CheckGPT: Detecting ChatGPT-Generated Contents in Academic Writing

Anonymous ACL submission

Abstract

We present a holistic investigation of the detection of LLM-generated academic writing by providing the dataset, user study, and algorithms, aiming to inspire more community effort to address the concern of LLM academic misuse. We first introduce GPABench2, a benchmarking dataset of 2.385 million samples of human-written, GPT-written, GPT-completed, and GPT-polished abstracts of research papers in subjects of computer science, physics, and humanities & social sciences. Through a user study of 155 participants, we show the complication for human users, including experienced faculty and researchers, to identify GPT-generated abstracts. Last, we present CheckGPT, a LLM-content detector consisting of a general representation module and an attentive-BiLSTM classification module, which is highly accurate and transferable.

1 Introduction

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 (Stokel-Walker, 2022; Khalil and Er, 2023), or in phishing and romance scams (Roy et al., 2023; Grbic and Dujlovic, 2023). 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 LLM-content.

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 perform poorly, especially in detecting GPT-polished text (Sec. 3.2). Some experiences in identifying LLM-content have been reported in the literature, e.g., Guo et al. (2023) and Liu et al. (2023b) noted that ChatGPT output tends to be more objective, formal, focused, and fluent than human-content. 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 2.385M 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 for visual inspection based on language appearance, with or without individual experiences of writing research articles. In addition, we test multiple state-of-the-art algorithmic detectors on GPABench2, e.g., GPTZero, and show that they demonstrate modest to poor performance, especially with GPT-polished text.

Finally, we develop and evaluate a language-model-based detection framework, named CheckGPT, to explore the possibility of building automated tools for LLM-content detection in a niche area. Specifically, 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 it 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. 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 in academia, and a valuable resource to assist in the design of new detecting methods.
- We evaluate the SOTA GPT detectors and show that they provide unsatisfactory performance with GPABench2. Meanwhile, with a user study of 155 participants, we show that even experienced faculty/researchers are unable to distinguish between human-written and GPT-generated academic writing.
- We present CheckGPT, a deep-learning-based and model-agnostic GPT-content detector with validated benefits of affordability, transferability, and interpretability. We demonstrate the outstanding performance of CheckGPT (∼99% average accuracy) with extensive experiments. We share CheckGPT at `https://anonymous.4open.science/r/CheckGPT-80B2`.

2 Background and Related Work

LLMs and LLM-Content Detection. ChatGPT is built on top of OpenAI's GPT-3.5 with fine-tuning through both supervised and reinforcement learning techniques. Benefiting from the large-scale autoregressive pre-training based on transformer networks and comprehensive fine-tuning based on reinforcement learning from human feedback (RLHF), ChatGPT is proven to mimic a versatile human conversationalist and succeed in many writing generation tasks. 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*; and (2) false and redundant information may flood the publication systems. Academic conferences started to ban LLM-generated texts (e.g., ICML) or enforce rules to require disclosure of LLM usage (e.g., ACL, Nature and RSC). However, such policies are hardly enforceable without an effective LLM-content detector. A small-scale experiment by Gao et al. (2022) showed that most of the GPT-generated abstracts were deemed as original by a web-based plagiarism detector.

LLM-Content Detection. LLM-content detection can be categorized into white-box and black-box approaches (Tang et al., 2023). Black-box detectors are further grouped into *feature-based* detectors, which examine hand-crafted statistical disparities, linguistic patterns, and facts (Tang et al., 2023), and *model-based* detectors, which learn another language model to discriminate linguistic characteristics between human-written and machine-generated texts. CheckGPT belongs to the second category. Table 1 summarizes the SOTA LLM-content detectors. Compared with them, CheckGPT collects and uses a significantly larger dataset, adopts a model-agnostic design for better affordability, upgradability, and flexibility, and achieves very high accuracy and transferability.

3 GPABenchmark: GPT Corpus for Academia

3.1 The GPABench2 Dataset

When ChatGPT is adopted for academic writing, such as essays, reports, and even research papers (Firat, 2023; Stokel-Walker, 2022), the generated text akin to academic writing style is often objective, formal, fluent, and focused (Guo et al., 2023; Liu et al., 2023b), posing more challenges to the detectors. In this paper, we introduce *GPABench2* (GPABenchmark version 2), a large-scale GPT-generated text corpus for academic writing.

We first collected published research papers (title and abstracts) from three disciplines: "computer science" (CS) abstracts from top-tier conferences and arXiv, "physics" (PHX) from arXiv, and "humanities and social sciences" (HSS) from Springer's SSRN that includes 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 GP-3). Eventually, we

Table 1: Summary of SOTA LLM-content Detectors. Tool: used online detection tools; stat: Statistical features.

	Study	Approach				Transfer-ability	# Human Assessors	Application Domain				Dataset	
		Tool	Stat.	Human	Train DL			News	QA	Essay	Paper	Size	Open
GPT-2&3	Gehrmann et al., 2019		●			–	–	●			●	300	
	Kushnareva et al., 2021		●		●	–	–	●				90k	●
	Mitchell et al., 2023	●				–	–	●		●		–	
	Theocharopoulos et al., 2023		●		●		–				●	28k	●
	Zellers et al., 2019			●	●		∗	●				20k	●
ChatGPT	Bleumink and Shikhule, 2023				●		–			●		100k	
	Gao et al., 2022	●		●		–	2				●	100	
	Guo et al., 2023		●	●	●		17		●			125k	●
	Liu et al., 2023b	●	●	●	●		43			●		8k	●
	Ours	●	●	●	●	●	**155**	●		●	●	**2.38M**	●

∗ The number of human evaluators is not explicitly provided.

collected 50,000 papers from each discipline.

We define three tasks based on the most representative scenarios where LLMs are used or misused in academic writing. In ***Task 1: GPT-written full abstracts (GPT-WRI)***, ChatGPT generates an abstract with only a given title. In ***Task 2: GPT-completed abstracts (GPT-CPL)***, with some user-provided seed text, ChatGPT completes the rest of the essay following the logic of the seed. 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. In ***Task 3: GPT-polished abstracts (GPT-POL)***, the entire abstract is provided to ChatGPT, which re-writes the text sentence-by-sentence.

In the rest of the paper, we denote the abstracts written by human authors and ChatGPT as *human-written* abstracts (HUM) and *GPT-Generated* abstract (GPT-GEN), respectively. The latter is a superset of 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 (White et al., 2023) and prompt guidelines (PlexPt, 2023; Akın et al., 2023; Amiri, 2023; Jaiswal, 2023) in the literature and crafted four distinct prompts for each task, denoted as Prompts 1 to 4 (presented in Appendix A.1): Prompt 1 is a 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 an increasing level of knowledge provided to ChatGPT.

We invoke ChatGPT (gpt-3.5-turbo) through OpenAI's API to generate the abstracts at the cost of 0.2 cents per 1,000 tokens. 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. 6.5). Eventually, GPABench2 contains 2.385M total samples (2.235M GPT-GEN and 0.15M HUM).

3.2 Benchmarking ChatGPT Detectors

Open-source and commercial ChatGPT detectors have been developed to detect AI-generated text. We evaluated the accuracy of three representative ChatGPT detectors, GPTZero (Tian, 2023), ZeroGPT (zer, 2023), and OpenAI's classifier (OpenAI, 2023a), over our academic abstract dataset. Due to a lack of API, slow responses, and expenses, 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 2. Note that, in Task 2 (GPT-completed abstracts), we only submitted the second half of each abstract to the detectors.

From the performance summary in Table 2 and the detailed results in Appendix B, we have three observations: (1) all three detectors demonstrated modest to poor detection 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). Given that these models are not explicitly trained with academic datasets, their inadequacy can be excused. However, the results show that generic detectors struggle in specific tasks, indicating a limited transferability. The gap highlights the need for effective detectors for this niche domain with the potential to transfer to related domains.

Table 2: Performance of open-source and commercial GPT detectors. Red: detection accuracy <50%, or average score on the wrong side of the decision threshold.

