How You Prompt Matters! Even Task-Oriented Constraints in Instructions Affect LLM-Generated Text Detection

Ryuto Koike[1] **Masahiro Kaneko**[2,1] **Naoaki Okazaki**[1]
[1]Tokyo Institute of Technology [2]MBZUAI
ryuto.koike@nlp.c.titech.ac.jp
masahiro.kaneko@mbzuai.ac.ae okazaki@c.titech.ac.jp

Abstract

To combat the misuse of Large Language Models (LLMs), many recent studies have presented LLM-generated-text detectors with promising performance. When users instruct LLMs to generate texts, the instruction can include different constraints depending on the user's need. However, most recent studies do not cover such diverse instruction patterns when creating datasets for LLM detection. In this paper, we reveal that even *task-oriented* constraints — constraints that would naturally be included in an instruction and are not related to detection-evasion — cause existing powerful detectors to have a large variance in detection performance. We focus on student essay writing as a realistic domain and manually create task-oriented constraints based on several factors for essay quality. Our experiments show that the standard deviation (SD) of current detector performance on texts generated by an instruction with such a constraint is significantly larger (up to an SD of 14.4 F1-score) than that by generating texts multiple times or paraphrasing the instruction. We also observe an overall trend where the constraints can make LLM detection more challenging than without them. Finally, our analysis indicates that the high instruction-following ability of LLMs fosters the large impact of such constraints on detection performance.[1]

1 Introduction

LLMs have exhibited human-level generative capabilities in response to various textual instructions (OpenAI, 2023b; Touvron et al., 2023). With such remarkable generative ability, malicious users might exploit LLMs for cheating on student homework or fabricating misinformation (Tang et al., 2023; Wu et al., 2023). To mitigate such potential misuse of LLMs, many recent works have presented LLM-generated-text detectors with highly promising detection performance (Kirchenbauer et al., 2023; Mitchell et al., 2023; Guo et al., 2023; Koike et al., 2024; Su et al., 2023).

When users instruct LLMs to generate texts, the instruction potentially includes various constraints (e.g., output format and style) (OpenAI, 2023c). Here, we call such constraints that would naturally be included in instruction and are not related to detection-evasion as *task-oriented* constraints. Despite being very natural, such differences in the instruction can have a large impact on the quality of the generated texts or on the downstream performance of various NLP tasks (Jiang et al., 2020; Zhang et al., 2023a; Feng et al., 2023). Most studies in LLM detection focus on the target LLM-generated text itself, analyzing its linguistic features (Mitrović et al., 2023; Li et al., 2023; Guo et al., 2023; Liu et al., 2023b) and not how the target texts are generated. Moreover, most previous works do not include such a variety of instructions to create their benchmarking datasets for LLM detection (Li et al., 2023; Guo et al., 2023; Liu et al., 2023b). This paper sheds light on the following question: ***Do the task-oriented constraints in generation instruction affect the LLM detection?***

Motivated by this question, this paper first demonstrates that even task-oriented constraints in instruction can lead to inconsistent detection performance of current significant detectors. In particular, as depicted on the right in Figure 1, we explore the standard deviation (SD) of detection performance on various datasets generated via instructions with each different constraint. We focus on student essay writing as one of the generation tasks to consider the constraints, and there is a recognized demand for its detection (OpenAI, 2023a). To generate essays via LLMs, we utilize essay questions created by Koike et al. (2024). Then, as listed in Table 1, we manually create the task-oriented constraints based on each factor of essay quality, defined by Ke and Ng (2019). To verify the impact

[1]Our detection datasets are available at `https://github.com/ryuryukke/HowYouPromptMatters`

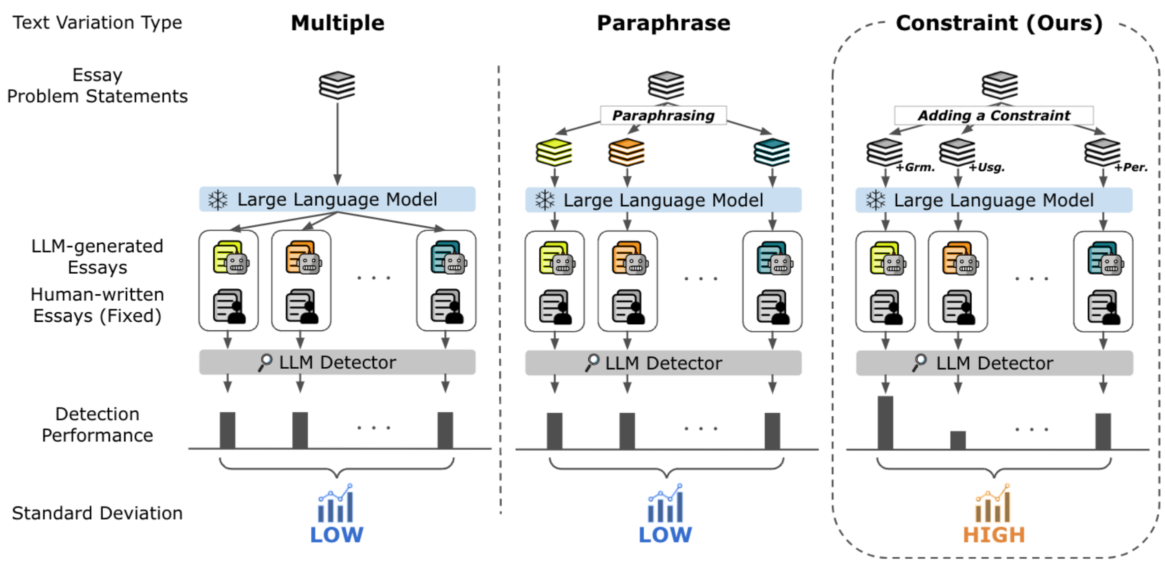


Figure 1: An overview of our `Constraint` setting and two baseline text variation types: `Multiple`, `Paraphrase`. To validate the impact of the constraint in instruction on LLM detection on the generated texts, we compare the SD of detection performance in our `Constraint` setting with that in two baseline settings: `Multiple`, `Paraphrase`. In the `Constraint` setting, `Grm.`, `Usg.`, and `Per.` are the abbreviation of factors for essay quality, listed in Table 1.

