Plagiarism in the Age of Generative AI: Cheating Method Change and Learning Loss in an Intro to CS Course

Binglin Chen
University of Illinois Urbana-Champaign
Urbana, IL, USA
chen386@illinois.edu

Colleen M. Lewis
University of Illinois Urbana-Champaign
Urbana, IL, USA
colleenl@illinois.edu

Matthew West
University of Illinois Urbana-Champaign
Urbana, IL, USA
mwest@illinois.edu

Craig Zilles
University of Illinois Urbana-Champaign
Urbana, IL, USA
zilles@illinois.edu

ABSTRACT

Background: ChatGPT became widespread in early 2023 and enabled the broader public to use powerful generative AI, creating a new means for students to complete course assessments.
Purpose: In this paper, we explored the degree to which generative AI impacted the frequency and nature of cheating in a large introductory programming course. We also estimate the learning impact of students choosing to submit plagiarized work rather than their own work.
Methods: We identified a collection of markers that we believe are indicative of plagiarism in this course. We compare the estimated prevalence of cheating in the semesters before and during which ChatGPT became widely available. We use linear regression to estimate the impact of students' patterns of cheating on their final exam performance.
Findings: The patterns associated with these plagiarism markers suggest that the quantity of plagiarism increased with the advent of generative AI, and we see evidence of a shift from online plagiarism hubs (e.g., Chegg, CourseHero) to ChatGPT. In addition, we observe statistically significant learning losses proportional to the amount of presumed plagiarism, but there is no statistical difference on the proportionality between semesters.
Implications: Our findings suggest that unproctored exams become increasingly insecure and care needs to be taken to ensure the validity of summative assessments. More importantly, our results suggest that generative AI can be detrimental to students' learning. It seems necessary for educators to reduce the benefit of students using generative AI for counterproductive purposes.

CCS CONCEPTS

• **Social and professional topics** → **Computing education**; **K-12 education**.

KEYWORDS

plagiarism detection, cheating, generative AI, LLM, CS 1

L@S '24, July 18–20, 2024, Atlanta, GA, USA

ACM ISBN 979-8-4007-0633-2/24/07
https://doi.org/10.1145/3657604.3662046

ACM Reference Format:
Binglin Chen, Colleen M. Lewis, Matthew West, and Craig Zilles. 2024. Plagiarism in the Age of Generative AI: Cheating Method Change and Learning Loss in an Intro to CS Course. In *Proceedings of the Eleventh ACM Conference on Learning @ Scale (L@S '24), July 18–20, 2024, Atlanta, GA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3657604.3662046

1 INTRODUCTION

The educational benefits of practice are arguably the most well established finding from education, psychology, and cognitive science. Specifically, practice involves retrieval practice, spaced practice, and feedback (see [5, 6, 35] for reviews). Retrieval practice is when a task requires someone to recall knowledge from long-term memory. Whether procedural or declarative knowledge, this act of retrieval is beneficial for student learning [32]. Spaced practice is beneficial and involves engaging in retrieval practice separated by breaks [32]. There is an additional benefit of such practice sessions being separated by sleep [28, 38]. Retrieval practice is most effective when it is accompanied by feedback [8, 36], either immediately or delayed, and the feedback is more useful if it provides more information than whether an answer is correct or incorrect [39]. Based upon this wealth of research, it is clear that ample opportunities for retrieval practice and feedback, which for narrative simplicity we will refer to simply as "practice", will enhance students' learning.

Despite the educational benefits of practice, students often circumvent educators' efforts to encourage such practice. For example, some students engage in unauthorized collaboration or plagiarize solutions from online sources such as Chegg and CourseHero. Students engage in such unauthorized behavior for a wide range of reasons, including misunderstanding academic norms, poor time management, external (e.g., parental) pressure, and a lack of self control [14, 27, 42]. Whatever their motivation, we expect that skipping practice will lead to less learning.

The recent, wide availability of generative AI in the form of large language models, such as ChatGPT, introduces a new method of cheating that adds to existing methods. This is particularly relevant to an introductory programming course, because ChatGPT has been shown to perform well solving introductory programming problems [15, 34]. In addition to ChatGPT providing an *additional* method of cheating, we hypothesize that ChatGPT may expand *who* cheats and not just *how* students cheat, because it can be done individually and for free. That is, we expect that ChatGPT

may increase the rate of plagiarism above levels of plagiarism in previous semesters.

It may also be the case that the learning impact of cheating with ChatGPT is different from cheating using other methods. For example, other methods of cheating may require students to review possible solutions in ways that provides them with opportunities to practice reading code. Whereas ChatGPT can provide explanations of code and these may function as worked examples, for which there is strong evidence of benefits to learning [4, 37, 40].

Given the substantial benefits from practice [20] and making mistakes [26] and that cheating circumvents students' opportunities to access these benefits, it is important to characterize the impact of ChatGPT as a new method of cheating. Towards this goal, we explore the following research questions:

- RQ1: Did the *quantity* of plagiarism increase after the wide availability of ChatGPT?
- RQ2: Did *sources* of plagiarism on introductory programming assignments change with the wide availability of ChatGPT?
- RQ3: Does plagiarism on homework assignments predict learning loss (i.e., lower performance on the final exam)?
- RQ4: Does plagiarism after the wide availability of ChatGPT lead to greater learning loss than previously available plagiarism methods?

To address these questions, we analyze data from two semesters of a large-enrollment introductory programming course ($N = 983$ across the two semesters). To support the course's scale, all of its assessments (homework and exams) are computer-based [41], providing digital records of student submissions. We use this data to identify markers of suspected plagiarism in the student work and perform statistical analysis to reason about our research questions.

We find a modest statistically significant increase in plagiarism after the wide availability of ChatGPT (RQ1). Our results suggests that the primary source of plagiarism *has* shifted from plagiarism hubs to ChatGPT (RQ2). Furthermore, we observe that higher rates of observed plagiarism correspond to larger losses of learning (RQ3). Roughly, a 25% increase in the amount of observed plagiarism on programming questions predicts a 10% reduction in the final exam score. However, we did not find differential rates of learning loss between the two semesters (RQ4).

As such, the primary contribution of this work is to affirm that the negative learning impact of plagiarism persists into the era of generative AI. With similar impacts to other forms of plagiarism and increased availability, the overall impact of plagiarism on learning may increase if we maintain the status quo. These findings both reinforce the need to ensure that our summative assessment is conducted in a trustworthy manner and reaffirm the many ongoing efforts to harness the power of generative AI to engage and support learners to mitigate their perceived need to engage in plagiarism.

2 RELATED WORK

2.1 Fraud triangle and cheating

The fraud triangle framework, originally developed by Cressey to explain fraudulent financial behaviors [13], has been adapted for educational contexts to analyze plagiarism and cheating [2, 7, 11–13]. According to this framework, three conditions are necessary for students to engage in cheating: (1) *pressure*, which arises from students feeling compelled to cheat, often due to fear of not achieving desired grades because of personal, academic, or time management issues, (2) *opportunity*, which presents itself when cheating appears to be risk-free, easy, or hard to detect, and (3) *rationalization*, where students justify their cheating behavior as compatible with their moral standards, which could be influenced by peer norms.