	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
HUM	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
HUM	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
HUM	0.09	0.13	0.52	0.08	0.04	0.47	0.20	0.14	0.64
(c.1) Detection accuracy (in %) of OpenAI's detector									
GPT	80.7	70	63	63.7	23.7	27.3	6.3	4.3	6
HUM	51.0	69.7	84.0	35.3	59.7	79.6	50.7	69.0	88.0
(c.2) 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
HUM	1.42	1.17	0.59	1.71	1.35	0.68	1.38	1.14	0.52

4 User Study: Identification of Human- and GPT-Generated Abstracts

In the user study, we attempt to answer three research questions: (1) Could (experienced) researchers distinguish between human-written and GPT-generated research abstracts? (2) Do prior experiences with reading/writing papers contribute to the capability of identifying GPT-generated abstracts? (3) Does the researchers' capability in identifying GPT content vary by discipline?

We designed a questionnaire as follows[1]: 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."

We distributed questionnaires to faculty members, researchers, and graduate students in the Departments of EECS, Physics, and College of Liberal Arts at our University (in the US) and a research organization in Europe. In four weeks, we received 155 responses with 465 annotated abstracts. The overall accuracy was 48.82%, which is slightly worse than random guesses. The detailed statistics are shown in Table 3. From the responses, we have the following observations: (1) It is very challenging for human users to distinguish between human-written and GPT-generated paper abstracts (only 21 users correctly identified all three abstracts). (2) Participants have the tendency to annotate abstracts as "human-written", i.e., 59.66% of GPT-generated abstracts were mistakenly labeled as "human". The results confirm the public opinion that ChatGPT achieves human-like language style and quality. (3) Users perform better in identifying GPT-written abstracts and worse in GPT-polished abstracts. (4) Users' self-claimed expertise only slightly affects their identification capability. And (5) users perform better in physics and significantly worse in humanity and social sciences.

Table 3: Detailed results of the user study. Pat.: number of participants; Abs.: number of annotated abstracts; Cor.: number of correct annotations; Acc.: accuracy; HUM: accuracy for human-written abstracts; GPT: accuracy for GPT-generated abstracts.

Category	Par.	Abs.	Cor.	Acc.	HUM	GPT
Role						
Faculty	44	132	65	49.2%	58.6%	41.9%
Researchers	30	90	45	50.0%	58.2%	37.1%
Students	81	243	117	48.1%	56.3%	40.3%
Discipline						
CS	57	171	86	50.3%	59.0%	43.0%
Physics	48	144	77	53.5%	65.1%	37.7%
HSS	50	150	64	42.7%	46.5%	39.2%
Self-claimed Familiarity with Research Papers						
Expert	52	156	80	51.3%	60.6%	43.5%
Knowledgable	56	168	80	47.6%	57.3%	34.7%
Somewhat	39	117	57	48.7%	56.0%	43.3%
No familiarity	8	24	10	41.7%	46.7%	33.3%
Published papers?						
Yes	106	318	155	48.7%	58.1%	39.2%
No	49	147	72	49.0%	55.6%	42.7%

5 CheckGPT: An Accurate Detector for ChatGPT-generated Academic Writing

5.1 The System Model and Assumptions

We denote the CheckGPT classifier as $\mathcal{H}$. The classification problem can be formulated as:

$$\hat{y} = \mathcal{H}(\mathrm{s}) \tag{1}$$

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

where s is an unstructured text snippet (abstract). Given s, $\mathcal{H}(\mathrm{s})$ generates the probability distribution $\hat{y}$ considering label space {'h', 'g'}, where 'h' indicates HUM and 'g' indicates GPT-GEN. The goal is to find an optimal set of parameter θ for

[1] This user study was reviewed and approved by the Human Research Protection Program at [Anonymized] University.

$\mathcal{H}$ to minimize the loss function $\mathcal{L}$ measuring the distance between prediction $\hat{y}$ and observation y.

CheckGPT is a *black-box detector*, which needs only the access to the observed samples, i.e., no insider knowledge of ChatGPT. We further make the following assumptions: (1) *Moderate data* – with the rate limit and cost of OpenAI's API, an ordinary user cannot have massive amounts (billions) of training samples. (2) *Affordability* – we develop a lightweight solution that smaller entities without requiring excessive computing power could easily adopt. And (3) *Local deployment* – the detector should be easily transferred to a new domain using a small amount of potentially private data from the target domain. Finally, all the models and datasets used in CheckGPT are publicly available.

5.2 The CheckGPT Framework

Input Representation. Our CheckGPT includes two stages: *representation* and *classification*. For text representation, CheckGPT adopts a model-agnostic design offering high affordability, upgradability, and flexibility. Our proof-of-concept prototype of CheckGPT uses the tokenizer and encoders of RoBERTa-large (Liu et al., 2019). The tokenization can be formalized as:

$$\mathbf{X} = \text{BPE}(\mathbf{s}) = \{x_i\}_{i=1}^{n} \tag{3}$$

where $\mathbf{X}$ denotes a sequence of length n consisting of individual tokens x_i, and BPE refers to the byte-level pairing encoding utilized by RoBERTa.

For the embedding layer, RoBERTa generates an embedding e_i of size 1024 per token. The text sequences are transformed into contextualized representations $\mathbf{E}$ with a $n \times 1024$ shape. The encoding can be formalized as:

$$\mathbf{E} = \text{TransformerEncoder}(\mathbf{X}) = \{e_i\}_{i=1}^{n} \tag{4}$$

LSTM Classification. The derived embeddings are fed into the bi-directional LSTM-based classifier (Hochreiter and Schmidhuber, 1997) f_θ. Our classifier consists of two layers, each with a hidden dimension of 256 and a following attention layer (Baziotis et al., 2018). The outputs from the two layers are concatenated and passed through a dropout layer with a rate of $p = 0.5$, and a dense layer. Details of the model architecture can be found in Appendix C.2. The softmaxed output indicates the conditional probability of the two classes: "GPT-generated" (y_g) or "Human-generated" (y_h). The function of our LSTM classifier $f_\theta(\mathbf{E})$ can be represented as follows:

$$\begin{aligned} h_1 = \text{LSTM}_1(\mathbf{E}),\quad r_1 = \text{ATTN}_1(h_1) \\ h_2 = \text{LSTM}_2(h_1),\quad r_2 = \text{ATTN}_2(h_2) \\ (\hat{y}_g, \hat{y}_h) = \text{Softmax}(\text{FC}(\text{Dropout}(r_1 \oplus r_2))) \end{aligned} \tag{5}$$

Model Training. The classifier f_θ with parameter θ is optimized independently with the RoBERTa frozen during the training. We adopt an AdamW optimizer (Loshchilov and Hutter, 2019), a CosineAnnealing learning rate scheduler (Loshchilov and Hutter, 2017), and a gradient scaler for efficient mixed-precision training (Micikevicius et al.). 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 cross-entropy loss of a data sample is defined as:

$$\mathcal{L}(\theta) = -\left[y_c \log(\hat{y}_c) + y_h \log(\hat{y}_h)\right] \tag{6}$$

Design Choices and Discussions. One alternative approach is directly applying RoBERTa by adding a RobertaClassificationHead (Huggingface). 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 (Yin et al., 2017). Refer to Section 6.1 for details of ablation study.

