A APPENDICES

A.1 Extended Method: Example Calculation

Here is a simple example of how the estimator works. We use c (compliers) to mark students who would become adopters if we offered them access. We use n to mark students who would not use GPT-4 even if we offered them. We have both types of students in the experiment and in the control group. We use N_{c1} to mark the number of adopters in the experiment group and N_{n1} to mark the number of students who did not use GPT-4 when given access. Analogously, we can define N_{c0} and N_{n0} for the control group as well. Note that we know N_{c1} and N_{n1} in our experiment because we have a detailed record of who used GPT-4 or not when they were given access, but we don't know N_{c0} and N_{n0} because we don't know who are the adopters in the control group. Let the total number of students in the experiment group be N_1 and the total number of students in control N_0. We use $i \in c0$ to mark students in the experiment group who are adopters (observed) and $i \in c1$ to mark students in the control group who are adopters (unobserved). Let Y be the student's diagnostic exam score. The learning benefit **E2** can be computed as:

$$\mathbf{E2} = \frac{1}{N_{c1}} \sum_{i \in c1} Y_i - \frac{1}{N_{c0}} \sum_{i \in c0} Y_i \tag{2}$$

We can rewrite **E1** in the same way as well:

$$\begin{aligned}
\mathbf{E1} &= \frac{1}{N_1}\left(\sum_{i \in c_1} Y_i + \sum_{i \in n_1} Y_i\right) - \frac{1}{N_0}\left(\sum_{i \in c_0} Y_i + \sum_{i \in n_0} Y_i\right) && (3)\\
&= \frac{1}{N_1}\sum_{i \in c_1} Y_i + \frac{1}{N_1}\sum_{i \in n_1} Y_i - \frac{1}{N_0}\sum_{i \in c_0} Y_i - \frac{1}{N_0}\sum_{i \in n_0} Y_i && (4)\\
&= \frac{1}{N_1}\sum_{i \in c_1} Y_i - \frac{1}{N_0}\sum_{i \in c_0} Y_i + \underbrace{\frac{1}{N_1}\sum_{i \in n_1} Y_i - \frac{1}{N_0}\sum_{i \in n_0} Y_i}_{=0} && (5)\\
&= \frac{1}{N_1}\sum_{i \in c_1} Y_i - \frac{1}{N_0}\sum_{i \in c_0} Y_i && (6)\\
&= \underbrace{\frac{N_{c1}}{N_1}}_{=P(c)} \frac{1}{N_{c1}}\sum_{i \in c_1} Y_i - \underbrace{\frac{N_{c0}}{N_0}}_{=P(c)} \frac{1}{N_{c0}}\sum_{i \in c_0} Y_i && (7)\\
&= P(c)\left(\frac{1}{N_{c1}}\sum_{i \in c_1} Y_i - \frac{1}{N_{c0}}\sum_{i \in c_0} Y_i\right) && (8)\\
&= P(c)\,\mathbf{E2} && (9)
\end{aligned}$$

As we can see, we can compute **E2** from **E1** from this derivation without ever needing to know exactly who the compliers would be in the control group. There are a few key assumptions that make this derivation work. Eq (5) holds from the exclusion principle (Assumption Assumption 3), which states that the student's exam score is independent of the nudge (advertisement) given their GPT-4 usage status. Second, Eq (7), the two terms $\frac{N_{c1}}{N_1}$ and $\frac{N_{c0}}{N_0}$ are estimators of the same probability – the probability of a student being an adopter in the experiment and the control group. As in Assumption Assumption 1, we assigned students to the experiment and control group randomly, and therefore the

percentage of compliers in the experiment and control group should be identical in expectation, $P(c)$. This term can be estimated as $P(c) = \frac{N_{c1}}{N_1}$.

