On the Detectability of ChatGPT Content: Benchmarking, Methodology, and Evaluation Through the Lens of Academic Writing

Zeyan Liu
EECS/I2S, The University of Kansas
Lawrence, KS, USA
zyliu@ku.edu

Zijun Yao
EECS/I2S, The University of Kansas
Lawrence, KS, USA
zyao@ku.edu

Fengjun Li
EECS/I2S, The University of Kansas
Lawrence, KS, USA
fli@ku.edu

Bo Luo
EECS/I2S, The University of Kansas
Lawrence, KS, USA
bluo@ku.edu

ABSTRACT

With ChatGPT under the spotlight, utilizing large language models (LLMs) to assist academic writing has drawn a significant amount of debate in the community. In this paper, we aim to present a comprehensive study of the detectability of ChatGPT-generated content within the academic literature, particularly focusing on the abstracts of scientific papers, to offer holistic support for the future development of LLM applications and policies in academia. Specifically, we first present GPABench2, a benchmarking dataset of over 2.8 million comparative samples of human-written, GPT-written, GPT-completed, and GPT-polished abstracts of scientific writing in computer science, physics, and humanities and social sciences. Second, we explore the methodology for detecting ChatGPT content. We start by examining the unsatisfactory performance of existing ChatGPT detecting tools and the challenges faced by human evaluators (including more than 240 researchers or students). We then test the hand-crafted linguistic features models as a baseline and develop a deep neural framework named CheckGPT to better capture the subtle and deep semantic and linguistic patterns in ChatGPT written literature. Last, we conduct comprehensive experiments to validate the proposed CheckGPT framework in each benchmarking task over different disciplines. To evaluate the detectability of ChatGPT content, we conduct extensive experiments on the transferability, prompt engineering, and robustness of CheckGPT.

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**; **Machine learning**; • **Security and privacy**;

KEYWORDS

AIGC Detection, Responsible AI, Large Language Models

CCS '24, October 14–18, 2024, Salt Lake City, UT, USA

ACM ISBN 979-8-4007-0636-3/24/10.
https://doi.org/10.1145/3658644.3670392

ACM Reference Format:
Zeyan Liu, Zijun Yao, Fengjun Li, and Bo Luo. 2024. On the Detectability of ChatGPT Content: Benchmarking, Methodology, and Evaluation Through the Lens of Academic Writing. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS '24), October 14–18, 2024, Salt Lake City, UT, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3658644.3670392

1 INTRODUCTION

The recently debuted Large Language Model (LLM) - ChatGPT has shown an impressive ability to generate sophisticated texts with human-like language style and quality. Concerns have been raised that the LLM-generated content (LLM-content) can be misused to abuse the trust systems we have, e.g., in cheating and plagiarism [49, 94], or in phishing and romance scams [33, 87]. While many academic institutes and publishers have announced policies on the usage of LLM-content, it is hard to enforce such policies unless we have a tool to accurately detect it.

LLM-content detection can be challenging due to the unique characters of LLM/ChatGPT: (1) like a human conversationalist, the output of LLM has a relevant, organized response with a low level of grammar errors; (2) the sampling mechanism of LLM output ensures that the choice of words is stochastic; therefore, the responses are distinct even with multiple repeated inquiries; and (3) the misuse of LLM-content can be stealthy since users can invoke ChatGPT to polish human writing. Facing these challenges, existing LLM detectors are less effective, especially in detecting GPT-polished text (Section 2.2). Some experiences in identifying LLM-content have been reported in the literature, e.g., ChatGPT output tends to be more objective, formal, focused, and fluent than human-content [35, 63]. However, a holistic investigation of the distinguishability of LLM-content is still missing.

To this end, in this paper, we first identify three typical cases of using or abusing ChatGPT in academic writing: *composing*, *completing*, and *polishing*. We pick three representative disciplines for investigation: *computer science* for technical/engineering writing, *physics* for science writing, and *humanities and social sciences* for liberal arts writing. To address a range of complex real-world scenarios, we used four different prompt patterns for each task across each discipline and collected a dataset, *GPABench2*, with 2.8 million human-written and ChatGPT-generated academic abstracts.

Next, we conducted an extensive field study with human evaluators to assess if they can distinguish LLM-content accurately provided with a mixture of true and false samples. The cohort of 155 evaluators, consisting of university faculty, researchers, and graduate students, proves that the recognition of LLM-content is difficult to visually inspect based on language appearance, with or without individual experiences of writing research articles. A second cohort of 87 evaluators was provided with ground truth data as references: pairs of human-written and GPT-generated abstracts. While their capacities to identify human- and GPT-generated text were improved, the accuracy is still too low for reliable detection. In addition, we test 10+ state-of-the-art online or open-source detectors on GPABench2, e.g., GPTZero, and show that they demonstrate modest to poor performance, except for the language-model-based detectors, like BERT and RoBERTa, which archives significantly better result but requires excessive training efforts.

Last, we develop and evaluate a language-model-based detection framework, named CheckGPT, to explore the feasibility of building automated tools for LLM-content detection in a niche area. CheckGPT has the following advantages: (1) it is a black-box solution that leverages deep learning frameworks to achieve a high accuracy compared to human and state-of-the-art (SOTA) LLM-content detectors. (2) CheckGPT adopts a model-agnostic setting that can be treated as a plugin to most pre-trained language models (e.g., BERT). As a result, the number of parameters to be trained can be largely reduced. (3) Due to its ability to learn generalized semantic patterns of LLM-content, CheckGPT shows a strong potential for domain transfer that only requires minimum fine-tuning efforts. Finally, we conduct comprehensive experiments to demonstrate CheckGPT's design goals and strengths: its performance on the GPABench2 dataset, its transferability to new domains, new models, and new prompts, and its robustness against continuous updates of ChatGPT and against human-modified GPT writing.

In summary, our main contributions are:

- We present GPABench2, a cross-disciplinary corpus consisting of human-written, GPT-written, GPT-completed, and GPT-polished research paper abstracts. GPABench2 has the potential to serve as a cornerstone for benchmarking GPT detectors and a valuable resource to assist in the design of new detecting methods.
- We evaluate the SOTA GPT detectors with GPABench2 and present their performance. Meanwhile, with a user study of 242 participants, we show that human evaluators are unable to distinguish between human-written and GPT-generated academic writing. This incapability persists regardless of experience, knowledge, and reference.
- We present CheckGPT, a deep-learning-based and model-agnostic ChatGPT-content detector with validated benefits of affordability, transferability, and Robustness[1]. We demonstrate the outstanding performance (~99% average accuracy) of CheckGPT with extensive experiments.

Ethical Considerations. The user study in Section 3 was reviewed and approved by the Human Research Protection Program at the University of Kansas under project ID STUDY00150100. All the paper abstracts collected in Section 2 are open to the public. We invoked ChatGPT's API (with payment) to collect the GPT-generated abstracts. The GPABench2 dataset and the CheckGPT tool have been open-sourced. The academic community is actively discussing how AI writing assistance tools may pose challenges to research and education [3, 70, 88]. OpenAI also posted their perspectives on the education-related risks and opportunities [76]. Last, the same as other ChatGPT detectors, CheckGPT should not be used as the *sole evidence* to accuse wrongdoings. In particular, false positives (human-written content being labeled as GPT-generated) may potentially harm the author. Such risk should be carefully managed in the adoption of CheckGPTor any other GPT detector.

The rest of the paper is organized as follows: We introduce the GPABench2 dataset and evaluate the open-source and commercial ChatGPT detectors in Section 2, followed by a user study of ChatGPT-content detection in Section 3. We present the technical details of CheckGPT in Section 4 and experimental results in Section 5. Finally, we survey the related literature in Section 6, and conclude the paper in Section 7.

2 GPABENCHMARK: GPT CORPUS FOR ACADEMIA

2.1 The GPABench2 Dataset

The concern of LLM/ChatGPT misuse has been raised widely in academia because (1) academic integrity violations such as cheating and plagiarism will become *easy-to-conduct* and *hard-to-detect*. (2) False and redundant information may flood the publication systems. Stack Overflow had to ban LLM-generated posts to ensure that visitors can find reliable answers efficiently [93]. Academic conferences started to ban LLM-generated texts (e.g., ICML) or require disclosure (e.g., ACL, Nature, and RSC). However, without effective mechanisms to detect GPT-generated content and benchmark datasets to train/evaluate the detectors, such discussions and policies become meaningless.

The state-of-the-art corpora for ChatGPT text classification mainly focus on question-and-answer (Q&A) dialogues [35, 37]. While the Q&A datasets align with the original design of ChatGPT as an interactive "Chat" interface, they become insufficient as the usage scenarios of ChatGPT have significantly expanded beyond chat. When ChatGPT is adopted for academic writing, such as essays, reports, and even research papers [25, 94], the generated text akin to academic writing style is often objective, formal, and focused [35, 63], posing more challenges to the detectors: (1) Human conversations often contain subjective opinions, personal biases, and emotions. However, such clues are significantly less observed in academic writing, which is generally formal and objective [7, 8, 35, 46]. (2) Grammatical errors and inconsistencies in human-generated texts may serve as meaningful indicators; however, they are less likely to occur in academic writing, which is expected to meet higher standards for fluency, clarity, and grammatical correctness [40, 67]. Also, academic writers typically adopt a comprehensive and organized style [15] akin to the one generated by ChatGPT [35]. (3) Academic abstracts typically explore domain-specific and highly-specialized topics [46], which lead to a significantly different term distribution from conversational dialogues.

With the unique characteristics of academic writing, a new ChatGPT-content corpus is necessary for benchmarking GPT detectors and assisting in the design of detectors. In this paper, we

[1]We share CheckGPT at https://github.com/liuzey/CheckGPT-v2

introduce *GPABench2* (GPABenchmark version 2), a large-scale GPT-generated text corpus for academic writing.

We first collected research papers (title and abstracts) from three disciplines: *computer science* (CS) abstracts from top-tier conference proceedings and arXiv, *physics* (PHX) from arXiv, and *humanities and social sciences* (HSS) from Springer's SSRN including history, philosophy, sociology, and psychology disciplines. The three fields spread across "hard science" (math-intensive) and "soft science" disciplines. For CS and Physics, we chose papers published or posted on or before 2019 (before the release of GPT-3) to ensure that they were all human-written, as researchers may have adopted GPT-3 to assist their writings before the web-based ChatGPT was released[2]. Eventually, we collected 50,000 papers from each discipline.

We define three tasks based on the most representative scenarios where LLMs are used/misused in academic writing:

- **Task 1. GPT-written full abstracts (GPT-WRI or WRI).** The author gives a title to ChatGPT and asks it to write the complete abstract from scratch.
- **Task 2. GPT-completed abstracts (GPT-CPL or CPL).** Text completion is considered a conventional function of LLMs: The author provides a few sentences to ChatGPT, who follows the logic to complete the rest of the paragraph. We mimic this scenario: for an abstract with s sentences, the first $s/2$ sentences are provided to ChatGPT, based on which it *completes the abstract with w words*, where w is the word count in the second half of the original abstract. Hence, the GPT-CPL abstract will have *approximately* the same length as the human-written abstract.
- **Task 3. GPT-polished abstracts (GPT-POL or POL).** We provide the entire abstract to ChatGPT for polishing. ChatGPT rewrites the text sentence-by-sentence and generates a revised abstract. Invoking ChatGPT multiple times will generate different results for the same seed abstract.

In this paper, we use *human-written* abstracts (HUM) to denote abstracts that are completely written by human authors. We use *GPT-Generated* abstract (GPT-GEN) to denote all three categories of GPT-content described above: GPT-WRI, -CPL, and -POL.

We applied prompt engineering in data collection to ensure a broad coverage of ChatGPT use cases. We studied popular prompt patterns [105] and prompt guidelines [1, 2, 48, 82] in the literature and crafted four distinct prompts for each task, denoted as Prompts 1 to 4 (presented in Appendix **??**): Prompt 1 is a popular but straightforward zero-shot prompt. Prompt 2 integrates the contexts to outline the scope of a specific discipline. Prompt 3 uses the role-playing technique to specify a "persona", e.g., "an expert paper writer in computer science". Prompt 4 provides detailed requirements and instructions to guide ChatGPT. These prompts represent four use cases with increasing levels of knowledge fed to ChatGPT.