Factor		Task-oriented constraint
Grammatically	(Grm.)	Your essay must be free of grammatical errors.
Usage	(Usg.)	Your essay must utilize a professional-level vocabulary.
Mechanics	(Mec.)	Your essay must be free of spelling and capitalization errors.
Style	(Sty.)	Your essay must include diverse word choices and sentence structures.
Relevance	(Rel.)	Your essay must follow the prompt.
Organization	(Org.)	Your essay must be logically organized.
Development	(Dev.)	Your essay must include concrete evidence that supports your opinion.
Cohesion	(Chs.)	Your essay must have a valid connection between paragraphs.
Coherence	(Chr.)	Your essay must have an effective transition throughout all paragraphs.
Thesis Clarity	(TC.)	Your essay must have a clear position through your essay.
Persuasiveness	(Per.)	Your essay must be persuasive to readers.

Table 1: Task-oriented constraints for essay writing based on each factor of essay quality.

of the constraint, we compare the SD with that of two baseline text variation types: generating texts multiple times (via sampling) and paraphrasing the instruction, denoted as `Multiple` and `Paraphrase` in Figure 1.

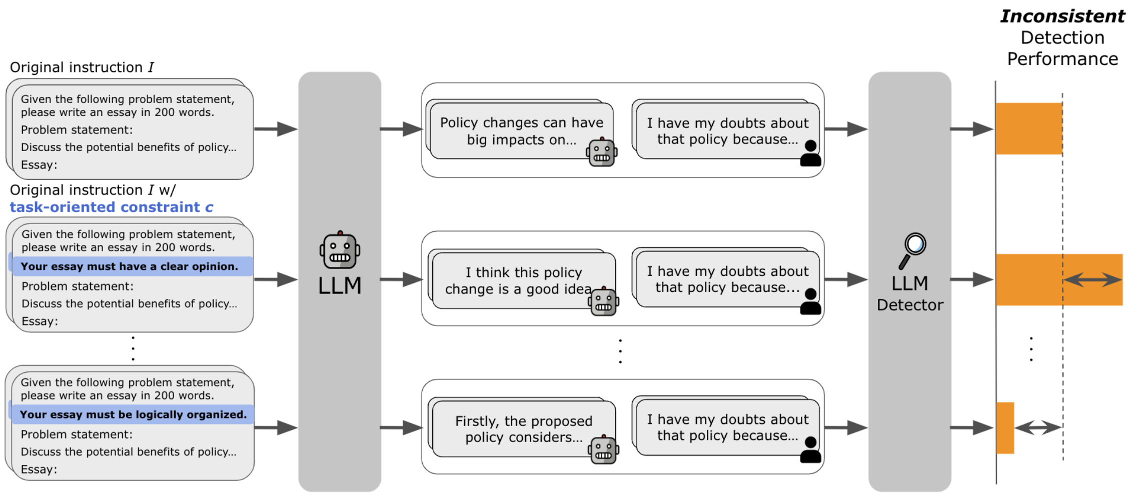
Indeed, our experiments show that a task-oriented constraint in the instruction has a more significant effect on the detection performance than the randomness caused by sampling texts or paraphrasing the instruction. Specifically, the SD of current detector performance on texts generated in our `Constraint` setting is substantially higher (up to an SD of 14.4 F1-score) than that in the `Multiple` and `Paraphrase` settings. We also observe an overall trend where the constraints can make LLM detection more challenging than without them. Finally, our analysis suggests that the high instruction-following ability of LLMs causes the large impact of such constraints on detection performance.

2 Methodology

This section describes our strategy for identifying and evaluating the impact of task-oriented constraints in the instruction on the performance of LLM-generated text detection.

2.1 Task Formulation

Our main task is LLM-generated text detection, specifically discerning LLM-generated essays from human-written essays. To evaluate the performance of current detectors, we utilize a mixture of human-written and LLM-generated essays as our test set. We employ pairs of essay problem statements s_j



Original instruction I
Given the following problem statement, please write an essay in 200 words.
Problem statement:
Discuss the potential benefits of policy...
Essay:
Original instruction I w/ task-oriented constraint c
Given the following problem statement, please write an essay in 200 words.
Your essay must have a clear opinion.
Problem statement:
Discuss the potential benefits of policy...
Essay:
Given the following problem statement, please write an essay in 200 words.
Your essay must be logically organized.
Problem statement:
Discuss the potential benefits of policy...
Essay:
LLM
Policy changes can have big impacts on...
I have my doubts about that policy because...
I think this policy change is a good idea...
I have my doubts about that policy because...
Firstly, the proposed policy considers...
I have my doubts about that policy because...
LLM Detector
Inconsistent Detection Performance

Figure 2: Even a task-oriented constraint in the instruction can cause inconsistent detection performance on the LLM-generated texts.

and human-written essays h_j that are part of the essay dataset created by Koike et al. (2024). Then, we instruct LLMs to generate essays based on the problem statements s_j. We elaborate on the details of our test set in §3.1.

2.2 Investigating the Impact of Task-Oriented Constraints on LLM Detection

In our work, we investigate the variation in the detection performance of texts generated by instruction with different constraints.

Constrained Generation We instruct an LLM to generate an essay and obtain a set of essays for each different constraint included in the instruction. Let I_i an instruction including the task-oriented constraint c_i, we instruct an LLM to generate an essay e_{ij} based on an essay problem statement s_j,

$$e_{ij} = LLM\left(I_i, s_j\right). \tag{1}$$

To facilitate our study, we manually[2] create the constraints c_i on essay writing based on each factor of essay quality, defined by Ke and Ng (2019). Table 1 lists the factors and our constraints. $I_i\,(i = 1)$ is the original instruction, not including any constraints:

```
Given the following problem statement,
please write an essay in {n} words.
Problem statement:
{problem_statement}
Essay:
```

[2]We create our task-oriented constraints as simply as possible to explain the factor, following the "Start Simple" philosophy in prompt engineering (Guide, 2024).

where `{n}` is the number of words in a human-written essay h_j paired with an essay problem statement s_j and `{problem_statement}` denotes the essay problem statement s_j.