A.2 Extended Discussion

A.2.1 Limitations. A Particular Course, with Particular Students, at a Particular Moment in Time

A common limitation of educational research, even randomized control trials, is that experiments are conducted in a particular context. In this case, we ran the student in a particular domain (computer science) with a particular cohort of students at a particular moment in time. Consequently, the findings from this study only directly apply to this particular context. The same intervention, if studied in different learning environments, may produce different results. Our course's unique characteristics likely shape the educational experience and outcomes observed: (1) The course is at-will in contrast to many k-12 or university experiences where learners must complete the whole experience in order to get a grade and/or credit. Students in traditional classrooms may have experiences that could lower their motivation, but they are less likely to drop-out. (2) The course centers on human teachers. As such, it likely attracts learners who are interested in education from humans. This may make our learner population different than that of a typical massive online course. (3) The impacts might be specific to programming education. Some of the causal mechanisms for why students disengage may play out differently in courses on literature, history, or even math. This is especially believable because coding is a computer-oriented domain in which LLMs particularly excel. (4) The experiment was conducted in 2023, a period marked by a distinctive zeitgeist regarding artificial intelligence in education. While perceptions and news differ by geographic location, broadly, the news was a mixture of excitement about the potential of large language models and fears over the role that LLMs may play in society.

These limitations are typical in the field of education. We urge the reader to be cautious about extrapolating these results into different contexts. Despite these contextual limitations, a large-scale randomized control trial case study remains one of the most useful ways of knowing within the educational domain. The insights gained, while specific, can be a helpful data point for those trying to pursue a broader understanding of educational interventions and their impacts.

A.2.2 What Caused Disengagement? In the randomized control trial, there was a notable difference in engagement on a two-week time scale between learners with and without access to ChatGPT. This effect is influenced by student age, coding experience, and HDI of the student's country. We note that it is surprising to see a drop in engagement in education simply from getting access to an optional tool. Many learning science experiments are "doomed to succeed" because of the strong novelty effect in education [35]. This novelty effect is seemingly non-existent when providing an LLM chat interface to students. On their own, these results do not suggest *why* there was a difference in engagement. In this section, we present a few theories on such a causal mechanism.

Job Threat Hypothesis

One hypothesis for the impact on engagement is that access and advertisement to GPT confronted learners with a threat to their prospects of getting a programming job. Several studies have shown that learners who make a connection between academic courses and activities and future versions of themselves experience increased motivation towards their learning [18, 32, 33], especially in intro to programming courses. Specifically, [40] showed that a difference in a student's perception of an instrumental connection between course participation and employment was a significant

predictor of standardized course grades. Under the Job Threat Hypothesis, interacting with ChatGPT decreases a student's instrumental connection between learning and obtaining employment. When the student either uses ChatGPT or reads the advertisement, the student supposes that there may be fewer jobs and there will be steeper competition for those limited roles. At the very least, a student might perceive a greater uncertainty around jobs.

This hypothesis is supported by the observation that the biggest engagement impact of access and advertising of GPT was on people who demographically match those who are applying for jobs – people in the job-seeking age (22-40) and middle-experience programmers. The instrumental connection between learning to code and jobs is especially pertinent in the Code in Place course. Of the students in the class, 46.8% of students list, "I want to get a job as a programmer" as a motivation for taking the class upon submitting their application. The percentage of students who listed jobs as motivation is even higher for learners of job-seeking age (49.0%). In contrast, only 43.7% of learners outside that range listed jobs as a motivation. As a corollary, there was a 1.26 percentage point decrease in exam participation among learners in the experimental condition who listed a job as a motivation for taking the course (43.4% exam participation) compared to those who did not list jobs (44.7% exam participation).