We invoke ChatGPT (gpt-3.5-turbo) through OpenAI's API to generate abstracts at the cost of 0.2 cents per 1,000 tokens. The default query rate is 200 requests per minute. In three months, we collected 50,000 samples for each prompt, task, and discipline, as the GPABench2 Main Dataset (1.8 million total GPT-GEN samples). We further adopted ten advanced prompting techniques, e.g., chain-of-thought and in-context prompt learning, to generate 435K additional testing samples (Sec. 5.8). Eventually, GPABench2 contains 2.385M total samples (2.235M GPT-GEN and 0.15M HUM).

[2]GPT-3 was first released in 06/2020, access to the test release was by-invitation-only until 11/2021 when the API was made publicly accessible.

Table 1: Performance of commercial GPT detectors on GPABench2. Red: detection accuracy <50%, or average score on the wrong side of the decision threshold. T1/2/3: Task 1/2/3. GPT-WRI/CPL/POL: GPT-written/completed/polished abstracts.

	T1. GPT-WRI			T2. GPT-CPL			T3. GPT-POL		
	CS	PHX	HSS	CS	PHX	HSS	CS	PHX	HSS
(a) Classification accuracy (in %) of GPTZero.									
GPT	30.3	25.3	72.0	17.0	6.0	43.7	1.7	2.3	20.3
Human	99.3	99.7	100	99.7	99.7	94.3	99.7	95.7	95.7
(b1) Detection accuracy (in %) of ZeroGPT									
GPT	67.4	68.4	92.3	25.3	10	62.4	3.3	2.7	24.7
Human	100	98.4	95	99.7	99.7	94.7	98.3	98.6	92.7
(b2) Average score reported by ZeroGPT. 0:human, 8:GPT									
GPT	5.43	5.39	7.41	2.26	0.97	4.97	0.35	0.29	2.15
Human	0.09	0.13	0.52	0.08	0.04	0.47	0.20	0.14	0.64
(c1) Detection accuracy (in %) of OpenAI's detector									
GPT	80.7	70	63	63.7	23.7	27.3	6.3	4.3	6
Human	51.0	69.7	84.0	35.3	59.7	79.6	50.7	69.0	88.0
(c2) Average score reported by OpenAI. 0:human, 4:GPT									
GPT	3.11	2.89	2.72	2.70	2.12	2.04	1.75	1.59	1.52
Human	1.42	1.17	0.59	1.71	1.35	0.68	1.38	1.14	0.52

Advanced Prompt Engineering and Additional Testing Data. Research efforts on prompt engineering aim to guide or improve the design of ChatGPT prompts [22, 105]. We adopt six approaches that are widely adopted in the community: (1) Zero-shot Chain-of-Thought Prompting (ZC, [52]) enforces step-by-step reasoning with specific trigger phrases like "Let's think step by step." (2) Automatic Prompt Engineer (APE, [112]) automates the creation and selection of prompts using iterative optimization. (3) Self-critique Prompting (SCP, [69]) employs GPT to evaluate its own responses and provide feedback. (4) Few-shot Prompting (FSP, [11]) conditions the model using examples or demonstrations. (5) Least-to-Most Prompting (LMP, [111]) parses a problem into simpler subproblems. (6) Generated Knowledge Prompting (GKP, [60]) starts the prompt with relevant information generation. We also adopt four prompt refinement methods: (1) Prompt Perfect (PP, [83]). (2) GPT-generated Prompts (GP, [92]). (3) Meta Prompts (MP, [31]). (4) Instruction Induction (II, [43], not for Task 3). We use each of these methods to write, complete, and polish 5,000 abstracts from each discipline (please refer to Appendix **??** for details). In summary, we collected 435K additional samples to be used for testing.

2.2 Benchmarking Online and Open-source ChatGPT Detectors

Several online commercial tools have been developed to detect AI-generated text. We are especially interested in them because they are the most accessible and easy-to-use detectors for ordinary users. We evaluate the accuracy of three representative online ChatGPT tools, GPTZero [99], ZeroGPT [110], and OpenAI's classifier [75],

over GPABench2. Due to a lack of API, slow responses, and high cost, we cannot run large-scale experiments. Instead, we randomly sampled 300 pairs of human-written and the corresponding GPT-generated abstracts for each task in each discipline, i.e., 2,400 pairs in total, and fed them to each detector. Their performance is summarized in Table 1. Note that, in Task 2 (GPT-completed abstracts), we only input the second half of each abstract to the detectors.

From the performance summary in Table 1, we have three observations: (1) all three detectors demonstrated modest to poor accuracy for GPT-GEN content; (2) the detectors have tendencies to classify GPT-generated text as human-written; and (3) the detection accuracy for GPT-GEN decreases significantly from Task 1 (GPT-WRI) to Task 3 (GPT-POL). Note that these experiments are not intended to compare with CheckGPT. Since these models are not explicitly trained with academic data, their inaccuracy can be excused, and directly comparing them with CheckGPT is unfair. Rather, we intend to use the results as a motivation for CheckGPT–generic detectors struggle in specific tasks, indicating limited transferability. The gap highlights the need for effective detectors for this niche domain, which has the potential to transfer to related domains.

A number of detectors have been proposed for LLM/ChatGPT-generated text. For a comparative study, we adopted 15 open-source detectors in the literature. Based on the design philosophy, we further categorize them into three groups: (a) *Pre-trained detectors*. We directly adopt the trained models: HC3-Perplexity (HC3-PPL) [35], HC3-GLTR [35], HC3-Roberta (HC3-RBT) [35], OpenAI-Roberta (OpenAI-RBT) [77, 91]. (b) *Statistics-based detectors*. They analyze the statistical differences between LLM and human-written text: Histogram-of-Likelihood Ranks (HLR) [30], Rank [73], Log-Rank [73], Total Probability (TP) [91] Perplexity (PPL) [35], Entropy [30, 73], DetectGPT [73]. (c) *Fine-tuned Language Models*. They fine-tune the pre-trained language models for detection: BERT [47], DistillBERT [37], and RoBERTa [35, 63, 68, 104]. We also include GPT-2, which is even larger with 355M parameters.

The statistics- and fine-tuning-based detectors are trained/tuned with the entire training set in GPABench2. We evaluate all the detectors on GPABench2 with samples generated by Prompt 1. From the results presented in Table 2, we observe the following: (1) The pre-trained detectors work poorly on GPABench2. The failure of the OpenAI detector can be attributed to the discrepancy in the target model, as it was designed for GPT-2. The poor performance of HC3 detectors indicates the ineffectiveness of statistics-based methods (HC3-PPL, HC3-GLTR) and the limited transferability of RoBERTa (HC3-RBT). (2) The statistics-based detectors provide satisfactory performance in Task 1; however, they become mostly ineffective in Tasks 2 and 3. These results also indicate that GPT-polished content demonstrates significantly more statistical similarities to human-written text, especially at the lexicon level, e.g., word distributions, complexity, etc. (3) Fine-tuned language models with a native classification layer provide outstanding performance in most tasks, with slightly lower accuracy in Task 3. However, training or tuning the full BERT, RoBERTa, or GPT models is computationally expensive, e.g., it takes over 1,000 seconds to fine-tune BERT or RoBERTa for one epoch on an NVIDIA 4090 GPU and more than 6,000 seconds for GPT-2.

Table 2: Performance (F1-score) of open-source detectors on GPABench2. Values in blue: the best performance.

	T1. GPT-WRI			T2. GPT-CPL			T3. GPT-POL		
	CS	PHX	HSS	CS	PHX	HSS	CS	PHX	HSS
(a) Pre-trained Detectors									
HC3-PPL	0.760	0.794	0.854	0.688	0.686	0.763	0.665	0.668	0.682
HC3-GLTR	0.679	0.680	0.735	0.670	0.669	0.700	0.666	0.667	0.670
HC3-RBT	0.710	0.788	0.795	0.726	0.753	0.786	0.722	0.746	0.805
OpenAI-RBT	0.072	0.106	0.110	0.159	0.219	0.192	0.072	0.113	0.112
(b) Statistics-based Detectors									
HLR	0.910	0.917	0.902	0.792	0.757	0.841	0.608	0.602	0.682
Rank	0.781	0.650	0.525	0.783	0.612	0.531	0.808	0.738	0.620
Log-Rank	0.911	0.919	0.895	0.794	0.764	0.843	0.626	0.608	0.679
TP	0.913	0.924	0.894	0.797	0.778	0.847	0.632	0.622	0.686
PPL	0.913	0.925	0.891	0.814	0.797	0.848	0.654	0.650	0.695
Entropy	0.803	0.851	0.745	0.697	0.708	0.738	0.558	0.593	0.633
DetectGPT	0.790	0.722	0.768	0.685	0.693	0.766	0.616	0.588	0.630
(c) Fine-tuned Language Models									
BERT	0.999	0.999	0.998	0.992	0.983	0.992	0.983	0.984	0.966
DistillBERT	0.999	0.999	0.999	0.990	0.992	0.975	0.977	0.989	0.973
RoBERTa	0.999	0.999	0.997	0.970	0.995	0.995	0.981	0.993	0.967
GPT2	0.998	0.995	0.998	0.982	0.987	0.947	0.969	0.974	0.956
CheckGPT	0.999	1.000	0.999	0.996	0.995	0.995	0.993	0.994	0.993

3 USER STUDY: IDENTIFICATION OF HUMAN- AND GPT-GENERATED ABSTRACTS

With all the news reports and online/informal discussions that human users are unable to distinguish ChatGPT-generated text from man-written text, we investigate this problem through a user study in a relatively well-defined domain: research publications. We aim to answer four research questions:

RQ1: Could (experienced) researchers distinguish between human-written and GPT-generated paper abstracts?

RQ2: Do prior experiences with reading/writing papers contribute to the capability of identifying GPT-generated abstracts?

RQ3: Does the researchers' capability in identifying GPT content vary by discipline?

RQ4: Will their capability improve if they have ground truth data (pairs of human- and GPT-generated content) as references?

We designed a questionnaire as follows: first, the landing page displays an IRB information statement and asks the participants to select their "most familiar discipline" among CS, Physics, and Humanities & Social Sciences (HSS). Then, the main questionnaire page asks the participants to provide basic background information, their roles, whether they have published research papers and self-claimed familiarity with research papers. Finally, each participant is presented with three abstracts and asked to annotate each as "human-written" or "GPT-GEN/POL". Each abstract is randomly sampled from HUM or GPT-WRI/POL abstracts from Tasks 1 and 3 of GPABench2. For Task 3, we display the following hint: "This abstract was completely written by humans OR written by humans and then polished by ChatGPT."

In the second experiment, we provide the participant with some ground truth data as reference, i.e., we display three pairs of labeled human- and GPT-generated abstracts from the same task and discipline, and instruct the participants to learn the writing styles from

Table 3: Results of the user study. Par.: number of participants; Acc.: accuracy; HUM: accuracy for human-written abstracts; GPT: accuracy for GPT-generated abstracts.

	Exp 1: w/o reference				Exp 2: with reference			
Category	Par.	Acc.	HUM	GPT	Par.	Acc.	HUM	GPT
Role								
Faculty	44	49.2%	58.6%	41.9%	28	59.5%	61.2%	57.1%
Researchers	30	50.0%	58.2%	37.1%	28	58.3%	56.3%	61.1%
Students	81	48.1%	56.3%	40.3%	31	66.7%	75.6%	58.3%
Discipline								
CS	57	50.3%	59.0%	43.0%	33	60.6%	59.6%	61.5%
Physics	48	53.5%	65.1%	37.7%	25	60.0%	71.1%	43.3%
HSS	50	42.7%	46.5%	39.2%	29	64.4%	62.0%	67.6%
Self-claimed Familiarity with Research Papers								
Expert	52	51.3%	60.6%	43.5%	32	61.5%	67.3%	53.7%
Knowledgable	56	47.6%	57.3%	34.7%	30	61.1%	56.5%	65.9%
Somewhat	39	48.7%	56.0%	43.3%	21	65.1%	72.7%	56.7%
No familiarity	8	41.7%	46.7%	33.3%	4	50.0%	50.0%	50.0%
Published papers?								
Yes	106	48.7%	58.1%	39.2%	56	60.7%	64.0%	57.0%
No	49	49.0%	55.6%	42.7%	31	63.4%	64.2%	62.5%

these examples before moving to the detection questions. Three unlabeled samples are then displayed for the user to annotate.

