Remove repeated words, phrases, and sentences. This is an editing example.
2. **Awkward phrasing.** The text contains awkward or unnatural phrasing that makes it difficult to read or understand. It contains generic and simplistic language or, on the contrary, uses unnecessarily sophisticated and outdated words and phrases.
   Do: Improve the response for natural flow. This is an editing example.
3. **Tone and style.** The tone and style of the text do not align with the intended audience and purpose.
   Do: Adjust the tone and style. This is an editing example.
4. **Texts written from the AI assistant's point of view.** AI assistants tend to add some unnecessary sentences to the beginning or the end of the text. These AI's introductory sentences are unnecessary and should be omitted.
   Do: Remove markers of a text generated by an AI assistant. This is an editing example.
5. **Grammar errors.** The text is ungrammatical and contains unacceptable sentences.
   Do: Edit so the text is grammatically correct, fix the sentence structure and correct the typos. This is an editing example.
6. **Relevance.** The response does not fully follow the provided instruction.
   Do: Remove any irrelevant or unnecessary information from the AI response. Add important details or context. If the AI response is completely irrelevant to the instruction, reject the text. This is an editing example.
7. **Personal touch.** The text is too generic.
   Do: You may want to add a personal touch to make the text more engaging. This is an editing example.
8. **Fact-check the AI response.** Check the facts provided by the AI assistant. If you can fix the facts based on common knowledge or your personal experience, you don't need to provide anything. If you searched online for the information to fix the facts, provide links to the found sources in the "**Comment**" that you used to fact-check the AI response.
   Do: Decide to what degree it is possible to edit the AI response to improve factual correctness. If it requires substantial changes, discard the text. This is an editing example.
9. **Incomplete response.** If the response is interrupted accidentally, complete it. This is an editing example.
10. **Other.** You may find other issues that you want to edit. Let us know about them in "**Comment**".

Example of web interface

Category This is a toy category.

Prompt This is a toy prompt.

AI Response This is a toy LLM's response.

○ Too bad
○ Already perfect

Edit the AI Response

Comment

Proofread before submission. Thank you!

Table 6: A shortened version of the annotation guidelines and an example of web interface for expert-based editing in §3.2 (Part 2). The full version with the editing examples will be provided in our GitHub repository upon acceptance.

B Details on LLM-based Editing

	Prompt	gemma-2b-it	Expert-edited	Llama3.1-70B-Instruct-edited	GPT-4o-edited
P1	You are given a prompt and a text generated by AI using this prompt. Your task is to edit the AI-generated text to make it sound human-like and error-free. Ensure your overall edits do not exceed 40% of the generated text and the edited text follows the user request. Output only the edited text and do not explain your edits. Prompt: {prompt} AI text: {model_output}	According to the statement, Chick-fil-A's core values are purpose, culture, and teamwork. They believe that through these values, they can provide a positive work experience for their staff and ensure that they deliver the best possible service to their customers.	According to this statement, Chick-fil-A does its best work through teamwork and collaboration.	According to the statement, Chick-fil-A's core values reflect their purpose, culture, and team experience. They believe that by putting the needs of their Operators, Team Members, and customers first, they can do their best work, particularly when achieved through teamwork and collaboration.	According to the statement, Chick-fil-A believes they do their best work through teamwork and collaboration. By fostering an inclusive culture and leveraging the strengths of their diverse talent, they aim to serve their Operators, Team Members, and customers effectively.
P2	You are given a pair containing two components: (1) a user prompt for an AI assistant and (2) the AI assistant's response. Refine the AI-generated response to make it sound more natural. Vary your editing patterns and the portions of text you choose to modify, and ensure your overall edits are 20-40% of the words in the response. Prompt: {prompt} AI-generated response: {model_output}	According to the statement, Chick-fil-A's core values are purpose, culture, and teamwork. They believe that through these values, they can provide a positive work experience for their staff and ensure that they deliver the best possible service to their customers.	According to this statement, Chick-fil-A does its best work through teamwork and collaboration.	According to Chick-fil-A's statement, they attribute their best work to a combination of their core values, which emphasize the importance of teamwork, a strong company culture, and staying true to their purpose. By prioritizing these values, they aim to foster a positive work environment for their staff while delivering exceptional service to their customers.	According to the statement, Chick-fil-A's best work is guided by their core values of purpose, culture, and teamwork. By embracing these values, they aim to create a positive work environment for their staff and deliver exceptional service to their customers.
P3	Modify a machine-generated response to a given prompt to make it appear more like it was written by a native English speaker. Ensure the revised version follows the user's intent. You should just give me the revised version without any other words. Prompt: {prompt} Machine-generated response: {model_output}	According to the statement, Chick-fil-A's core values are purpose, culture, and teamwork. They believe that through these values, they can provide a positive work experience for their staff and ensure that they deliver the best possible service to their customers.	According to this statement, Chick-fil-A does its best work through teamwork and collaboration.	According to Chick-fil-A's statement, they do their best work when they operate as a team, leveraging the strengths of their diverse talent through collaboration and inclusivity.	According to the statement, Chick-fil-A believes they do their best work through their core values of service, teamwork, purpose-driven actions, and continuous improvement. These values help create a positive work experience for their staff and ensure they deliver the best possible service to their customers.

