Generative AI and Distributed Work: Evidence from Open Source Software

Manuel Hoffmann* Sam Boysel* Frank Nagle* Sida Peng† Kevin Xu‖

*Harvard Business School, Harvard University

†Microsoft Corporation

‖GitHub Inc.

This version: *March 21, 2024*

Please do not circulate or cite without permission!

Abstract: Artificial intelligence (AI) is at the beginning of a technological evolution that is slowly and steadily reaching all sectors of the economy. However, due to the novelty and breadth of AI, we still know relatively little about the labor impacts of AI, including both the effects on strategic human capital decisions as well as decentralized, often voluntary, contributions to public goods. Therefore, it imperative to understand how different work processes can be affected by this transformational technology. Using the context of open source software, we study the individual consequences of using a generative AI tool designed to help developers work more efficiently and effectively. We exploit a natural experiment arising from the deployment of GitHub Copilot, a generative programming AI released by the social coding platform in June 2022. We study the impact of generative AI for a whole year on millions of work activities by exploiting threshold eligibility requirements through a quasi-experimental regression discontinuity approach. We find that after having access to Copilot, top maintainers are able to code more and spend less time on project management tasks. Similar to other modern technologies, generative AI also induces maintainers to substitute away from high cost interactions (i.e. comments) towards low cost interactions (i.e. emoticons). Overall, a back-of-the-envelope calculation suggests that around thirty percent of the efficiencies arising from the programming AI are reallocated onto the platform.

JEL-Classification: H4, O3, J0

Keywords: Generative Artificial intelligence, Digital Work, Digital Economy, Open Source Software

Acknowledgement: The authors are grateful for financial and administrative support from GitHub and, in particular, from Peter Cihon. The authors are also indebted for comments by seminar participants at the research seminars at Harvard LISH in Cambridge, MA.

1 Introduction

Throughout human history, there have been a handful of technological innovations that fundamentally shift how the economy works. The printing press, internal combustion engine, and computers are oft-cited examples of this phenomenon. Although artificial intelligence (AI) technology has existed for some time, many have argued that recent advances may push it into this elite category of technologies that alter the course of history. However, if AI — broadly defined as the use of computers and machines to mimic human intelligence – is destined to have such a substantial impact, we are likely still at the beginning of this technological revolution that is slowly and steadily reaching all sectors of the economy. Importantly, the highest economic impact of AI is predicted to be on productivity growth through the labor market (Bughin and Manyika 2018, Sachs 2023). However, due to the novelty and breadth of AI, research is only starting to scratch the surface of understanding the impact of AI on the labor market. This is particularly true of generative AI — a subset of AI built on large-language machine-learning models (LLMs) — which exploded on to the scene in 2022 and currently represents the cutting-edge of AI. These models, including OpenAI's GPT4, Google's Gemini[1], Meta's LLaMa, and numerous others, are trained on massive, Internet-scale databases and use billions of parameters to construct a probabilistic model that predicts what the next word in an answer to a prompt from a user should be. These models can also be trained on datasets that are more focused on specific contexts — e.g., health, finance, customer service, etc. In this paper, we consider one such context-specific generative AI system, the software development tool GitHub Copilot, and seek to answer the research questions of how access to this tool changes labor activities in a distributed work context and who benefits most from using it.

Answering such questions is of the utmost importance to better understand how distributed digital work processes are affected when AI is introduced and how it can assist in workforce optimization as a goal for strategic human capital management (Wright, Coff, and Moliterno 2014). It is likely that AI will impact nearly all categories of labor, so focusing on a category like software

[1]Formerly known as Bard.