$I_i\,(2 \leq i \leq 12)$ is the original instruction with an added constraint from Table 1:

```
Given the following problem statement,
please write an essay in {n} words.
{constraint_i}
Problem statement:
{problem_statement}
Essay:
```

where `{n}` and `{problem_statement}` are the same as the above and `{constraint_i}` denotes the constraint c_i in Table 1. For instance, "Your essay must be logically organized."

Impact Evaluation As depicted in Figure 2, we leverage LLMs to generate essays via instructions without and with each different constraint c_i and thus obtain the original instruction dataset and multiple constraint-based datasets $DS_i = \{(h_j, e_{ij})\}_{j=1}^{N}$. To quantify the impact of the constraint on LLM detection, let p_i be the F1-score detection performance on DS_i, we calculate the SD of the detection performance on the multiple datasets,

$$\sigma = \sqrt{\frac{\sum_{i=1}^{12}\left(p_i - \mu\right)^2}{12}}. \tag{2}$$

Here, μ is the average of the detection performances $\{p_i\}_{i=1}^{12}$. To validate the impact of such constraints on LLM detection, we use two randomnesses as baseline text variation types: `Multiple` and Paraphrase. Finally, we compare the SD of the detection performance in our Constraint setting of adding a constraint to instruction and Multiple and Paraphrase. We delve into the Multiple and Paraphrase settings in §3.1.

3 Experiments and Results

Our experiment investigates the answer to the following question: *Can current detectors consistently capture LLM-generated text variations caused by even task-oriented constraints in the instruction?*

3.1 Experimental Setup

Essay Generation Models We employ ChatGPT (gpt-3.5-turbo-0613) and GPT-4 (gpt-4-0613), which are commonly used LLMs, as our essay generation model. Additional configuration details of the essay generator models are in Appendix A.

Evaluation Metric and Dataset In our experiment, as described below, all LLM-generated-text detectors output a binary label for an input text. Thus, our evaluation metric for LLM detectors is the F1-score on LLM-generated texts, which is a common evaluation metric in binary classification tasks. As our evaluation dataset, we employ pairs of essay problem statements and human-written essays from the essay dataset created by Koike et al. (2024). We also prepare a set of LLM-generated essays based on the same essay problem statements. Finally, we evaluate LLM detectors on a mixture of 500 human-written and 500 LLM-generated essays from the problem statements by each LLM.

LLM-generated Text Detectors To verify the stability of existing representative detectors[3], we target the HC3 ChatGPT detector[4] and the ArguGPT[5] as supervised classifiers, and the in-context learning (ICL) approach[6] of Koike et al. (2024). The HC3 detector is a RoBERTa-base detector fine-tuned with the Human ChatGPT Comparison Corpus (HC3) dataset for detecting ChatGPT-generated texts covering diverse domains (Guo et al., 2023). The ArguGPT is a RoBERTa-large detector fine-tuned for catching LLM-generated argumentative essays, including various domains such as homework exercises, TOEFL, and GRE writing tasks (Liu et al., 2023b). Following the setting of Koike et al. (2024), in the ICL approach, we leverage ChatGPT (gpt-3.5-turbo-0613) with 5 ChatGPT-generated and 5 human-written essays as in-context examples from their training set for each essay to be detected. Further configuration details of the in-context learning approach are in Appendix A.

Text Variation Type In our experiment, to verify the consequent impact of a task-oriented constraint on LLM detection, we have two text variation types as the baseline: Multiple and Paraphrase, which can influence the stability of detection performance.

- **Multiple:** As depicted in Figure 1, we instruct LLMs to generate 12 texts from each original instruction to form 12 datasets. To generate multiple texts from an instruction, we utilize an argument to control the number of outputs in the OpenAI Chat Completion API[7]. The SD of detection performance is calculated on the 12 datasets.

- **Paraphrase:** As shown in Figure 1, we obtain 12 datasets by generating a text from each of 12 different paraphrases of the original instruction. The SD of detection performance is computed on the 12 datasets. To paraphrase the original instruction, we employ ChatGPT (gpt-3.5-0613). Specifically, we paraphrase the beginning of the original instruction thus, the instruction in this setting is as follows,

```
{paraphrased_instruction}
Problem statement:
{problem_statement}
Essay:
```

  where {paraphrased_instruction} is a paraphrase of the beginning of the original instruction, which is "Given the following problem statement, please write an essay in {n} words.". For instance, {paraphrased_instruction} can be "I kindly request you to compose an essay that adheres to the given problem statement, ensuring that it contains {n} words.". The examples of the paraphrases are in Appendix A.

[3]The logit information of our essay generators (ChatGPT and GPT-4) is not publicly available, thus our study does not cover statistical outlier detectors.