Table 7: An example of edits made by the experts, Llama3.1-70B-Instruct, and GPT-4o for a gemma-2b-it's response on Chick-fil-A Values Statement. Blue boxes indicate individual words that remain unchanged from the original Model Output. Pink boxes indicate individual words that were changed or added in the edited versions.

C Details on Benchmark Analysis

Model	# Examples	# Tokens	# Stopwords	FKG	LD(M, H)	LD(M, E)	LD(M, L)	LD(M, G)
zephyr-7b-beta	230	203.56 ± 143.09	98.43 ± 76.85	9.42 ± 6.27	211.22 ± 151.6	144.93 ± 111.01	189.86 ± 151.39	160.77 ± 151.02
tulu2-2b	216	139.76 ± 95.21	70.9 ± 50.5	8.41 ± 3.63	164.76 ± 126.19	104.53 ± 95.91	127.43 ± 91.15	97.92 ± 92.36
tulu2-13b	211	149.95 ± 113.37	74.15 ± 60.96	9.44 ± 7.08	170.25 ± 142.93	92.55 ± 93.53	136.29 ± 112.99	103.84 ± 111.87
gemma-2b-it	204	153.79 ± 137.75	69.59 ± 65.49	7.73 ± 3.65	188.56 ± 172.0	120.98 ± 130.15	157.91 ± 141.57	129.9 ± 139.24
gemma-7b-it	209	140.81 ± 111.14	63.71 ± 52.87	8.01 ± 4.06	175.59 ± 145.4	101.6 ± 106.03	143.42 ± 114.69	113.28 ± 110.16
Llama2-7b-chat-hf	236	227.41 ± 155.22	107.67 ± 81.07	8.63 ± 3.71	239.5 ± 170.31	160.38 ± 132.92	212.13 ± 163.73	183.9 ± 164.31
Llama2-13b-chat-hf	235	211.98 ± 150.72	102.86 ± 77.34	7.76 ± 3.44	217.17 ± 163.6	166.83 ± 131.14	194.08 ± 155.4	164.98 ± 153.9
Llama2-70b-chat-hf	225	238.97 ± 146.51	118.68 ± 81.02	7.97 ± 4.56	256.45 ± 157.17	190.53 ± 134.77	226.63 ± 154.74	198.96 ± 154.56
Mistral-7B-Instruct-v0.1	202	145.59 ± 120.75	72.48 ± 64.95	8.94 ± 7.43	153.63 ± 128.54	101.21 ± 102.59	134.58 ± 120.5	105.77 ± 120.43
Mixtral-8x7B-Instruct-v0.1	227	192.51 ± 137.21	92.37 ± 73.09	8.57 ± 3.83	203.58 ± 154.21	117.11 ± 99.72	174.14 ± 142.85	140.6 ± 143.09
Overall	2195	182.15 ± 137.82	87.93 ± 71.87	8.49 ± 4.98	199.43 ± 155.3	131.35 ± 119.36	171.08 ± 141.29	141.43 ± 140.77

Table 8: General statistics by LLM. **FKG**=Flesch-Kincaid Grade; **H**=human-written; **M**=machine-generated; **E**=expert-edited; **L**=`Llama3.1-70B-Instruct`-edited; **G**=`GPT-4o`-edited; **FKG**=Flesch-Kincaid Grade; **LD(M, H/E/L/G)**=Levenshtein distance. **L** and **G** are aggregated over three prompts (§3.3).