Another alternative approach is to fine-tune the entire pre-trained model (Ott et al., 2019, 2020) on the new dataset. However, our CheckGPT design has distinct advantages: (1) **Efficiency**: CheckGPT significantly reduces the parameters to save both time and computing resources. Given the parameters of language models ranging from 66M (DistilledBERT, Sanh et al., 2019) to 355M (RoBERTa-large, Liu et al., 2019) and 1750M (GPT-3, Brown et al., 2020a), our efficient model only has 4M parameters (during training). The drop in model size also reduces the risks of over-fitting and catastrophic forgetting (Mosbach et al., 2021; Kirkpatrick et al., 2017), especially with small datasets (Uchendu et al., 2020; Bakhtin et al., 2019). (2) **Applicability**: Our framework is model-agnostic and compatible with various representation approaches (e.g., BERT(Devlin et al., 2018), BART(Lewis et al., 2019)), making it a lightweight and universal detector, as detailed in Section 6.1. This feature is especially valuable considering deployment and customization in academia and education. (3) **Versatility**: By freezing the LLM's well-crafted parameters, we retain the meta-knowledge to the greatest extent to

Table 4: 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
From top to bottom: Prompt 1, 2, 3, 4									
TPR	99.95	99.98	99.88	99.59	99.30	99.38	99.19	99.08	99.23
TNR	99.97	99.99	99.98	99.60	99.58	99.63	99.15	99.28	99.04
ACC	99.96	99.98	99.93	99.60	99.44	99.50	99.17	99.18	99.14
TPR	99.99	99.99	99.99	99.54	99.50	99.61	98.80	99.49	99.22
TNR	99.99	99.99	99.96	99.68	99.54	99.63	99.17	99.34	99.14
ACC	99.99	99.99	99.96	99.61	99.52	99.62	98.98	99.42	99.18
TPR	99.97	99.99	99.95	99.72	99.58	99.58	99.26	99.35	99.31
TNR	100.0	100.0	99.94	99.72	99.74	99.63	98.64	99.48	99.36
ACC	99.98	100.0	99.94	99.72	99.66	99.60	98.95	99.42	99.34
TPR	100.0	99.99	99.96	99.69	99.69	99.65	99.09	99.43	99.22
TNR	99.99	100.0	99.99	99.73	99.83	99.73	99.42	99.64	99.55
ACC	100.0	100.0	99.98	99.71	99.76	99.69	99.26	99.54	99.38

improve CheckGPT's transferability for new domains, as detailed in Section 6.2, which is challenging for finetuned RoBERTa (Wang et al., 2023c).

6 Experiments

We implement CheckGPT with PyTorch 1.13.1 in Python 3.9.1 on Ubuntu 22.04, running on an Nvidia 2080Ti GPU and an Intel i9-9900k CPU. The pre-trained RoBERTa is adopted from Huggingface. See Appendix C.1 for more details.

6.1 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 4. TPR denotes the proportion of correctly identified GPT-GEN abstracts out of all GPT-GEN abstracts. TNR is the proportion of correctly identified HUM abstracts out of all HUM abstracts.

CheckGPT achieves very high performance 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 may be explained by shorter text lengths and better writing by ChatGPT given more seed information. The classification accuracy of Task 3, which is most challenging for the open-source and commercial detectors (Sec. 3.2), is between 98.9% and 99.5%.

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

Table 5: Comparison with other design choices.

Model	Para Size	Acc(%) Task 1	Task 2	Task 3
(a) Other representation module + CheckGPT classifier				
GLoVe	-	99.77	98.34	95.90
BERT	-	99.90	99.28	97.81
(b) CheckGPT representation module + other classifier				
RCH	1.05M	99.80	97.70	94.08
MLP-Pool	1.05M	99.87	98.62	95.93
CNN	3.33M	99.80	98.47	96.49
BiLSTM w/o attention	4.21M	99.91	99.54	98.92
CheckGPT(ours)	4.21M	99.96	99.60	99.17

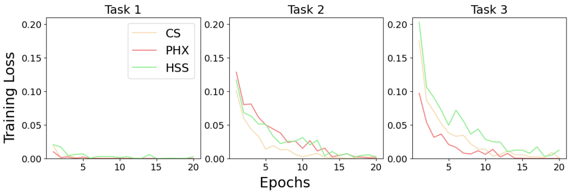


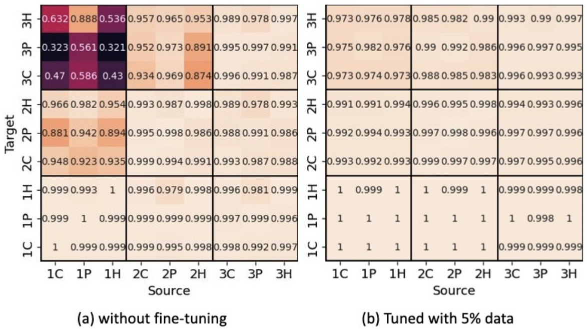
Figure 1: Training loss of the task-specific and discipline-specific classifiers.

Finally, we use t-Distributed Stochastic Neighbor Embedding (t-SNE) to analyze the vector representations extracted from the last layer and discuss the observations in Appendix D.1.

Ablation Study. We compare the current design of CheckGPT with several alternatives. As the baseline, GLoVe with a two-layer MLP (size 1024) yields an F1-score of 0.755 (Task 3). Next, we keep the attentive-BiLSTM classification head in CheckGPT and replace the representation module with GLoVe6B-100d (Pennington et al., 2014) or pre-trained BERT-base. Last, we keep the representation module and replace the classifier with: (1) the default classification head for RoBERTa (RCH) in Huggingface, and its variant with global average pooling (MLP-Pool; Lin et al., 2013; El-Nasr, 2023); (3) an AlexNet-like CNN (Krizhevsky et al., 2012) with five convolutional layers, and (4) a basic BiLSTM classifier without attention. As shown in Table 5, CheckGPT achieves the best accuracy.

6.2 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 6 (a)) to evaluate testing samples from



(a) without fine-tuning

(b) Tuned with 5% data

Figure 2: 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 6: 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	99.98	99.99	99.98	99.84	99.77	99.78	99.70	99.78	99.70
TNR	99.96	100.0	99.97	99.52	99.55	99.51	98.44	98.85	98.80
	(b) Cross-prompt Cross-disciplinary Classifiers								
TPR	99.97	99.99	100.0	99.85	99.72	99.84	99.76	99.81	99.78
TNR	99.98	100.0	99.93	99.41	99.67	99.06	98.93	99.23	99.01
	(c) Cross-prompt & -task & -disciplinary Classifier								
TPR	100.0	100.0	100.0	99.89	99.90	99.90	99.69	99.82	99.82
TNR	98.88	99.07	98.59	99.04	98.85	98.70	98.88	99.07	98.59

Table 7: 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

Table 8: 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
From top to bottom: Wiki, Essay-C, Essay-P, BBC						
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
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
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
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

other tasks and disciplines, *without model fine-tuning*. In Figure 2 (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. CheckGPT achieves ≥0.978 accuracy on *cross-discipline* data from the *same task*. However, CheckGPT is less adaptable *across tasks*. In particular, Task 1 models perform the worst on data from Task 3. Task 3 models demonstrate solid performance on other Tasks. It implies that CheckGPT learns subtle but inherent features of AIGC in the most challenging Task 3.

We then fine-tune the last linear layer of each model with the data in the target domain. As shown in Figure 2(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 2(a).

The Unified Classifiers. We evaluate CheckGPT by pooling data from all prompts (train with 160K GPT samples for each task/discipline) and show the classification accuracy in Table 6 (a). We then combine data across disciplines (Table 6 (b)) and further across all tasks (Table 6 (c)). In summary, unified training slightly improves TPR, especially for difficult tasks, e.g., GPT-POL in CS.

Prompt Transferability. To assess CheckGPT's generalizability over the domain shifts caused by ChatGPT prompts, we train CheckGPT with data from 3 prompts and test it with samples from the fourth prompt. As shown in Table 7, CheckGPT is very transferable across prompts.

6.3 Transferability to New Domains

We evaluate CheckGPT with three NLP datasets: Wiki Abstracts (1,500 random samples from Brümmer et al., 2016), ASAP Essays (Foundation, 2012), and BBC News (Greene and Cunningham, 2006). 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 (2012) for Task 1 and designed the prompts for the other tasks and datasets (see Appendix C.3 for details).

We apply the cross-prompt cross-discipline CheckGPTclassifiers (Sec. 6.2) on the new domains. As shown in Table 8, 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.

6.4 Transferability to New LLMs

We invoke Bard and GPT-4 with the same prompts in Sec. 3.1 to generate LLM-WRI, LLM-CPL, and LLM-POL abstracts for 100 and 2000 random samples, respectively (small sample size due to a lack

Table 9: CheckGPT's TPR (in %) for Bard and GPT-4.