AI Mistrust Hypothesis

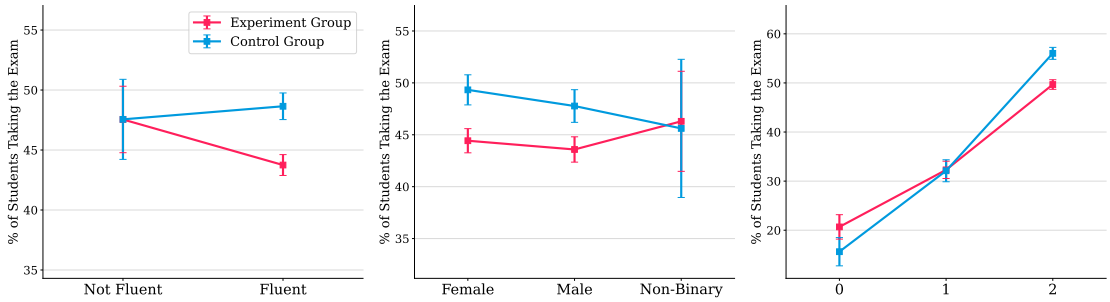
Artificial intelligence's integration into society has engendered polarized opinions. IPSOS ran a global study of perceptions of AI at the same time that the Code in Place experiment ran, where they interviewed 22,816 adults under the age of 75 across 31 countries [21]. They found that 52% of respondents were nervous regarding AI, and 46% did not agree that there were more benefits than drawbacks. These broad phenomena appeared to be playing out inside the course. In conversations with learners in Code in Place, we noticed a surprisingly high level of distrust in AI. Students voiced concerns which range from concerns over (1) privacy, (2) concerns about the trustworthiness of AI, as well as (3) more existential worries about the role of AI in society [26]. As such, as evidenced by the IPSOS survey and conversations with our students, it is reasonable to assume that a subset of our learners (large but of unknown size) do not trust AI. We also assume that there is a separate subset of learners who are excited about the potential. We hypothesize that those with a negative association towards AI are more impacted by a teaching chatbot than those with a positive attitude. As such, the polarization surrounding AI, fueled by these concerns, may contribute to a decrease in engagement among students when AI tools are introduced into the learning environment. The country-level variance in trust seems to support this hypothesis. In the IPSOS survey, they found that "Trust in AI varies widely by region; it is generally much higher in emerging markets ... than in high-income countries." This ordering of countries by IPSOS trust in AI reflects the ordering by country of effect-sizes that the advertisement of GPT had on students. Specifically, we also note that learners from emerging markets (largely those with lower HDI) were positively impacted by access to ChatGPT.

People are more comfortable with AI automating mechanical tasks than human or relationship labor, such as caring for children. Where does teaching fit into this spectrum? There is a broad set of literature that argues, "The need for social belonging—for seeing oneself as socially connected—is a basic human motivation" and as such, " social connectedness ... has a dramatic impact on course completion" [45]. Providing more AI assistance likely induces fewer human-to-human interactions. Could this reduction in interpersonal engagement lead to feelings of loneliness and alienation? These concerns are particularly potent in courses like Code in Place, designed to emphasize a human-centric learning experience. In such environments, the introduction of AI tools such as ChatGPT might be perceived as especially intrusive or displacing, contributing to a decrease in student engagement.

A.3 What Caused Improved Exam Scores?

Our analysis of the adopters' interactions with GPT-4 shows that those who used GPT-4 were generally using it in a constructive way to better their understanding of the material and the field at large. This could have improved their understanding, which transferred to test performance gains. An interesting question for future study is how the type of use of GPT-4, and the amount, might impact learning outcomes. This particular result is less surprising. There is a long-held belief in education that direct one-on-one tutoring substantially improves student ability [8]. LLM chat seems to be a reasonably useful autonomous tutor. One concern was that access to an AI tutor at any point could serve as too much help, sometimes labeled the "Scaffolding Paradox" [17]. Instead of grappling with problems, iterating through solutions, and learning from errors– a crucial cycle in developing programming acumen– students may shortcut this process by seeking immediate answers from AI, thereby steepening their learning curve in the long run. Fortunately, the strong exam performance of students who adopted ChatGPT suggests that overreliance on AI, to the detriment of learning, was not a dominant mechanism in our setting.

A.4 Additional Student Covariate Interaction Effect



(a) Student English Fluency **(b)** Student Gender **(c)** Student Section Attendance

Fig. 8. We present exploratory analyses for the experiment-control exam participation gap by looking at different student demographics. The vertical line is the standard error.

We show additional interactions between student covariates and their exam participation in Figure 8. We note that there doesn't seem to be a difference in exam participation for students who are not fluent in English. Nonetheless, as the main instructional materials for this course are in English, most students who decide to apply already have high proficiency in the language, regardless of whether it is the primary language in their home countries. For gender, we notice a similar gap between experiment and control. For section attendance, we are not able to draw meaningful conclusions beyond what we have observed so far.

A.5 Additional Details on Trial Assignment

At the beginning of April 24th, 8,762 students enrolled in the class. However, we don't want to offer access to ChatGPT too early because some empirical work in education showed that providing too many hints too early might hurt a student's learning progress [17]. We determine a student to be "active" by looking at whether they have completed all Week 1 assignments. We end up with 5,831 students in our randomized control trial. We then sent an email to 3,581 students. 2,778 students (77.6%) opened the email. 539 students (15.1%) clicked on the link to our custom ChatGPT interface. We obtained institutional review board (IRB) approval for conducting this experiment.