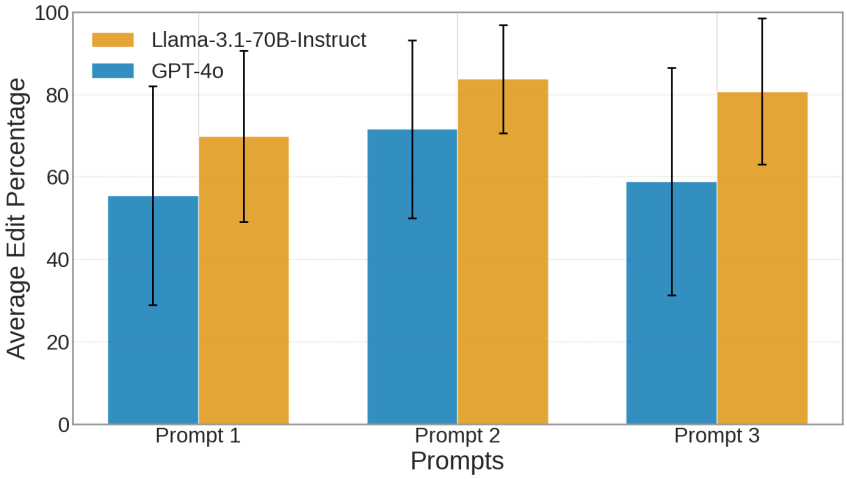

Figure 5: Comparison of average edit percentages between `GPT-4o` and `Llama3.1-70B-Instruct` across three different prompts. The bars represent the mean edit percentage for each prompt, with error bars indicating the standard deviation.

D Details on Annotation Survey

Survey: Editing AI-generates Responses

Answer the following questions related to editing AI-generated responses. The questions are divided into two groups. First, you are asked to judge general issues in the AI-generated responses based on how often you have observed them. Second, you are asked to share your strategies in the editing AI-generated responses in the free form.

How often do you observe the following issues in the AI-generated responses?

Rate each issue on a scale from 1 to 5, where 1 stands for "never," 3 stands for "sometimes," and 5 stands for "very often."

Q1: Repetitions. The same word, phrase, or sentence is repeated in the AI-generated text multiple times.

	1	2	3	4	5	
Never	○	○	○	○	○	**Very often**

Q2: Awkward phrasing. The AI-generated text contains awkward or unnatural phrasing that makes it difficult to read or understand. It contains generic and simplistic language or, on the contrary, uses unnecessarily sophisticated and outdated words and phrases.

	1	2	3	4	5	
Never	○	○	○	○	○	**Very often**

Q3: Tone and style. The tone and style of the AI-generated text do not align with the user intent.

	1	2	3	4	5	
Never	○	○	○	○	○	**Very often**

Q4: Grammar errors. The AI-generated text is ungrammatical and does not sound natural.

	1	2	3	4	5	
Never	○	○	○	○	○	**Very often**

Q5: Relevance. The AI-generated text does not fully follow the user's prompt or complete the target task.

	1	2	3	4	5	
Never	○	○	○	○	○	**Very often**

Q6: Code fragments. The AI-generated text includes fragments that resemble programming language and are not contextually relevant.

	1	2	3	4	5	
Never	○	○	○	○	○	**Very often**

Q7: Incomplete response. The AI-generated text is incomplete.

	1	2	3	4	5	
Never	○	○	○	○	○	**Very often**

Q8: Factual inconsistency. The facts described in the AI-generated texts are wrong.

	1	2	3	4	5	
Never	○	○	○	○	○	**Very often**

Q9: Incoherence. Segments of the AI-generated text are not connected or contradict each other.

	1	2	3	4	5	
Never	○	○	○	○	○	**Very often**

Table 9: A survey on AI-generated response issues and editing strategies (Part 1).

Answer the following questions in 2-5 sentences.

1. Do you have specific strategies for editing AI-generated texts?

2. In your experience, how does editing AI-generated texts compare with editing texts written by humans?

3. What difficulties or challenges have you encountered when editing the AI-generated texts?

Thank you for your time and efforts.

Leave a comment if you want to add something.

Table 10: A survey on AI-generated response issues and editing strategies (Part 2).