[4]https://huggingface.co/Hello-SimpleAI/chatgpt-detector-roberta

[5]https://huggingface.co/SJTU-CL/RoBERTa-large-ArguGPT

[6]https://github.com/ryuryukke/OUTFOX

[7]https://platform.openai.com/docs/api-reference/chat/create#chat-create-n

Essay Generator	Variation Type	LLM Detector		
		HC3	ArguGPT	ICL
ChatGPT	Multiple	1.02	0.30	0.48
	Paraphrase	4.07	0.84	0.58
	Constraint (Ours)	**12.76**	**6.69**	**1.15**
	The *deviation* for each factor			
	Grammar	5.43	2.43	0.13
	Usage	34.78	19.77	1.73
	Mechanics	2.23	1.33	0.98
	Style	15.88	5.67	0.83
	Relevance	5.13	2.53	0.23
	Organized	6.23	2.83	1.03
	Development	3.33	3.33	1.98
	Cohesion	11.23	3.43	0.63
	Coherence	7.03	2.83	0.03
	Thesis Clarity	4.03	2.43	2.08
	Persuasive	0.53	1.63	0.23
GPT-4	Multiple	1.09	1.14	0.68
	Paraphrase	3.42	2.43	0.69
	Constraint (Ours)	**4.13**	**14.38**	**1.26**
	The *deviation* for each factor			
	Grammar	2.26	8.11	0.85
	Usage	8.34	34.39	1.75
	Mechanics	2.96	8.01	0.45
	Style	7.54	24.39	0.55
	Relevance	1.96	7.01	0.15
	Organized	4.26	7.01	0.05
	Development	0.44	9.61	0.35
	Cohesion	3.56	4.81	0.05
	Coherence	1.44	4.89	1.05
	Thesis Clarity	0.26	6.21	1.05
	Persuasive	0.74	4.21	3.25

Table 2: A comparison of the SD of detection performance on essays generated by ChatGPT and GPT-4 on three variation types: `Multiple`, `Paraphrase`, and `Constraint`. It includes the *deviation* of detection performance for each factor in our `Constraint` setting.

3.2 Results

Table 2 presents the comparison of the SD of detection performance in the three text variation settings: `Multiple`, `Paraphrase`, and `Constraint`. In addition, it shows the deviation of detection performance for each factor in our `Constraint` setting. The LLM detectors include the HC3 detector, ArguGPT, and the ICL approach. Throughout all configurations of the generator and the detector, the SD on texts via instruction with the constraints is significantly larger than the two baseline variation types, reaching up to an SD of 14.4 F1-score for ArguGPT. These results imply that even a task-oriented constraint in instruction has a more significant effect on the detection performance of current detectors than the effect of generating texts multiple times and paraphrasing the instruction. We also observe an overall trend where the constraints make the detection more challenging: there is a decrease in detection performance in most constraints with up to a 40.3 drop in F1-score. We provide the detection performance itself in Appendix B.

Especially in the HC3 detector and ArguGPT, we can observe that the SD of detection performance is relatively large. This may be partially because the two detectors are trained with benchmarking datasets created without considering a variety of instructions and are prone to the difference of constraint in instruction. On the other hand, in the ICL approach, the effect of the constraint in instruction is relatively small. We could assume that the ICL approach might inherently consider a wide variety of expressions as in-context examples for detection, alleviating the effect of the constraint.

Looking into the impact of constraint for each factor, throughout all settings of the generator and the detector, the deviation of detection performance is notably larger in the factors of “Usage” and “Style”. As shown in Table 1, since both constraints on the two factors explicitly instruct to change the lexical distribution of the output text, this result aligns with our expectation. In our pilot study, we calculate the average of distinct-n $(= 1, 2, 3)$[8] (Zhao et al., 2017; Li et al., 2016) between two output texts from instructions without and with constraints for each factor through our test set. As a result, the top two factors with distinct-n values are “Usage” and “Style”. This implies that the constraints of “Usage” and “Style” in instruction may cause relatively large differences in the expression of the output texts, leading to such a large impact on the detection performance.

4 High Instruction-Following Ability Leads to Inconsistent Detection

Our experiments demonstrate that even task-oriented constraints in instruction induce notably inconsistent detection performance of the generated texts. In this section, to further examine how the constraint causes such an effect, we verify our hypothesis: *The high instruction-following ability of LLMs as a generator causes the large impact of constraints in the instruction on LLM detection.*

[8]The distinct-n is a metric for expression diversity in multiple texts, calculating the ratio of unique n-grams in the total word count. We apply the distinct-n to the setting of two texts to measure the difference in expression between them.

Essay Generator	Text Variation Type	LLM Detector			Instruction-Following Score (%)
		HC3	ArguGPT	ICL	
ChatGPT	Multiple	1.02	0.30	0.48	**87.1**
	Paraphrase	4.07	0.84	0.58	
	Constraint	**12.76**	**6.69**	**1.15**	
GPT-4	Multiple	1.09	1.14	0.68	
	Paraphrase	3.42	2.43	0.69	
	Constraint	**4.13**	**14.38**	**1.26**	
Davinci-002	Multiple	1.07	0.15	0.78	49.3
	Paraphrase	**4.14**	**0.51**	**1.51**	
	Constraint	1.44	0.32	1.17	

Table 3: A comparison of the SD of detection performance on essays generated by ChatGPT, GPT-4, and Davinci-002 on three variation types: `Multiple`, `Paraphrase`, and `Constraint` (as described in §3.1). The instruction-following score is a ratio of texts that follow a constraint in the generated texts by instruction with a constraint. It shows the overall instruction-following score across all constraints.

Essay Generator	Factor											
	Chs.	Chr.	Org.	Rel.	Sty.	Usg.	Dev.	TC.	Per.	Grm.	Mec.	Overall
ChatGPT/GPT-4	100	100	100	95.6	93.3	91.1	88.9	88.9	84.4	71.1	44.4	87.1

Table 4: The ratio of essays that follow each constraint in a mixture of essays generated by ChatGPT and GPT-4 with the instruction including each constraint. The scores are sorted in descending order.

4.1 Verification Setup

To verify our hypothesis, we investigate the relationship between the level of instruction-following ability of a generator and the extent of the impact of a constraint. In particular, we compare the impact of a constraint when using generators with low and high instruction-following ability as we measure.

Evaluating the Instruction-Following Ability To evaluate the instruction-following ability of an LLM, we calculate the *instruction-following score*: the ratio of texts that follow a constraint in generated texts by instruction with a constraint. Here, we call the texts generated by instruction without and with constraint as *plain* and *constrained* texts each. We prompt GPT-4[9] to classify whether a constrained text follows the constraint or not, compared with a plain text,

```
Please classify whether the following
texts follow the constraint.
Constraint: {constraint}
Text: {plain_text}
Answer (just Yes or No): No
Text: {constrained_text}
Answer (just Yes or No):
```

where `{constraint}` is a task-oriented constraint we create, for instance including "Your essay must be logically organized." and `{plain_text}` and `{constrained_text}` denotes a plain text and constrained text based on the same essay problem statement, respectively. To eliminate the randomness of the evaluation by GPT-4, we configure temperature and top_p parameters to be 0.