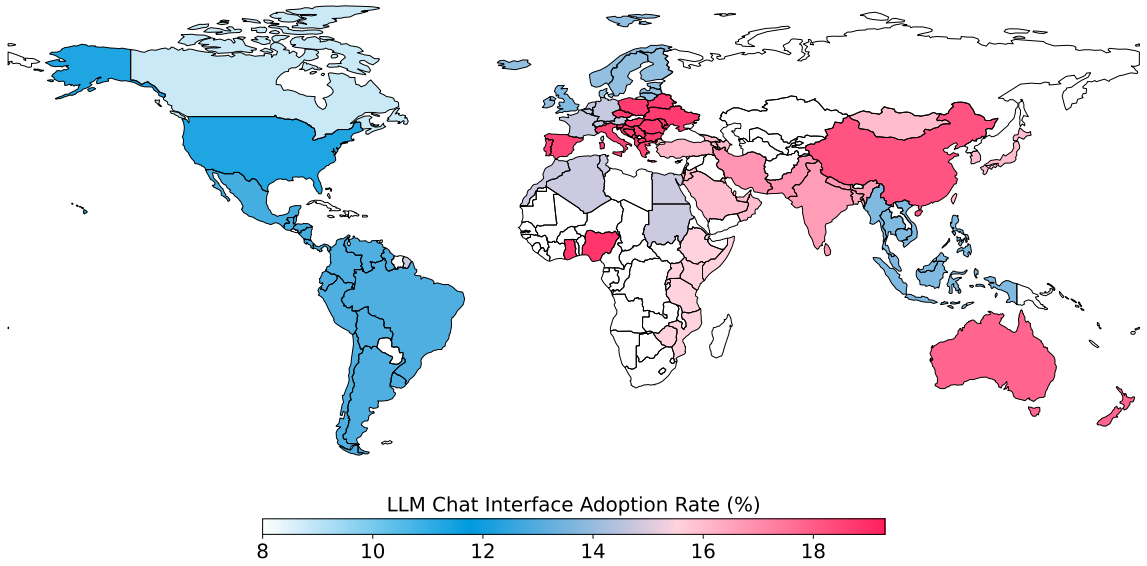


Fig. 9. We show the adoption rates of GPT-4, defined as the percentage of students in each country in the experiment who were offered access to GPT-4 and chose to use it. We cap the adoption rate on display between 8%-25%. We refer to these students as "adopters". The average adoption rate for all students, regardless of country, is 14.2%. We calculate the adoption rate directly for countries with more than 100 students enrolled in class. For countries that have less than 100 students in class, we merge them into their United Nations-defined global subregions. If such a subregion has more than 50 students in total, we calculate and plot the subregion adoption rate.

A.6 Full Description of Student Covariates

We report the basic statistics of the student population in our randomized control trial in Table 1. Here, we provide some additional information about them. Note that we are not reporting these distributions over the entire course's student population. They are only computed on the 5,831 students who were deemed active by week 1 and were included in our randomized control trial. In order to make sure all of the covariates we use are independent of the treatment (access to our custom ChatGPT), we only use either demographic information or a record of the student prior to the start of the experiment (week 3).

- **Application Score** (Mean=48.2, SD=11.4, Median (IQR)=47.2 (41.0-55.0), Max=103.0): This is an aggregate score computed by the course staff to rank student applications to enroll in this class. It is a weighted combination of many factors. We are not able to share what this equation is.
- **Age** (Mean=31.4, SD=10.4, Median (IQR)=29.0 (23.0-37.0), Max=84.0): Student self-reported age.
- **Gender** ("Female": 51.59%, "Male": 45.58%, "Non-Binary/Other/NA": 2.83%): Student self-reported gender. We provide five categories in our application.
- **English Fluency** (Mean=13.8, SD=2.5, Median (IQR)=14.0 (12.0-16.0), Max=20.0): This course iteration provided all lectures and materials exclusively in English. Applicants were categorized based on the English fluency demonstrated in their application materials. In future iterations, the course will offer materials in multiple languages to accommodate those who may not be fluent in English.