	Task 1	Task 2	Task 3
Bard	99.00	96.00	82.00
GPT-4	99.95	96.90	96.15

of API (Bard) and strict rate limits). We use the unified classifiers to evaluate all the samples and show the TPRs in Table 9. CheckGPT achieves >96% TPR in 5 experiments. The TPR for Bard-polished text is relatively low. Further investigation shows that Bard makes minimal changes (e.g., copy editing a few words) for some samples. Therefore, these misclassifications appear reasonable.

6.5 Prompt Engineering

Research efforts on prompt engineering aim to guide or improve the design of ChatGPT prompts (White et al., 2023; Ekin, 2023). We select six approaces that are widely adopted in the community: (1) Zero-shot Chain-of-Thought Prompting (ZC, Kojima et al., 2022) enforces step-by-step reasoning with specific trigger phrases like "Let's think step by step." (2) Automatic Prompt Engineer (APE, Zhou et al., 2023b) automates the creation and selection of prompts using iterative optimization. (3) Self-critique Prompting (SCP, Madaan et al., 2023) employs GPT to evaluate its own responses and provide feedback. (4) Few-shot Prompting (FSP, Brown et al., 2020b) conditions the model using examples or demonstrations. (5) Least-to-Most Prompting (LMP, Zhou et al., 2023a) parses a problem into simpler subproblems. (6) Generated Knowledge Prompting (GKP, Liu et al., 2022) starts the prompt with relevant information generation. We also adopt four prompt refinement methods: (1) Prompt Perfect (PP, pro, 2023). (2) GPT-generated Prompts (GP, solrevdev, 2023). (3) Meta Prompts (MP, Goodman, 2023). (4) Instruction Induction (II, Honovich et al., 2023, not for Task 3). We use each method to write, complete, and polish 5,000 abstracts from each discipline (please refer to Appendix A.2 for details).

We evaluate the new dataset with task-specific, discipline-specific, and cross-prompt classifiers. As shown in Table 10, 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% (detailed in Appendix D.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 10: TPR (in %) for 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	-	-	-

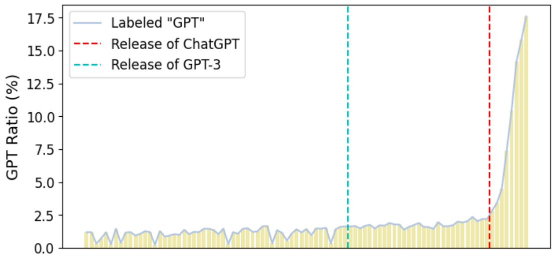


Figure 3: Detecting ChatGPT usage in arXiv papers.

6.6 Use of ChatGPT in arXiv Papers

Finally, we ask "*How many authors are using ChatGPT to write/polish their research papers*?" We collected all the arXiv abstracts in CS from 01/2016 to 07/2023 (∼400K 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 3. There is a significant increase in ChatGPT usage with a peak of 17.59% in July 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.

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 abstracts of research papers. Second, we show that the existing ChatGPT detectors and human users are incapable of identifying GPT-content in GPABench2. Finally, we 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.

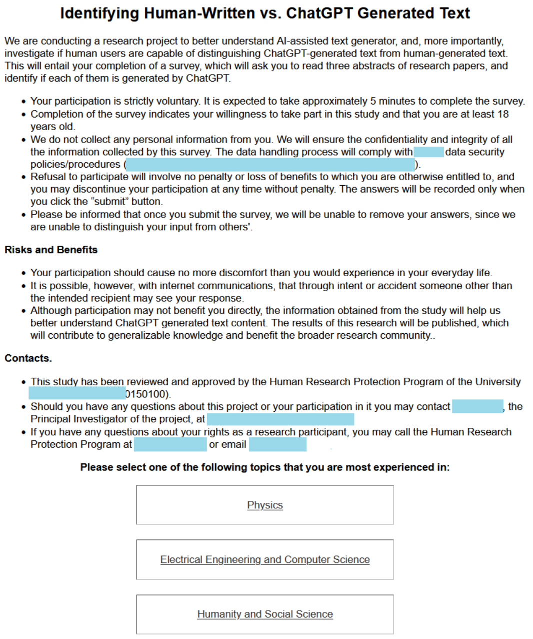
8 Ethical Considerations

Data Collection. All the research paper abstracts collected in Section 3 are open to the public. We invoked ChatGPT's API (@ 0.2 cents per 1,000 tokens) to collect the GPT-generated abstracts. OpenAI's terms allow the use of the generated content for research and publication: "*OpenAI hereby assigns to you all its right, title and interest in and to Output. This means you can use Content for any purpose, including commercial purposes such as sale or publication, if you comply with these Terms.*" The GPABench2 dataset and the CheckGPT tool will be shared with the community.

User Study. The user study in Section 4 was reviewed and approved by the Human Research Protection Program at the [Anonymized] University. An IRB information statement is displayed on the landing page, as shown in Figure 4. The median time spent with the questionnaire was 110 seconds. Faculty, researchers, and students participating in this survey do not get paid.

General Discussions on Security and Ethics in AIGC Application. AI-generated content (AIGC) has been used in adversarial activities even before LLMs were introduced (Ferrara et al., 2016), while the recent release of ChatGPT may have provided the malicious actors with a powerful tool (Renaud et al., 2023; Derner and Batistič, 2023). The rise of LLMs/ChatGPT introduces new opportunities (Heidari et al., 2021; Heidari and Jones, 2020; Dukić et al., 2020; Garcia-Silva et al., 2019; Hoes et al., 2023; Wang et al., 2023a) and challenges (Gradonm, 2023; De Angelis et al., 2023; Mansfield-Devine, 2023), e.g., ChatGPT may be used in scamming or phishing (Grbic and Dujlovic, 2023; Roy et al., 2023; Hazell, 2023). Open AI has enforced internal mechanisms to prohibit the unethical use of ChatGPT, however, the restrictions could be evaded through prompt engineering (jailbreaking) (Li et al., 2023; Liu et al., 2023a).

The academic community is actively discussing how AI writing assistance tools may pose potential challenges to research and education (Sallam, 2023; Malinka et al., 2023; Stokel-Walker, 2022; Willems, 2023; Firat, 2023), especially on authorship and plagiarism (Stokel-Walker, 2023; Flanagin et al., 2023; Khalil and Er, 2023; Anders, 2023). OpenAI also posted their perspectives on the education-related risks and opportunities (OpenAI, 2023c). In this paper, we provide a detection tool for LLM-Content. *The impact of ChatGPT and other AI writing assistance tools on academic integrity is outside of the scope of the paper.*



Identifying Human-Written vs. ChatGPT Generated Text

We are conducting a research project to better understand AI-assisted text generator, and, more importantly, investigate if human users are capable of distinguishing ChatGPT-generated text from human-generated text. This will entail your completion of a survey, which will ask you to read three abstracts of research papers, and identify if each of them is generated by ChatGPT.

- Your participation is strictly voluntary. It is expected to take approximately 5 minutes to complete the survey.
- Completion of the survey indicates your willingness to take part in this study and that you are at least 18 years old.
- We do not collect any personal information from you. We will ensure the confidentiality and integrity of all the information collected by this survey. The data handling process will comply with [redacted] data security policies/procedures ([redacted]).
- Refusal to participate will involve no penalty or loss of benefits to which you are otherwise entitled to, and you may discontinue your participation at any time without penalty. The answers will be recorded only when you click the "submit" button.
- Please be informed that once you submit the survey, we will be unable to remove your answers, since we are unable to distinguish your input from others'.

Risks and Benefits

- Your participation should cause no more discomfort than you would experience in your everyday life.
- It is possible, however, with internet communications, that through intent or accident someone other than the intended recipient may see your response.
- Although participation may not benefit you directly, the information obtained from the study will help us better understand ChatGPT generated text content. The results of this research will be published, which will contribute to generalizable knowledge and benefit the broader research community..

Contacts.