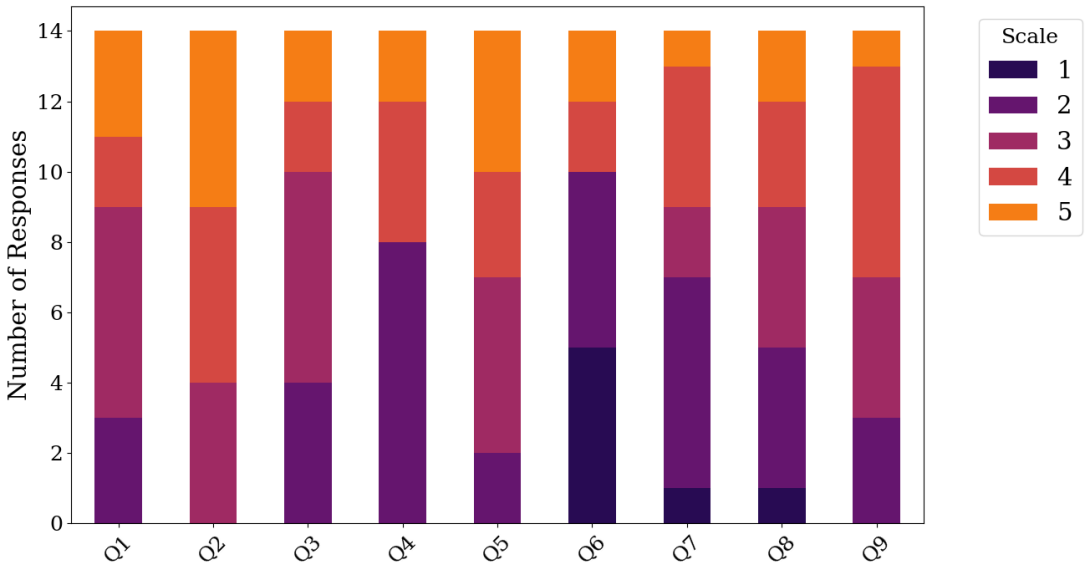

Figure 6: Results of the survey on issues in the AI-generated responses. **Q1**=Repetitions; **Q2**=Awkward phrasing; **Q3**=Tone and style. **Q4**=Grammar errors; **Q5**=Relevance; **Q6**=Code fragments; **Q7**=Incomplete response; **Q8**=Factual inconsistency; **Q9**=Incoherence.

E Details on MGT Detectors

Zero-shot MGT Detectors

- Binoculars utilizes the ratio of perplexities of a text obtained using two pre-trained LLMs to compute cross-perplexity. This relative measure of perplexity from the lens of two LLMs helps detect LLM generated texts.
- Log Probability utilizes the average of the log probabilities of each token in a text and classifies text with higher average log probability to be machine-generated. This follows the notion that the probabilities of the tokens as measured by the language model that generated it should be higher.
- Rank. In this approach, text with higher average of the ranks of each token in the text is determined to be machine-generated text. Rank is calculated by sorting the vocabulary tokens in decreasing order of likelihood.
- Log-Rank. By calculating the average of the observed log-ranks of all tokens, this method classifies text with higher average values as machine-generated.
- Entropy attributes higher entropy text as machine-generated. Entropy is calculated as the negative sum of the probability times log-probabilities of all tokens at each position of the text.
- DetectLLM has two variations: DetectLLM-LRR which replaces log-likelihood computation with log-rank, and DetectLLM-NPR which computes normalized perturbed log rank. The former is fast and efficient, while the latter is slow but with higher detection accuracy.
- DetectGPT leverages the structure of LLMs' probability curve to identify machine-generated texts. Specifically, it assumes that LLMs' generations tend to occupy negative curvature regions.

Pretrained MGT Detectors

- RADAR uses an adversarial learning setup in which a paraphraser LM learns to alter machine-generated text to evade detection. A detector LM then learns from this adversarially paraphrased text to better its performance.
- AIGC MPU augments LLM training with an additional length-sensitive multiscale positive-unlabeled (MPU) loss term, as well as a text multiscaling module to enhance the training data. The method works with the assumption that MGT can be formulated as a partial Positive-Unlabeled (PU) problem by assuming short-length machine texts as partially unlabeled.
- MAGE is a fine-tuned LongFormer model on 447K human-written and machine-generated texts collected from a wide range of sources.
- OpenAI's RoBERTa-based detectors: OpenAI released pretrained detectors using RoBERTa models (base and large) trained on text generated by GPT-2.