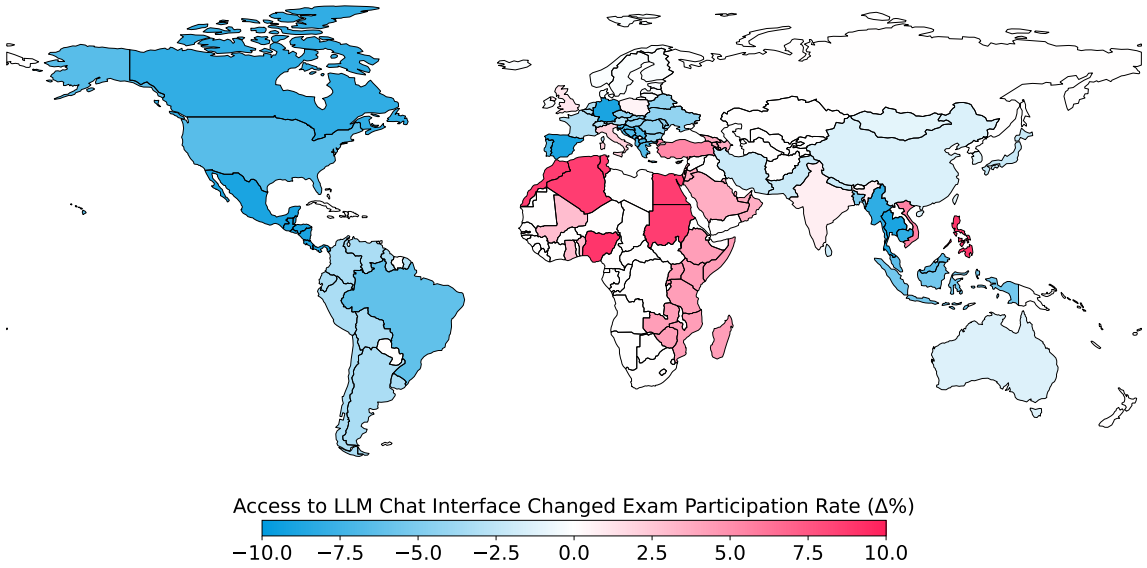


Fig. 10. We show the change in average exam participation between the experiment and control group by student country, which is the effect we studied in Section 3.1. The overall average change in exam participation is -4.4% across all countries. We capped the greatest positive or negative change to +10% and -10% for the plot. We calculate the change in participation rate directly for countries with more than 50 students enrolled in class. For countries that have less than 50 students in class, we merge them into their United Nations-defined global subregions. If such a subregion has more than 50 students in total, we calculate and plot the subregion adoption rate.

- **Application Effort** (Mean=4.0, SD=1.3, Median (IQR)=5.0 (3.0-5.0), Max=5.0): The course staff computed a score that captures how much effort an applicant spent on their application. The exact computation is kept confidential.

- **Coding Score** (Mean=8.2, SD=3.8, Median (IQR)=10.0 (10.0-10.0), Max=10.0): Applicants were required to complete a lesson and then tackle a programming exercise. Their performance on this exercise determined their coding score. In this scoring system, a lower score indicates a lower proficiency in programming, whereas a higher score demonstrates better programming skills.

- **Prior Experience** (Mean=-5.1, SD=4.4, Median (IQR)=-4.0 (-8.0–2.0), Max=0.0): This course is designed for those with little or no prior programming experience. During the admission process, course instructors evaluate applicants' previous programming knowledge to determine if they are overqualified. Applicants detail their prior programming experience in their applications, which is then used to categorize them. A lower, or more negative, score indicates more prior programming experience, which is disadvantageous for their overall evaluation as the course is designed for beginners. Whereas, a higher score indicates less prior programming experience, indicating that the applicant might be better suited for the course.

- **Friend Score** (Mean=4.6, SD=18.3, Median (IQR)=0.0 (0.0-0.0), Max=288.2): Student is asked to tell us if they know other people in the class. We compute a score based on this information. The more people they know, the higher their friend score will be.

- **Section Attendance** (Mean=1.7, SD=0.6, Median (IQR)=2.0 (1.0-2.0), Max=2.0): We calculate how many sections the student has attended prior to receiving access to our custom ChatGPT interface. At most, they could have attended 2 sections.
- **Country HDI** (Mean=0.8, SD=0.1, Median (IQR)=0.9 (0.8-0.9), Max=1.0): Students are asked to self-report their residing country. We map the self-reported country to the United Nations Human Development Index score. This score is a measure of a country's progress on key elements of human development, including health, education, and economic situation[7].