developers – who are likely to embrace such a tool earlier than other types of workers – can yield insights that apply to a broader set of workers. Although some early studies on generative AI have shown positive high-level productivity impacts (Brynjolfsson, Li, and Raymond 2023, Dohmke, Iansiti, and Richards 2023, Noy and Zhang 2023 Peng et al. 2023), it is less clear what the mechanisms behind these improvements are. Does the use of generative AI lead users to focus on particular types of tasks that lead to those productivity improvements? If so, which tasks? How exactly does the work process change when using generative AI? How do social interactions change when this novel technology is introduced in the distributed work context? How are developers impacted by AI? And overall, how is one of the most important public goods, open source software, affected by generative AI? Understanding these impacts will allow firms to better adjust their labor hiring policies as well as their upskilling and reskilling efforts (Tamayo et al. 2023, and for policy makers and managers to better anticipate the future of the labor market (and potentially offer work training programs to allow for the best possible outcomes for the entire labor force or to strategically hire workers with the right skillset at the right point in time as managers). Finally, our context allows us to be the first (to our knowledge) to also consider the impact of whether a generative AI tool is freely available or if users must pay to access it. Understanding this dichotomy will have substantial implications for the roll out of future AI tools of all types.

We study the use of generative AI in the context of GitHub's Copilot – a software development focused generative AI tool based on OpenAI's Generative Pre-trained Transformer (GPT) models, which also underlie ChatGPT. Copilot was developed jointly by OpenAI (an AI research organization) and GitHub, a subsidiary of Microsoft that offers cloud-based software development and version control services and is also the world's largest host of open source software (OSS) projects. Copilot was trained on vast amounts of publicly available software code, including that hosted as OSS on GitHub itself. Originally released to a subset of users as a technical preview starting in June 2021, it became more broadly available as a full product in June 2022. As part of this roll out process, GitHub offered free access to a subset of users, including students and maintainers of top OSS projects hosted on GitHub. This roll-out process assists us in identifying the causal effects of

access to Copilot.

We exploit aspects of the general access launch of Github Copilot to the broader public. In particular, a segment of developers who maintain the top OSS project on GitHub receive free Copilot access. Eligibility is determined by an internal ranking known to GitHub, and not the individual developers, allowing us to make credible claims about non-manipulation to establish a causal identification strategy to recover the effect of generative AI on developer activity. In the most basic version, maintainers with a ranking below 0 become eligible to receive free access to Copilot and those above do not. We use a panel of 175,990 distinct maintainers observed weekly from July 2022 through July 2023, which results in millions of maintainer-week observations for activity levels in public GitHub repositories and Copilot usage.

Within the data set of top maintainers, we find those with access to Copilot AI during the general access period are engaging in more coding activities while reducing time in activities required for project management. We also find that the quality of social interactions drops with the new technology. The developers with access are more likely to use low-quality, low-cost interaction methods such as emoticons, and less likely to use high-quality, high-cost means of communication via comments and text. Overall activities on the platform are dropping which indicates that the efficiency and time gains from coding for the individual maintainer are too a large proportion allocated outside of the platform.

Our results contribute to a growing literature on the productivity impacts of AI in important ways. Early work in this area posited general productivity gains (Agrawal, Gans, and Goldfarb 2019 Corrado, Haskel, and Jona-Lasinio 2021, Raj and Seamans 2018), but that the gains may not be evenly distributed (Brynjolfsson, Rock, and Syverson 2018; Furman and Seamans 2019). Empirical work confirmed such predictions and found wide-ranging productivity benefits to using AI, at both the firm level (e.g., Czarnitzki, Fernández, and Rammer 2023) and the individual level (e.g., Fügener et al. 2022). Our work is consistent with this prior research but adds additional nuance to the labor augmenting technical change literature (Acemoglu 2003) as we provide one of the largest natural experiments of generative AI and it's impact on granular measured distributed

work processes for over a one year time horizon ”in the wild”.