F Details on Empirical Evaluation Results

Detector		H vs. M	E vs. M	H vs. E	L vs. M	H vs. L	G vs. M	H vs. G
Zero-shot MGT Detectors								
Binoculars		93.64	82.84	76.19	75.26	81.80	74.44	77.03
Log Probability	GPT2-XL	74.41	67.21	63.02	72.21	56.50	68.20	61.55
	OPT-1.3B	82.13	70.03	70.28	76.52	63.23	70.35	71.00
	Falcon-7B	90.44	71.53	83.59	77.07	80.20	71.01	84.83
	Qwen2-7B	91.14	75.85	81.57	58.62	87.95	67.23	84.76
Rank	GPT2-XL	67.87	61.35	59.02	67.60	49.58	62.81	54.87
	OPT-1.3B	75.29	60.95	68.88	72.03	54.29	64.33	63.56
	Falcon-7B	83.47	62.21	78.61	72.42	65.54	64.37	74.84
	Qwen2-7B	81.74	64.23	73.14	54.74	78.87	59.49	76.01
Log-Rank	GPT2-XL	76.28	68.08	63.65	74.53	54.43	69.61	61.05
	OPT-1.3B	80.64	69.97	68.07	77.28	58.28	71.04	67.01
	Falcon-7B	89.56	71.44	82.15	77.95	76.94	71.55	82.50
	Qwen2-7B	90.44	74.75	80.79	58.25	87.22	66.50	84.01
Entropy	GPT2-XL	36.66	42.43	43.56	27.97	60.51	33.93	53.34
	OPT-1.3B	33.92	42.95	39.53	26.89	58.58	33.40	50.31
	Falcon-7B	19.50	37.62	23.40	25.86	38.92	30.69	32.09
	Qwen2-7B	8.26	26.53	14.41	42.18	10.31	34.35	12.36
DetectLLM Likelihood Log-Rank Ratio	GPT2-XL	77.86	69.39	63.26	78.35	47.81	71.95	57.65
	OPT-1.3B	71.68	67.46	57.28	77.28	40.81	71.08	50.44
	Falcon-7B	83.28	68.72	73.11	78.37	60.35	71.06	68.87
	Qwen2-7B	84.26	68.21	73.92	56.07	80.81	62.14	77.37
DetectLLM Normalized Perturbed Log-Rank	GPT2-XL	71.76	66.26	69.49	58.19	70.65	53.20	71.37
	OPT-1.3B	73.27	66.76	70.56	61.37	71.64	54.51	72.79
	Falcon-7B	77.85	68.01	73.84	63.70	75.43	56.75	77.00
	Qwen2-7B	81.68	71.68	75.23	63.20	79.00	56.41	80.81
DetectGPT	GPT2-XL	72.51	68.19	71.51	70.82	71.81	68.17	71.98
	OPT-1.3B	74.50	70.40	73.04	77.60	72.83	72.93	73.30
	Falcon-7B	76.74	69.53	74.95	77.85	74.29	74.60	74.68
	Qwen2-7B	78.34	73.19	75.48	78.25	75.29	75.53	75.62
RADAR		64.13	55.84	60.73	22.19	81.38	27.00	78.27
MAGE		78.66	59.90	71.01	67.36	64.61	72.90	59.47
AIGC MPU		75.30	71.70	55.86	56.28	72.17	47.66	77.83
OpenAI RoBERTa-base		74.60	71.41	52.85	71.32	56.89	70.81	58.30
OpenAI RoBERTa-large		70.45	69.87	51.16	63.07	60.52	56.74	66.00

Table 11: The AUROC scores (%) of the 33 MGT detectors' configurations on Beemo in the **Generation** category. A random classifier has an AUROC of 50%. **H**=human-written; **M**=machine-generated; **E**=expert-edited; **L**=`Llama3.1-70B-Instruct`-edited; **G**=`GPT-4o`-edited. **L** and **G** are aggregated over three prompts (§3.3).