For 11 task-oriented constraints each, we sample 45 pairs of plain text and constrained text from our test set generated by an LLM. Finally, we compute the instruction-following score on 495 $(= 45 \times 11)$ texts generated by an LLM.

Comparing the Impacts of a Constraint between LLMs Besides ChatGPT and GPT-4, we explore the effect of a constraint when using an LLM with relatively low instruction-following ability as a generator. As such a generator model with low instruction-following ability, we select Davinci-002 (GPT-3), which is not explicitly optimized for instruction-following.[10] The parameter configuration of Davinci-002 is in Appendix A.

[9]GPT-4 as an evaluator has been recently reported to exhibit promising alignment with humans in evaluation across various downstream tasks (Liu et al., 2023a; Chiang and Lee, 2023).

[10]OpenAI declares that GPT base models, including Davinci-002, are not optimized for instruction-following: `http://tinyurl.com/gpt-base-model`.

Essay Problem Statement	`Should schools require a minimum grade requirement for students to participate in sports? Discuss the potential benefits and drawbacks of such a policy, taking into consideration the impact on students' academic performance, physical health, and personal interests. Use evidence and examples to support your argument.`
Plain Essay	`Education and sports are two integral parts of a student's life. However, the debatable question regarding the minimum grade for students to participate in sports has been a contentious topic of discussion. Supporters of the minimum grade requirement argue that it presents students with a target to work towards, thus promoting focus and dedication towards their academic performance. With this policy in place, students are motivated to perform well acadically because participation in sports depends on it. This, in turn, helps them develop better time management and prioritization skills. Moreover, this prerequisite can drive students to have greater self-discipline, which`
Constrained Essay	`Educational institutions universally grapple with the conundrum of whether to impose a minimum grade requirement for students wishing to partake in sports. While the underlying purpose of such a provision is to ensure academic excellence, it may inadvertently suppress the sporting interests of students, thus bringing their physical health into question. The most salient benefit of imposing a minimum grade requirement is undoubtedly the enhancement of academic performance. By establishing a tangible prerequisite for sports participation, it incentivizes students to strive for scholastic excellence. Additionally, this measure can circumvent the pervasiveness of overemphasis on athletics sometimes observed in schools, inadvertently`

Table 5: Examples of essays generated by GPT-4 via instruction without and with a constraint on "Usage". Both essays are cut off at the first 100 words. Red in the table are low-frequency words based on the Wikipedia dump.

4.2 Results

Overall Table 3 provides a comparison of the SD of detection performance on essays generated by ChatGPT, GPT-4, and Davinci-002, in the three variation settings: `Multiple`, `Paraphrase`, and `Constraint`. It also includes the instruction-following score of Davinci-002 essays and a mixture of essays generated by ChatGPT and GPT-4, each across all constraints[11].

We can observe that the effect of a constraint in the case of ChatGPT and GPT-4 is significantly large, but the effect in the case of Davinci-002 is quite small. Furthermore, the instruction-following score in a group of ChatGPT and GPT-4 across all factors is 87.1%, which is notably larger than 49.3% in Davinci-002.[12] These results imply that the high instruction-following ability of LLMs reinforces the effect of a constraint on LLM detection, supporting our hypothesis. We provide a discussion on the change in detection difficulty caused by the constraints, showing the detection performance itself in Appendix B.

[11]We group LLMs based on their instruction-following ability and treated ChatGPT and GPT-4 as a relatively high-performing group compared with Davinci-002.

[12]To confirm the validity of the evaluation by GPT-4, we use Amazon Mechanical Turk to crowdsource the human agreement rate with the evaluation. We get an 87% human agreement rate, ensuring the validity of the evaluation to some degree. The details of this validation are in Appendix C.

Details Table 4 shows the instruction-following score for each constraint in a group of ChatGPT and GPT-4. We sort the scores in descending order. Here, the top three constraints with the deviation in detection performance, averaged between the detectors and essay generators, are "Usage", "Style", and "Cohesion", while the bottom three constraints are "Persuasive", "Mechanics", and "Thesis Clarity". All top three constraints obtain over 90% of the instruction-following score and are ranked relatively higher in Table 4, while all bottom three constraints have less than 90% of the instruction-following score and are ranked relatively lower. This suggests that our hypothesis is reasonable to some extent, not only across all constraints but also for each constraint.

Table 5 showcases example essays generated by GPT-4 via instruction without and with a constraint on "Usage", which is "Your essay must utilize a professional-level vocabulary". Professional words tend to have low frequencies in a corpus. Thus, we identify low-frequency words[13] in each text and observe that the constrained text contains more low-frequency words than the plain text. It implies that the constrained text might follow the constraint.

[13]We leverage the Wikipedia dump extracted on April 23, 2023: `https://github.com/IlyaSemenov/wikipedia-word-frequency`. We define a word whose number of occurrences in the corpus is below the average number of occurrences of all words.

5 Related Work

LLM-Generated Text Detection Algorithms In this section, we briefly outline current LLM-generated text detection algorithms. The detection algorithms are mainly divided into three groups: watermarking, statistical outlier approach, and supervised classifiers. The watermarking embeds token-level markers into output texts that are hard to recognize by humans and utilizes the ratio of the markers in a text for detection (Kirchenbauer et al., 2023). Our work only focuses on non-watermarked LLMs that are mainly for our daily use. The statistical outlier approaches capture a probability deviation of a text from the predicted distribution of LLMs. These include token log probabilities (Solaiman et al., 2019), entropy (Lavergne et al., 2008), perplexity (Beresneva, 2016), and negative curvature of perturbed text probabilities (Mitchell et al., 2023). The supervised classifiers are basically neural-based models trained to distinguish human-written and LLM-generated texts with labeled datasets (Uchendu et al., 2020; Rodriguez et al., 2022; Guo et al., 2023). In addition to the above three groups, there has recently been a new direction: leveraging in-context learning for LLM-generated text detection (Koike et al., 2024). They utilize in-context learning of LLMs with retrieved few-shot human-written and LLM-generated examples, showing promising detection performance.