In addition to identifying an increase in productivity from AI adopters, we also find changes in digital work processes. We show that when software developers become more productive by using AI, they spend their newly found free time more on coding activities and less on auxiliary project management activities that require lengthy human interactions with other maintainers. This is a sign that the maintainers intensify contributions to the public goods of open source software through AI. It is also consistent with reducing frictions during the problem solving process of work and a change in the way voluntary workers interact with each other on the platform. We complement the current literature that leverages IT and consultancy chat support AIs and measures them through experimentation (Brynjolfsson, Li, and Raymond 2023, Dell'Acqua et al. 2023) by investigating changes in work and social interaction processes over a year after the introduction of the programming LLM. While it has been recognized that some new technology may reduce social interactions, including social media and the internet in general (Allcott et al. 2020, Mosquera et al. 2020, Kraut et al. 1998), our findings highlight that a reduction in high quality social interactions may be accompanied by this novel technology as well. This finding has important implications for the strategic human capital literature (Wright, Coff, and Moliterno 2014) since it implies that managers could strategically leverage programs at work that heighten social interactions to accompany the proliferation of generative AI allowing for an offset of a potential drop in social interactions. In aggregate, our results build upon existing literature to answer questions related to how access to generative AI changes labor and social activities.

Beyond the identification of causal effects that generative AI has on decentralized work, our results suggest important implications for the future of OSS. OSS has become an increasingly critical part of the modern economy, such that 96% of corporate codebases include some OSS (Synopsys 2023). Further, recent studies estimate the value of OSS to be on the order of billions of dollars (Blind et al. 2021, Robbins et al. 2021) or trillions of dollars when accounting for usage (Hoffmann, Nagle, and Zhou 2024). Additionally, firm usage of, and contribution to, OSS has important implications for firm productivity (Nagle 2018, Nagle 2019) and entrepreneurial activity

(Wright, Nagle, and Greenstein 2023). However, despite the importance of OSS, many critical projects are under-resourced (Nagle et al. 2020b) as numerous firms free-ride on the efforts of others without giving back (Lifshitz-Assaf and Nagle 2021) leaving volunteer maintainers burnt out and overwhelmed (Raman et al. 2020). As our results show, generative AI may offer a solution to help address these concerns and allow top maintainers to more easily contribute to the common good by solving issues faster. Prior research has shown that OSS developers and maintainers generally contribute to OSS because it gives them a creative outlet and they do not want to spend their time on managerial tasks like security and documentation (Nagle et al. 2020a). AI-powered tools may make it easier to quickly address such managerial tasks, so developers can spend time in a manner they prefer, while still ensuring the security, stability, and usability of OSS.

The remainder of this paper proceeds as follows. In Section 2 we discuss the environment within which the generative AI is used. In Section 3 we characterize our empirical setting and discuss the construction of our sample. We home into the set of developers that are obtaining Copilot eligibility for free via an internal score from GitHub and present our estimation strategy in Section 4. In a next step we present our results using a regression discontinuity design (Section 5) while exploring mechanisms on the determinants of Gen AI and its consequences. We discuss the limitations, implications and a back-of-the-envelope calculation in Section 6. Section 7 concludes.

2 Institutional Background

2.1 The GitHub Platform

GitHub was established in 2008 and is the world's largest hub for OSS development.[2] It is a "social coding" platform that offers cloud-based software development and version control services. Importantly, it is specifically designed for dispersed teams to collaborate on software de-

[2]According to `https://6sense.com/tech/source-code-management/github-market-share` 6Sense GitHub had a market share of 78.81% on March 18, 2024 followed by GitKraken, and SourceTree and with around 8.70% and 8.58%. Further, 93% of individuals are using Git as a version control system which underlies GitHub.

velopment projects, and therefore chronicles all activities performed on the system to ensure any contributor can observe all prior activity. Activity on the GitHub platform can therefore provide the researcher unique and granular insights into patterns of distributed work, which are increasingly becoming the norm in all areas of knowledge work. Furthermore, we can observe the decentralized production of OSS as a public good. Although the details can be quite intricate, the primary workflow of a GitHub contributor is straightforward.