Detector		H vs. M	E vs. M	H vs. E	L vs. M	H vs. L	G vs. M	H vs. G
Zero-shot MGT Detectors								
Binoculars		85.97	81.31	56.46	74.09	69.67	70.06	69.48
Log Probability	GPT2-XL	76.15	75.18	50.95	70.37	60.05	60.10	70.79
	OPT-1.3B	79.22	76.83	53.02	73.49	61.33	62.16	73.14
	Falcon-7B	92.30	75.66	80.35	72.94	86.18	58.85	92.81
	Qwen2-7B	93.68	77.76	83.45	59.25	90.27	68.51	86.86
Rank	GPT2-XL	63.60	62.21	51.78	62.24	51.17	56.24	57.96
	OPT-1.3B	66.54	64.21	53.64	65.54	51.78	56.92	61.35
	Falcon-7B	77.41	62.01	68.83	65.28	64.02	54.43	75.68
	Qwen2-7B	77.53	60.33	72.00	53.44	75.69	56.89	73.84
Log-Rank	GPT2-XL	77.45	76.13	51.31	72.50	59.18	61.92	70.68
	OPT-1.3B	78.12	76.84	51.38	74.30	58.45	63.33	70.66
	Falcon-7B	91.79	75.65	79.41	73.96	84.35	60.07	91.81
	Qwen2-7B	93.30	77.78	82.40	59.26	89.67	68.52	86.03
Entropy	GPT2-XL	35.06	34.36	50.88	35.35	48.77	44.33	39.67
	OPT-1.3B	37.08	35.25	51.85	34.43	51.53	42.63	43.20
	Falcon-7B	12.26	32.29	20.55	31.01	19.34	40.37	13.64
	Qwen2-7B	6.10	24.96	12.57	41.65	8.26	33.31	10.41
DetectLLM Likelihood Log-Rank Ratio	GPT2-XL	74.80	73.75	50.88	72.67	54.49	62.71	65.20
	OPT-1.3B	69.69	72.35	46.67	72.13	48.71	62.79	59.14
	Falcon-7B	84.28	71.15	69.35	72.70	70.31	61.47	79.23
	Qwen2-7B	86.58	72.14	71.58	57.38	81.58	64.76	76.58
DetectLLM Normalized Perturbed Log-Rank	GPT2-XL	55.89	67.17	53.56	59.94	54.54	50.94	55.97
	OPT-1.3B	56.81	68.31	54.02	64.00	54.75	52.58	56.60
	Falcon-7B	63.13	68.05	58.78	64.72	59.87	52.75	62.89
	Qwen2-7B	68.05	72.53	60.61	64.39	64.04	51.52	67.94
DetectGPT	GPT2-XL	58.80	69.55	55.31	62.10	56.40	58.44	57.30
	OPT-1.3B	61.25	75.15	56.14	69.21	57.05	63.20	58.59
	Falcon-7B	65.51	71.80	60.04	66.81	60.93	62.49	62.44
	Qwen2-7B	68.14	74.24	61.11	62.56	64.31	62.64	64.70
RADAR		47.78	46.69	51.43	37.84	57.84	40.62	55.58
MAGE		76.05	64.39	63.31	58.99	70.21	62.23	67.34
AIGC MPU		75.33	79.87	40.69	85.27	31.55	80.63	39.59
OpenAI RoBERTa-base		76.68	76.55	48.95	80.72	41.85	78.99	43.93
OpenAI RoBERTa-large		72.60	71.87	51.46	78.94	40.87	73.95	47.36

Table 12: The AUROC scores (%) of the 33 MGT detectors' configurations on Beemo in the **Open QA** category. A random classifier has an AUROC of 50%. **H**=human-written; **M**=machine-generated; **E**=expert-edited; **L**=`Llama3.1-70B-Instruct`-edited; **G**=`GPT-4o`-edited. **L** and **G** are aggregated over three prompts (§3.3).

Detector		H vs. M	E vs. M	H vs. E	L vs. M	H vs. L	G vs. M	H vs. G
Zero-shot MGT Detectors								
Binoculars		80.25	76.05	55.97	38.28	88.65	39.19	85.97
Log Probability	GPT2-XL	70.21	64.29	57.11	54.74	67.61	59.37	61.70
	OPT-1.3B	74.95	66.61	60.53	57.90	70.88	59.25	67.11
	Falcon-7B	87.88	63.56	81.00	54.90	88.29	55.22	84.99
	Qwen2-7B	90.41	67.70	81.50	55.90	87.44	61.80	84.47
Rank	GPT2-XL	57.24	51.93	55.43	53.79	54.16	53.28	54.16
	OPT-1.3B	62.02	53.70	58.76	57.24	55.81	52.42	59.73
	Falcon-7B	70.81	49.83	70.69	55.17	66.15	49.63	70.25
	Qwen2-7B	70.71	51.04	70.21	50.35	70.54	50.69	70.38
Log-Rank	GPT2-XL	69.88	63.95	57.05	56.47	65.67	60.01	60.51
	OPT-1.3B	73.26	65.56	59.45	58.76	67.94	59.99	64.40
	Falcon-7B	87.02	62.35	80.41	56.08	86.54	56.16	83.39
	Qwen2-7B	90.04	66.86	81.06	55.62	87.05	61.24	84.05
Entropy	GPT2-XL	35.56	42.34	42.83	42.84	41.56	44.49	40.63
	OPT-1.3B	34.05	42.23	41.07	41.60	40.85	44.02	39.29
	Falcon-7B	16.02	45.02	18.24	45.45	15.79	47.90	16.29
	Qwen2-7B	5.78	29.58	13.81	43.19	8.46	36.39	11.13
DetectLLM Likelihood Log-Rank Ratio	GPT2-XL	64.75	59.97	55.57	59.14	56.95	59.89	54.74
	OPT-1.3B	64.29	60.76	54.67	60.52	55.65	60.95	53.84
	Falcon-7B	78.55	56.93	73.60	58.83	73.77	58.20	71.78
	Qwen2-7B	83.14	62.02	74.66	54.01	80.31	58.01	77.49
DetectLLM Normalized Perturbed Log-Rank	GPT2-XL	61.93	56.39	60.78	45.90	62.60	49.25	62.08
	OPT-1.3B	63.48	59.90	61.74	51.23	63.23	49.81	63.60
	Falcon-7B	68.11	58.33	66.39	49.17	68.32	47.17	68.82
	Qwen2-7B	72.17	66.17	68.07	45.76	72.91	47.08	72.89
DetectGPT	GPT2-XL	60.80	59.71	59.88	46.92	62.09	53.66	60.98
	OPT-1.3B	62.98	66.23	60.66	52.93	62.99	55.29	62.72
	Falcon-7B	67.54	67.08	64.56	53.33	67.41	56.32	66.91
	Qwen2-7B	70.40	72.84	65.65	48.42	71.07	56.65	69.44
RADAR		41.12	46.81	44.16	41.96	46.88	45.31	44.75
MAGE		69.88	59.94	61.09	46.81	72.77	48.72	70.96
AIGC MPU		71.85	68.09	54.00	70.64	50.94	64.91	56.09
OpenAI RoBERTa-base		49.03	55.56	42.30	55.24	42.88	55.23	43.43
OpenAI RoBERTa-large		47.66	52.87	44.33	55.00	42.13	52.18	45.54