2.2 Academic performance and cheating

Numerous survey-based studies have found negative correlations between self-reported academic performance and academic cheating [18, 19, 22, 24, 33]. However, few have directly investigated cheating behaviors and their correlation with academic outcomes. We highlight two studies that have undertaken this approach.

Pierce and Zilles analyzed submissions to programming assignments from 2,409 students in a data structures course over six semesters. By combining similarity metrics and manual inspections to identify plagiarism, they compared the academic outcomes of students who have plagiarized at least one assignment (cheaters) to students who have not plagiarized (non-cheaters). Their findings revealed that cheaters' final course grades were 0.24 letter grades lower than those of non-cheaters ($p = 0.019$). Furthermore, cheaters also underperformed in a subsequent systems programming course by 0.30 letter grades ($p < 0.001$) [30].

Palazzo et al. studied 428 MIT students' submissions to an online physics tutoring system in a mechanics course without random parameterization of questions. By monitoring the speed of submissions, they classified submissions as either original or copied from peers. They z-scored final exam score on analytical problems and regressed it against fraction of homework copied, and found a slope of -2.42 ± 0.23. This regression result suggested a significant negative correlation between copying and final exam performance on analytical problems, with a decrease of 2.42 standard deviations per 100% of answers copied [29].

2.3 Technological advancement and cheating

Technological advancements, notably the invention of the Internet, have significantly reduced the cost of information sharing, thereby potentially facilitating plagiarism and cheating by expanding opportunities, according to the fraud triangle framework [13]. While direct evidence comparing the prevalence of cheating before and after the Internet became widespread is scarce, various indirect pieces of evidence support this notion. As illustrative examples, we highlight two studies that observed increases in cheating and plagiarism during the shift to online instruction and exams amidst the COVID-19 pandemic, and one study that did not observe a notable shift in cheating due to the introduction of ChatGPT.

Lancaster and Cotarlan analyzed the increase in questions posted on Chegg across five STEM subjects before and after the shift to online instructions due to the COVID-19 pandemic [21]. They observed a 196.25% increase in the number of questions posted between April and August 2020 compared to the same period in 2019. Upon manual review, they noted that many questions likely originated from exams, aligning with a peak in April-May, coinciding with universities' final assessment periods.

Emerson and Smith investigated the impact of question searchability on the performance of intermediate accounting students in online quizzes [16]. They found that students performed significantly worse on an online quiz that prevented them from accessing external websites than one without this restriction. On the online quiz where students could access external websites, they performed significantly better on questions with easily searchable answers than those without.

Lee et al. investigated the impact of ChatGPT availability on cheating behaviors in US high schools, using anonymous surveys of students [23]. In contrast to the present paper, they did not find a notable increase in cheating or a clear shift in the mode of cheating towards the use of ChatGPT or other generative AI tools.

3 DATA COLLECTION AND PREPARATION

3.1 Course context

The data was collected from an introductory Python programming course for non-technical majors in a large R1 university in the United States during Fall 2022 ($N = 550$) and Spring 2023 ($N = 433$). The course was taught by the same instructor in both semesters. The course includes weekly homework assignments, unproctored quizzes, and exams taken in a proctored computer lab (see [43, 44]).

3.2 Data collection

With IRB approval, we collected data composed of all students' submissions to online programming questions. While other types of questions exist on homework, quizzes, and exams, we focus exclusively on programming questions because their large potential answer spaces is more conducive to detecting cheating through analysis of the submitted responses. For our analysis, we only looked at the first correct submission that a student made to each question.

3.3 Markers of plagiarism

In the Spring 2023 semester, we observed a number of students that completed their homework remarkably quickly relative to previous semesters (e.g., 17 seconds to read a multi-line programming question prompt and produce a 7 line program). A number of these students were accused of and admitted to cutting and pasting responses from ChatGPT to complete their homework, which was disallowed by course policy.

We identified features indicative of plagiarism in a two-stage process. In the first stage we sought to identify likely examples of plagiarism for manual inspection. We isolated potentially plagiarized, correct student code for each question based on their distance to other students' code. Specifically, the steps were (1) generating an abstract syntax tree (AST) from each student's code, (2) standardizing variable names within the AST, (3) converting the AST back into code, (4) calculating the distance between each pair of students' code by taking the string edit distance divided by the length of the longer piece of code in that pair, (5) computing a mean distance for each student's code relative to the rest of the class, and (6) flagging code that was two standard deviations away from the class mean based on the measure computed in step (5). These flagged solutions formed the set that we manually inspected to identify features of plagiarism.

Through manual review of the flagged solutions obtained above, we identified four binary features that are potentially indicative of plagiarism: advanced syntax, extra comment, extra print, and extra code. Answers that demonstrate each of these markers can be found in Table 1.

Advanced Syntax marker: The advanced syntax marker is present if there is any appearance of list/set/dictionary comprehensions, generator expressions, map, reduce, or lambda. These elements of Python syntax were not covered in the course and none of the programming questions on the exams would require use of these elements.

Extra Comment marker: The extra comment marker is present if there is any appearance of comments. Our manual inspection identified clear and accurate comments longer than the accompanying code, which we also observe in responses from ChatGPT to our programming prompts and solutions from online plagiarism hub.[1] As an introductory CS course for non-majors, the course neither emphasizes documentation of code nor penalizes students for a lack of documentation. Additionally, even the most complicated programming question in the course does not require more than ten lines of code to solve.

Extra Print marker: The extra print marker is present if there is any print statement in a question that does not require print to receive full credit. Many questions ask for return values or parameters to be modified without printing.

Extra Code marker: The extra code marker is present if there is any code that is outside the scope of the function that the question is asking the students to write, except import. Our manual inspection found that such code is typically test code that calls the function students were asked to write. Even though students are encouraged to test their code before submitting, the course only grades the specified function. Therefore we do not expect students to include test code in their submissions, and plagiarized responses often include a test call.

We created a script to detect the presence or absence of each marker in a correct submitted solution. Since a solution submitted to a programming question may include multiple markers, we use the marker *Any* to indicate that a submitted solution includes one or more of the markers, which is essentially an indicator of whether a submitted solution is considered plagiarized.

3.4 Plagiarism Ratio

We calculate a *plagiarism ratio* for each marker and each student as the proportion of submitted solutions with the marker divided by the total number of programming questions.[2] This *plagiarism ratio* is calculated over a set of assignments, such as all homework between two exams.

[1] Interestingly, we observe that students sometimes first submit an answer with comments, then they resubmit the same code removing all the comments.

[2] We only looked at the first correct submission that students made to any question in any assessment, therefore multiple correct submissions to the same question were not counted multiple times.

Table 1: Example code for each marker of plagiarism, along with the criterion and rationale. For all of the example solutions in the table, we highlighted lines of the code that contain the feature each marker detects. The question prompt was: "Define a function below called `decrease_elements_by_x`, which takes two arguments — a list of numbers and a single positive number (you might want to call it `x`). Complete the function such that it returns a copy of the original list where every value is decreased by the second argument. For example, given the inputs `[1,2,5]` and `2`, your function should return `[-1,0,3]`."

Marker name and example code with the marker	Criterion	Rationale

No marker
`def decrease_elements_by_x(nums, x):`
`  new_nums = []`
`  for n in nums:`
`  new_nums.append(n-x)`
`  return new_nums`
	Has none of the four markers below	This is what we would like students to be able to write independently.