A user who wants to start a new project creates a repository and then writes their code within this repository.[3] Alternatively, a user may "fork" another repository, which entails copying everything in another repository to have the same exact information, but allow them to take the project into a different direction than the primary repository. When the user modifies project code in a local copy of the repository on their machine, these changes to the codebase are condensed into a "commit" that attributes authorship to a user. Uploading these commits from the local repository to the remote GitHub repository is called a "push". GitHub also has popularized the "pull request" paradigm for OSS contribution in which users contribute to projects by requesting the project's maintainer to integrate their proposed changes. For example, if user A maintains a repository and user B wants to add a new feature to it but does not have permission to edit the code directly, user B can issue a "pull request" which includes the suggested changes as a sequence of commits to a fork of the original project. If user A accepts the proposed change, the changes proposed by user B are integrated or merged into the codebase. Finally, any user can report an "issue" for a particular repository (e.g., identifying a bug or asking for a new feature) and when the issue is addressed, it is considered "closed."

[3] A project "repository" is a focal point for collaboration over a particular codebase. While a repository technically refers to the collection of source code files for the project, the GitHub platform adds important social and project management features to a repository's profile, including a detailed version control viewer, a forum to raise issues and discussions, and a formal contribution system based on "pull requests". Throughout this paper, we will refer to a "repository", "project", or "codebase" somewhat interchangeably.

2.2 The GitHub Copilot AI

The primary focus of this paper is to study the introduction of the generative AI coding tool GitHub Copilot. Copilot was built collaboratively between OpenAI and Microsoft/GitHub and was based on models similar to those that underlie ChatGPT.[4] The version of the Copilot AI under consideration in this study is based on the Generative Pre-trained Transformer 3 series (GPT-3) from Open AI. Copilot can be used by programmers to obtain code-snippets for work while they are coding, and it allows them to easily integrate those snippets into their coding projects. As a large language model, Copilot operates on the idea of next Word prediction and instead of a text completion tool it is a code completion tool with the idea to assist programmers to code faster, solve problems more quickly, and learn code that they previously did not know. Figure A1 provides an example of how the generative AI Copilot was used to create a summation function.

— Figure A2 about here —

Figure A2 shows the timeline of the introduction of the AI for programmers. GitHub's Copilot was first launched on June 29, 2021, as a "technical preview" (TP), and then fully launched for "general access" (GA) on June 21, 2022. GitHub users can access Copilot through several pathways. During the technical preview time, every individual was eligible to access Copilot for free. During the general access time, individuals could access Copilot by obtaining a free trial for 60 days. After the trial period, they must pay $10 per month (or $100 dollars per year) for continued access. For some individuals, GitHub provides free access after the technical preview period. Students can obtain Copilot and pay $0 per month. Similarly, GitHub rewards top maintainers with free Copilot access based on an internal eligibility score. Finally, individuals can obtain free access through their company since February 14, 2023. Companies can sponsor their employee access to Copilot with the employer paying $19 per month per employee.

In this study, we leverage the internal eligibility score from GitHub which determines that a sub-population of top maintainers receive complimentary Copilot access which we leverage as a

[4]GitHub Copilot is not the same as MSFT Copilot which was launched after GitHub Copilot. In June 2023, 92% of programmers have used some coding AI tool according to a GitHub programmer survey.

natural experiment. GitHub ranks selected repositories based on an internal score that is unknown to the community. Any individual who has been added as a "collaborator" to those repositories is considered a maintainer and is therefore potentially eligible for free access to GitHub Copilot.[5] Maintainers for projects below a given threshold of the eligibility score are considered top maintainers and are granted complimentary Copilot access for one year. After a year, GitHub will check if you are still eligible through the top maintainer channel and continue granting access if that is the case.

A key feature of this program is that the exact scoring system is largely unknown to maintainers. Public discussions between users suggests there exists a considerable degree of uncertainty over what makes a given project eligible.[6] As GitHub does not reveal the exact composition and ingredients of the score, the maintainers can only guess. This vagueness in the composition and ingredients of the eligibility score lends additional credibility of our identification strategy as it is virtually impossible to manipulate the ranking as a maintainer. Moreover, GitHub did not engage in any additional messaging to communicate the eligibility status to maintainers. Each maintainer has to check whether they are eligible to access the AI for free when they apply for Copilot.