A.7 Diagnostic Exam Details

A diagnostic exam is administered near the end of the course. All students enrolled in the class received an email notifying them that the exam is available on May 26th, 2023, at 9:43 am EDT. The email was sent to 9,573 students. 7,084 students opened the email (74.4%), and 1,748 students clicked on the diagnostic exam link in the email (18.4%). The student will see a welcome message once they open the exam page:

> This diagnostic has **five** questions. Complete each question, to the best of your ability. When you are done, hit the blue Submit button. You can change questions using the numbers in the navbar above. You may go back and forth between questions.
> You have 3 hours to complete it from the time you hit start. The diagnostic is designed to only take 50 minutes. Time does not pause if you close the diagnostic and come back to it. For pedagogical purposes, we do allow you to run your programs, but we will not be giving you live feedback as to whether or not your program works.

Questions cover basic concepts of Python knowledge, control flow, arithmetic, and using Python canvas to animate objects. No official score is given to the students. The exam was graded with unit tests that verified each part of the student code, and a rubric system was used to create detailed feedback.

Diagnostic	Q1	Q2	Q3	Q4	Q5	Total
Rubric Items	12	12	14	29	6	73

We use a simple normalization rule to convert a student's diagnostic exam feedback to a score that has a range of 0 to 100. If a student has completed all exam questions and received 0 feedback, they get a score of 100. If a student did not submit a particular question, we count it as if the student received the maximal number of feedback from that question (i.e., if a student misses Q2, we would treat it as if they received 12 feedback). If a student gets s number of feedback in total, their score is $1 - s/73$. We verified our conversion with the course staff and obtained their approval.

A.8 Additional Details on Statistics

In the main result section, we report two p-values. One p-value is the family-wise error rate (FWER) controlled p-value, which we denote as P in the main text. We use Bonferroni correction to control for family-wise error rate. In addition, we report the unadjusted p-value per comparison, which we denote as "unadjusted P". In all the figures, we report the

[7] https://hdr.undp.org/data-center/human-development-index

significance level based on the Bonferroni-corrected P, which is calculated by multiplying 15 to unadjusted p-values. For GPT-4 usage patterns reported in Section 3.3, we follow the guideline that the p-values for logistic regression analysis do not need to be additionally adjusted.

For confidence interval, when we use a difference-in-means (DM) estimator for computing Δ between two groups without missing values (Section 3.1), we use the standard confidence interval calculation for the DM estimator discussed in [44]. When we have to deal with missing data, for both the DM estimator and local average treatment effect (LATE) estimator, we use the bias-corrected and accelerated (BCa) bootstrap interval (Section 3.2). The number of bootstrap samples we used is 1000, and we sampled with replacement.

A.9 Impute for Missingness (Regression Model)

All of these features are standardized, which means for feature X with empirical mean μ and standard deviation σ, we use $\tilde{X} = \frac{X-\mu}{\sigma}$. We first discuss how we conduct our model selection (hyperparameter search) and then discuss how we imputed for missing values. We conducted a search over a few model classes: linear regression, Ridge regression, Lasso regression, a 2-layer neural network with 128-dimension hidden size and tanh activation function, and a random forest regressor. All the models are implemented in sklearn. We first split the dataset into a training and a holdout set and used 5-fold cross-validation to select the best model from the training set. We then compute the mean-squared error (MSE) on the holdout test set. Because we do not have access to students who did not participate in the exam, we only train and evaluate our models on students who took the exam (on both the training and holdout set).

Model Name	MSE (Holdout set)
Linear	0.0370
Ridge	**0.0355**
Lasso	0.0356
Neural Net	0.0391
Random Forest	0.0397

Table 3. Model Selection: We use Ridge regression as our model class. And we use 5-fold cross-validation to choose the best model for imputation in our algorithm.

We use *cross-fitting* to impute for missing values. This procedure is inspired by works in econometrics and double machine learning [3, 11]. Cross-fitting allows us to use part of the dataset to estimate nuisance parameters of the imputation model and use the holdout dataset to estimate the parameter of interest for the causal effect estimation model. We refer readers to Page 22 in [44] for a more detailed discussion.