Advanced syntax marker
`def decrease_elements_by_x(numbers, x):`
`  return [num - x for num in numbers]`
	Appearance of comprehensions, generator expression, map, reduce, or lambda	The course neither explicitly covers these features in class nor has any questions specifically designed to test students' mastery of these features, therefore we do not expect students to employ any of these features in programming questions.

Extra comment marker
`def decrease_elements_by_x(nums, x):`
`  #an empty list`
`  new_nums = []`
`  #traverse the input list`
`  for n in nums:`
`  #add x to every element and append to new list`
`  new_nums.append(n-x)`
`  #return the list`
`  return new_nums`
	Appearance of comments	The course neither emphasizes documentation of code nor penalizes students for poor documentation, additionally even the most complicated programming question in the course does not require more than ten lines of code to solve, therefore we do not expect students to comment their code.

Extra print marker
`def decrease_elements_by_x(nums, x):`
`  new_nums = []`
`  for n in nums:`
`  new_nums.append(n-x)`
`  return new_nums`
`  numbers = [1, 2, 5]`
`  new_list = decrease_elements_by_x(numbers, 2)`
`  print(new_list)`
	Appearance of print statement for questions that do not ask for printing	The course explicitly states in all programming questions whether functions should return, print, or modify arguments without returning anything, and the course emphasize the distinction between return and print during classes, therefore we expect students to briefly confuse these concepts, not regularly, and manual inspections suggest that these prints are often associated with testing code.

Extra code marker
`def decrease_elements_by_x(nums, x):`
`  new_nums = []`
`  for n in nums:`
`  new_nums.append(n-x)`
`  return new_nums`
`numbers = [-1, 0, 3]`
`new_list = decrease_elements_by_x(numbers, 2)`
	Appearance of code at the same indentation level of the function that students were asked to write (except import)	Manual inspection suggests such code is typically test code. Even though students are encouraged to test their code before submitting, the course only grades the specified function. We, therefore, do not expect students to frequently include testing code in their submitted answers.

Any marker	Has any of the four markers above	This marker is essentially an aggregated indicator of plagiarism.

3.5 These markers as a proxy for plagiarism

The presence or absence of these markers does not provide certainty about whether a particular submission was plagiarized. In fact, we suspect that our data set includes many plagiarized submissions that do not include these markers. Nevertheless, we believe these markers provide a reasonable estimate of the amount of plagiarism in which a student has engaged.

We estimate false positive rate of around 10% and false negative rate of around 15%. Our estimate of the false positive rate comes from the the results in Appendix A, where we observe that plagiarism ratios on homework are roughly ten times higher than they are on the computer-based exams where students do not have access to plagiarism hubs or ChatGPT. Because students' submissions in exams are likely their own work, this ten times ratio suggests that about 10% of answers on homework with the markers could be students' original work. Our estimate of the false negative rate comes from the results in Section 3.6, where we observe that known-plagiarized samples from plagiarism hubs and ChatGPT are both detected about 85% of the time using our markers. This indicates that about 15% of plagiarized answers would be missed by our technique.

3.6 Manual collection of plagiarized solutions

In an attempt to identify the provenance of these plagiarized solutions, we created a dataset of solutions both produced by ChatGPT queries and available from popular plagiarism hubs. We selected 23 programming questions from homework where at least 10% of the solutions to them were tagged with one or more markers. For each programming question, we generated 10 solutions using ChatGPT 3.5 by using the question prompt verbatim. For each programming question, we also searched for solutions on five popular plagiarism hubs: Chegg, CourseHero, Quizlet, Brainly, and Numerade.

3.7 Exam design and security

The primary summative evaluation in the course occurs on four proctored exams that occur in the 6th, 9th, 12th, and 15th (finals) week of the semester (see Figure 1). These computerized exams are conducted in a proctored environment on university computers with isolated file systems and restricted access to the internet (i.e., no access to ChatGPT) [43, 44]. As such, we have high confidence that there is minimal cheating on these assessments.

As a further cheating mitigation effort, these exams use pools of questions [10]. Each pool of programming questions attempts to assess a specific learning objective at a specific difficulty. However, despite efforts to balance question difficulty in a pool of questions, slight variations are unavoidable. The exams in the two semesters studied were almost identical in structure and pool construction, but question pools did differ slightly between semesters.

In the week preceding each exam, the course conducts an unproctored quiz that has a structure that is almost identical to the proctored exam that follows it. These unproctored quizzes contribute to the final grade, but significantly less than the proctored ones. These unproctored quizzes are timed, but students complete them on their own machines at a time of their choosing. In spite of students agreeing to an honor statement at the beginning of the quiz, we believe (and the data supports) that some students cheat on these quizzes.

3.8 Computing exam performance

Because students receive different exam questions from question pools, students' raw exam scores may not be directly comparable. Thus our analysis does not use students' raw exam scores directly.

To obtain a more reliable measure of student exam performance, we calculate a predicted score for the student on each exam. To do this, we first fit the following three parameter logistic model (3PL) according to item response theory (IRT):

$$p_{ijk} = c_i + \frac{1 - c_i}{1 + e^{-a_i(\theta_{jk} - b_i)}}. \quad (1)$$

p_{ijk} is the observed score of student j on question i in exam k (every unproctored and proctored exam is treated as a unique exam). p_{ijk} is a real number between 0 and 1, and $a_i, b_i, c_i, \theta_{jk}$ are the coefficients that we want to estimate, which have standard interpretations in IRT:

- a_i: discrimination of question i,
- b_i: difficulty of question i,
- c_i: probability of a successful guess on question i,
- θ_{jk}: ability of student j on exam k.

IRT assumes binary scoring. However, the exams include questions with partial credit. We adapted the optimization process by minimizing cross entropy loss instead of maximizing log-likelihood. We chose to adopt cross entropy loss rather than rounding partial credits to fit the standard 3PL model because cross entropy loss is a natural extension for situations allowing partial credit without the information loss incurred by rounding. As there is no unique minimum for 3PL models,[3] we follow the common practice of bounding $a_i \in [0, 2]$, $b_i \in [-3, 3]$, $c_i \in [0, 1]$, and $\theta_{jk} \in [-3, 3]$ in the optimization process [3]. We have shared our implementation of this optimization process on GitHub.[4]

After the above model was fit, we used the model to predict each student's score on every question that appeared on an exam. We then computed a predicted score for each question pool based on the questions in the pool by taking the mean. The predicted scores of question pools were aggregated to produce a predicted score for each exam. Since each exam in two semesters have different question pools, we predict students' performance on the exam from both semesters. We use the mean of these as the final predicted score for the student on the exam.

4 ANALYSIS AND RESULTS

4.1 More cheating post ChatGPT release (RQ1)

We see three indicators of modestly higher levels of plagiarism with the advent of ChatGPT. First, we see larger differences between the scores on the unproctored quizzes than on the proctored exams. Figure 2 plots how much better students perform (using the predicted performance discussed in Section 3.8) on the unproctored assessment relative to the proctored assessment that follows

[3] Multiplying all a_i by a non-zero constant and then dividing all θ_{jk} and b_i by the same constant results in the same loss.