We distributed questionnaires to faculty members, researchers, and graduate students in the Department of EECS, Department of Physics, and College of Liberal Arts at the University of Kansas. Physics faculty members also shared the questionnaire with collaborators in the European Organization for Nuclear Research (CERN). For Experiment 1, we received 155 responses with 465 annotated abstracts in approximately four weeks. The overall accuracy, defined as the proportion of correctly identified abstracts out of all abstracts, was 48.82%, which is slightly worse than random guesses. For Experiment 2, we received 87 responses with 261 annotated abstracts in two weeks. The overall accuracy improved to 61.69%. Note that the median time spent on each questionnaire was 108 seconds in Experiment 1 and 229 seconds in Experiment 2, showing that the participants did spend time reading and comprehending the ground truth samples. The detailed statistics of the responses are shown in Table 3. From the responses, we have the following observations:

- It is very challenging for human users to distinguish between human-written and GPT-generated paper abstracts. Only 21 users in Experiment 1 and 24 users in Experiment 2 correctly identified all three abstracts. If all participants were making random selections, 19.38 users would have scored 3 correct selections in Experiment 1.
- Participants have the tendency to annotate abstracts as "human". 57.33% and 64.08% of human-written abstracts were correctly labeled as "human-written" in Experiments 1 and 2, respectively, while 59.66% and 41.2% of GPT-generated abstracts were mistakenly labeled as "human-written". The result confirms the public opinion that ChatGPT achieves human-like language style and quality.
- Users are better at identifying fully GPT-written abstracts with the accuracy of 43.81% and 66.13%, while they perform worse with GPT-polished abstracts with the accuracy of 37.5% and 50.88%.
- Users' self-claimed expertise appears to very slightly affect their capability to identify human-written and GPT-generated abstracts. For example, participants with "No familiarity" with papers performed worse than the others. However, most of the differences are *not* statistically significant.
- Users become slightly better at identifying GPT-generated content when they are provided with references. However, the accuracy is still too low for reliable identification. The largest improvement is witnessed in HSS, with an accuracy improvement of 21.5%.

4 CHECKGPT: AN ACCURATE DETECTOR FOR CHATGPT-GENERATED ACADEMIC WRITING

4.1 The System Model and Assumptions

Our objective is to build a classifier, CheckGPT, to determine whether a given text snippet is generated by ChatGPT. We denote our classifier as $\mathcal{H}$, and the classification problem can be formulated as:

$$\hat{y} = \mathcal{H}_\theta(\mathbf{s}) \tag{1}$$

$$\text{argmin}_\theta \ \mathcal{L}(y, \hat{y}) \tag{2}$$

where $\mathbf{s}$ is an unstructured text snippet (i.e., paper abstract). Given $\mathbf{s}$, $\mathcal{H}(\mathbf{s})$ generates the probability distribution $\hat{y}$ considering label space {'h', 'g'}, where 'h' indicates human-written text and 'g' indicates ChatGPT-generated text. The goal is to find an optimal set of parameter θ for $\mathcal{H}$ to minimize the loss function $\mathcal{L}$ measuring the distance between prediction $\hat{y}$ and observation y.

We consider a *black-box defender*, who only has access to the observed LLM outputs, and does not have insider knowledge of the LLM that generates these samples (weights, structures, and gradients). This assumption is practical, considering OpenAI has not open-sourced its LLMs since the GPT-3.5.

The primary goal of this work is to detect *raw* ChatGPT-generated text, i.e., the unaltered output from ChatGPT. In the community, there have been discussions on the boundary between GPT-generated text and Human-generated text in the context of partial paraphrasing [10], where the boundary can become murky if a piece of GPT-generated text is further edited by a human editor. Therefore, to avoid the controversy of data labeling, we mainly focus on the clean boundary between purely human-generated text and raw ChatGPT output in problem setting[3].

Based on the application of the task, we further assume the following demands in addition to the defender's goals: (1) *Moderate Data Availability*. We do not assume the defender's privileged usage of ChatGPT. Therefore, the training samples are collected strictly following OpenAI's policy. With the efficiency of regular queries, an ordinary user usually can not collect massive amounts (e.g., tens of millions to billions) of data. (2) *Affordability*. We do not assume the defender's access to excessive computing power, which is only affordable to large organizations in the real world. We aim to offer a lightweight solution that smaller entities could conveniently obtain and deploy in a daily operational environment. And (3) *Privacy-Preserving Local Deployment*. The end users may not agree to share their data with the detector providers due to privacy, intellectual property, or policy concerns (e.g., manuscripts being reviewed).

[3]We will relax this assumption in Section 5.10 and evaluate CheckGPT's robustness against *post facto* human interventions

Table 4: Comparison between GPT-generated and human-written abstracts: complexity and syntactic features.

	word-len	flesch-kincaid	smog-index	coleman-liau	lix	ttr	VBG	DT	PRP	report-verbs
GPT-WRI	6.3	3.6	18.8	20.0	72	0.63	0.05	0.11	0.012	0.010
GPT-CPL	6.1	3.0	18.2	18.6	68	0.68	0.04	0.12	0.013	0.011
GPT-POL	5.7	1.8	16.4	16.4	61	0.66	0.03	0.12	0.014	0.013
HUM	5.5	2.4	16.3	15.1	60	0.64	0.02	0.11	0.019	0.015
HUM-T2	5.5	2.6	16.4	15.1	60	0.73	0.02	0.11	0.020	0.017

Table 5: Comparison between GPT-generated and human-written abstracts: semantic, pragmatic, and sentiment features.

	dates	factives	implicatives	stops	assertatives	hedges	pos-opinion	neg-opinion
GPT-WRI	0.0022	0.0012	0.0026	0.29	0.0018	0.0031	0.037	0.018
GPT-CPL	0.0040	0.0013	0.0044	0.31	0.0029	0.0058	0.037	0.020
GPT-POL	0.0070	0.0014	0.0049	0.31	0.0039	0.0081	0.027	0.021
HUM	0.0078	0.0013	0.0051	0.32	0.0041	0.0091	0.026	0.021
HUM-T2	0.0077	0.0014	0.0057	0.33	0.0045	0.0100	0.028	0.020

Therefore, the detector should have the potential to be transferred to a new domain with a small amount of target-domain data.

4.2 Linguistic & Semantic-Based Detection

We first explore a non-deep learning method using hand-crafted linguistic & semantic features as our baseline approach. The raw texts are transformed into vector representations using the NELA (News Landscape) features [44]. The NELA features were initially proposed for analysis in news articles. They comprise six linguistic and semantic features: styles, complexity, biases, affects, morals, and events. In Tables 4 and 5, we compare the statistics of HUM and GPT-GEN abstracts based on various features: average word length, readability (using Flesch-Kincaid Grade Level, SMOG index, Coleman-Liau Index, and LIX), lexical diversity (TTR), syntactic features (VBG, DT, PRP, and report verbs), semantic features (dates, factives, and implicatives), pragmatic features (stop words, assertatives, and hedges), and sentiment (positive/negative opinion words). The results show that GPT-POL abstracts are linguistically and semantically the most similar to the HUM ones. As more information is provided to ChatGPT in Tasks 2 and 3, the GPT-GEN abstracts become less complex in word length and readability and contain fewer gerunds (VBG) and positive opinion words. They also include more personal pronouns (PRP), report verbs, dates, factives, implicatives, stop words, assertatives, hedges, and negative opinion words, making them more similar to human-written ones.

We adopt Gradient Boosting (GB) decision trees to distinguish between human-written and GPT-generated abstracts. For each task and discipline, the models are trained with 50,000 human-written and GPT-generated samples, respectively, and tested with 10,000 samples from each class, i.e., an 80/20 train-test split ratio.

The classification performance of the baseline models is shown in Table 6. We have the following observations: (1) For Task 1, GB performs well in distinguishing GPT-written abstracts in CS and physics. (2) For Task 2, the F1-scores drop to about 0.9, indicating an increased difficulty. (4) The GPT-polished abstracts in Task 3 are more challenging to detect. The performance decreases significantly to <0.8 for all three disciplines.

Table 6: The baseline approach: classification F1-score for the NELA classifier.

T1. GPT-WRI			T2. GPT-CPL			T3. GPT-POL		
CS	PHX	HSS	CS	PHX	HSS	CS	PHX	HSS
0.965	0.980	0.963	0.901	0.918	0.896	0.774	0.794	0.798

We inspect the feature importance within the GB model to analyze the differences between the GPT-GEN and human abstracts. We find that: (1) ChatGPT tends to compose longer sentences and use longer words; (2) ChatGPT habitually uses more gerunds as adverbials; (3) ChatGPT uses more determiners and fewer personal pronouns than human writers.

While the baseline approach lacks capabilities in some tasks, it still significantly outperforms the human evaluators tested in Section 3, and the existing detecting tools in Table 1. It showcases the different habitual patterns of GPT-generated and human-written abstracts. Prominently, ChatGPT exhibits a tendency of increased complexity and a higher frequency of third-person singulars and gerunds. While these habitual patterns identified provide valuable insights, they alone cannot guarantee reliable and generalizable classification, as shown by the model's insufficiency on Task 2 and 3. This motivates us to adopt DNN-based detectors, which are more capable of capturing subtler and deeper semantic and/or linguistic patterns that the hand-crafted NELA features may overlook.

4.3 The CheckGPT Framework

Preliminaries. The Bidirectional Encoder Representations from Transformers (BERT) [20] family, including but not limited to BERT itself and RoBERTa, have shown extraordinary capabilities in a wide range of NLP tasks. RoBERTa (Robustly Optimized BERT approach) [62] is the state-of-the-art member of this family built upon BERT by Meta. Models like RoBERTa are pre-trained on a massive corpus from diverse disciplines. Such extensive training allows them to capture and represent various linguistic patterns, syntactic structures, and semantic relationships in the texts. Its tokenization and encoding enable the transformation of raw data into effective representations, which can be used for downstream tasks. In this work, we utilize the pre-trained RoBERTa to preprocess the text data. The pre-training of the RoBERTa utilizes a masked language modeling (MLM) objective, which can be formalized as:

$$\mathcal{L}_{\text{MLM}} = -\mathbb{E}_{\mathbf{s}\sim\mathcal{D}_s} \log P(m|\mathbf{s}) \tag{3}$$

where $\mathcal{D}_s$ is the corpus, $\mathbf{s}$ denotes an input sequence, and m is a masked token. The representations extracted by RoBERTa serve as the features for our downstream classification head. Long-Short-Term Memory (LSTM) networks [41], a variant of Recurrent Neural Network (RNN) that incorporates a gating mechanism to effectively retain information, can improve feature learning over long sequences. In this work, we adopt LSTM to build the classification head to aggregate the sequences of RoBERTa encoded tokens, and to fine-tune the final representation for task-specific prediction.

Representation. CheckGPT framework consists of two stages: representation and classification, as shown in Figure 1. For representation, CheckGPT proposes a model-agnostic design for text encoding

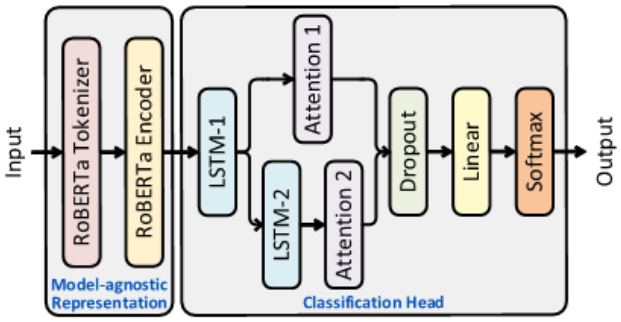

Figure 1: The architecture of CheckGPT.

where any pre-trained representation model (e.g., other variants of BERT) could be directly employed as is. This design achieves higher affordability, upgradability, and flexibility since (1) adopting a sophisticated pre-trained embedding model can save tremendous efforts and computations compared with training one from scratch, which is often beyond the capability of regular users or organizations; (2) a plugin design allows future upgrades by seamlessly accommodating new representation models; and (3) the lightweight classification head is easier to fine-tune when new data or domains are added. In CheckGPT, the representation stage is completed using the tokenizer and encoders of RoBERTa-large[4]. For tokenization, the pre-trained RoBERTa-large enforces a limit of 512 tokens. The tokenization can be formalized as:

$$\mathbf{X} = \text{Tokenizer}(\mathbf{s}) = \{x_i\}_{i=1}^{n} \tag{4}$$

where $\mathbf{X}$ denotes a sequence of length n consisting of individual tokens x_i, and Tokenizer refers to the Byte-level Pairing Encoding (BPE) utilized by RoBERTa.