Table 13: The AUROC scores (%) of the 33 MGT detectors' configurations on Beemo in the **Summarize** category. A random classifier has an AUROC of 50%. **H**=human-written; **M**=machine-generated; **E**=expert-edited; **L**=`Llama3.1-70B-Instruct`-edited; **G**=`GPT-4o`-edited. **L** and **G** are aggregated over three prompts (§3.3).

Detector		H vs. M	E vs. M	H vs. E	L vs. M	H vs. L	G vs. M	H vs. G
Zero-shot MGT Detectors								
Binoculars		83.83	77.66	61.90	49.35	82.51	46.54	81.91
Log Probability	GPT2-XL	66.56	61.63	56.03	63.39	54.23	62.44	54.77
	OPT-1.3B	72.73	62.53	62.20	65.91	59.42	62.31	62.24
	Falcon-7B	84.17	62.26	77.70	64.38	77.68	61.58	78.08
	Qwen2-7B	84.78	65.91	75.18	55.30	81.58	60.61	78.38
Rank	GPT2-XL	56.44	52.87	54.43	57.68	48.82	56.91	49.23
	OPT-1.3B	64.11	53.05	62.36	61.82	53.20	56.43	58.19
	Falcon-7B	72.94	53.61	71.12	61.98	62.50	56.47	67.63
	Qwen2-7B	73.23	56.84	68.20	52.28	71.55	54.56	69.88
Log-Rank	GPT2-XL	66.19	61.05	56.18	64.29	52.51	62.85	53.66
	OPT-1.3B	70.04	61.91	59.80	66.24	55.44	62.98	58.26
	Falcon-7B	82.13	61.80	75.54	65.03	73.98	62.24	74.56
	Qwen2-7B	83.28	65.37	73.57	55.12	80.04	60.25	76.81
Entropy	GPT2-XL	45.41	47.10	48.15	36.49	58.94	41.54	53.82
	OPT-1.3B	42.17	47.53	44.18	34.54	57.56	40.72	51.19
	Falcon-7B	26.84	46.09	28.43	36.53	37.46	41.45	33.24
	Qwen2-7B	14.62	34.95	22.56	44.98	17.27	39.97	19.91
DetectLLM Likelihood Log-Rank Ratio	GPT2-XL	63.75	58.94	55.98	65.69	47.73	63.35	49.88
	OPT-1.3B	58.88	58.20	51.28	65.42	43.10	63.18	45.27
	Falcon-7B	71.86	58.70	65.68	65.17	58.81	63.02	60.02
	Qwen2-7B	74.21	61.41	65.15	53.80	71.19	57.61	68.17
DetectLLM Normalized Perturbed Log-Rank	GPT2-XL	64.30	62.54	62.24	50.66	64.13	50.25	64.31
	OPT-1.3B	65.81	63.51	63.35	52.73	65.25	50.47	65.87
	Falcon-7B	70.27	63.41	66.98	53.06	69.74	51.08	70.34
	Qwen2-7B	73.57	69.32	67.83	48.16	74.28	49.63	74.00
DetectGPT	GPT2-XL	64.00	63.18	61.76	60.89	62.23	61.87	61.97
	OPT-1.3B	66.43	66.27	63.51	66.98	63.61	65.36	63.79
	Falcon-7B	69.91	64.56	67.07	69.00	66.68	68.00	66.68
	Qwen2-7B	71.48	69.51	67.43	64.94	68.96	67.85	68.12
RADAR		57.24	54.46	53.01	24.69	77.32	27.91	74.72
MAGE		70.67	59.56	62.05	59.23	62.77	60.29	61.34
AIGC MPU		71.29	69.80	52.30	52.49	70.62	47.35	73.71
OpenAI RoBERTa-base		65.61	64.11	51.08	58.16	59.77	58.70	58.83
OpenAI RoBERTa-large		63.01	62.77	49.84	56.49	57.51	52.17	61.49