- This study has been reviewed and approved by the Human Research Protection Program of the University [redacted]0150100).
- Should you have any questions about this project or your participation in it you may contact [redacted], the Principal Investigator of the project, at [redacted]
- If you have any questions about your rights as a research participant, you may call the Human Research Protection Program at [redacted] or email [redacted]

Please select one of the following topics that you are most experienced in:

Physics
Electrical Engineering and Computer Science
Humanity and Social Science

Figure 4: The landing page of the user study.

9 Limitations

The main limitations of the proposed CheckGPT mechanism are discussed as follows. (1) While CheckGPT demonstrates good transferability to new domains and new models (Sec. 6.3 and 6.4), it is not yet evaluated in large-scale assessments in broader scopes. As a general discussion, the detection accuracy of CheckGPT will decrease when the content in the target domain differs from academic writing since CheckGPT is specifically designed and trained for this niche domain. (2) We also plan to investigate how the users may manipulate the prompts or re-edit the GPT-generated text to escape the detectors. Post-processing may present a significant challenge, as knowledgeable users with insights into the detector may purposefully revise GPT-GEN text to evade detection. (3) As discussed in the literature and in Section 2, feature-based LLM-content detectors are more explainable, since the hand-crafted features are often comprehensible to human users. However, they often suffer from limited accuracy and scalability. Meanwhile, the model-based detectors, including CheckGPT, suffer from explainability. While CheckGPT provides outstanding detection accuracy, it is difficult to provide a human-comprehensible justification of the decisions. It is our future work to investigate explainable AI solutions for CheckGPT.

References

2023. Prompt Perfect. Available at: https://promptperfect.xyz/.

2023. ZeroGPT: AI Text Detector. Available at: https://www.zerogpt.com.

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

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

Brent A Anders. 2023. Is using chatgpt cheating, plagiarism, both, neither, or forward thinking? *Patterns*, 4(3).

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

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. *arXiv preprint arXiv:1906.03351*.

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 *Proceedings of the 12th International Workshop on Semantic Evaluation*, pages 613–621, New Orleans, Louisiana. Association for Computational Linguistics.

Arend Groot Bleumink and Aaron Shikhule. 2023. Keeping ai honest in education: Identifying gpt-generated text.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020a. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020b. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Martin Brümmer, Milan Dojchinovski, and Sebastian Hellmann. 2016. Dbpedia abstracts: a large-scale, open, multilingual nlp training corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3339–3343.

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. *Frontiers in Public Health*, 11:1567.

Erik Derner and Kristina Batistič. 2023. Beyond the safeguards: Exploring the security risks of chatgpt. *arXiv preprint arXiv:2305.08005*.

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

David Dukić, Dominik Keča, and Dominik Stipić. 2020. Are you human? detecting bots on twitter using bert. In *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 631–636. IEEE.

Sabit Ekin. 2023. Prompt engineering for chatgpt: A quick guide to techniques, tips, and best practices.

Mohammed Abu El-Nasr. 2023. Sentence embeddings using siamese roberta-networks.

Emilio Ferrara, Onur Varol, Clayton Davis, Filippo Menczer, and Alessandro Flammini. 2016. The rise of social bots. *Communications of the ACM*, 59(7):96–104.

Mehmet Firat. 2023. What chatgpt means for universities: Perceptions of scholars and students. *Journal of Applied Learning and Teaching*, 6(1).

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. *Jama*, 329(8):637–639.

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

Deep Ganguli, Amanda Askell, Nicholas Schiefer, Thomas Liao, Kamilė Lukošiūtė, Anna Chen, Anna Goldie, Azalia Mirhoseini, Catherine Olsson, Danny Hernandez, et al. 2023. The capacity for moral self-correction in large language models. *arXiv preprint arXiv:2302.07459*.

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. *bioRxiv*, pages 2022–12.

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 *Proceedings of the 4th Workshop on*

Representation Learning for NLP (RepL4NLP-2019), pages 148–155.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. 2019. Gltr: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 111–116.

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

Kacper T Gradonm. 2023. Electric sheep on the pastures of disinformation and targeted phishing campaigns: The security implications of chatgpt. *IEEE Security & Privacy*, 21(3):58–61.

Dijana Vukovic Grbic and Igor Dujlovic. 2023. Social engineering with chatgpt. In *2023 22nd International Symposium INFOTEH-JAHORINA (INFOTEH)*, pages 1–5. IEEE.

Derek Greene and Pádraig Cunningham. 2006. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proc. 23rd International Conference on Machine learning (ICML'06)*, pages 377–384. ACM Press.

Stanford NLP Group. 2022. GloVe: Global Vectors for Word Representation. Available at: https://nlp.stanford.edu/projects/glove/.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. *arXiv preprint arXiv:2301.07597*.

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

Maryam Heidari and James H Jones. 2020. Using bert to extract topic-independent sentiment features for social media bot detection. In *2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 0542–0547. IEEE.

Maryam Heidari, Samira Zad, Parisa Hajibabaee, Masoud Malekzadeh, SeyyedPooya 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 *2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 0103–0109. IEEE.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Emma Hoes, Sacha Altay, and Juan Bermeo. 2023. Using chatgpt to fight misinformation: Chatgpt nails 72% of 12,000 verified claims.

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

Huggingface. RoBERTa. Available at: https://huggingface.co/docs/transformers/model_doc/roberta.

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

Mohammad Khalil and Erkan Er. 2023. Will chatgpt get you caught? rethinking of plagiarism detection. *arXiv preprint arXiv:2302.04335*.

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. *Proceedings of the National Academy of Sciences of the United States of America*, 114(13):3521–3526.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

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 *ACL Anthology*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Haoran Li, Dadi Guo, Wei Fan, Mingshi Xu, and Yangqiu Song. 2023. Multi-step jailbreaking privacy attacks on chatgpt. *arXiv preprint arXiv:2304.05197*.

Min Lin, Qiang Chen, and Shuicheng Yan. 2013. Network in network. *arXiv preprint arXiv:1312.4400*.

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 *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3154–3169.

Yi Liu, Gelei Deng, Zhengzi Xu, Yuekang Li, Yaowen Zheng, Ying Zhang, Lida Zhao, Tianwei Zhang, and Yang Liu. 2023a. Jailbreaking chatgpt via prompt engineering: An empirical study. *arXiv preprint arXiv:2305.13860*.

Yikang Liu, Ziyin Zhang, Wanyang Zhang, Shisen Yue, Xiaojing Zhao, Xinyuan Cheng, Yiwen Zhang, and Hai Hu. 2023b. Argugpt: evaluating, understanding and identifying argumentative essays generated by gpt models. *arXiv preprint arXiv:2304.07666*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*.

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? *arXiv preprint arXiv:2303.11146*.

Steve Mansfield-Devine. 2023. Weaponising chatgpt. *Network Security*, 2023(4).

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. In *International Conference on Learning Representations*.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature. *arXiv preprint arXiv:2301.11305*.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. In *9th International Conference on Learning Representations*.

OpenAI. 2023a. AI Text Classifier. Available at: https://platform.openai.com/ai-text-classifier.

OpenAI. 2023b. Chatgpt plugins.

OpenAI. 2023c. Educator considerations for ChatGPT. Available at: https://platform.openai.com/docs/chatgpt-education.

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 *Proceedings of NAACL-HLT 2019: Demonstrations*.

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.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

PlexPt. 2023. Awesome ChatGPT Prompts. Available at: https://github.com/PlexPt/awesome-chatgpt-prompts.

Karen Renaud, Merrill Warkentin, and George Westerman. 2023. From chatgpt to hackgpt: Meeting the cybersecurity threat of generative ai. *MIT Sloan Management Review*, 64(3):1–4.

Sayak Saha Roy, Krishna Vamsi Naragam, and Shirin Nilizadeh. 2023. Generating phishing attacks using chatgpt. *arXiv preprint arXiv:2305.05133*.

Malik Sallam. 2023. Chatgpt utility in healthcare education, research, and practice: systematic review on the promising perspectives and valid concerns. In *Healthcare*, volume 11, page 887. MDPI.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. 2022. Self-critiquing models for assisting human evaluators. *arXiv preprint arXiv:2206.05802*.

solrevdev. 2023. chatgpt-prompt-gen.txt. Available at: https://gist.github.com/solrevdev/c9ada9de794237acdd5028418cea8ec5.