For the encoder, the RoBERTa uses embeddings of size 1024 to represent each token. In this way, the tokenized texts are encoded into contextualized representations with a shape of $n \times 1024$. The encoding can be formalized as:

$$\mathbf{E} = \text{Encoder}(\mathbf{X}) = \{e_i\}_{i=1}^{n}, \quad e_i \in \mathbb{R}^{1024} \tag{5}$$

where $\mathbf{E}$ denotes a sequence consisting of individual embedding e_i, and Encoder refers to the transformer encoder utilized in RoBERTa.

Classification. The derived embeddings are fed into the LSTM-based classifier. The classification head consists of two LSTMs (with an embedding size of 256), and each is followed by an attention layer [5] to aggregate the entire sequence into a single vector. The outputs from the two LSTM modules are later concatenated and passed through a dropout layer with a rate of $p = 0.5$ for overfitting prevention, and a dense layer (FC) for final task-specific fine-tuning.

The softmaxed output indicates the conditional probability of the two classes: "GPT-generated" (y_g) or "Human-generated" (y_h). The functions of our LSTM classifier $f_\theta(\mathbf{E})$ are as follows:

$$\begin{aligned} h_1 &= \text{LSTM}_1(\mathbf{E}), \quad r_1 = \text{ATTN}_1(h_1) \\ h_2 &= \text{LSTM}_2(h_1), \quad r_2 = \text{ATTN}_2(h_2) \\ (\hat{y}_g, \hat{y}_h) &= \text{Softmax}(\text{FC}(\text{Dropout}(r_1 \oplus r_2))) \end{aligned} \tag{6}$$

Model Training. The classifier f_θ with parameter θ is optimized independently with the RoBERTa frozen (as is) during the training. We adopt an AdamW optimizer [65], a CosineAnnealing learning rate scheduler [64], and a gradient scaler for efficient mixed-precision training [72]. Given the model's predicted probabilities $\hat{y} = (\hat{y}_h, \hat{y}_c)$ and one-hot encoded ground truth $y = (y_h, y_c)$, the Binary Cross Entropy (BCE) loss of each training sample is defined as:

$$\mathcal{L}(\theta) = -\left[y_c \log(\hat{y}_c) + y_h \log(\hat{y}_h)\right] \tag{7}$$

Design Choices and Discussions. One alternative approach is directly applying RoBERTa by adding a RobertaClassificationHead [45]. However, experiments show that CheckGPT incur a higher accuracy, which can be attributed to LSTM's capability to track the sequential dependencies over long periods in the text sequences [108]. Please refer to Section 5.2 for details of the ablation study.

Another alternative approach is to include the entire pre-trained language representation model [78, 79] for task-specific fine-tuning. As shown in Table 2, CheckGPT consistently outperforms the approaches of fine-tuning the entire language representation models (BERT, DistillBERT, RoBERTa, and GPT-2) on GPABench2.

Comparing with both alternative approaches in framework design, CheckGPT offers the following advantages: (1) **Efficiency**: CheckGPT significantly reduce the parameters to save both time and computing resources. Given the parameters of language models ranging from 66M (DistilledBERT [89]) to 355M (RoBERTa-large, GPT-2-medium) and 1750M (GPT-3 [12]), our model only maintain 4M parameters and achieve satisfactory accuracy. In the experiments, tuning RoBERTa on the full unified dataset takes 9.75 hours, while training CheckGPT takes 21 minutes to exceed RoBERTa's accuracy. (2) **Affordability**: Tuning RoBERTa's 355M parameters consumes 22GB GPU memory, while training CheckGPT's LSTM (4M parameters) uses 164MB (batch=16 for both). (3) **Applicability**: CheckGPT is model-agnostic and thus accepts various representation approaches (e.g., BERT[20], BART [56]), making it a lightweight and universal detector, as detailed in Section 5.2. This feature can be especially valuable considering deployment and customization in the real world. (4) **Versatility**: By freezing RoBERTa's well-crafted parameters trained on a broad range of domains, we retain the framework's generalizability to a great extent to use CheckGPT in different subjects, which is challenging for fine-tuned RoBERTa [104]. More details are presented in Section 5.3. (5) **Transferability**: In transferring the pre-trained classifier to a new domain (e.g., from paper abstracts to news reports), finetuning a very large model using a small dataset may cause overfitting and catastrophic forgetting [50, 74]. We noticed obvious overfitting on RoBERTa during transfer learning with 2000 samples. The small model size in CheckGPT also reduces the risks of over-fitting and catastrophic forgetting, especially with small datasets [4, 101].

5 EXPERIMENTS

5.1 Settings and Metrics

We implement CheckGPT with PyTorch 1.13.1 in Python 3.9.1 on Ubuntu 22.04. The pre-trained RoBERTa is adopted from [45]. All the experiments were conducted on an Nvidia 2080Ti GPU and an Intel i9-9900k CPU. We use GPABench2 for most of the experiments. CheckGPT is trained with an initial learning rate of 2e-4, a batch size of 256, and an early-stop strategy to finish training when the validation loss stays flat for a number of epochs (8 by default).

[4]https://github.com/facebookresearch/fairseq/tree/main/examples/roberta

Table 7: CheckGPT's performance (in %) for each task, discipline, and prompt: TPR, TNR, accuracy (Acc).

	T1. GPT-WRI			T2. GPT-CPL			T3. GPT-POL		
	CS	PHX	HSS	CS	PHX	HSS	CS	PHX	HSS
Prompt 1									
TPR	99.95	100.0	99.94	99.58	99.50	99.48	99.23	99.31	99.22
TNR	99.97	100.0	100.0	99.70	99.57	99.67	99.33	99.40	99.39
Acc	99.96	100.0	99.97	99.64	99.54	99.58	99.28	99.36	99.31
Prompt 2									
TPR	99.99	100.0	100.0	99.63	99.50	99.65	99.23	99.50	99.32
TNR	99.99	100.0	99.98	99.75	99.63	99.68	99.33	99.51	99.40
Acc	99.99	100.0	99.99	99.69	99.57	99.67	99.28	99.51	99.36
Prompt 3									
TPR	99.97	99.99	99.96	99.78	99.68	99.67	99.33	99.35	99.46
TNR	100.0	100.0	99.98	99.76	99.79	99.71	99.34	99.68	99.36
Acc	99.98	100.0	99.97	99.77	99.74	99.69	99.34	99.52	99.41
Prompt 4									
TPR	100.0	99.99	99.96	99.76	99.71	99.67	99.41	99.61	99.47
TNR	99.99	100.0	99.99	99.77	99.78	99.88	99.60	99.69	99.50
Acc	100.0	100.0	99.98	99.77	99.75	99.78	99.51	99.65	99.49

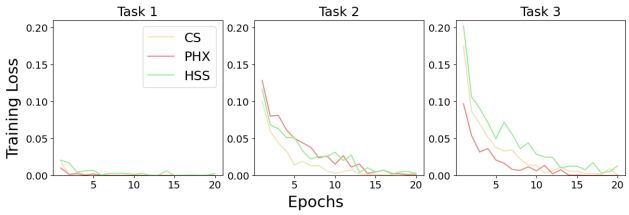

Figure 2: Training losses of the task-specific and discipline-specific classifiers.

When we consider CheckGPT as the GPT-generated content detector, the *true positive* rate ($TPR = \frac{TP}{TP+FN}$) is the proportion of correctly detected GPT-generated abstracts out of all GPT-generated abstracts, i.e., the accuracy in classifying GPT-generated text. The *true negative* rate ($TNR = \frac{TN}{TN+FP}$), is the proportion of correctly identified human-written abstracts out of all human-written abstracts, i.e., the accuracy in classifying human-written text. The overall accuracy is defined as the proportion of correctly classified samples over all the testing samples: $Acc = \frac{TP+TN}{TP+FP+TN+FN}$.

5.2 Task- and Discipline-specific Classifiers

We first evaluate CheckGPT at the finest granularity: one classifier for each discipline, task, and prompt combination. We use an 80%-20% train-test split on the main GPABench2 dataset: 80K samples (40K each of GPT and HUM) for training and 20K for testing. Training takes an average of 120s per epoch, while testing takes about 0.03s per sample. We report the classification accuracy in Table 7.

CheckGPT achieves very high accuracy in all cases. The detection accuracy for Task 1 (abstracts entirely written by ChatGPT) is higher than 99.9% across all disciplines/prompts. Task 2, where only the second halves of the abstracts are checked, has slightly lower accuracy, which is explained by shorter text lengths and better writing by ChatGPT given more seed data. The accuracy of Task 3, which is most challenging for the open-source and commercial detectors (Sec. 2.2), is between 99.28% and 99.65%.

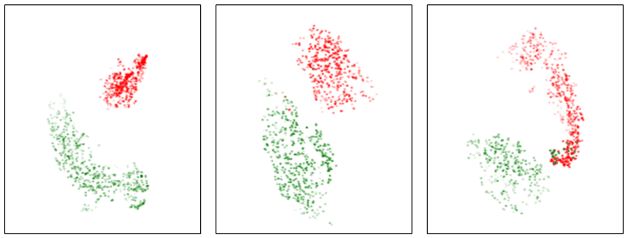

Figure 3: Feature space distribution of human-written (green) and GPT-generated (red) abstracts.

Table 8: Comparison with other design choices.

Model	Para Size	Acc(%)		
		Task 1	Task 2	Task 3
GLoVe + CheckGPT classifier	-	99.77	98.34	95.90
BERT + CheckGPT classifier	-	99.90	99.28	97.81
CheckGPT representation + RCH	1.05M	99.80	97.70	94.08
CheckGPT representation + MLP-Pool	1.05M	99.87	98.62	95.93
CheckGPT representation + CNN	3.33M	99.80	98.47	96.49
CheckGPT rep. + BiLSTM w/o attention	4.21M	99.91	99.54	98.92
CheckGPT	4.21M	99.98	99.72	99.39

Note: CheckGPT representation = RoBERTa

Figure 2 shows the training losses for Prompt 1. Task 1 models rapidly grasp simple features like lexical characteristics and reached convergence. Tasks 2 and 3 models are more difficult to train. In most cases, HSS is more challenging than the other two disciplines, which implies that ChatGPT does a better job mimicking human-written style in HSS topics. Task 2 is the outlier, where the samples in PHX are significantly shorter than the other disciplines, and thus harder to distinguish.

We randomly select 2,000 CS abstracts from each task, and then use t-Distributed Stochastic Neighbor Embedding (t-SNE) [102] to map the 1024-dimensional feature vectors from the last dense layer of the BiLSTM module into a 3-dimensional space, as shown in Figure 3. The figure shows that: (a) GPT-written abstracts form a dense cluster (consistent vocabulary, writing style, and semantic features), while human-written samples demonstrate significantly more diverse distributions. (b) GPT-completed abstracts (Task 2) are clearly more diverse than the GPT-written ones. While their representations are closer to the human-written samples, a distinct gap still remains. (c) GPT-polished samples are scattered and intertwined with human-written samples, demonstrating the challenges in distinguishing human-written and GPT-CPL abstracts in Task 3.

Ablation study. We conduct an ablation study to compare the current design of CheckGPT with several alternatives. We first keep the classification head in CheckGPT and replace the representation module (RoBERTa) with GLoVe6B-100d [80] or pre-trained BERT. As shown in Table 8, there is a slight performance drop. We then keep CheckGPT's representation module (RoBERTa) and replace the classifier with: the default classification head for RoBERTa (RCH) Huggingface [45], and its variant with global average pooling (MLP-Pool; 23, 58); an AlexNet-like CNN [53] with five convolutional layers, and a basic BiLSTM without attention. As shown in Table 8, CheckGPT achieves the best accuracy.

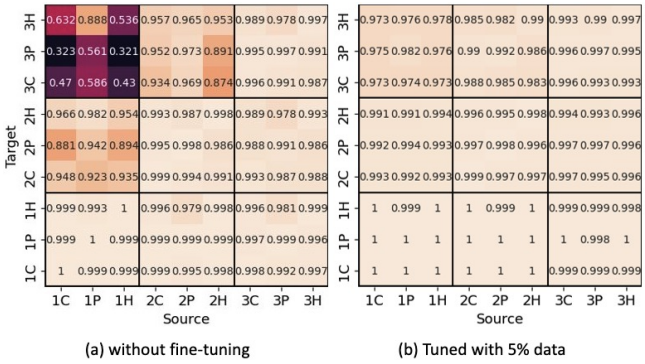

Figure 4: CheckGPT's transferability across disciplines and tasks: (a) without fine-tuning, (b) tuned with 5% data from the train set. 1C: Task 1 GPT-WRI+CS; 2P: Task 2 GPT-CPL+PHX; 3H: GPT-POL+HSS.