The Sensitivity of Prompting Prompting is a way of steering LLMs to generate texts via textual instruction without updating the model's parameters (Liu et al., 2021). Although prompting has shown promising performance on various tasks (Kamalloo et al., 2023; Zhang et al., 2023b,a), the quality of output text is very sensitive to how the instruction is expressed (Jiang et al., 2020). For instance, in machine translation, Zhang et al. (2023a) observed that a small difference in generation instruction causes a significant difference in BLEU score of 23.1 points.

Regardless of the substantial effect of instruction patterns on text quality, most studies on LLM detection overlook the subsequent effect of instruction patterns in text generation on the detection performance. Our work investigates the impact of instruction patterns, encompassing not only the surface patterns but also the difference of task-oriented constraints in the instruction, on LLM detection.

Benchmarking datasets for LLM Detection Many studies recently have established benchmarking datasets to identify LLM-generated texts. As representing examples, Guo et al. (2023) targets question answering and builds the Human ChatGPT Comparison Corpus (HC3) dataset for identifying ChatGPT-generated texts on diverse domains. Liu et al. (2023b) focuses on argumentative essay writing and creates a corpus consisting of about 4,000 pairs of human-written and LLM-generated essays. Highlighting how LLM-generated texts in such benchmarking datasets are generated, most studies make LLMs generate texts with one fixed instruction pattern. For instance, Liu et al. (2023b) targets on one instruction pattern: "`{essay_topic}` Do you agree or disagree? Use specific reasons and examples to support your answer. Write an essay of roughly `{n}` words.".

Considering the above sensitivity of prompting, there is a gap between the instruction pattern for generation and LLM detection on the generated texts. Our work bridges this gap by quantifying the effect of the difference in instructions on detection performance and showing a significant impact of the difference of task-oriented constraints in instructions.

6 Conclusion

This study investigates how much impact even task-oriented constraints in instruction can have on the current detector's performance to the generated texts. Our experiments in the domain of student essay writing demonstrate that even task-oriented constraints in instruction have a more significant effect on the detection performance than the effect of sampling texts and paraphrasing the instruction. Furthermore, there is an overall trend where the constraints can make LLM detection more challenging than without them. Our analysis suggests that the high instruction-following ability of an LLM as a generator leads to a noteworthy effect of the constraint.

Taking into account the remarkable speed of recent LLM development, the instruction-following ability of LLMs would be much better, amplifying the effects of the constraint. Therefore, in an era of evolving LLMs, our finding more strongly calls for further development of robust LLM detectors against such distribution shifts caused by a constraint in instruction.

Limitations

Our work shows that even task-oriented constraints in generation instruction cause existing detectors to have a large variance in detection performance. We focus on student essay writing because 1) There is a recognized demand for LLM detection against academic dishonesty (OpenAI, 2023a) with less discussion of such demand in other domains, 2) Due to the nature of being graded, the student essay domain has more established criteria we can refer to create the constraints than other domains (e.g., scientific writing and story generation). Establishing such criteria for other domains could be another line of research, and constraints can vary from the criteria. Thus, we encourage the research community to further investigate the impact of constraints in other generation tasks on LLM detection.

Ethical Considerations

Our goal in this paper is not to propose a method to deceive detectors. Instead, we aim to improve the robustness of LLM-generated text detection and raise awareness in the research community that how the generation instruction is written has a large impact on detection performance. Furthermore, we provoke the research community to develop new robust LLM detectors against distribution shifts caused by constraints in generation instruction.

Acknowledgements

These research results were obtained from the commissioned research (No.22501) by National Institute of Information and Communications Technology (NICT), Japan. In addition, this work was supported by JST SPRING, Japan Grant Number JPMJSP2106. The authors are greatly thankful to Mengsay Loem, Marco Cognetta, and Youmi Ma for their detailed feedback on this paper.

References

Daria Beresneva. 2016. Computer-generated text detection using machine learning: A systematic review. In *21st International Conference on Applications of Natural Language to Information Systems, NLDB*, pages 421–426. Springer.

Cheng-Han Chiang and Hung-yi Lee. 2023. Can large language models be an alternative to human evaluations? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15607–15631, Toronto, Canada. Association for Computational Linguistics.

Yutao Feng, Jipeng Qiang, Yun Li, Yunhao Yuan, and Yi Zhu. 2023. Sentence simplification via large language models.

Prompt Engineering Guide. 2024. General tips for designing prompts. Accessed: 2024-02-10.

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.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know?

Ehsan Kamalloo, Nouha Dziri, Charles L. A. Clarke, and Davood Rafiei. 2023. Evaluating open-domain question answering in the era of large language models.

Zixuan Ke and Vincent Ng. 2019. Automated Essay Scoring: A Survey of the State of the Art. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 6300–6308. ijcai.org.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A Watermark for Large Language Models.

Ryuto Koike, Masahiro Kaneko, and Naoaki Okazaki. 2024. OUTFOX: LLM-Generated Essay Detection Through In-Context Learning with Adversarially Generated Examples. In *Proceedings of the 38th AAAI Conference on Artificial Intelligence*, Vancouver, Canada.

Thomas Lavergne, Tanguy Urvoy, and François Yvon. 2008. Detecting Fake Content with Relative Entropy Scoring. In *Proceedings of the ECAI'08 Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse*, CEUR Workshop Proceedings.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Yafu Li, Qintong Li, Leyang Cui, Wei Bi, Longyue Wang, Linyi Yang, Shuming Shi, and Yue Zhang. 2023. Deepfake text detection in the wild.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023a. G-eval: Nlg evaluation using gpt-4 with better human alignment.