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

Chris Stokel-Walker. 2023. Chatgpt listed as author on research papers: many scientists disapprove. *Nature*.

Ruixiang Tang, Yu-Neng Chuang, and Xia Hu. 2023. The science of detecting llm-generated texts. *arXiv preprint arXiv:2303.07205*.

Panagiotis C Theocharopoulos, Panagiotis Anagnostou, Anastasia Tsoukala, Spiros V Georgakopoulos, Sotiris K Tasoulis, and Vassilis P Plagianakos. 2023. Detection of fake generated scientific abstracts. *arXiv preprint arXiv:2304.06148*.

Edward Tian. 2023. GPTZero. Available at: https://gptzero.me/.

Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship attribution for neural text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8384–8395.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Hong Wang, Xuan Luo, Weizhi Wang, and Xifeng Yan. 2023a. Bot or human? detecting chatgpt imposters with a single question. *arXiv preprint arXiv:2305.06424*.

Rui Wang, Hongru Wang, Fei Mi, Yi Chen, Ruifeng Xu, and Kam-Fai Wong. 2023b. Self-critique prompting with large language models for inductive instructions. *arXiv preprint arXiv:2305.13733*.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Alham Fikri Aji, et al. 2023c. M4: Multi-generator, multi-domain, and multi-lingual black-box machine-generated text detection. *arXiv preprint arXiv:2305.14902*.

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. *arXiv preprint arXiv:2302.11382*.

Jurgen Willems. 2023. Chatgpt at universities–the least of our concerns. *Available at SSRN 4334162*.

Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. 2017. Comparative study of cnn and rnn for natural language processing. *arXiv preprint arXiv:1702.01923*.

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

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. 2023a. Least-to-most prompting enables complex reasoning in large language models. In *ICLR*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023b. Large language models are human-level prompt engineers. In *ICLR*.

Radim Řehůřek. 2022. Gensim: Topic modelling for humans. Available at: https://radimrehurek.com/gensim/.

A Prompts used in GPABench2

A.1 Prompts Used in the Main Dataset of GPABench2

The complete prompts used in GPABench2 data collection are listed as follows:

1. Prompt 1: zero-shot prompt.
   Task 1: Here is the title of an academic research paper. Please write a paper abstract about it: {input}.
   Task 2: Here is the first half of the abstract of an academic research paper. Please complete its second half with approximate {X} words: {input}.
   Task 3: Here is the abstract of an academic research paper. Please rewrite it for clarity: {input}.
2. Prompt 2: Prompt with context.
   Task 1: Write an abstract of a research paper in {discipline} with first-person, clear, and academic language about "{title}".
   Task 2: Write a well-written and coherent continuation, with approximately {X} words, of the following first half of the abstract of a research paper in {discipline}: "input"
   Task 3: Write a polished and refined version of the following abstract of a research paper in {discipline} to improve its overall quality and readability: "{input}"
3. Prompt 3: Role-playing prompt.
   Task 1: I want you to act as an academic paper writer. You are familiar with the topics in {discipline}. You will be responsible for writing a paper abstract. Your task is to generate an abstract for a paper with a given title. Please only include the written abstract in your answer. Here is the title of the paper: "{input}"
   Task 2: I want you to act as an academic paper writer. You are familiar with the topics in {discipline}. You will be responsible for completing an unfinished paper abstract. Your task is to create a seamless and well-written continuation with approximately {X} words for the second half, given the provided first half of the abstract. Please only include the second half in your answer. Here is the first half of the abstract: "{input}"
   Task 3: I want you to act as an academic paper writer. You are familiar with the topics in {discipline}. You will be responsible for rewriting a paper abstract. Your task is to improve the writing and clarity of the abstract. Please only include the rewritten abstract in your answer. Here is the original abstract of the paper: "{input}"
4. Prompt 4: detailed user requirements and instructions.
   Task 1: Please act as an expert paper writer and write the abstract section of a paper from the perspective of a paper reviewer to make it fluent and elegant. Please only include the written abstract in your answer. Here are the specific requirements: 1. Enable readers to grasp the main points or essence of the paper quickly. 2. Allow readers to understand the important information, analysis, and arguments throughout the entire paper. 3. Help readers remember the key points of the paper. 4. Please clearly state the innovative aspects of your research in the abstract, emphasizing your contributions. 5. Use concise and clear language to describe your findings and results, making it easier for reviewers to understand the paper. Here is the title of the paper: "{input}"
   Task 2: Please act as an expert paper writer and complete the second half of the given first half of an abstract section from the perspective of a paper reviewer to make it fluent and elegant. Please only include the second half of the abstract in your answer. Here are the specific requirements: 1. The length of the second half should be about {X} words. 2. The existing content should serve as the foundation, and the new portion should seamlessly integrate with it. 3. Use your expertise and maintain its technical accuracy and clarity. 4. Ensure a coherent and logical flow between the first and second halves. 5. Use clear and academic language, making it easier for reviewers to understand the paper. Here is the first half of the paper abstract section: "{input}"
   Task 3: Please act as an expert paper editor and revise the abstract section of the paper from the perspective of a paper reviewer to make it more fluent and elegant. Please only include the revised abstract in your answer. Here are the specific requirements: 1. Enable readers to grasp the main points or essence of the paper quickly. 2. Allow readers to understand the important information, analysis, and arguments throughout the entire paper. 3. Help readers remember the key points of the paper. 4. Please clearly state the innovative aspects of your research in

the abstract, emphasizing your contributions. 5. Use concise and clear language to describe your findings and results, making it easier for reviewers to understand the paper. Here is the original abstract section of the paper: "{input}"

A.2 Prompts used in the Additional Testing Samples

The details of the prompt techniques covered in Sec 6.5 are as follows.

1. **Zero-shot Chain-of-Thought Prompting (ZC).** Zero-shot Chain-of-Thought (Zero-shot CoT) Prompting (Kojima et al., 2022) utilizes a trigger phrase like "Let's think step by step." to guide the model through a sequence of necessary reasoning steps for the problems. Each prompt has two parts: the first generates a chain of thought, and the second extracts the final answer. In our experiment, we adhered the trigger phrase to our original prompts.

2. **Automatic Prompt Engineer (APE).** APE (Zhou et al., 2023b) automates the process of generation and selection of the prompts for LLMs with an iterative scoring and resampling mechanism. We simplify this process by directly adopting the optimal trigger phrase "Let's work this out in a step by step way to make sure that we have the correct (good) answer." from (Zhou et al., 2023b).

3. **Self-critique Prompting (SCP).** This method (Madaan et al., 2023) engages LLMs in a self-evaluation process to enhance model performance (Wang et al., 2023b; Ganguli et al., 2023; Bai et al., 2022; Saunders et al., 2022). The LLMs provide self-reflective feedback or suggestions on their own responses and improve them. In our experiment, we instruct ChatGPT to perform self-critique and self-improvement subsequently.

4. **Few-shot Prompting (FSP).** Few-shot prompting (Brown et al., 2020b), also widely recognized as few-shot in-context learning, involves a set of demonstrations or examples to condition the LLMs to the context. In our experiment, we provide three paper abstracts each time to facilitate ChatGPT's understanding of academic writing styles.

5. **Least-to-Most Prompting (LMP).** This method (Zhou et al., 2023a) consists of two stages: decomposing the problem into easier subproblems and solving them subsequently. In our experiment, we asked ChatGPT to decompose our original question and respond following the devised recipe.

6. **Generated Knowledge Prompting (GKP).** Generated Knowledge Prompting (Liu et al., 2022) includes two stages: initial queries asking the LLM to give relevant information, which is subsequently refined as the context for further instructions. This recursive prompting technique leverages the LLM's knowledge-generation capability.