Table 9: TPR and TNR (in %) of the unified classifiers.

	T1. GPT-WRI			T2. GPT-CPL			T3. GPT-POL		
	CS	PHX	HSS	CS	PHX	HSS	CS	PHX	HSS
(a) Cross-prompt Classifiers									
TPR	99.99	99.99	99.99	99.84	99.87	99.78	99.74	99.75	99.74
TNR	99.97	100.0	99.97	99.52	99.43	99.51	98.87	99.31	98.90
(b) Cross-prompt Cross-disciplinary Classifiers									
TPR	99.99	99.99	99.99	99.82	99.72	99.72	99.62	99.73	99.62
TNR	99.99	100.0	99.99	99.66	99.67	99.72	99.10	99.58	99.24
(c) Cross-prompt & -task & -disciplinary Classifiers									
TPR	100.0	100.0	100.0	99.81	99.77	99.80	99.58	99.76	99.65
TNR	99.16	99.46	99.27	99.47	99.45	99.55	99.16	99.46	99.27

5.3 Transferability across Tasks, Disciplines, and Prompts

We evaluate CheckGPT's capability of cross-prompt, cross-task, and cross-disciplinary generalization. First, we train nine cross-prompt models (one model for each task and discipline as shown in Table 9 (a)) to evaluate testing samples from **other** tasks and disciplines, *without model fine-tuning*. In Figure 4 (a), each value demonstrates the F1-score using the model from the task/discipline denoted on the x-axis to test samples from the task/discipline on the y-axis. From the figure, we observe the following:

- CheckGPT is adaptable *across disciplines*. CheckGPT achieves ≥0.978 F1-score on *cross-discipline* data from the *same task*.
- CheckGPT is less adaptable *across tasks*. In particular, the models trained in Task 1 demonstrate low performance with testing samples from the other tasks, while the models from Task 2 are also less capable of handling Task 3 (GPT-POL) data.
- The models trained in Task 3 demonstrate solid performance with testing samples from Tasks 1 and 2. This implies that Task 3 could be the most difficult, and the models have learned the subtle but inherent features of AIGC.

We then fine-tune the last linear layer of each model with the data in the target domain. As shown in Figure 4 (b), tuning with as few as 5% of data (2K samples) increases the classification F1-score to >0.97 in all cases, while the distribution patterns of the F-1 scores remain similar to Figure 4(a).

Table 10: TPR and TNR (in %) of cross-prompt testing.

	T1. GPT-WRI			T2. GPT-CPL			T3. GPT-POL		
	CS	PHX	HSS	CS	PHX	HSS	CS	PHX	HSS
(a) Train with Prompts 2, 3, 4; test with Prompt 1									
TPR	99.73	99.91	99.63	99.43	99.31	99.54	97.73	98.47	98.00
TNR	99.89	99.99	99.95	99.35	99.42	99.19	98.54	99.00	98.99
(b) Train with Prompts 1, 3, 4; test with Prompt 2									
TPR	99.46	99.83	99.59	99.63	99.50	99.55	98.82	99.18	99.59
TNR	99.89	99.81	99.85	99.14	99.40	99.20	99.23	99.39	98.87
(c) Train with Prompts 1, 2, 4; test with Prompt 3									
TPR	99.99	99.99	99.97	99.31	99.51	99.60	99.34	99.75	99.68
TNR	99.87	99.92	99.89	99.49	99.46	99.29	98.86	98.98	98.80
(d) Train with Prompts 1, 2, 3; test with Prompt 4									
TPR	99.98	99.95	99.95	99.67	99.39	99.51	99.75	99.63	99.79
TNR	99.79	99.91	99.82	99.06	99.41	99.35	98.47	98.99	98.79

Prompt Transferability. To assess CheckGPT's generalizability over domain shifts caused by ChatGPT prompts, we train it with data from 3 prompts and test it with samples from the fourth prompt. As shown in Table 10, CheckGPT is highly transferable across prompts. Notably, testing accuracy for Prompt 1 is usually the lowest, which is explained by the fact that Prompt provided minimal context to ChatGPT so that the outputs are most diverse.

The Unified Classifiers. We evaluate CheckGPT by sampling data from all prompts (train with 80K GPT samples for each task/discipline) and show the classification accuracy in Table 9 (a). We then combine data across disciplines (Table 9 (b)) and further across all tasks (Table 9 (c)). In summary, unified training slightly improves TPR, especially for difficult tasks, e.g., GPT-POL in CS.

The Multi-Task Classifier. Finally, we investigate whether CheckGPT is able to distinguish the difference among the three tasks. We turn CheckGPT into a unified, 4-ary classifier to distinguish HUM, GPT-WRI, GPT-CPL, and GPT-POL abstracts. CheckGPT achieved a 98.51% accuracy, with 98.75%, 99.39%, 97.69%, and 98.24% for HUM, GPT-WRI, GPT-CPL, and GPT-POL, respectively. CheckGPT's accuracy drops slightly due to the challenges of multi-label classification, but it can still catch the subtle differences in the three tasks well.

5.4 Transferability to New Domains

Other academic writing purposes. While our GPABench2 focuses on abstracts of research papers, ChatGPT can be used for other writing purposes, e.g., other sections of papers. While paper abstracts are almost always publicly available, the full papers are often restricted by copyright. Web scraping is not allowed even for open-access papers. Thus, we do not attempt to include human-written full papers in GPABench2. Instead, we provide ChatGPT with titles and ask it to write an introduction, background, and conclusion section for each title (similar to Task 1). For each discipline and section, we collect 10,000 samples using the titles from GPABench2 (90,000 in total). We apply the cross-prompt cross-discipline CheckGPT (Sec. 5.3) on the new sections. As shown in Table 11, CheckGPT maintains 99.9% TPR for all the sections. We also manually gathered and evaluated human-written sections from 1,000 papers. The TNRs for the introduction, background, and conclusion sections are 98.24%, 99.35%, and 99.61%, respectively. This shows the high potential of CheckGPT for general academic writing.

Table 11: TPR (in %) of other sections of research papers.

	Introduction			Background			Conclusion		
	CS	PHX	HSS	CS	PHX	HSS	CS	PHX	HSS
TPR	99.99	100.0	100.0	99.99	100.0	99.98	100.0	100.0	100.0

Classic NLP Datasets. We evaluate CheckGPT with three NLP datasets: Wikipedia Abstracts (1,500 random samples from 13), ASAP Essays [27], and BBC News [34]. In ASAP Essays, we selected two different tasks: "letters stating opinions on computers" (Essay-C), and "stories about patience" (Essay-P). We adopted the original instructions in Foundation [27] for Task 1 and designed the prompts for the other tasks and datasets for details). We apply the cross-prompt cross-discipline CheckGPT classifiers (Sec. 5.3) on the new domains. As shown in Table 13, CheckGPT shows solid performance, especially on objective, structural, or argumentative writing like news and opinions. When the last layer of CheckGPT is tuned with 100 samples (50 for each label) from each domain, it achieves significantly higher accuracy.

SOTA ChatGPT Datasets in the Literature. We further evaluate CheckGPT on five ChatGPT datasets in the literature that cover 15 domains. We adopted the following datasets: student essays in ArguGPT [63]; Finance, Medicine, OpenQA, Reddit, and Wikipedia in HC3 [35]; arXiv papers, PeerRead reviews, Reddit, wikiHow manuals, and Wikipedia in M4 [104]; news articles in MULTITuDE [68]; and MGTBench [37] that covers short Q&A from NarrativeQA [51], SQuAD1.0 [85] and TruthfulQA [59].

We evaluate CheckGPT with three experiments: (1) directly validate CheckGPT without any fine-tuning, (2) tuning the last layer of the classification head with 150 samples, and (3) tuning the whole classification head with the entire training set. We also compare CheckGPT with tuning the entire language models as reported in these papers. As shown in Table 12, CheckGPT (without tuning) reaches an F1-score of >0.95 for 9 out of the 15 domains, and reaches >0.9 for two more domains. Unsatisfactory performance are shown on wikiHow and the short Q&A in MGTBench. Unlike all other writing tasks, the text in wikiHow is highly informal, with many imperatives as tips or advice. For the MGTBench, most of the answers consist of a single sentence, and many of them even come with one or two words. For these data, the gaps are too large for CheckGPT to transfer effectively. Notably, the fully-tuned CheckGPT outperforms fine-tuned language models in almost all the domains except M4-Wikipedia, where CheckGPT is 0.012 lower than fine-tuned RoBERTa. Note full tuning of CheckGPT takes approximately 120s per epoch, which is considerably faster than tuning RoBERTa (1049s per epoch) or DistillBERT (548s per epoch).

5.5 CheckGPT Performance Over Time

Since its first release, OpenAI has made several major updates to ChatGPT. Hence, we ask the question: *Will CheckGPT remain effective over time?* We evaluate CheckGPT on ChatLog-HC3 [100], which consists of ChaGPT-generated answers for the same questions collected every month starting from 03/2023. In this experiment, we pre-train CheckGPT with GPABenchmark v1, which was collected before March 2023. As shown in Figure 5, CheckGPT's performance has stayed high over the past ten months. Note that the

Table 12: F1-score of evaluating CheckGPT on SOTA datasets in different domains. Val.: Direct Validation. FT-L: Fine-tuning the last layer. FT-A: Fine-tuning all the layers in the CheckGPT classifier. All SOTA detectors were trained/tuned with data in the target domain.

Dataset	CheckGPT			SOTA	
	Val.	FT-L	FT-A	Model	F1-score
ArguGPT-WECCL	1.000	1.000	1.000	RoBERTa	0.994
HC3-Finance	0.957	0.971	0.998	RoBERTa	0.993
HC3-Medicine	0.985	0.985	0.998	RoBERTa	0.995
HC3-OpenQA	0.936	0.948	0.999	RoBERTa	0.986
HC3-Reddit	0.965	0.973	1.000	RoBERTa	1.000
HC3-Wiki	0.945	0.971	0.991	-	-
M4-arXiv	0.995	0.999	1.000	RoBERTa	1.000
M4-PeerRead	0.962	0.968	1.000	RoBERTa	0.961
M4-Reddit	0.968	0.986	0.999	RoBERTa	0.907
M4-wikiHow	0.767	0.899	0.998	RoBERTa	0.997
M4-Wiki	0.964	0.983	0.996	RoBERTa	0.997
MULTITuDE	0.951	0.956	0.989	RoBERTa	0.984
MGTBench-N	0.398	0.938	1.000	DistillBERT	0.948
MGTBench-S	0.407	0.949	1.000	DistillBERT	0.989
MGTBench-T	0.817	0.919	1.000	DistillBERT	1.000

Table 13: TPR and TNR (in %) in new domains.

	w/o fine-tuning			w/ fine-tuning		
	Task 1	Task 2	Task 3	Task 1	Task 2	Task 3
(a) Wiki						
TPR	100.0	99.86	98.13	99.86	98.76	94.08
TNR	81.13	96.50	81.13	99.23	99.54	93.85
(b) Essay-C						
TPR	91.09	97.13	86.82	100.0	100.0	100.0
TNR	100.0	100.0	100.0	100.0	100.0	100.0
(c) Essay-P						
TPR	83.36	68.82	79.09	99.82	99.82	99.82
TNR	99.92	99.77	99.92	99.69	98.75	99.37
(d) BBC News						
TPR	100.0	99.43	90.72	100.0	99.57	97.50
TNR	99.86	99.93	99.86	100.0	99.86	98.79

Table 14: TNR (in %) on ChatLog-HC.