3 Data

We use a mixture of openly available and proprietary data from GitHub to understand the effects of the AI Copilot program. The general activity data is publicly available on GitHub while any AI interactions of the users with Copilot is proprietary. In collaboration with GitHub, we link the public platform activity to Copilot usage for a set of pseudonymized users to form a panel of maintainer-week observations. From the granular activities data observed on the GitHub platform, we develop a set of four broader classifications of maintainer activities essential to collaborative software development. Precise definitions for each classification can be found in Appendix B.

[5]The nominal owner of the repository, i.e. the GitHub user who created the repository, can add collaborators in the repository's settings page.

[6]See YCombinator and GitHub on the discussions.

We consider the extent to which maintainers engage in activities related to (1) coding, (2) project management, (3) communicating via written comments, and (4) communicating with short form reactions. We detail these categories in turn in the following sections.

First, coding includes maintainer activities that form the core of OSS contribution: pushing commits from local repositories to GitHub, forking existing projects to begin new development directions, and submitting pull requests for other maintainers to integrate proposed changes. These activities characterize the more technical process of writing lines of software code. Second, one can think of all other activities as non-coding activities. We classify an important subset of these non-coding activities as "project management", process-oriented community interactions that seek to progress project development and require a more than purely technical skill set. A critical component of OSS collaboration is engagement within the repository's "issues board". It is here where project maintainers engage with the wider community to assist with software issues, introduce new ideas, and discuss longer term project objectives.[7] Other activities related to project management include the creation of project boards, which assist maintainers with project organization and "road-mapping", and systematically reviewing the proposed contributions stemming from pull requests. Third, we consider activity related to platform communication in two distinct categories: low-cost, low-quality interactions versus relatively high-cost, high-quality digital social interactions. Written comments occur in project discussions, issue sections, and during the pull request review process. They are commonly social interactions through sentences that require a little bit more thought on the sender side. On the other hand, users can also communicate via quick reactions, emoticon responses including images of "thumbs up", "thumbs down", or hearts. Finally, we aggregate all items to obtain a measure of the total activities for each maintainer on the platform.

Table A1 shows univariate statistics for the Copilot AI treatment, the work activities and the ranking.[8] We use a sample of active maintainers who are a collaborator on a repository that receives an eligibility score ranking during the general access period. To leverage meaningful variation for

[7]For example, maintainers can open issues themselves, assign other maintainers to investigate certain issues, give issues labels for organization, and close the issue itself, thereby signaling that it is no longer an outstanding concern.

[8]More granular statistics on work items can be found in Table A2.

the sub-sample of maintainers who received access for the first time, we exclude any maintainers with some Copilot experience during the technical preview. All individuals in our top maintainer dataset engaged in at least one of the four activities mentioned above during our time window (and thus are considered active users). This leads to a sample of 175,990 unique "top maintainers" (those who are maintainers of OSS projects that are amongst the most popular and widely used on GitHub). In this dataset, the average user, used Copilot for approximately 10 days and on average 19% of the maintainer used Copilot at some point within the general access period from July 2022 to July 2023. During each week, usage was relatively low, with 3% weekly exposure and 0.21 days of Copilot adoption per week. For the four work activities, coding, project management, commenting, and reactions, we create a cumulative share measure defined as the cumulative sum of the activity over the maintainers overall total cumulative activity.[9] By conditioning on individual activity levels overall, we are better able to directly compare time allocation across maintainers and identify any potential reallocation in response to AI adoption. The average maintainer spends 44% of their time coding, 24% on issues, 9% on commenting and 5% on reactions.[10]. Finally, our measure for our natural experiment is the best (minimum) ranking across eligible repositories that maintainers are connected to. The average maintainer rank is around 737 which is beyond the threshold ranking of 0 and the ranking distribution is right skewed.