7. **GPT-generated Prompts (GP).** Following (solrevdev, 2023), we appoint ChatGPT as a prompt generator. We assign the task of drafting and improving the prompts to optimally align with user needs while ensuring their clarity, conciseness, and comprehensibility for ChatGPT. Here is the prompt used in this paper: "I want you to become my prompt generator. Your goal is to help me craft the best possible prompt for my needs. The prompt will be used by you, ChatGPT. You will follow the following process: 1. Your first response will be to ask me what the prompt should be about. I will provide my answer, but we will need to improve it. 2. Based on my input, you will generate the revised prompt. It should be clear, concise, and easily understood by you."

8. **Prompt Perfect (PP).** Prompt Perfect (pro, 2023), a third-party plugin supported in the OpenAI GPT-4 interface (OpenAI, 2023b), rephrases user inputs to improve the quality of ChatGPT's responses. In our experiments, we use Prompt Perfect to rephrase our original prompt.

9. **Meta Prompts (MP).** Similar to self-critique prompting, meta prompts instruct LLMs to revise both the answer and the prompt (Goodman, 2023). At the end of the process, LLMs generate an additional response based on the refined prompt.

10. **Instruction Induction (II).** This method (Honovich et al., 2023) searches the natural language space for an apt description of the target task. It introduces a paradigm where the model is provided with a few input-output

pairs and then prompted to infer a fitting instruction. Task 3 was omitted in our experiments due to the lack of abstracts before and after proficient polishing. For Task 1 and 2, we use the original title and abstract as examples. The prompts inferred by ChatGPT are "Given the title of a research paper, please generate an abstract that outlines the main contributions, methodology, and results of the paper." and "Given an abstract or introduction discussing the motivation and problem definition of a research paper, provide a continuation which describes the proposed solution, methodology, and results.".

B Benchmarking Open and Commercial ChatGPT Detectors

B.1 Benchmarking ZeroGPT

GPTZero. For each text paragraph, GPTZero (Tian, 2023) reports a binary decision of “human” or “GPT”. As shown in Table 2 (a), GPTZero demonstrates very high accuracy with human-written abstracts (98.1% average accuracy across all the topics). However, its detection accuracy for GPT-generated abstracts appears to be very low, with an average accuracy of 24.3%. That is, GPTZero has a very strong tendency to classify an input abstract as “human-written”. From Task 1 to Task 3, the detection performance decreases significantly (from 42.5% to 8.1%). That is, when more information is given to ChatGPT, the generated text appears to be more “human-like” in the eyes of GPTZero.

B.2 Benchmarking ZeroGPT

For each input text snippet, ZeroGPT (zer, 2023) returns one of the nine possible decisions. We assign an integer score of [0, 8] as follows:

0. Your text is Human written
1. Your text is Most Likely Human written
2. Your text is Most Likely Human written, may include parts generated by AI/GPT
3. Your text is Likely Human written, may include parts generated by AI/GPT
4. Your text contains mixed signals, with some parts generated by AI/GPT
5. Your text is Likely generated by AI/GPT
6. Your text is Most Likely AI/GPT generated
7. Most of Your text is AI/GPT Generated
8. Your text is AI/GPT Generated

The distribution of the scores for each task and each discipline is shown in Table 11. For instance, for GPT-polished abstracts (Task 3) in CS, 88.3% were annotated as “human” by ZeroGPT, while 4.7% were annotated as “Most likely human written”.

When we converted the 9-point scores to binary decisions of “GPT”/“Human”, a threshold of 4 was used. While we can also make the case that categories 2, 3, 4 should be categorized as “GPT” in Task 3, since the decision statements indicate that they “may include parts generated by AI/GPT,” which is the case for Task 3. However, changing the decision threshold will not significantly change the observations and conclusions in Section 3.2, since only a very small portion of the samples in Task 3 were annotated with those three labels, as shown in Table 11. For Tasks 1 and 2, the text snippets we sent to ZeroGPT were completely written by ChatGPT, hence, a threshold of 4 is the most reasonable choice.

ZeroGPT’s average detection accuracy for each task and each discipline was presented in Table 2 (b1), and the average score for each experiment in Table 2 (b2). ZeroGPT’s detection accuracy for human-written abstracts is close to 100% in CS and physics, and slightly lower (∼95%) in humanities and social sciences (HSS). Its accuracy with fully GPT-written abstracts is also high, especially for HSS (92.3%). However, the detection accuracy for GPT-completed and GPT-polished abstracts in CS and physics appears to be very low (in the range of [5%, 25.3%]), while the accuracy for HSS appears to be relatively higher. While ZeroGPT claims a detection accuracy of 98%, it appears to be less effective in academic writing. Similar to GPTZero, ZeroGPT also has a tendency to classify GPT-generated text as human-written.

B.3 Benchmarking OpenAI’s Text Classifier

For each input text snippet, the OpenAI text classifier (OpenAI, 2023a) returns a decision from one of the five classes. We map them to an integer score of [0, 4] as follows:

0. The classifier considers the text to be very unlikely AI-generated.
1. The classifier considers the text to be unlikely AI-generated.
2. The classifier considers the text to be unclear if it is AI-generated.
3. The classifier considers the text to be possibly

Table 11: Distribution of detection score generated by the ZeroGPT: 0: human-written; 8: GPT-generated. The largest score category for each experiment is shown in bold.

	T1. GPT-ERI			T2. GPT-CPL			T3. GPT-POL		
	CS	PHX	HSS	CS	PHX	HSS	CS	PHX	HSS
(a) Score distribution (in %) of GPT-generated abstracts.									
0	16.7	21	1.7	**52.7**	**75.7**	18	**88.3**	**93**	**52**
1	4.7	3	2	1.3	1	1	4.7	2	6.7
2	6	5	2	13.3	11.3	13.7	2	1	8.3
3	2.7	1	2	6.7	0.7	3	1.3	0.7	5.3
4	2.7	1.7	0	0.7	1.3	2	0.3	0.7	3
5	4.3	0.7	0.3	5.3	2	7.7	1.3	0	6.7
6	4.7	5.7	2.7	4	3.3	7.3	1	0.7	5
7	8.7	17.3	3.3	3.3	1	9.7	0	0.3	3.3
8	**49.7**	**44.7**	**86**	12.7	3.7	**37.7**	1	1.7	9.7
(b) Score distribution (in %) of human-written abstracts.									
0	**93.7**	**97.7**	**79**	**96.3**	**98.7**	**87.3**	**92**	**96**	**79**
1	3.3	0	9	0.7	0	0.7	4	1	6.7
2	3	0.7	5	2.7	1	5.7	2	1.3	5
3	0	0	2	0	0	1	0.3	0.3	2
4	0	0.3	1.7	0	0	1.3	0.3	0	1.7
5	0	0	1	0	0.3	1	0.3	0.3	2
6	0	0	1.3	0	0	1	0	0	2
7	0	0.3	0.7	0.3	0	0.3	0	0.3	0.3
8	0	1	0.3	0	0	1.7	1	0.7	1.3

Table 12: Distribution of detection score generated by the OpenAI text classifier: 0: very unlikely AI-generated; 2: unclear if it is AI-generated; 4: likely AI-generated. The largest score category for each experiment is shown in bold.

	T1. GPT-ERI			T2. GPT-CPL			T3. GPT-POL		
	CS	PHX	HSS	CS	PHX	HSS	CS	PHX	HSS
(a) Score distribution (in %) of GPT-generated abstracts.									
0	0	0	0	0	0	8	4	5	11.3
1	0.3	0.3	3.3	1.3	12.3	11	23.7	35.3	31.3
2	19	29.7	33.7	35	**64**	**53.7**	**66**	**55.3**	**51.3**
3	**50**	**50.7**	**51**	**56**	22.7	24	6.3	4	6
4	30.7	19.3	12	7.7	1	3.3	0	0.3	0
(b) Score distribution (in %) of human-written abstracts.									
0	11	15.7	**60**	4.3	7.7	**56.3**	12.7	18	**62.0**
1	40	**54**	24	31	**52**	23.3	38	**51**	26
2	**45.3**	28.3	14	**54**	38	16.7	**48.3**	29.7	10.7
3	3.7	2	1.3	10.7	2	3.3	1	1.3	1
4	0	0	0.7	0	0.3	0.3	0	0	0.3

AI-generated.