	Finance	Medicine	OpenQA	Reddit	Wiki
Val.	99.87	100	83.12	99.79	89.29
FT-L	98.60	99.60	91.56	98.72	95.83
Full Re-train	99.75	100	95.36	100	95.83

TNRs (Table 14) do *not* change over time since the human answers are the same. Additionally, we have the following observations:

- While we only train/tune CheckGPT with data prior to 03/2023, it yields stable performance over 10 months. Moreover, CheckGPT's accuracy significantly increased for all the domains on 06/2023. A possible explanation is that an update may have introduced more consistent writing styles and stronger patterns to ChatGPT.
- CheckGPT performs worse on OpenQA compared to the original HC3 collected in 12/2022 (Table 12). In a closer look, we find that ChatGPT generated significantly shorter answers in ChatLog. The average length was 133 tokens for HC3-OpenQA but 71 tokens for ChatLog in 03/2023. However, ChatGPT started to generate longer

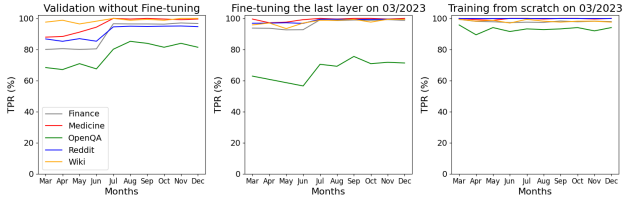

Figure 5: TPR on ChatLog-HC3 with direct validation, fine-tuning the last layer, and full re-train.

Table 15: Performance (F1-score) of CheckGPT on non-GPT LLMs. SOTA: the performance of fine-tuned language models.

Dataset	Other LLMs	FT-L		FT-A		SOTA	
		Min.	Avg.	Min.	Avg.	Min.	Avg.
M4-arXiv	BloomZ, Cohere, Dolly, Flant-T5/Llama	0.903	0.927	0.998	0.999	-	-
M4-PeerRead		0.820	0.868	0.996	0.999	-	-
M4-Reddit		0.872	0.987	0.984	0.996	-	-
M4-wikiHow	BloomZ, Cohere, Dolly	0.834	0.915	0.988	0.995	-	-
M4-Wiki		0.798	0.847	0.978	0.989	-	-
MULTITuDE	Alpaca, LLaMA, OPT, OPT-IML, Vicuna	0.656	0.758	0.937	0.965	0.883	0.941
MGTBench-N	ChatGLM, Dolly, StableLM	0.909	0.934	1.000	1.000	0.875	0.928
MGTBench-S		0.908	0.919	1.000	1.000	0.939	0.965
MGTBench-T		0.920	0.956	1.000	1.000	0.966	0.988

answers later in 2023, with an average length of 98 tokens in December, and the performance of CheckGPT increases accordingly.

5.6 Transferability to New LLMs

GPT-4. We invoke GPT-4, which is the most up-to-date and powerful member of GPT models, with the same prompts in Sec. 2.1 to generate GPT-WRI, GPT-CPL, and GPT-POL abstracts for 2000 random samples, respectively (small sample size due to strict rate limits). We use the unified classifiers to evaluate all the samples, and CheckGPT achieves >96% TPR in all three experiments (99.95%, 96.90%, and 96.15%, respectively).

Non-GPT LLMs. Other LLMs might adopt unique model architectures, training datasets, and training methodologies, which may differ significantly from ChatGPT. In this paper, we do not claim or expect the transferability of CheckGPT to non-GPT LLMs. Nevertheless, we still apply CheckGPT (without tuning) to the content generated by 11 non-GPT LLMs: M4 [104], MULTITuDE [68], and MGTBench [37]. As shown in Figure 15, CheckGPT is highly adaptable to content generated by all the LLMs. Note that the F1-scores for models other than ChatGPT in M4 are not reported in [104].

5.7 Advanced Prompt Engineering

As presented in Section 2.1, GPABench2 contains 435K additional testing samples using advanced prompt engineering. We evaluate the new dataset with task-specific, discipline-specific, and cross-prompt classifiers. As shown in Table 16, CheckGPT's TPRs are consistently high. Moderate decreases are only noticed in LMP, SCP, GKP, MP, and II for Task 2, and LMP for Task 3. However, when a prompt-specific (P1) model is used for the new data, the average TPR decreases by 0.85%, and the maximum decrease is 8.2%. This suggests the robustness of the cross-promote models, i.e., the models learned GPT-specific features that are transferable instead of prompt-specific bias.

Table 16: TPR (in %) on advanced prompts.

	T1. GPT-WRI			T2. GPT-CPL			T3. GPT-POL		
	CS	PHX	HSS	CS	PHX	HSS	CS	PHX	HSS
ZC	100.0	100.0	100.0	99.38	98.91	99.21	99.79	99.84	99.79
APE	100.0	100.0	99.96	99.11	99.21	99.21	99.47	99.26	99.27
SCP	99.95	99.94	99.98	99.15	98.43	98.67	99.64	99.81	99.73
FSP	100.0	99.98	99.92	99.68	99.61	99.44	99.45	99.24	99.54
LMP	99.94	99.98	99.94	98.60	98.97	98.85	99.01	98.99	98.89
GKP	99.96	99.98	100.0	98.50	98.62	98.70	99.78	99.73	99.80
PP	100.0	100.0	99.98	99.66	99.90	99.54	99.84	99.88	99.83
GP	99.64	99.59	99.83	99.78	99.59	99.67	99.19	99.48	99.33
MP	99.96	99.98	99.98	97.78	97.75	97.97	99.58	99.65	99.73
II	100.0	99.98	99.96	99.21	98.53	99.00	-	-	-

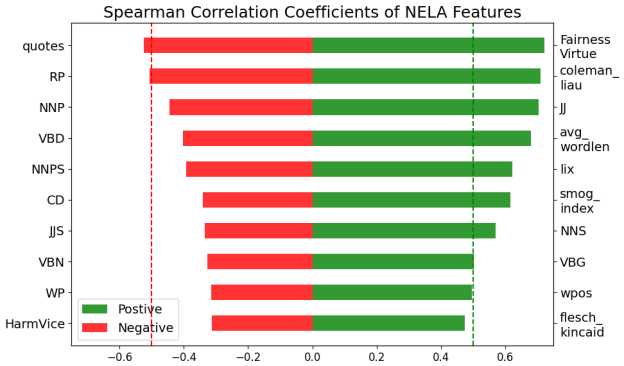

Figure 6: Correlations between domain-specific features and CheckGPT's transferability.

5.8 Discussions on CheckGPT's Transferability

With all the task-, prompt-, model-, and domain-transferability experiments, we raise the critical questions: *What makes CheckGPT trained on GPABench2 (non-)transferable to other domains? How can GPABench2 contribute to general-purpose GPT-text detection?* To investigate this, we again adopt the NELA feature to measure the domain gaps between GPABench2 and all the 39 target domains in the transferability experiments, e.g., Wiki, ASAP Essays, BBC News, etc. We examine the correlation of each feature with CheckGPT's transferability, i.e., the performance of CheckGPT in validations without fine-tuning. We set a threshold at 0.5 on Spearman's Rank Correlation Coefficient. Eventually, we obtain the positive and negative factors influencing the transferability of CheckGPT, as shown in Figure 6. The top six positive features are (from high to low): Fairness Virtue, Readability (Coleman-Liau Index, average word length, LIX Readability, and Smog Index), adjectives, plural nouns, gerunds. The top two negative factors are quotes and particles.

Based on the positive factors, we conclude that *CheckGPT will transfer better to objective, complex, detailed, narrative, or journalistic writing.* Firstly, the fairness virtue indicates an objective tone, while higher readability indicates longer and more complex sentences and word usage. Secondly, descriptive language plays an important role in writing styles with more adjectives. Adjectives add details and expressiveness to the writing by qualifying nouns

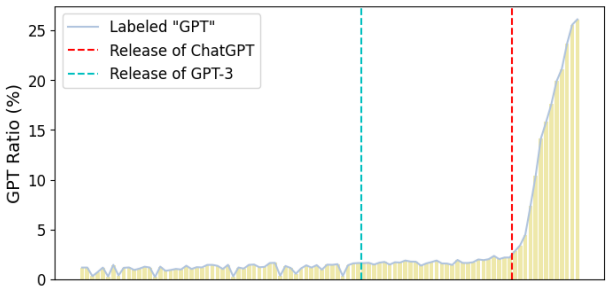

Figure 7: Detecting ChatGPT usage in arXiv papers.

and pronouns. More adjectives improve the clarity and vividness of the text, indicating creative or detailed writing. Thirdly, writing styles that frequently use plural nouns typically involve discussion of general concepts, discovery, groups, or disciplines rather than focusing on individual or specific instances. They usually serve a technical, scientific, analytical, or political purpose. Lastly, as we have shown in Section 4.2, ChatGPT favors gerunds and CheckGPT will transfer to descriptive or narrative writing with a sense of continuous actions and processes. Similarly, based on the negative factors, we conclude that *CheckGPT will possibly show less transferability to informal writing and conversations,* as particles and quotes will appear more frequently in these writing styles.

5.9 Use of ChatGPT in arXiv Papers

Finally, we ask "*How many authors are using ChatGPT to write/polish their research papers*?" We collect all the arXiv abstracts in CS from January 2016 to December 2023 (~450k samples, excluding those in GPABench2). We evaluate each abstract with the unified cross-task cross-prompt cross-disciplinary classifier and show the monthly average positive rates in Figure 7. There is a significant increase in ChatGPT usage, with a peak of 26.1% in December 2023. The average positive rates before, between, and after the releases of GPT-3 and ChatGPT are 1.12%, 1.78%, and 7.83%, respectively. The exponential growth started in December 2022, right after ChatGPT's release on 11/30/2022. Our model also annotates 0.23~1.66% of the abstracts posted before GPT3 as GPT-GEN, which may be explained by CheckGPT's 1% FPR, while LLMs like GPT-2 might also be used by the early adopters.

5.10 Robustness against Sanitized GPT Output

We consider the scenario where the user (attacker) sanitizes the raw output from ChatGPT to make it more human-like in an attempt to escape CheckGPT. We discuss and implement four attacks:

Automatic Rephrasing. The attacker prompts ChatGPT to rephrase its answers twice. In each prompt, ChatGPT is asked to rephrase its output to make it "more like human-written."

Mixed Human/GPT Writing (The Mixing Attack). The attacker substitutes some of the ChatGPT-generated text with human-written text to confuse the detector. In particular, ChatGPT-written abstracts are usually 6-8 sentences long, where the first and last sentences are more generic and less technical. We substitute *each* of the top 3 and last 3 sentences with the corresponding sentence from the HUM abstract. We further attempt to substitute two sentences (top 2 or last 2) from GPT-GEN abstracts. In the end, each attack sample is a hybrid text with GPT-GEN and HUM content.

Copyediting. We make a strong attack assumption that the attacker knows the vocabulary distribution of GPT- and HUM-generated text, and sanitizes GPT text accordingly. To mimic this attack, we examine word and bi-gram distribution in HUM- and GPT-generated abstracts in GBABench2, and identify the words and phrases with the largest frequency discrepancy. i.e., words that are popular in GPT-GEN but rarely used in HUM. We drop the attributives and substitute the other words/bigrams that are popular in ChatGPT text with synonyms that are heavily used by humans. Based on the attackers' knowledge and costs, we define three attack levels: Top-3, Top-5, and Top-10. We extract the Top-N words and phrases in each discipline, task, and prompt and combine them in one set. Eventually, there are 76 items in the Top-10 set and 52 items in the Top-3 set. Some examples of substitutions in all three levels include removing or replacing "this paper", "the paper", "this study", "in this paper", "in this study", "additionally", "furthermore", etc.

Prompt Engineering. Same as the Copyediting Attack, the attacker knows the term distributions of human- and GPT-generated texts. The attacker employs prompt engineering to ask ChaGPT to avoid using the frequent words and phrases in GPT-GEN text.

We evaluate our cross-prompt cross-discipline cross-task classifier against 10,000 attack samples for each combination of task, discipline, and prompt. The TPRs are shown in Table 17. We have the following observations:

- CheckGPT's accuracy becomes higher on rephrased text, i.e., nearly 100% TPR for all the tasks/discipline. It implies that the inherent patterns of ChatGPT get stronger after iterative rewriting.
- CheckGPT is robust when any of the first/last 3 sentences are replaced with a human sentence. Performance for a few tasks dropped to ~75% when two sentences (about 1/4 to 1/3 of the abstract) are replaced, while the other tasks still perform >90%. That is, CheckGPT's detection is not dominated by any single sentence-level indicator. Instead, every sentence contributes to the decision.
- CheckGPT's detection accuracy decreases slightly when the unique terms used by ChatGPT (sometimes referred to as "artifacts") are removed or replaced with "human terms". However, the average TPR remains at 83.1% (with a minimum of 70.45%) after removing/replacing 76 top words and phrases in GPT-GEN text. This implies that CheckGPT learns and utilizes the lexical features in GPT-generated content, but it does not completely rely on them.
- CheckGPT reaches the lowest accuracy with Copyediting Attacks in Task 3. In this polishing task, ChatPGT mostly maintains the structure of the paragraph and the sentences but makes word-level tuning. Therefore: (1) The semantic and structural features of GPT-POL abstracts are very similar to HUM abstracts. And (2) the lexicon features of GPT-POL abstracts are effectively eliminated by the manual Copyediting Attack. Both factors contribute to the low accuracy of CheckGPT against this attack.
- CheckGPT's performance stays high against the Prompt Engineering Attacks. This shows that the ChatGPT's habitual patterns persist even under specially crafted instructions, which aligns with our findings in Section 5.7.