Algorithm 1: Cross-fitting based Impute for Missing Values

Input: Dataset $\mathcal{D}$, imputation algorithm $\mathcal{A}$, factor k

Output: Estimated Local Average Treatment Effect $\hat{\gamma}$

$\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_k$ = `Split` $(\mathcal{D})$

for $i \leftarrow 1...k$ **do**

  $\hat{D} = \emptyset$ ▷ Missing value imputed dataset

  $\hat{f}$ = `Cross-Validate` $(\mathcal{A}, \mathcal{I}_i)$ ▷ Nuisance model

  for $x \leftarrow \mathcal{I}_{j \neq i}$ **do**

    $\hat{y} = \hat{f}(x)$ ▷ Imputation

    $\hat{D} = \hat{D} \cup \{\hat{y}\}$

  end

  $\hat{\gamma}$ = `LATE` $(\hat{D})$ ▷ Causal Effect Estimation

end

$\hat{\gamma} = \frac{1}{k} \sum_{i=1}^{k} \hat{\gamma}_i$

return $\hat{\gamma}$

We describe our algorithm above, and we choose $k = 2$ for a 2-fold cross-fitting, a standard choice for most settings. The imputation algorithm $\mathcal{A}$ is pre-selected during the model selection phase (the algorithm with the lowest MSE on the holdout is chosen, which, in our case, is the Ridge regression model).

A.10 Logistic Regression Analysis of GPT Usage

In Section 3.3, we used a logistic regression model to understand what kind of students are more likely to become adopters of LLMs if we offer to them in a massive online class like ours. Each feature has been standardized. We report the coefficients in Table 4.

Variable	Coef.	Std. Err.	z	$P > \|z\|$	[0.025	0.975]
Intercept	-1.8312	0.049	-37.091	0.000	-1.928	-1.734
Application Score	-0.0238	0.063	-0.376	0.707	-0.148	0.100
Gender (Female)	0.0988	0.049	2.019	0.043	0.003	0.195
Age	0.1422	0.052	2.738	0.006	0.040	0.244
English Fluency	-0.0447	0.058	-0.765	0.444	-0.159	0.070
Application Effort	0.0482	0.061	0.797	0.426	-0.070	0.167
Coding Score	0.0445	0.054	0.831	0.406	-0.060	0.150
Friend Score	0.0456	0.046	0.984	0.325	-0.045	0.136
Section Attendance	0.2126	0.055	3.840	0.000	0.104	0.321
Country HDI	-0.1856	0.052	-3.561	0.000	-0.288	-0.083
Prior Experience	0.0093	0.055	0.169	0.866	-0.098	0.117

Table 4. Logistic regression for predicting student GPT usage. Pseudo R-squred is 0.01414. Number of observations is 3581.

A.11 Deployment Implementation Details

Student Access to GPT Interface within and outside the course. We do not record students' browsing histories and, unfortunately, do not know if they have accessed the publicly available GPT interface provided by OpenAI. However,

a survey conducted by a concurrent study on the same course asked if the students had used the ChatGPT interface during the course period. Only 2% of the students responded that they did.

Using GPT-4 to Cheat on the Diagnostic Exam. Despite our effort to give a stern warning to the students and let them know that their conversation with the GPT is monitored and visible to course staff, we conducted an analysis to see if the students copied and pasted exam questions into our custom interface. We did not find any evidence that the students used our custom interface to cheat. It is worth pointing out that, unlike a university course, this free online class does not incentivize students to cheat on exams – the course does not offer a letter grade, and the exam does not provide students a score – only feedback on how they did on each of the problems. The final course completion certificate only mentions if they had made an attempt on the exam.

Dear Students,

Happy Monday everyone!! Hope you had a great weekend. We have an announcement to make! Have you heard of GPT and Large Language Model? GPT is a new and advanced tool that generates human-like responses in a conversation. It can answer questions, provide helpful suggestions, and carry out a natural conversation with users. Can you chat with GPT in Code in Place? Yes!