[4] https://github.com/chen386/generative-ai-plagiarism-study

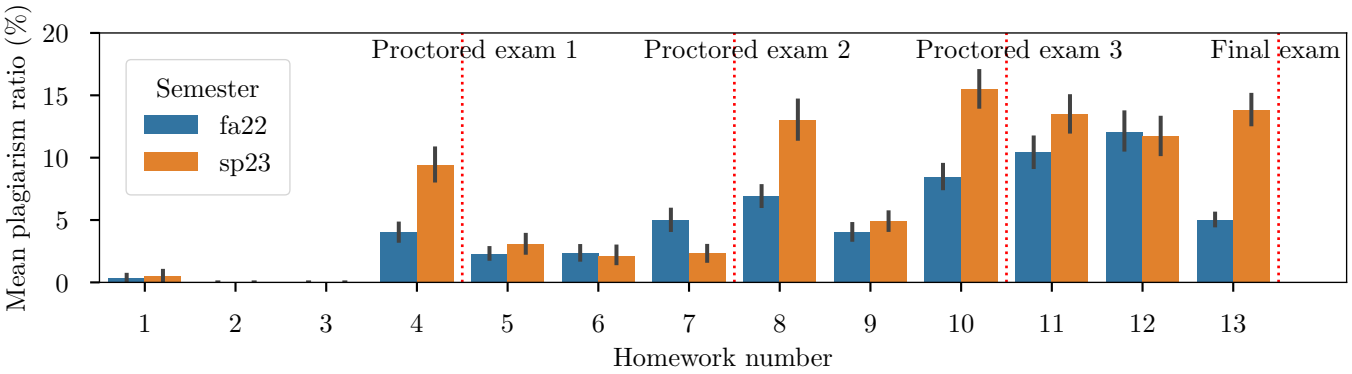

Figure 1: Mean plagiarism ratio on each homework assignment in each semester. The vertical red dotted lines indicate when each proctored exam takes place. The zero bars for homework 2 and 3 are due to the absence of any programming questions on those homework. The detail of how plagiarism ratio is computed is described in Section 3.3. The error bars show 95% confidence intervals of the means.

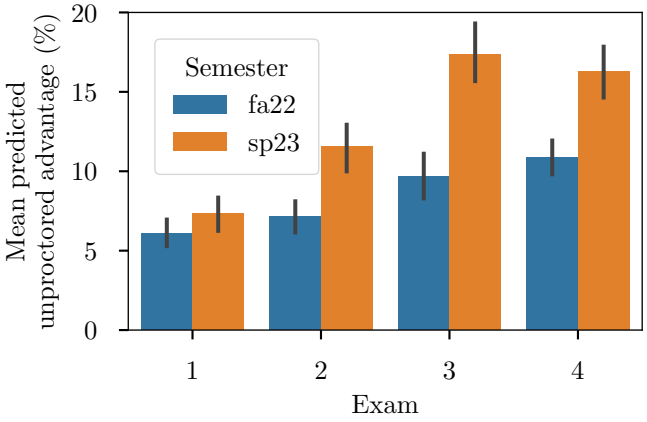

Figure 2: Mean predicted advantage of unproctored quizzes over proctored exams for each quiz/exam pair. We use this as a measure of plagiarism. The plot shows the predicted performance as per Section 3.8, but we see similar trends in the raw data. The error bars show 95% confidence intervals of the means.

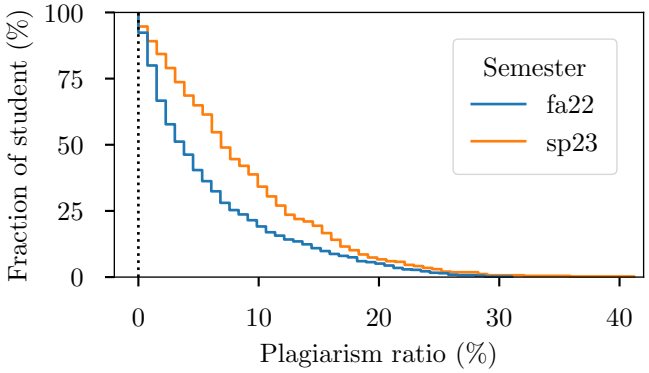

Figure 3: The empirical complementary cumulative distribution of students with respect to plagiarism ratio on homework. Each point along the curve indicates the amount of students that have a plagiarism ratio greater than or equal to the plagiarism ratio at that point.

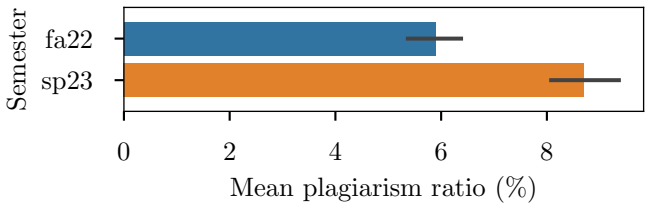

Figure 4: Average plagiarism ratios on homework.

it. This "unproctored advantage" is larger by a statistically significant amount ($p < 0.001$) in the Spring 2023 semester after the popularization of ChatGPT.

Second, a larger fraction of the students appear to be plagiarizing in Spring 2023. Figure 3 plots a complementary cumulative distribution that shows what fraction of students have a plagiarism ratio of at least a given fraction. For example, approximately 20% of the Fall 2022 students have plagiarism markers in at least 10% of their submitted homework programming question answers. As can be seen in the figure, the Spring 2023 line is consistently above the Fall 2022 line, indicating that, at every level of observed plagiarism, a larger fraction of the class was incriminated in the semester after the popularization of ChatGPT.

Third, we see a larger overall number of plagiarized submissions to homework programming questions in Spring 2023. Figure 4 shows the average plagiarism ratios across the two semesters.

While these measures suggest an increase in plagiarism, the increase seems to be relative modest with an effect size of about 0.43 standard deviations, as shown in Figure 4.

4.2 Change in plagiarism sources (RQ2)

In this section, we demonstrate that the distribution of plagiarism markers changes between the two semesters. This change suggests that the main effect of ChatGPT's availability has been that the

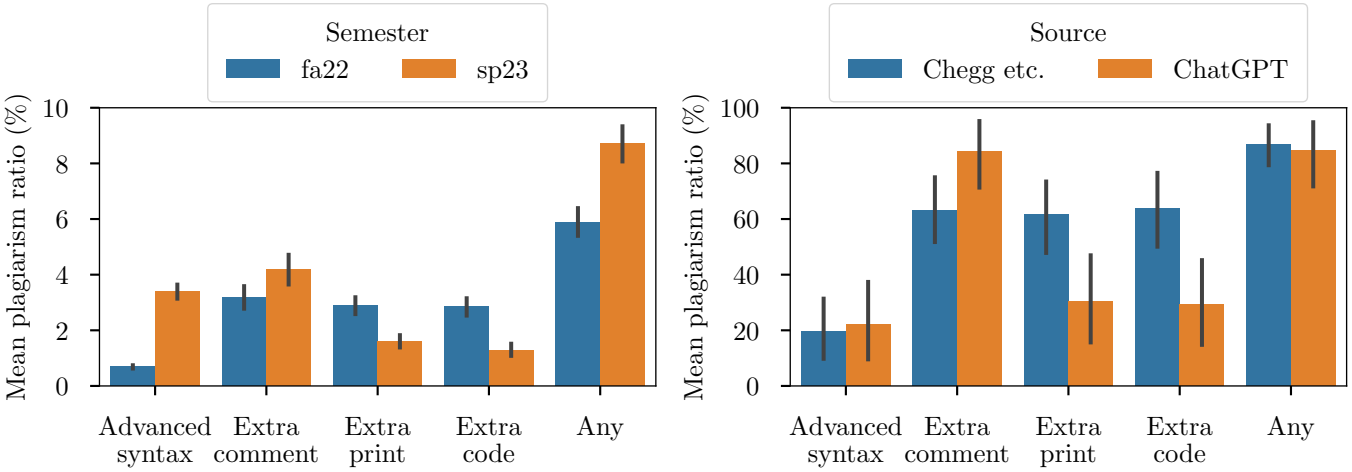

Figure 5: The left plot shows the mean plagiarism ratio on each of the markers over all programming questions on homework. The right plot shows the mean plagiarism ratio on answers collected from popular plagiarism hubs and ChatGPT over a set of questions sampled from the homework, as described in Section 3.6. The error bars show 95% confidence intervals of the means.

students that are prone to plagiarize are merely changing the way that they plagiarize.