4. The classifier considers the text to be likely AI-generated.

The distribution of the scores for each task and each discipline is shown in Table 12. For instance, for human-written CS abstracts, 11% are classified as "very unlikely AI-generated, 40% are classified as "unlikely AI-generated", 45.3% are classified as "unclear if it is AI-generated", and the remaining 3.7% are classified as "possibly AI-generated".

We use a threshold of 2 to generate a binary decision for each test. Note that a classification of "(2) unclear if it is AI-generated" is considered wrong for both GPT-generated and human-written inputs. We present OpenAI's classification accuracy in Table 2 (c1) and the average scores in Table 2 (c2). OpenAI's classifier shows slightly different patterns from GPTZero and ZeroGPT. It demonstrates moderate performance in classifying abstracts that are fully written by humans or GPT. However, its accuracy for GPT-completed and GPT-polished abstracts appears inadequate (but slightly better than GPTZero and ZeroGPT). We also noticed that this classifier is very sensitive to the length of text. While it requires a minimum of 1,000 characters for each input text snippet, a shorter input (e.g., input in Task 2 GPT-CPL) is more likely to yield a wrong or "unclear" decision.

Note that OpenAI has taken its detector offline in July 2023, "*due to its low rate of accuracy.*" [2] This is another indication that distinguishing human-written and GPT-generated text is a very challenging task even for the owner of GPT.

C Model and Dataset

C.1 Implementation Details

CheckGPT is trained with an initial learning rate of 2e-4, a batch size of 256, and an early-stop strategy to terminate when the validation loss does not improve for 10 epochs. The default random seed and maximum epochs are set at 100 and 200. The pre-trained BERT and RoBERTa paras are obtained from Huggingface, and we utilize Řehůřek, 2022 and Group, 2022 for GLOVE embeddings.

C.2 CheckGPT Architecture

BERT. The Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018) family of models, 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) (Liu et al., 2019), 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

[2] https://openai.com/blog/new-ai-classifier-for-indicating-ai-written-text

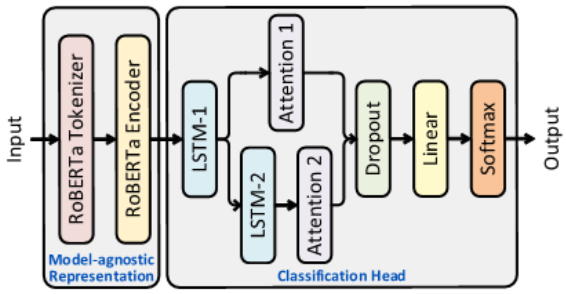
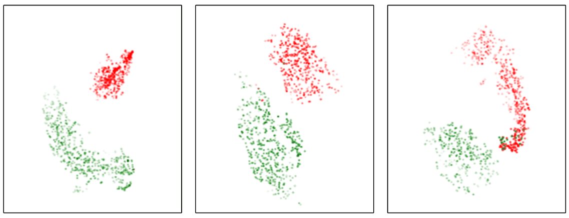


Figure 5: The architecture of the CheckGPT model.



(a) Task 1 GPT-WRI (b) Task 2 GPT-CPL (c) Task 3 GPT-POL

Figure 6: Feature space distribution of human-written (green) and GPT-generated (red) abstracts.

Table 13: TPR (in %) for advanced prompts.

	T1. GPT-WRI			T2. GPT-CPL			T3. GPT-POL		
	CS	PHX	HSS	CS	PHX	HSS	CS	PHX	HSS
ZC	99.96	100.00	99.98	98.67	97.81	98.60	98.01	99.53	99.08
APE	99.96	99.98	99.90	98.14	98.06	98.69	97.05	97.81	98.00
SCP	99.91	99.75	99.81	97.78	97.40	97.56	98.35	99.28	99.08
FSP	99.88	99.98	99.87	99.01	98.82	99.02	97.19	97.93	98.50
LMP	99.82	99.88	99.64	96.95	97.19	97.93	95.47	97.13	97.49
GKP	99.27	99.93	99.98	96.74	97.26	97.87	98.08	99.17	99.37
PP	94.70	100.00	99.96	98.71	99.40	99.00	98.99	99.48	99.31
GP	98.75	99.03	98.71	99.23	99.24	99.48	97.61	99.03	98.68
MP	99.78	99.90	99.83	95.43	96.38	97.01	97.92	99.46	99.12
II	99.95	99.88	99.90	96.20	96.18	90.79	-	-	-

representations, which can be used for downstream tasks. The pre-training of the RoBERTa utilizes a masked language modeling (MLM) objective, which can be formalized as:

$$\mathcal{L}_{\text{MLM}} = -\mathbb{E}_{\mathbf{s}\sim\mathcal{D}_s} \log P(m|\mathbf{s}) \qquad (7)$$

where $\mathcal{D}_s$ is the corpus, s denotes an input sequence, and m is a masked token.

LSTM. Long-Short-Term Memory (Hochreiter and Schmidhuber, 1997) is a variant of Recurrent Neural Networks (RNNs) that has gained incredible success in natural language processing by handling sequential information. LSTM mitigates the gradient vanishing problem and improves model performance over long sequences by incorporating the gating mechanism, which enables it to effectively and selectively retain or update information.

Model Pipeline. In this work, we utilize the pre-trained RoBERTa to preprocess the text data. The representations extracted by RoBERTa serve as the inputs of our downstream classifier, an LSTM network. The pipeline of the model is shown in Figure 5.

C.3 The Datasets in New Domains

Note that the same data samples are used for testing before and after fine-tuning in Section 6.3.

- **Wikipedia Abstracts**. The dataset contains the first introductory section of Wiki articles. We revise the ChatGPT prompts to avoid terms such as "research" and "paper". For example, we use the prompt "*Please generate a brief introduction of ...*" in Task 1.
- **ASAP Essays**. We use two types of essays from the Hewlett Foundation Automated Essay Scoring dataset (Foundation, 2012): [Essay-C] Essay set 1 contains 1,785 essays of 350 words on average. We adopt the original prompt from the dataset in Task 1: "*Write a letter to your local newspaper in which you state your opinion on the effects computers have on people. Persuade the readers to agree with you.*". [Essay-P] Essay set 7 contains 1,730 stories about patience. We refer to the original prompts from the dataset to design ChatGPT prompts e.g., "*write a story in your own way about patience*" is used in Task 1. We design prompts for Tasks 2 and 3 accordingly. We remove essays that are shorter than 70 words.
- **BBC News Article Dataset**. This dataset contains 1,454 BBC news articles from 2004 to 2005 in five topical areas: business, entertainment, politics, sport, and technology (Greene and Cunningham, 2006). We use prompts to emphasize "news articles" to ChatGPT, e.g., "*Please generate a news article titled ...*".

D Additional Experimental Results

D.1 Visualization of t-SNE

Finally, we randomly select 2,000 CS abstracts from each task and each label, and then use t-Distributed Stochastic Neighbor Embedding (t-SNE) (Van der Maaten and Hinton, 2008) to map each 1024-dimension feature vector from the last dense layer of the BiLSTM module into a 3-D space. The visualization is shown in Figure 6. From the figure, we observe that: (a) GPT-written abstracts form a dense cluster, which is different from the varied distribution of the human-written samples, suggesting a consistent vocabulary, writing style, and semantic features. (b) GPT-completed abstracts are significantly 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 the human-written samples, demonstrating the difficulty in separating these two categories.

D.2 Advanced Prompt Engineering

In Table 13, we show the testing accuracy of prompt-specific models on the advanced prompts. As discussed in Section 6.5, the prompt-specific models perform worse than the cross-prompt models. Our interpretation is that the prompt-specific models may have learned some prompt-specific bias, i.e., linguistic features that are only generated by certain prompts. Meanwhile, the cross-prompt models are more likely to learn ChatGPT-specific features, i.e., features that consistently appear in ChatGPT-generated content from different prompts.