4 Methodology

4.1 Correlations

Our methodological approach begins with simply comparing the activity levels of maintainers that use Copilot relative to those who do not. We are interested in the association between AI adoption and work activities during the general access period, controlling for two particular sources

[9]Specifically, use the cumulative share version $\frac{\sum_{s \leq t} Activity_{it}}{\sum_{s \leq t} TotalActivities_{it}}$.

[10]The sum does not add up to one, because total activity contains more items then the four activities.

of unobservable characteristics. These correlations are estimated with a simple linear model:

$$Activity_{it} = \gamma_0 + \gamma_1 AIAdoption_{it} + \iota_i + \iota_t + \iota_{it}$$

where $Activity_{it}$ captures the outcome variable of interest and $AIAdoption_{it}$ the treatment variable of Copilot usage. Hence γ_1 captures the correlation between activity and adoption after controlling for maintainer fixed effects ι_i as well as time-varying shocks common to all maintainers with week fixed effects ι_t. However, programmers are a group of individuals that are learning substantially over time. It may therefore be the case that unobserved characteristics varying across both maintainers and time may be related to platform activity levels and AI adoption. Therefore it is likely the case that neither of these purely correlational approaches are sufficient to recover the true causal effects generative AI.

4.2 Natural Experiment

To more credibly establish a causal effect of Copilot on OSS activity, we leverage a natural experiment in a next step. GitHub awarded free Copilot access to maintainers of the most popular public repositories according to an internal ranking. The top maintainer program was launched at the beginning of the general access period and was at least partially motivated by a desire to support critical OSS infrastructure. GitHub determines that a certain number of repositories are worthy of ranking and those repositories receive a score. The exact calculation of the score that determines which repositories receive rankings is not publicly disclosed by GitHub.

Since the ranking occurs at the project level, we assign each maintainer a weekly rank from the projects for which they are designated as collaborators. Specifically, the rank assigned to each maintainer is the running best (i.e. cumulative minimum) rank received across all repositories through the current observation period.[11] Maintainers who receive a ranking below 0 are eligible for free access to Generative AI Copilot while others are not. Maintainers only become aware

[11] Ranks are recalculated each week beginning three weeks after GA.

of their eligibility status when applying for Copilot access. Since the maintainers have to check themselves whether they are eligible for free Copilot access in contrast to receiving a message from the platform, we expect the adoption of the AI to be less sharp than it would have been under alternative program implementations. However, we believe that the regression discontinuity design is the best methodology in this environment allowing us to not only test its underlying assumptions while coming close to a randomized control trial from an internal validity perspective but also being able to study developers in their natural experiment over a long time horizon.

This setting allows us to employ a regression discontinuity design (RDD) which in practice can be interpreted as a localized randomized control trial close to the threshold of AI eligibility. Whether a maintainer receives a score just above or just below the threshold can be considered to be as-good-as-random. Fortunately, the regression discontinuity design allows us to test these assumptions. For example, it is important to have no manipulation across the ranking variable for the RDD to be valid.[12] We have checked the credibility of this assumption through frequency plots and statistical tests of the running variable as well as covariate checks along the running variable and find evidence that is consistent with this assumption.[13]

To estimate the first stage of Copilot adoption, we employ the following model:

$$Copilot_{it} = \alpha_0 + \alpha_1 \mathbb{1}Eligible_{it} + \alpha_2 Ranking_{it} + \alpha_3 Eligible_{it} \times Ranking_{it} + \epsilon_{it}$$

where $Ranking_{it}$ is the cumulative minimum Ranking across each eligible repository of a maintainer. $Eligible_{it}$ is defined as $\mathbb{1}(Ranking < 0)$ while considering a bandwidth close to the eligibility threshold of $h \in [100, 100]$.