The left plot of Figure 5 show the relative frequency of the plagiarism markers in the student submissions by semester. We found the Fall 2022 (pre-ChatGPT) semester to have statistically significantly lower fraction of answers with advanced syntax ($p < 0.001$) and extra comments ($p = 0.003$),[5] and higher fractions of answers with extra prints ($p < 0.001$) and extra code ($p < 0.001$).[6] As such, it appears that the source of the plagiarism might be different in the two semesters.

Using the collection of plagiarized solutions described in Section 3.6, the right plot of Figure 5 shows the ratio of each marker found in the ChatGPT-produced solutions as compared to the solutions that we retrieved from popular plagiarism hubs. We see that answers found on plagiarism hubs have statistically significantly lower extra comment ($p = 0.020$), and statistically significantly higher extra code ($p = 0.004$) and extra print ($p = 0.002$). The similarity of this pattern to that observed in student submissions supports the hypothesis that students moved from plagiarism hubs (Fall 2022) to ChatGPT (Spring 2023).

While we saw a statistically significant difference between the semesters on the advanced syntax marker, we do not see one between plagiarism hubs and ChatGPT ($p = 0.791$). We discuss possible explanations for this in Section 5.

These results are consistent with a significant shift in the source of plagiarism from plagiarism hubs to ChatGPT. Furthermore, this shift is also supported anecdotally by students admission of using ChatGPT in the academic integrity cases mentioned in Section 3.3.

[5]Overlap of 95% confidence intervals of two means does not automatically imply insignificance, but 95% confidence interval of one mean containing the other mean does.

[6]Both subplots in Figure 5 also show that the extra code marker and extra print marker have similar plagiarism ratios. Our manual inspection found that print often occurs in testing code, which would have both of the markers. We decided to keep both markers as they only overlap about 75% of the time.

4.3 Plagiarism correlated to less learning (RQ3)

In this section, we present results indicating that a student's degree of plagiarism is correlated with less learning. In the next section, we use the same results to reason about whether the shift from plagiarism hubs to ChatGPT impacts this learning loss.

These results use a linear regression to predict final exam scores using the student's "baseline performance" and their amount of observed plagiarism between the measurement of baseline performance and the final exam. We first present the details of this regression, then why the students' Exam 1 performance is an acceptable measure of baseline performance, and, finally, the results.

Method: We fit a linear regression of the following form (note the negative sign before γ):

$$finalExam_i = \alpha + \beta \cdot firstExam_i - \gamma \cdot plagiarismRatio_i \quad (2)$$

where $finalExam_i$, $firstExam_i$, and $plagiarismRatio_i$ are observed values from the data, defined as follows:

- $finalExam_i$: student i's predicted (see Section 3.8) final exam score, a number between 0 and 100.
- $firstExam_i$: student i's predicted first proctored exam score, a number between 0 and 100; a measure of the student's baseline performance before significant plagiarism has occurred.
- $plagiarismRatio_i$: student i's plagiarism ratio of programming questions on homework between the first proctored exam and the final exam, a number between 0 and 1.

α, β, γ are the coefficients that we want to estimate, which can be interpreted as:

- α: the intercept, i.e., how much a student would be predicted to score on the final exam if the student scored 0 on the first proctored exam and did not plagiarize,
- β: the effect of the predicted first proctored exam score on the predicted final exam score, i.e., how much better a student would be predicted to score on the final exam for every

percentage point the student scored on the first proctored exam,
- γ: the effect of plagiarism on the predicted final exam score, i.e., how much worse a student would be predicted to score on the final exam if the student's plagiarism ratio of programming questions on homework between the first exam and the final exam is 1, in other words if the student plagiarized every single programming question on homework between the first proctored exam and the final exam.

Exam 1 as baseline performance: In order to analyze the learning impact of cheating, our regression presumes there is a performance that plagiarizing students would have achieved on the final exam in the scenario where they did not plagiarize, and the learning loss is the difference between their actual performance and this counterfactual performance. Our regression uses a control for the student's baseline ability in estimating this performance.

We did not have access to the students standardized (e.g., ACT) test scores, so we used their performance on the first proctored exam (Exam 1) as our measure of baseline performance. Two pieces of evidence suggest that Exam 1 occurs before the most of the plagiarism in the course takes place.

First, the difference between students' unproctored Quiz 1 performance and their proctored Exam 1 performance (shown in Figure 2) is smaller (5%) than for the other quiz-exam pairs, which is typically around 10%. The small unproctored advantage for Quiz 1 over Exam 1 suggests that either students were less likely to attempt to plagiarize significantly on Quiz 1 or they were not yet effective at doing so. Similar trend has been observed previously [9].

Second, the material leading up to Exam 1 is simpler than later material, likely necessitating less plagiarism. Figure 1 shows the plagiarism ratio for each homework assignment individually; exams are situated after the homework assignments up to which they cover. Homework assignments 1–3 focus on building a mental model of execution (through tracing problems) and syntax (through questions that ask student to write a single line of code; e.g., using interfaces of built-in data structures like lists). In fact Homeworks 2 and 3 include none of the multi-line programming questions analyzed in this paper, and thus have no plagiarism markers. As can be seen, the bulk of the observed plagiarism happens later in the course, as the number and complexity of the programming questions increase.

Based on this evidence, we feel that Exam 1 is an acceptable baseline measure of student ability.

Results: We fit the linear regression for the two semesters separately. The results can be found in Table 2.

The key output of the regression is the coefficient γ (plotted in Figure 6), which relates learning loss to the degree of observed plagiarism. In both semesters, this parameter is statistically significantly positive. The Fall 2022 data suggests that a student observed to plagiarize *every* assignment would perform 47 percentage points lower than they would if they had not cheated. Spring 2023 data suggests the drop would only be 36 percentage points. Because we usually detect plagiarism on only a fraction of a student's submissions (see Figure 3), the observed learning losses are smaller than these numbers would suggest.

Table 2: Regression coefficients and p-values for the linear regression described in Section 4.3.