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.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. 2023. DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature.

Sandra Mitrović, Davide Andreoletti, and Omran Ayoub. 2023. Chatgpt or human? detect and explain. explaining decisions of machine learning model for detecting short chatgpt-generated text.

OpenAI. 2023a. How can educators respond to students presenting ai-generated content as their own? Accessed: 2023-11-10.

OpenAI. 2023b. Introducing ChatGPT. Accessed on 2023-05-10.

OpenAI. 2023c. Prompt engineering guide. Accessed: 2023-10-10.

Juan Diego Rodriguez, Todd Hay, David Gros, Zain Shamsi, and Ravi Srinivasan. 2022. Cross-domain detection of GPT-2-generated technical text. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1213–1233, Seattle, United States. Association for Computational Linguistics.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, Miles McCain, Alex Newhouse, Jason Blazakis, Kris McGuffie, and Jasmine Wang. 2019. Release Strategies and the Social Impacts of Language Models.

Jinyan Su, Terry Yue Zhuo, Di Wang, and Preslav Nakov. 2023. Detectllm: Leveraging log rank information for zero-shot detection of machine-generated text.

Ruixiang Tang, Yu-Neng Chuang, and Xia Hu. 2023. The science of detecting llm-generated texts.

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

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, Online. Association for Computational Linguistics.

Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Derek F. Wong, and Lidia S. Chao. 2023. A survey on llm-generated text detection: Necessity, methods, and future directions.

Yu Yu, Abdul Rafae Khan, and Jia Xu. 2022. Measuring robustness for NLP. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3908–3916, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Biao Zhang, Barry Haddow, and Alexandra Birch. 2023a. Prompting large language model for machine translation: A case study.

Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B. Hashimoto. 2023b. Benchmarking large language models for news summarization.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 654–664, Vancouver, Canada. Association for Computational Linguistics.

A Configuration Details

Parameter Configurations of Generators For the essay generator models, we set the `temperature` parameter of ChatGPT to be 1.3, GPT-4 to be 1.0, and Davinci-002 to be 0.6. For the paraphraser to rephrase the instruction in the `Paraprhase` setting, we set the `temperature` parameter of ChatGPT to be 1.3.

Details of the ICL Approach Following the setting of Koike et al. (2024), we leverage ChatGPT (gpt-3.5-turbo-0613) as a detector of the ICL approach. To eliminate the randomness of the detection, we configure `temperature` and `top_p` parameters of ChatGPT to be 0. As a dataset for retrieving examples for the ICL approach, we employ the training set[14] of Koike et al. (2024), containing 14,400 triplets of essay problem statements, human-written essays[15], and ChatGPT-generated essays. Regardless of the type of essay generators (ChatGPT, GPT-4, and Davinci-002), we retrieve ChatGPT-generated essays.

Examples of Paraphrased Instructions In the `paraphrase` setting, we employ ChatGPT to paraphrase the beginning of the original instruction,

[14] `https://github.com/ryuryukke/OUTFOX`

[15] Written by native English speaking 6th to 12th graders from the U.S.

Please compose a {n}-word essay based on the provided problem statement.
I kindly request you to compose an essay that adheres to the given problem statement, ensuring that it contains {n} words.
Could you kindly compose an essay containing {n} words based on the provided problem statement?
Please compose an essay of {n} words based on the given prompt.
Please compose an essay with a word count of {n}, based on the provided problem statement.
Please compose an essay consisting of {n} words based on the provided problem statement.
I kindly request you to compose an essay with {n} words, based on the subsequent problem statement.
I kindly request you to compose an {n}-word essay based on the aforementioned problem statement.
Please compose an essay of {n} words based on the provided problem statement.
I am requesting an essay to be written in {n} words using the provided problem statement.
Please compose an essay in which you discuss the given problem statement, utilizing {n} to express your thoughts.
I kindly request you to compose an essay consisting of {n} words, using the problem statement provided below.

Table 6: Examples of the paraphrased instructions in the `paraphrase` setting.

which is "Given the following problem statement, please write an essay in {n} words." Table 6 lists the paraphrased instructions.

B Do the Constraints Make Detection Easier or Harder?

Our study mainly focuses on the SD of detection performance to elucidate the behavior of current significant detectors against task-oriented constraints in generation instruction. This is because there is a common understanding that when building robust NLP systems, the performance of the system itself should be consistent, regardless of the scale of their performance (Yu et al., 2022). Similarly, an ideal LLM detector should also have a consistent detection performance against the effect of task-oriented constraints, regardless of whether the performance improves or degrades.

On the other hand, we also acknowledge the worth of discussing whether LLMs can generate texts easier or harder to detect via instruction with the constraints. Table 7 showcases the detection performances of LLMs, including ChatGPT, GPT-4, and Davinci-002, with and without each constraint (Plain) in the generation instruction. In most constraints, it is observed that the detection performance degrades (in gray parts) compared to the setting of Plain. We can also see up to a 40.3 F1-score drop in blue parts with the lowest detection performance. Finally, overall, there is a greater decrease in detection performance among ChatGPT and GPT-4 with relatively high instruction-following abilities compared to Davinci-002. It suggests that instruction-following on the constraints could lead to not only higher detection performance deviation but also **more challenging** detection.

C Validation of GPT-4 Evaluation

In §4.1, we leverage GPT-4 to evaluate the instruction-following ability of LLMs. Specifically, we instruct GPT-4 to decide whether the constrained text follows the constraint compared with the plain text. Here, the plain text and the constrained text are generated based on the same essay problem statement.