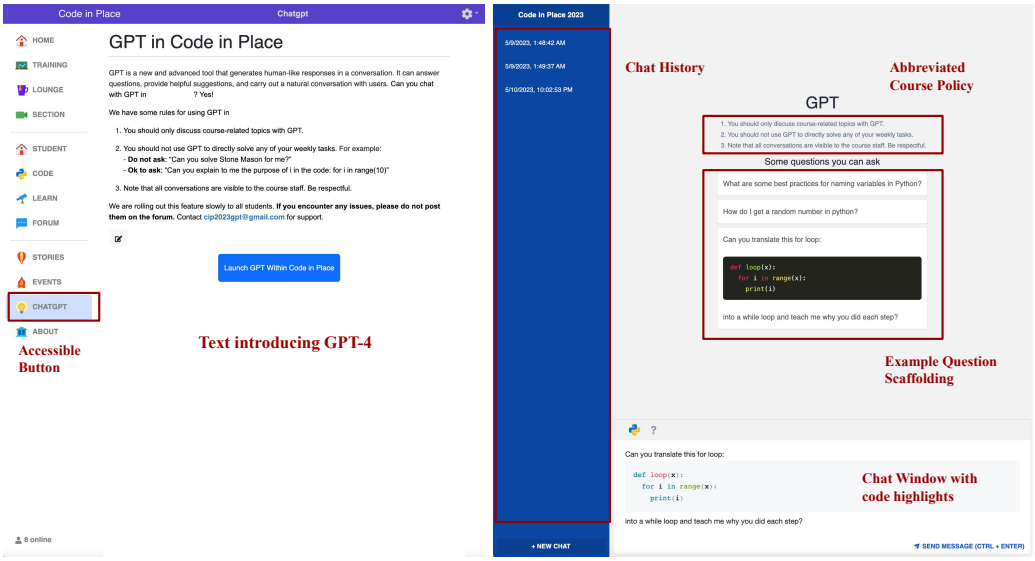
This is a new feature we are slowly rolling out to all students, and you have been selected for early access. You can find a button, "ChatGPT," on the sidebar of your course homepage. If you have any questions or need further assistance, please do not hesitate to reach out to our support team cip2023gpt@gmail.com. We're here to help!

Happy coding!

Best regards,

Code-in-Place Course Staff

Fig. 11. The email sent to alert the students who were granted access to GPT-4.



(a) (b)

Fig. 12. (a) The initial landing page of GPT-4. Students click on the button and have a chance to read through the text introducing GPT-4 before they start using GPT (See Figure 13); (b) The actual chat interface we built. Students can select a pre-generated example question or type in their own question in the chat window. The students who weren't in the treatment group do not see the lightbulb button on their home page.

GPT in Code in Place

GPT is a new and advanced tool that generates human-like responses in a conversation. It can answer questions, provide helpful suggestions, and carry out a natural conversation with users. Can you chat with GPT in Code in Place? Yes!

We have some rules for using GPT in Code in Place:

(1) You should only discuss course-related topics with GPT.
(2) You should not use GPT to directly solve any of your weekly tasks. For example:
    - **Do not ask**: "Can you solve Stone Mason for me?"
    - **Ok to ask**: "Can you explain to me the purpose of i in the code: for i in range(10)"

Note that all conversations are visible to the course staff. Be respectful.

Launch GPT Within Code-in-Place

Here is what we think you should know before using a tool like GPT to learn:

GPT may provide:

- **Instant Support**: You can ask questions and receive feedback instantly at any time of day.
- **Context-Based Understanding**: GPT can recall all topics within one conversation, so it can answer you based on what was said before in this interaction. This makes GPT's answers more helpful and personalized to your learning experience in each conversation.
- **Practice using new tools that may become standard use**: There are more and more tools trying to use GPT to make programming easier. By using GPT in Code in Place, you could get a sense of what this future might look like!

However, it could also harm your learning. Here are a few things to consider when using it: Depending too much on GPT can actually make learning how to program more difficult. When GPT does all the work, you might miss an opportunity to develop your understanding of important programming concepts. Another important thing to know is that GPT is sometimes wrong. You will need your strong programming skills to know if the code is correct, and it is your responsibility as a programmer to understand what your code does line by line. Lastly, GPT was trained on data from the entire internet. It may have strong biases and could give harmful or unsettling responses, especially when used outside of the context of the course.

In general, platforms like GPT are powerful but experimental tools. They are not intended as a substitute for professional teaching advice. If you choose to use them, do so at your own risk and with due diligence because they might generate incorrect results.

Fig. 13. The paragraphs we display for the student before they enter a custom-built GPT interface.