Coefficient	Semester	Value	95% CI		p-value
α	fa22	−5.21	−12.45	2.03	0.158
	sp23	−15.15	−23.56	−6.74	< 0.001
β	fa22	1.03	0.95	1.11	< 0.001
	sp23	1.10	1.01	1.20	< 0.001
γ	fa22	47.05	34.50	59.61	< 0.001
	sp23	35.91	20.89	50.93	< 0.001

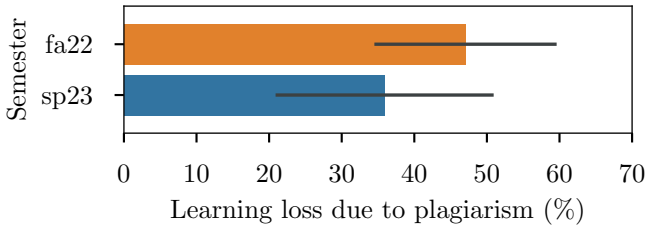

Figure 6: Learning loss due to plagiarism, measured by how much worse a student would be predicted to perform on the final exam if the student plagiarized every programming question on homework. These values correspond to γ in the linear regression described in Section 4.3 when two semesters' data were fitted separately. The error bars show 95% confidence intervals of the regression estimates.

4.4 Learning impact is independent of type of plagiarism (RQ4)

While our regressions computed different values for the plagiarism learning loss (the coefficient γ) in the two semesters, the values are not statistically significantly different ($p = 0.258$). As such, our data does not support the hypothesis that one method of plagiarism (hubs vs. ChatGPT) is more detrimental than the other.

5 DISCUSSION

In hindsight, that plagiarism has increased after the release of ChatGPT is not surprising, since the accessible nature of ChatGPT not only lowers the cost of plagiarism, but also reduces the waiting time for an answer. The measured increase, however, is modest. Again, in hindsight, this makes intuitive sense in the context of fraud triangle theory [13], as generative AI only influences the opportunity aspect directly and not, for example, the rationalization aspect.

The data for three of our four markers (extra comment, print, and code) is consistent with an almost complete shift in the mode of plagiarism. This finding is consistent with a concurrent loss of revenue by commercial plagiarism hubs [25].

We were surprised that the semester trend for the fourth marker (advanced syntax) did not reflect what we found in our manual collection of plagiarized code. We can think of three possible reasons for this. First, students copying code from plagiarism hubs might bias their selection to code that they can understand if presented with multiple solutions, leading to the observed lower frequency of

advanced syntax markers in Fall 2022. Second, ChatGPT 3.5 was updated at least twice during 2023, so the code samples that we obtained from ChatGPT 3.5 might not reflect what students received from ChatGPT 3.5 during the Spring 2023 semester.[7] Third, our ChatGPT prompts were just question prompts copied verbatim, which could differ substantially from those used by students, thus leading to this difference.

Consistent with our hypothesis discussed in Section 1, our analysis strongly suggests that plagiarism leads to significant learning loss. We initially theorized that plagiarizing with ChatGPT would be worse than with plagiarism hubs such as Chegg, because ChatGPT may not be conscientious of the context and could provide solutions that students would not be able to understand, such as those flagged by the advanced syntax marker. However, our findings do not support this hypothesis.

Interestingly, in spite of differences between the kinds of markers that show up on samples from plagiarism hubs compared to those from ChatGPT, the overall ("any") plagiarism ratios are remarkably consistent between plagiarism hubs and ChatGPT. The ability to detect plagiarism consistently across different sources is important to many pieces of our analysis, including our estimates of the relative amount of plagiarism between semesters (RQ1) as well as the relative learning losses due to the two sources of plagiarism (RQ4).

We caution readers from using these markers to build tools that try to determine if an individual student's work has been plagiarized. Independent of possible false positives and negatives, students will likely learn to prompt generative AI to generate solutions that would be hard to discern from original work. Future generative AI might also provide answers without some of these markers (e.g., advanced syntax), even if students do not explicitly prompt them to do so.

6 LIMITATIONS

The primary limitation of this work is our method of detecting plagiarism. Our method flags student answers based only on the answer itself.[8] As noted in Section 3.5, we believe that this leads to an underestimate of the amount of plagiarism (and consequently an overestimate of γ), but that the underestimate is consistent between sources.[9] Considering other data (e.g., time it takes a student to make a response) could be used to improve the identification and, perhaps, be used to measure plagiarism for question types whose correct answer space is very small (e.g., write a line of code to remove the element at index 2 of a list called `foods`), which are not considered in the current paper.

In addition, this research carries the usual constraints of a study focused on a single course. While we imagine the observed trends (more plagiarism, different source, and learning loss independent of source) generalize to other topics and other courses, many aspects of a course (institution, demographics, course delivery) could impact the specific numeric values found. One counterpoint is that Lee et al. [23] did not find a notable increase in cheating in US high schools after the introduction of ChatGPT. However, their data was collected from March to May in 2023 and high school students might take longer than university students to adopt new technologies for cheating. It would be best to generalize the results in the current paper through a meta-analysis of multiple studies in different contexts.

7 CONCLUSION

The advent of generative AI facilitates students' plagiarism because it can provide students with answers quickly, freely, and without having to interact with other people. By identifying markers associated with plagiarism in one particular class, we observed that the popularization of generative AI lead to (1) a modest increase in plagiarism, from cheating hubs such as Chegg to ChatGPT, (2) a substantial shift in the source of plagiarism, and (3) no significant change in the already substantial learning loss due to plagiarism.

We suspect that future advances in generative AI and students' increasing aptitude in using it could make identifying plagiarism nearly impossible, which places teachers and instructors in a challenging position when grading out-of-class work. The solution to this problem may be one that is already being championed for equity. Feldman suggests that students grades be computed entirely or almost entirely from summative assessment, treating formative assessment (the bulk of out-of-class work) as a means to an end, rather than an end itself [17]. Because summative assessment is typically small relative to formative assessment, we have the potential to make it secure and trustworthy.

Furthermore, the future of summative assessment might significantly include evaluating students' ability to perform tasks in conjunction with generative AI, for example using GitHub Copilot to solve programming tasks [31]. While we may never completely eliminate summative assessment of "un-augmented" humans, this portion of assessment might be even smaller than it is now, further facilitating our ability to make it trustworthy.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under grant numbers 1725729, 2121424, and 2144249.

A APPENDIX: VALIDITY OF MARKERS AS PLAGIARISM INDICATORS

In this section, we provide evidence demonstrating that the markers shown in Table 1 are probable indicators of plagiarism under the context of homework submissions. We first visualize how final exam score correlates differently with homework plagiarism ratio versus exam plagiarism ratio. Figure 7 shows the scatter plot of final exam score against the plagiarism ratio of each marker on homework and exam. As the slope of the fitted regression lines and their corresponding 95% confidence intervals suggest, the correlations are all significantly negative on homework, but not on exams.

We report correlations between final exam score and all combinations of marker, assessment type, and semester in Table 3. This table

[7] While an API of older ChatGPT 3.5 models (e.g., gpt-3.5-turbo-0301) are available, the behavior of ChatGPT 3.5's web interface, which is presumably what students used, and the ChatGPT 3.5 API are far from similar based on our experience and anecdotes reported on the OpenAI developer's forum. OpenAI has never disclosed what additional prompting to include in an API call to enable the behavior observed on the ChatGPT web interface.

[8] Tools like MOSS [1] compare a student's answer to other students' answers.

[9] The 15% false negative rate quoted in Section 3.5 may be an underestimate because that data was collected for a subset of the questions whose answers appeared to be the most frequently plagiarized.