To ensure the validity of the GPT-4 evaluation, we utilize Amazon Mechanical Turk (AMT) to examine the ratio of the decisions made by GPT-4 that align with human consensus. Figure 3 shows the AMT interface we use for our human agreement test. Particularly, we show a constraint, an essay problem statement, the shuffled pair of plain and constrained texts based on the problem statement, and the GPT-4 decision to the AMT workers and ask if they agree with the decision. We perform the test on 495 $(= 45 \times 11)$ texts, which is a mixture of essays generated by ChatGPT and GPT-4 with the instruction including 11 constraints each. We set workers' qualifications where the HIT[16] approval rate is over 99 %, and the number of approved HITs is greater than 10,000. In our test, one worker is assigned per HIT, and workers are paid \$0.03 per HIT.

D Computational Budget

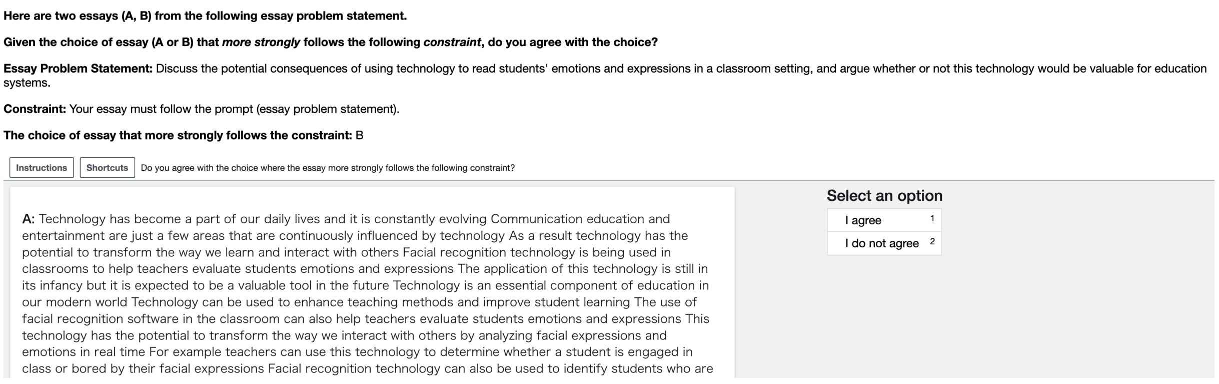
We run all the experiments on AI Bridging Cloud Infrastructure (ABCI)[17], Compute Node (V), whose CPUs are two Intel Xeon Gold 6148, and GPUs are four NVIDIA V100 SXM2. The total processing time is approximately 20 hours.

[16]A human intelligence task (HIT) in AMT workplace refers to one single task. In our case, the HIT would be to decide if they agree or not with one of the GPT-4 decisions.

[17]https://abci.ai/

Generator	**Detector**		**Factor**												
		Plain	**Grm.**	**Usg.**	**Mec.**	**Sty.**	**Rel.**	**Org.**	**Dev.**	**Chs.**	**Chr.**	**TC.**	**Per.**	**Avg.**	**Diff.**
ChatGPT	HC3	78.2	78.1	37.9	74.9	56.8	77.8	78.9	76.0	83.9	79.7	76.7	73.2	72.2	-6.03
	ArguGPT	96.4	96.2	74.0	95.1	88.1	96.3	96.6	97.1	97.2	96.6	96.2	95.4	93.5	-2.87
	ICL	94.3	94.2	95.8	93.1	94.9	94.3	95.1	92.1	94.7	94.1	92.0	94.3	94.1	-0.25
GPT-4	HC3	12.3	11.3	0.70	12.0	1.50	11.0	13.3	8.60	12.6	7.60	9.30	8.30	8.75	-3.55
	ArguGPT	84.0	83.4	40.9	83.3	50.9	82.3	82.3	84.9	80.1	70.4	81.5	79.5	74.5	-9.50
	ICL	92.2	92.6	93.5	91.3	92.3	91.6	91.7	92.1	91.7	92.8	90.7	88.5	91.7	-0.49
Davinci	HC3	87.2	90.1	87.7	87.9	89.2	85.4	89.8	88.4	87.9	89.6	86.4	87.0	88.1	0.93
	ArguGPT	97.8	97.6	97.3	97.8	97.6	97.2	97.5	97.4	97.7	97.7	96.7	97.2	97.4	-0.37
	ICL	87.9	85.8	88.8	87.4	85.1	86.5	89.0	86.7	87.9	88.0	87.8	88.1	87.4	-0.53

Table 7: A comparison of detection performance on essays generated via instructions with and without task-oriented constraints (Plain). The gray parts indicate a lower detection performance in the setting with the constraints than the setting of Plain. It depicts the lowest detection performance for each combination of generator and detector in the blue parts. Avg. indicates the mean detection performance in the setting with the constraints. Diff. implies the difference in detection performance between Avg. and Plain.



Here are two essays (A, B) from the following essay problem statement.

Given the choice of essay (A or B) that *more strongly* follows the following *constraint*, do you agree with the choice?

Essay Problem Statement: Discuss the potential consequences of using technology to read students' emotions and expressions in a classroom setting, and argue whether or not this technology would be valuable for education systems.

Constraint: Your essay must follow the prompt (essay problem statement).

The choice of essay that more strongly follows the constraint: B

Instructions
Shortcuts
Do you agree with the choice where the essay more strongly follows the following constraint?
Select an option
I agree 1
I do not agree 2

A: Technology has become a part of our daily lives and it is constantly evolving Communication education and entertainment are just a few areas that are continuously influenced by technology As a result technology has the potential to transform the way we learn and interact with others Facial recognition technology is being used in classrooms to help teachers evaluate students emotions and expressions The application of this technology is still in its infancy but it is expected to be a valuable tool in the future Technology is an essential component of education in our modern world Technology can be used to enhance teaching methods and improve student learning The use of facial recognition software in the classroom can also help teachers evaluate students emotions and expressions This technology has the potential to transform the way we interact with others by analyzing facial expressions and emotions in real time For example teachers can use this technology to determine whether a student is engaged in class or bored by their facial expressions Facial recognition technology can also be used to identify students who are

Figure 3: AMT interface for our human agreement test.