Table 14: The AUROC scores (%) of the 33 MGT detectors' configurations on Beemo in the **Rewrite** category. A random classifier has an AUROC of 50%. **H**=human-written; **M**=machine-generated; **E**=expert-edited; **L**=`Llama3.1-70B-Instruct`-edited; **G**=`GPT-4o`-edited. **L** and **G** are aggregated over three prompts (§3.3).

Detector		H vs. M	E vs. M	H vs. E	L vs. M	H vs. L	G vs. M	H vs. G
Zero-shot MGT Detectors								
Binoculars		74.83	76.09	49.49	52.07	75.44	47.29	76.63
Log Probability	GPT2-XL	70.06	63.69	57.60	54.76	68.95	54.42	67.08
	OPT-1.3B	71.61	65.11	58.35	56.31	69.54	54.34	68.67
	Falcon-7B	83.29	60.60	76.37	53.72	83.81	51.56	82.89
	Qwen2-7B	87.08	64.19	76.77	54.73	83.64	59.46	80.21
Rank	GPT2-XL	54.86	54.46	50.74	52.09	53.42	52.24	52.94
	OPT-1.3B	58.22	57.86	51.15	55.57	54.44	53.75	55.08
	Falcon-7B	64.51	55.45	59.75	54.25	61.84	50.23	64.21
	Qwen2-7B	65.20	55.52	59.51	51.84	63.30	53.68	61.41
Log-Rank	GPT2-XL	70.84	63.71	58.69	56.00	68.61	54.63	67.50
	OPT-1.3B	70.90	64.72	57.83	56.64	68.26	54.80	67.49
	Falcon-7B	82.52	60.46	75.92	54.05	82.97	52.14	81.66
	Qwen2-7B	87.11	64.87	76.75	54.96	83.66	59.91	80.20
Entropy	GPT2-XL	31.34	42.60	37.21	46.67	31.46	48.52	31.83
	OPT-1.3B	31.09	41.61	38.04	46.15	31.67	48.47	31.60
	Falcon-7B	18.33	46.14	19.95	49.14	16.26	50.68	16.97
	Qwen2-7B	10.18	38.04	15.74	46.01	12.03	42.03	13.89
DetectLLM Likelihood Log-Rank Ratio	GPT2-XL	69.18	60.24	59.98	57.93	64.22	54.47	65.69
	OPT-1.3B	65.08	61.43	55.58	56.15	61.35	54.87	61.09
	Falcon-7B	74.48	58.24	68.55	53.86	73.54	53.23	72.05
	Qwen2-7B	80.44	62.98	70.09	54.33	76.99	58.65	73.54
DetectLLM Normalized Perturbed Log-Rank	GPT2-XL	62.14	62.06	60.72	56.56	61.28	55.24	61.61
	OPT-1.3B	62.62	61.25	61.30	56.50	61.69	52.51	62.34
	Falcon-7B	64.85	58.15	63.51	52.37	64.32	52.22	64.59
	Qwen2-7B	68.00	66.66	64.54	52.47	67.06	55.54	67.17
DetectGPT	GPT2-XL	63.87	60.33	61.14	52.12	62.50	53.01	62.68
	OPT-1.3B	65.05	63.49	62.05	55.13	63.12	52.66	63.94
	Falcon-7B	67.20	60.53	64.05	50.31	65.62	52.63	65.71
	Qwen2-7B	68.80	66.55	64.67	46.95	67.81	55.01	66.92
RADAR		39.41	52.57	34.99	71.24	13.34	74.10	11.02
MAGE		71.88	64.21	58.50	46.43	74.24	43.57	77.66
AIGC MPU		76.62	73.45	52.60	68.83	56.73	65.70	58.58
OpenAI RoBERTa-base		57.17	62.46	42.64	65.45	38.65	70.30	33.11
OpenAI RoBERTa-large		48.72	57.38	40.69	74.68	22.41	69.15	28.56

Table 15: The AUROC scores (%) of the 33 MGT detectors' configurations on Beemo in the **Closed QA** category. A random classifier has an AUROC of 50%. **H**=human-written; **M**=machine-generated; **E**=expert-edited; **L**=`Llama3.1-70B-Instruct`-edited; **G**=`GPT-4o`-edited. **L** and **G** are aggregated over three prompts (§3.3).