In summary, CheckGPT is highly robust against *post facto* human interventions of ChatGPT-generated content at word/phrase and sentence levels. As we have observed, CheckGPT employs many

Table 17: TPR (in %) under attacks. CE: Copyediting. PromptEng: Prompt Engineering Attacks. Mixing-F/L: substituting the first/last sentences.

	T1. GPT-WRI			T2. GPT-CPL			T3. GPT-POL		
	CS	PHX	HSS	CS	PHX	HSS	CS	PHX	HSS
Rephrasing	100.0	100.0	100.0	99.99	99.98	100.0	99.99	99.97	99.95
Mixing-F1	99.98	100.0	99.93	99.52	99.54	99.33	93.10	97.31	91.52
Mixing-F2	99.98	100.0	99.98	99.21	99.44	99.09	95.73	98.00	95.28
Mixing-F3	99.89	99.97	99.77	97.59	98.62	96.83	90.52	96.22	90.73
Mixing-L3	99.96	99.97	99.85	99.02	99.54	98.57	96.71	98.18	96.79
Mixing-L2	99.83	100.0	99.87	98.14	99.24	98.03	94.63	98.17	94.52
Mixing-L1	99.80	100.0	99.78	95.21	97.33	96.24	96.44	98.39	95.83
Mixing-F12	99.77	99.97	99.31	96.90	97.63	95.62	75.78	87.55	73.02
Mixing-L12	95.60	98.20	96.43	78.05	85.57	82.67	75.68	94.14	85.75
CE-Top3	96.55	98.47	92.79	93.06	95.35	91.88	83.77	91.88	85.85
CE-Top5	92.92	96.47	88.83	89.06	92.25	87.35	78.36	89.84	81.17
CE-Top10	89.27	92.62	82.02	85.75	88.52	82.54	70.45	84.20	72.56
PromptEng	99.99	99.94	99.99	99.19	99.02	99.62	89.92	89.30	95.92

Table 18: Summary of SOTA LLM-content Detectors.

Study		Approach				Transfer-ability	#Hum. Evalu.	Domain				Dataset	
		Tool	Stat	Hum	DNN			News	QA	Essay	Res.	Size	Open
Pre-ChatGPT	[30]		●			–	–	●			●	300	
	[55]		●		●	–	–	●				90k	●
	[73]	●				–	–	●		●		–	
	[98]		●		●		–				●	28k	●
	[109]			●	●		[a]	●				20k	●
ChatGPT	[9]				●		–			●		100k	
	[28]	●		●		–	2				●	100	
	[35]		●	●	●		17		●			125k	●
	[37]	●	●		●	●			●			6k	●
	[54]	●	●						●		●	134k	
	[63]	●	●	●	●		43			●		8k	●
	[68]	●	●		●			●				74k	●
	[104]		●		●	●		●	●	●	●	122k	●
	Ours	●	●	●	●	●	**242**[b]	●		●	●	**>2.8m**	●

Pre-ChatGPT: Grover and GPT-2/3. Tool: if the mechanism used or evaluated online detection tools. Sat: Statistical approach. Hum: human evaluation. # Hum. Evalu.: number of human evaluators. Res.: research papers/abstracts. Open: if the mechanism is open-sourced.

[a] The number of human evaluators is not explicitly provided.

[b] 155 evaluators in the first experiment and 87 evaluators in the second experiment.

"weak indicators" from the lexicon, structural, and semantic features that are scattered across the entire text snippet to collectively support a classification decision with relatively strong confidence.

6 RELATED WORK

Neural Language Models and LLM. Neural networks for word probabilistic modeling have been developed since the 2000s [6, 16, 96]. Recently, pre-trained language models with general but effective word representation have been widely used, e.g., BERT [20], RoBERTa [62], ELMo [81], GPT-2 [84], and BART [56]. The large language models (LLMs) are trained on massive amounts of data with deep learning frameworks that consist of an ultra-large number of parameters. ChatGPT is built on top of OpenAI's GPT-3.5 with fine-tuning through supervised and reinforcement learning.

LLM-Content Detection. The detection of LLM-texts can be categorized into white-box and black-box approaches [97]. Prior works [30, 55, 73, 98, 109] study LLM-content detection for pre-ChatGPT models. [28] evaluated 50 ChatGPT-written biomedical research abstracts with human reviewers and a RoBERTa-based classifier. Their findings show that 34% of the abstracts are labeled as likely human-written. [9] trained a transformer-based deep learning model to distinguish between AI-generated and human-written essays in a range of different education levels. [35] and [63] conduct comprehensive studies, including human evaluators. [35] analyzes a series of question-answering datasets in both English and Chinese, and [63] targets essays written by students and English learners. [37] establishes the first machine-generated text benchmark evaluating a number of detection approaches.

The detailed comparisons of existing GPT detectors are listed in Table 18. Compared with the other approaches, CheckGPT (1) collects/uses a significantly larger dataset, (2) uses a model-agnostic design for higher affordability, upgradability, and flexibility, and (3) achieves very high accuracy, transferability, and interoperability.

DEMASQ [54] is the most recent work that is similar to CheckGPT. It designed an energy-based model to identify the inherent differences between GPT- and human-generated text. Experiments across 8 sub-datasets achieved an accuracy of 73.9% to 100%. DEMASQ employs a very different attack model as it assumes alterations to ChatGPT output made by humans to evade detection. Its detailed alteration algorithm, datasets, and models are not yet open-sourced. Therefore, we are unable to run a comparative study.

Security and Ethics in AIGC Application. AI-generated content (AIGC) has been used in adversarial activities before LLMs were introduced [24], while ChatGPT may provide a powerful tool to malicious actors [19, 86]. The detection of AI-backed bots, scams, and misinformation has been extensively studied in the literature [17, 66, 90, 107], while the rise of ChatGPT introduces both new opportunities [21, 29, 38, 39, 42, 103] and challenges [14, 18, 32, 33, 36, 71, 87]. While Open AI has enforced internal mechanisms to prohibit the unethical use of ChatGPT, the restrictions could be evaded [57, 61]. Finally, there are also discussions and concerns with ChatGPT's potential impact on education and research [25, 94, 106], especially on authorship and plagiarism [3, 26, 49, 95].

7 CONCLUSION

In this paper, we first present GPABench2, a benchmarking dataset with 2.385 million samples of human-written, GPT-written, GPT-completed, and GPT-polished research paper abstracts. Next, we show that the existing ChatGPT detectors and human users are incapable of identifying GPT-content in GPABench2. We further present CheckGPT, a deep learning-based detector for GPT-generated academic writing. With extensive experiments, we show that CheckGPT is highly accurate, affordable, flexible, and transferable.

ACKNOWLEDGMENTS

The authors were supported in part by US National Science Foundation (NSF) under grant numbers IIS-2014552 and DGE-1565570, and the Ripple University Blockchain Research Initiative. The authors would like to thank all the volunteers who participated in the user study. The authors would like to thank the anonymous reviewers for their valuable comments and suggestions.

REFERENCES

[1] Fatih Kadir Akın et al. 2023. Awesome ChatGPT Prompts. Available at: https://github.com/f/awesome-chatgpt-prompts.

[2] Kevin Amiri. 2023. A collection of ChatGPT, GPT-3.5, GPT-4 prompts. Available at: https://github.com/kevinamiri/Instructgpt-prompts.

[3] Brent A Anders. 2023. Is using ChatGPT cheating, plagiarism, both, neither, or forward thinking?. In *Patterns*, Vol. 4. Elsevier.

[4] Anton Bakhtin, Sam Gross, Myle Ott, Yuntian Deng, Marc'Aurelio Ranzato, and Arthur Szlam. 2019. Real or fake? learning to discriminate machine from human generated text. In *arXiv:1906.03351*.

[5] Christos Baziotis, Athanasiou Nikolaos, Pinelopi Papalampidi, Athanasia Kolovou, Georgios Paraskevopoulos, Nikolaos Ellinas, and Alexandros Potamianos. 2018. NTUA-SLP at SemEval-2018 Task 3: Tracking Ironic Tweets using Ensembles of Word and Character Level Attentive RNNs. In *International Workshop on Semantic Evaluation*.

[6] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2000. A neural probabilistic language model. In *NeurIIPS*, Vol. 13.

[7] Douglas Biber. 1991. *Variation across speech and writing*. Cambridge University Press.

[8] Douglas Biber and Bethany Gray. 2010. Challenging stereotypes about academic writing: Complexity, elaboration, explicitness. In *Journal of English for Academic Purposes*, Vol. 9. Elsevier, 2–20.

[9] Arend Groot Bleumink and Aaron Shikhule. 2023. Keeping AI Honest in Education: Identifying GPT-generated text. Available at: https://www.aicheatcheck.com/.

[10] Jordan Boyd-Graber, Naoaki Okazaki, and Anna Rogers. 2023. ACL 2023 Policy on AI Writing Assistance. https://2023.aclweb.org/blog/ACL-2023-policy/.

[11] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *NeurIIPS*.

[12] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *NeurIIPS*.

[13] Martin Brümmer, Milan Dojchinovski, and Sebastian Hellmann. 2016. DBpedia abstracts: a large-scale, open, multilingual NLP training corpus. In *International Conference on Language Resources and Evaluation*.

[14] Enrico Cambiaso and Luca Caviglione. 2023. Scamming the Scammers: Using ChatGPT to Reply Mails for Wasting Time and Resources. In *arXiv:2303.13521*.

[15] Margaret Cargill and Patrick O'Connor. 2021. *Writing scientific research articles: Strategy and steps*. John Wiley & Sons.

[16] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*.

[17] Stefano Cresci. 2020. A decade of social bot detection. In *Communications of the ACM*, Vol. 63. 72–83.

[18] Luigi De Angelis, Francesco Baglivo, Guglielmo Arzilli, Gaetano Pierpaolo Privitera, Paolo Ferragina, Alberto Eugenio Tozzi, and Caterina Rizzo. 2023. ChatGPT and the rise of large language models: the new AI-driven infodemic threat in public health. In *Frontiers in Public Health*, Vol. 11. Frontiers, 1567.

[19] Erik Derner and Kristina Batistič. 2023. Beyond the Safeguards: Exploring the Security Risks of ChatGPT. In *arXiv:2305.08005*.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *arXiv:1810.04805*.

[21] David Dukić, Dominik Keča, and Dominik Stipić. 2020. Are you human? Detecting bots on Twitter Using BERT. In *DSAA*.

[22] Sabit Ekin. 2023. Prompt Engineering For ChatGPT: A Quick Guide To Techniques, Tips, And Best Practices. TechRxiv.

[23] Mohammed Abu El-Nasr. 2023. Sentence embeddings using Siamese RoBERTa-networks. Available at: https://keras.io/examples/nlp/sentence_embeddings_with_sbert/.

[24] Emilio Ferrara, Onur Varol, Clayton Davis, Filippo Menczer, and Alessandro Flammini. 2016. The rise of social bots. In *Communications of the ACM*, Vol. 59. 96–104.

[25] Mehmet Firat. 2023. What ChatGPT means for universities: Perceptions of scholars and students. In *Journal of Applied Learning and Teaching*, Vol. 6.

[26] Annette Flanagin, Kirsten Bibbins-Domingo, Michael Berkwits, and Stacy L Christiansen. 2023. Nonhuman "authors" and implications for the integrity of scientific publication and medical knowledge. In *Jama*, Vol. 329. American Medical Association, 637–639.

[27] The Hewlett Foundation. 2012. The Hewlett Foundation: Automated Essay Scoring. Kaggle, available at: https://www.kaggle.com/c/asap-aes.