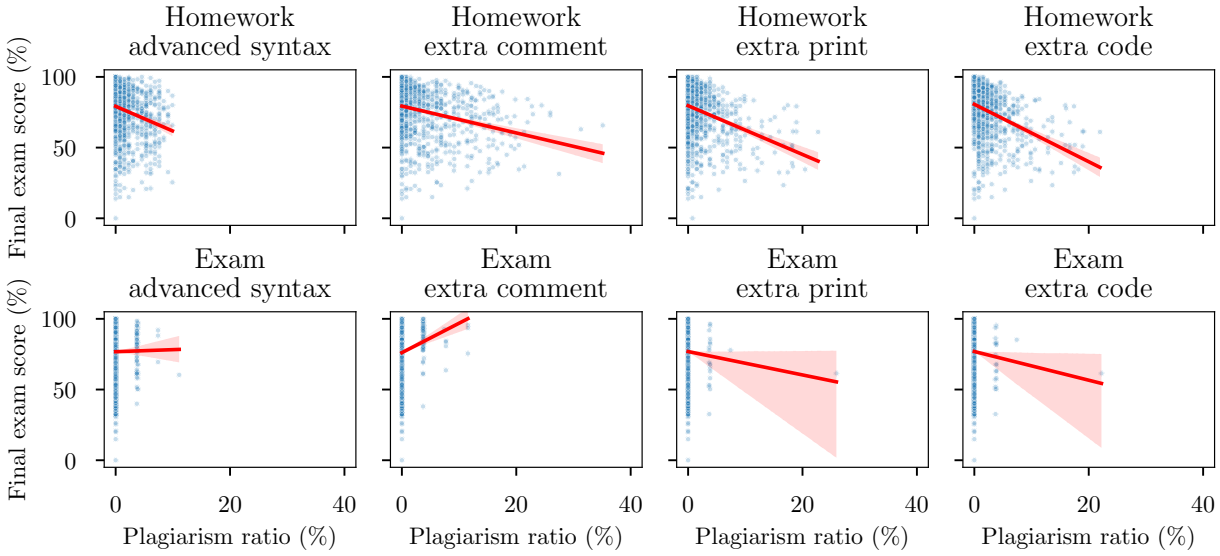

Figure 7: Scatter plot of final exam score against plagiarism ratio of each marker on homework and exam. Each data point corresponds to one student. We aggregated the semesters for this plot because the per-semester plots were very similar. The red line is the result of linear regression fitted on the data. The red band visualizes the 95% confidence interval of the regression fit. A negative slope indicates that higher plagirism rations are associated with lower final exam scores.

Table 3: Correlations between final exam scores and marker ratios of each marker on each assessment type. Numbers in parentheses are p-values.

Marker	Semester	Homework	Quiz	Exam
Advanced syntax	fa22	-0.18 ($< 0.001^{***}$)	-0.09 (0.041^{*})	-0.03 (0.497)
	sp23	-0.25 ($< 0.001^{***}$)	-0.25 ($< 0.001^{***}$)	0.05 (0.267)
Extra comment	fa22	-0.31 ($< 0.001^{***}$)	-0.24 ($< 0.001^{***}$)	0.14 (0.001^{**})
	sp23	-0.23 ($< 0.001^{***}$)	-0.13 (0.008^{**})	0.19 ($< 0.001^{***}$)
Extra print	fa22	-0.51 ($< 0.001^{***}$)	-0.38 ($< 0.001^{***}$)	-0.06 (0.157)
	sp23	-0.23 ($< 0.001^{***}$)	-0.15 (0.002^{**})	-0.05 (0.287)
Extra code	fa22	-0.49 ($< 0.001^{***}$)	-0.41 ($< 0.001^{***}$)	-0.04 (0.378)
	sp23	-0.16 ($< 0.001^{***}$)	-0.09 (0.069)	-0.06 (0.254)
Any	fa22	-0.43 ($< 0.001^{***}$)	-0.37 ($< 0.001^{***}$)	0.11 (0.010^{*})
	sp23	-0.32 ($< 0.001^{***}$)	-0.25 ($< 0.001^{***}$)	0.09 (0.069)

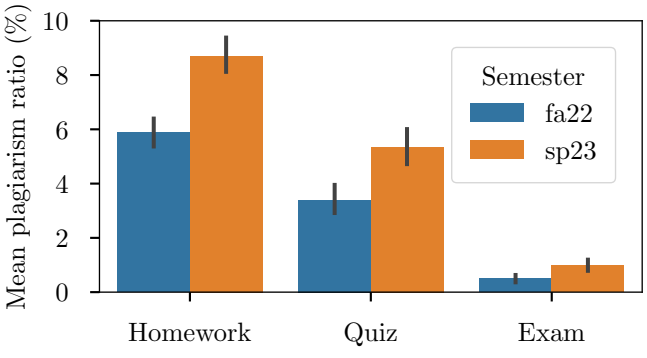

Figure 8: Mean plagiarism ratio of each assessment type.

reveals consistently significantly negative correlations for homework, predominantly significant negative correlations for quizzes, but not for exams—some correlations even turn significantly positive in exam contexts. These patterns imply that students demonstrating these markers during exams are likely employing good programming practices, unlike in homework settings where high marker appearance likely indicates plagiarism.

Lastly, we plotted the average plagiarism ratio across different types of assessments in Figure 8. As the figure shows, homework plagiarism ratios are significantly higher than those on quizzes, which are in turn significantly higher than those on exams. This suggest that students who submit code with markers on homework are likely plagiarizing rather than following good programming practices since they are not doing the same under exam conditions.

REFERENCES

[1] Alex Aiken. 1994. *MOSS: A System for Detecting Software Similarity.* https://theory.stanford.edu/~aiken/moss/

[2] Ibrahim Albluwi. 2019. Plagiarism in programming assessments: a systematic review. *ACM Transactions on Computing Education (TOCE)* 20, 1 (2019), 1–28.

[3] Frank B Baker. 2001. *The basics of item response theory.* ERIC.

[4] Christina Areizaga Barbieri, Dana Miller-Cotto, Sarah N Clerjuste, and Kamal Chawla. 2023. A meta-analysis of the worked examples effect on mathematics performance. *Educational Psychology Review* 35, 1 (2023), 11.

[5] John Bransford, National Research Council (U.S.), and National Research Council (U.S.) (Eds.). 2000. *How people learn: brain, mind, experience, and school* (expanded ed ed.). National Academy Press, Washington, D.C.

[6] Peter C Brown, Henry L Roediger III, and Mark A McDaniel. 2014. *Make it stick: The science of successful learning.* Harvard University Press.

[7] Debra D Burke and Kenneth J Sanney. 2018. Applying the fraud triangle to higher education: Ethical implications. *J. Legal Stud. Educ.* 35 (2018), 5.

[8] Andrew C Butler and Henry L Roediger. 2008. Feedback enhances the positive effects and reduces the negative effects of multiple-choice testing. *Memory & cognition* 36, 3 (2008), 604–616.

[9] Binglin Chen, Sushmita Azad, Max Fowler, Matthew West, and Craig Zilles. 2020. Learning to cheat: quantifying changes in score advantage of unproctored assessments over time. In *Proceedings of the Seventh ACM Conference on Learning@ Scale*. 197–206.