[28] Catherine A Gao, Frederick M Howard, Nikolay S Markov, Emma C Dyer, Siddhi Ramesh, Yuan Luo, and Alexander T Pearson. 2022. Comparing scientific abstracts generated by ChatGPT to original abstracts using an artificial intelligence output detector, plagiarism detector, and blinded human reviewers. In *bioRxiv:2022.12.23.521610*.

[29] Andres Garcia-Silva, Cristian Berrio, and José Manuel Gómez-Pérez. 2019. An empirical study on pre-trained embeddings and language models for bot detection. In *Workshop on Representation Learning for NLP*.

[30] Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. 2019. GLTR: Statistical Detection and Visualization of Generated Text. In *ACL*.

[31] Noah Goodman. 2023. Meta-Prompt: A Simple Self-Improving Language Agent. Available at: https://noahgoodman.substack.com/p/meta-prompt-a-simple-self-improving.

[32] Kacper T Gradonm. 2023. Electric Sheep on the Pastures of Disinformation and Targeted Phishing Campaigns: The Security Implications of ChatGPT. In *IEEE S&P*.

[33] Dijana Vukovic Grbic and Igor Dujlovic. 2023. Social Engineering with chatgpt. In *International Symposium INFOTEH-JAHORINA*.

[34] Derek Greene and Pádraig Cunningham. 2006. Practical Solutions to the Problem of Diagonal Dominance in Kernel Document Clustering. In *ICML*.

[35] 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. In *arXiv:2301.07597*.

[36] Julian Hazell. 2023. Large language models can be used to effectively scale spear phishing campaigns. In *arXiv:2305.06972*.

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

[38] Maryam Heidari and James H Jones. 2020. Using bert to extract topic-independent sentiment features for social media bot detection. In *IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference*.

[39] Maryam Heidari, Samira Zad, Parisa Hajibabaee, Masoud Malekzadeh, Seyyed-Pooya HekmatiAthar, Ozlem Uzuner, and James H Jones. 2021. Bert model for fake news detection based on social bot activities in the covid-19 pandemic. In *IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference*.

[40] Eli Hinkel. 2003. *Teaching academic ESL writing: Practical techniques in vocabulary and grammar*. Routledge.

[41] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. In *Neural computation*, Vol. 9. MIT press, 1735–1780.

[42] Emma Hoes, Sacha Altay, and Juan Bermeo. 2023. Using ChatGPT to Fight Misinformation: ChatGPT Nails 72% of 12,000 Verified Claims. PsyArXiv.

[43] Or Honovich, Uri Shaham, Samuel R Bowman, and Omer Levy. 2023. Instruction induction: From few examples to natural language task descriptions. In *ACL*.

[44] Benjamin D Horne, Jeppe Nørregaard, and Sibel Adali. 2019. Robust fake news detection over time and attack. In *ACM Transactions on Intelligent Systems and Technology*, Vol. 11. 1–23.

[45] Huggingface. [n. d.]. RoBERTa. Available at: https://huggingface.co/docs/transformers/model_doc/roberta.

[46] Ken Hyland. 2005. Stance and engagement: A model of interaction in academic discourse. In *Discourse studies*, Vol. 7. SAGE Publications, 173–192.

[47] Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. Automatic Detection of Generated Text is Easiest when Humans are Fooled. In *ACL*.

[48] Ashish Jaiswal. 2023. Smart ChatGPT Prompts. Available at: https://github.com/asheeshcric/smart-chatgpt-prompts.

[49] Mohammad Khalil and Erkan Er. 2023. Will ChatGPT get you caught? Rethinking of plagiarism detection. In *International Conference on Human-Computer Interaction*.

[50] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. In *Proceedings of the National Academy of Sciences of the United States of America*, Vol. 114. JSTOR, 3521–3526.

[51] Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. In *TACL*, Vol. 6. MIT Press, 317–328.

[52] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *NeurIIPS*.

[53] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *NeurIIPS*.

[54] Kavita Kumari, Alessandro Pegoraro, Hossein Fereidooni, and Ahmad-Reza Sadeghi. 2024. DEMASQ: Unmasking the ChatGPT Wordsmith. In *NDSS*.

[55] Laida Kushnareva, Daniil Cherniavskii, Vladislav Mikhailov, Ekaterina Artemova, Serguei Barannikov, Alexander Bernstein, Irina Piontkovskaya, Dmitri Piontkovski, and Evgeny Burnaev. 2021. Artificial Text Detection via Examining the Topology of Attention Maps. In *EMNLP*.

[56] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*.

[57] Haoran Li, Dadi Guo, Wei Fan, Mingshi Xu, and Yangqiu Song. 2023. Multi-step Jailbreaking Privacy Attacks on ChatGPT. In *Findings of EMNLP*.
[58] Min Lin, Qiang Chen, and Shuicheng Yan. 2014. Network in network. In *ICLR*.
[59] Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring How Models Mimic Human Falsehoods. In *ACL*.
[60] Jiacheng Liu, Alisa Liu, Ximing Lu, Sean Welleck, Peter West, Ronan Le Bras, Yejin Choi, and Hannaneh Hajishirzi. 2022. Generated Knowledge Prompting for Commonsense Reasoning. In *ACL*.
[61] Yi Liu, Gelei Deng, Zhengzi Xu, Yuekang Li, Yaowen Zheng, Ying Zhang, Lida Zhao, Tianwei Zhang, and Yang Liu. 2023. Jailbreaking ChatGPT via Prompt Engineering: An Empirical Study. In *arXiv:2305.13860*.
[62] 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. In *arXiv:1907.11692*.
[63] Yikang Liu, Ziyin Zhang, Wanyang Zhang, Shisen Yue, Xiaojing Zhao, Xinyuan Cheng, Yiwen Zhang, and Hai Hu. 2023. ArguGPT: evaluating, understanding and identifying argumentative essays generated by GPT models. In *arXiv:2304.07666*.
[64] Ilya Loshchilov and Frank Hutter. 2017. SGDR: Stochastic Gradient Descent with Warm Restarts. In *ICLR*.
[65] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *ICLR*.
[66] Zhuoran Lu, Patrick Li, Weilong Wang, and Ming Yin. 2022. The Effects of AI-based Credibility Indicators on the Detection and Spread of Misinformation under Social Influence. In *Proceedings of the ACM on Human-Computer Interaction*.
[67] Tony Lynch and Kenneth Anderson. 2013. Grammar for academic writing. In *English Language Teaching Centre*. 1–6.
[68] Dominik Macko, Robert Moro, Adaku Uchendu, Jason Lucas, Michiharu Yamashita, Matúš Pikuliak, Ivan Srba, Thai Le, Dongwon Lee, Jakub Simko, et al. 2023. MULTITuDE: Large-Scale Multilingual Machine-Generated Text Detection Benchmark. In *EMNLP*.
[69] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2024. Self-refine: Iterative refinement with self-feedback. In *NeurIPS*.
[70] Kamil Malinka, Martin Perešíni, Anton Firc, Ondřej Hujňák, and Filip Januš. 2023. On the educational impact of ChatGPT: Is Artificial Intelligence ready to obtain a university degree?. In *Conference on Innovation and Technology in Computer Science Education*.
[71] Steve Mansfield-Devine. 2023. Weaponising ChatGPT. In *Network Security*, Vol. 2023. MA Business London.
[72] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. 2018. Mixed Precision Training. In *ICLR*.
[73] Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature. In *ICML*.
[74] Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the Stability of Fine-tuning BERT: Misconceptions, Explanations, and Strong Baselines. In *ICLR*.
[75] OpenAI. 2023. AI Text Classifier. Available at: https://platform.openai.com/ai-text-classifier.
[76] OpenAI. 2023. Educator considerations for ChatGPT. Available at: https://platform.openai.com/docs/chatgpt-education.
[77] OpenAIcommunity. 2019. roberta-base-openai-detector. Available at: https://huggingface.co/openai-community/roberta-base-openai-detector.
[78] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In *NAACL*.
[79] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2020. Finetuning RoBERTa on a custom classification task. Available at: https://github.com/facebookresearch/fairseq/blob/main/examples/roberta/README.custom_classification.md.
[80] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.
[81] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *NAACL*.
[82] PlexPt. 2023. Awesome ChatGPT Prompts. Available at: https://github.com/PlexPt/awesome-chatgpt-prompts.
[83] Prompt Perfect. 2023. Available at: https://promptperfect.xyz/.
[84] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. In *OpenAI blog*, Vol. 1. 9.
[85] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *EMNLP*.
[86] Karen Renaud, Merrill Warkentin, and George Westerman. 2023. From ChatGPT to HackGPT: Meeting the Cybersecurity Threat of Generative AI. In *MIT Sloan Management Review*, Vol. 64. Massachusetts Institute of Technology, 1–4.
[87] Sayak Saha Roy, Krishna Vamsi Naragam, and Shirin Nilizadeh. 2023. Generating Phishing Attacks using ChatGPT. In *arXiv:2305.05133*.
[88] Malik Sallam. 2023. ChatGPT utility in healthcare education, research, and practice: systematic review on the promising perspectives and valid concerns. In *Healthcare*, Vol. 11. MDPI, 887.
[89] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *NeulIPS Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing*.
[90] Wajiha Shahid, Bahman Jamshidi, Saqib Hakak, Haruna Isah, Wazir Zada Khan, Muhammad Khurram Khan, and Kim-Kwang Raymond Choo. 2022. Detecting and mitigating the dissemination of fake news: Challenges and future research opportunities. In *IEEE Transactions on Computational Social Systems*.
[91] 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. In *arXiv:1908.09203*.
[92] solrevdev. 2023. chatgpt-prompt-gen.txt. Available at: https://gist.github.com/solrevdev/c9ada9de794237acdd5028418cea8ec5.
[93] StackOverflow. 2023. Use of ChatGPT generated text for content on Stack Overflow is temporarily banned. Available at: https://meta.stackoverflow.com/questions/421831/temporary-policy-chatgpt-is-banned.
[94] Chris Stokel-Walker. 2022. AI bot ChatGPT writes smart essays-should academics worry?. In *Nature*.
[95] Chris Stokel-Walker. 2023. ChatGPT listed as author on research papers: many scientists disapprove. In *Nature*.
[96] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. LSTM neural networks for language modeling. In *Annual conference of ISCA*.
[97] Ruixiang Tang, Yu-Neng Chuang, and Xia Hu. 2024. The science of detecting llm-generated texts. In *Communications of the ACM*, Vol. 67. 50–59.
[98] Panagiotis C Theocharopoulos, Panagiotis Anagnostou, Anastasia Tsoukala, Spiros V Georgakopoulos, Sotiris K Tasoulis, and Vassilis P Plagianakos. 2023. Detection of Fake Generated Scientific Abstracts. In *IEEE International Conference on Big Data Computing Service and Applications*.
[99] Edward Tian. 2023. GPTZero. Available at: https://gptzero.me/.
[100] Shangqing Tu, Chunyang Li, Jifan Yu, Xiaozhi Wang, Lei Hou, and Juanzi Li. 2023. ChatLog: Recording and Analyzing ChatGPT Across Time. In *arXiv:2304.14106*.
[101] Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship attribution for neural text generation. In *EMNLP*.
[102] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE.. In *JMLR*, Vol. 9.
[103] Hong Wang, Xuan Luo, Weizhi Wang, and Xifeng Yan. 2023. Bot or Human? Detecting ChatGPT Imposters with A Single Question. In *arXiv:2305.06424*.
[104] Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Alham Fikri Aji, et al. 2024. M4: Multi-generator, Multi-domain, and Multilingual Black-Box Machine-Generated Text Detection. In *EACL*.
[105] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C Schmidt. 2023. A prompt pattern catalog to enhance prompt engineering with chatgpt. In *arXiv:2302.11382*.
[106] Jurgen Willems. 2023. ChatGPT at universities–the least of our concerns. In *SSRN:4334162*.
[107] Kai-Cheng Yang, Onur Varol, Pik-Mai Hui, and Filippo Menczer. 2020. Scalable and generalizable social bot detection through data selection. In *AAAI*.
[108] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. 2017. Comparative study of CNN and RNN for natural language processing. In *arXiv:1702.01923*.
[109] Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *NeurIPS*.
[110] ZeroGPT. 2023. AI Text Detector. Available at: https://www.zerogpt.com.
[111] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. 2023. Least-to-most prompting enables complex reasoning in large language models. In *ICLR*.
[112] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. In *ICLR*.