[10] Binglin Chen, Matthew West, and Craig Zilles. 2018. How much randomization is needed to deter collaborative cheating on asynchronous exams?. In *Proceedings of the fifth annual ACM conference on learning at scale*. 1–10.

[11] Freddie Choo and Kim Tan. 2008. The effect of fraud triangle factors on students' cheating behaviors. In *Advances in accounting education*. Vol. 9. Emerald Group Publishing Limited, 205–220.

[12] Janice Connolly, Paula Lentz, Joline Morrison, et al. 2006. Using the business fraud triangle to predict academic dishonesty among business students. *Academy of Educational Leadership Journal* 10, 1 (2006), 37.

[13] Donald R Cressey. 1953. Other people's money; a study of the social psychology of embezzlement. (1953).

[14] Deborah F Crown and M Shane Spiller. 1998. Learning from the literature on collegiate cheating: A review of empirical research. *Journal of business ethics* 17 (1998), 683–700.

[15] Paul Denny, Viraj Kumar, and Nasser Giacaman. 2023. Conversing with copilot: Exploring prompt engineering for solving cs1 problems using natural language. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*. 1136–1142.

[16] David J Emerson and Kenneth J Smith. 2022. Student use of homework assistance websites. *Accounting Education* 31, 3 (2022), 273–293.

[17] Joe Feldman. 2023. *Grading for equity: What it is, why it matters, and how it can transform schools and classrooms.* Corwin Press.

[18] Kristin Voelkl Finn and Michael R Frone. 2004. Academic performance and cheating: Moderating role of school identification and self-efficacy. *The journal of educational research* 97, 3 (2004), 115–121.

[19] Valerie J Haines, George M Diekhoff, Emily E LaBeff, and Robert E Clark. 1986. College cheating: Immaturity, lack of commitment, and the neutralizing attitude. *Research in Higher education* 25 (1986), 342–354.

[20] Cindy E Hmelo-Silver. 2004. Problem-based learning: What and how do students learn? *Educational psychology review* 16 (2004), 235–266.

[21] Thomas Lancaster and Codrin Cotarlan. 2021. Contract cheating by STEM students through a file sharing website: a Covid-19 pandemic perspective. *International Journal for Educational Integrity* 17 (2021), 1–16.

[22] Mark M Lanier. 2006. Academic integrity and distance learning. *Journal of criminal justice education* 17, 2 (2006), 244–261.

[23] Victor R. Lee, Rosalia C. Zarate, Denise Pope, and Sarah B. Miles. submitted, under review. Cheating in the Age of Generative AI: A High School Survey Study of Cheating Behaviors before and after the Release of ChatGPT. (submitted, under review).

[24] Donald L McCabe and Linda Klebe Trevino. 1997. Individual and contextual influences on academic dishonesty: A multicampus investigation. *Research in higher education* 38 (1997), 379–396.

[25] Bill McColl. 2023. *Chegg Shares Plunge After Company Warns That ChatGPT Is Impacting Growth.* https://www.investopedia.com/chegg-shares-plunge-after-company-warns-that-chatgpt-is-impacting-growth-7487968

[26] Janet Metcalfe. 2017. Learning from errors. *Annual review of psychology* 68 (2017), 465–489.

[27] Paula J Miles, Martin Campbell, and Graeme D Ruxton. 2022. Why students cheat and how understanding this can help reduce the frequency of academic misconduct in higher education: a literature review. *Journal of Undergraduate Neuroscience Education* 20, 2 (2022), A150.

[28] Jaap MJ Murre and Joeri Dros. 2015. Replication and analysis of Ebbinghaus' forgetting curve. *PloS one* 10, 7 (2015), e0120644.

[29] David J Palazzo, Young-Jin Lee, Rasil Warnakulasooriya, and David E Pritchard. 2010. Patterns, correlates, and reduction of homework copying. *Physical Review Special Topics-Physics Education Research* 6, 1 (2010), 010104.

[30] Jonathan Pierce and Craig Zilles. 2017. Investigating student plagiarism patterns and correlations to grades. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*. 471–476.

[31] Leo Porter. 2024. *Learn AI-Assisted Python Programming: With Github Copilot and ChatGPT.* Simon and Schuster.

[32] Katherine A. Rawson, John Dunlosky, and Sharon M. Sciartelli. 2013. The Power of Successive Relearning: Improving Performance on Course Exams and Long-Term Retention. *Educational Psychology Review* 25, 4 (Dec. 2013), 523–548. https://doi.org/10.1007/s10648-013-9240-4

[33] Clara Sabbagh. 2021. Self-reported academic performance and academic cheating: Exploring the role of the perceived classroom (in) justice mediators. *British Journal of Educational Psychology* 91, 4 (2021), 1517–1536.

[34] Jaromir Savelka, Arav Agarwal, Marshall An, Chris Bogart, and Majd Sakr. 2023. Thrilled by Your Progress! Large Language Models (GPT-4) No Longer Struggle to Pass Assessments in Higher Education Programming Courses. In *Proceedings of the 2023 ACM Conference on International Computing Education Research*.

[35] Daniel L. Schwartz, Jessica M. Tsang, and Kristen P. Blair. 2016. *The ABCs of how we learn: 26 scientifically proven approaches, how they work, and when to use them* (first edition ed.). W.W. Norton & Company, Inc., New York, NY. OCLC: 954134221.

[36] Valerie J Shute. 2008. Focus on formative feedback. *Review of educational research* 78, 1 (2008), 153–189.

[37] Ben Skudder and Andrew Luxton-Reilly. 2014. Worked examples in computer science. In *Proceedings of the Sixteenth Australasian Computing Education Conference-Volume 148*. 59–64.

[38] Robert Stickgold and Matthew P Walker. 2005. Memory consolidation and reconsolidation: what is the role of sleep? *Trends in neurosciences* 28, 8 (2005), 408–415.

[39] Fabienne M Van der Kleij, Remco CW Feskens, and Theo JHM Eggen. 2015. Effects of feedback in a computer-based learning environment on students' learning outcomes: A meta-analysis. *Review of educational research* 85, 4 (2015), 475–511.

[40] Tamara Van Gog, Liesbeth Kester, and Fred Paas. 2011. Effects of worked examples, example-problem, and problem-example pairs on novices' learning. *Contemporary Educational Psychology* 36, 3 (2011), 212–218.

[41] Matthew West, Geoffrey L Herman, and Craig Zilles. 2015. Prairielearn: Mastery-based online problem solving with adaptive scoring and recommendations driven by machine learning. In *2015 ASEE Annual Conference & Exposition*. 26–1238.

[42] Hongwei Yu, Perry L Glanzer, Byron R Johnson, Rishi Sriram, and Brandon Moore. 2018. Why college students cheat: A conceptual model of five factors. *The Review of Higher Education* 41, 4 (2018), 549–576.

[43] Craig Zilles, Matthew West, David Mussulman, and Tim Bretl. 2018. Making testing less trying: Lessons learned from operating a Computer-Based Testing Facility. In *2018 IEEE Frontiers in Education Conference (FIE)*. IEEE, 1–9.

[44] Craig B Zilles, Matthew West, Geoffrey L Herman, and Timothy Bretl. 2019. Every University Should Have a Computer-Based Testing Facility.. In *CSEDU (1)*. 414–420.