By construction, a change in access and, therefore, ultimately an exogenous change in adoption of GitHub Copilot is the only policy intervention operating on a maintainer as they cross the ranking threshold of 0. We can therefore interpret any activity change across this threshold as caused

[12] In other words, neither the maintainer can manipulate the score to receive Copilot nor GitHub should manipulate the score to the benefit of some maintainers.

[13] For example, Figure A3 shows a the distribution of the ranking without bunching at the threshold of 0.

by Copilot through the following intent-to-treat (ITT) estimation:

$$Activity_{it} = \beta_0 + \beta_1 \mathbb{1} Eligible_{it} + \beta_2 Ranking_{it} + \beta_3 Eligible_{it} \times Ranking_{it} + \epsilon_{it}$$

where β_1 shows the causal effect of crossing the threshold — and through it, increased copilot access — on the work activity of interest.

5 Main Results

5.1 Correlations of AI with Activities

To get an understanding about potential biases and selection, we first estimate the associations of AI adoption with our activity categories of interest.

— Table 1 about here —

Table 1 shows the associations of Copilot adoption with open source relevant work activities. After controlling for maintainer and weekly fixed effects, we find very small positive correlations for all activities. The project management, commenting, and total activity correlations are highly significant while coding and reactions are insignificant. The positive correlation with Copilot eligibility and total activity in particular suggests the potential for positive selection: maintainers that are more active overall are also more likely to adopt Copilot. Taking those estimates at face-value has the potential to overestimate the true effect of generative AI and therefore, we leverage the regression discontinuity design and the internal GitHub ranking as a natural experiment.

5.2 AI Adoption

As a society, we are at the beginning of the S-shaped adoption curve of generative AI overall. This is not any different for the programming AI Copilot from GitHub. 19 percent of maintainers

used Copilot at least once (see Table A1). Further, GitHub was not engaging in any kind of advertising informing the top maintainers about their eligibility for free Copilot access. Top maintainers had to visit the GitHub Copilot website and check by themselves whether they were eligible or not. Hence, we expect the differences in adoption not to be very large. Fortunately, due to the large scale of our data based on the weekly level, we are able to identify even small changes.

— Figure 1 about here —

Figure 1 shows how crossing the threshold of 0 from left to right alters programming AI adoption. Maintainers that are below the threshold of 0 are eligible to receive Copilot for free with certainty while those above are not eligible for free through this channel. We find a significant drop of AI adoption based on the total number of days that AI has been used over the sample period when we cross the threshold, which implies that maintainers with free access are more likely to use the generative AI on the social coding platform.[14]

— Table 2 about here —

Table 2 shows the exact coefficients on generative AI adoption of the top maintainer population across those who are eligible to access Copilot for a prize of zero dollars (i.e. ranking below 0) and those who are ineligible to access Copilot through the top maintainer channel (i.e. ranking above 0). The first two columns show estimates of the cross-section and the last two columns estimates of the panel. We find an increase of AI adoption independent of the measure that is employed. For the cross-section, we observe an increase of 5.71 days (164% relative to the baseline) overall in AI adoption and an increase of 12.1 pp (146% relative to the baseline) of maintainers that ever adopted Copilot. For the panel, we find an increase of 0.08 days per week (55% relative to the baseline) and an exposure increase of 1.1 pp. (50% relative to the baseline).

Despite not widely advertising Copilot maintainer access, we find very strong effects ranging from 50 to 164% adoption, depending on the definition of the first stage. Independent of the

[14]The first stage is robust to a polynomial of degree two (see Figure C1

definition of AI adoption, we find that our first stage is relevant with an F-value substantially beyond the recommended threshold of 10 where the cross-section F-values range from 32 to 65 and the panel from 100 to 121.

5.3 Consequences of the Copilot Generative AI

In this step, we will analyze how generative AI causally changes work behavior in the digital world, namely the distributed work behavior of programmers that contribute to the public good of open source software.

— Table 3 about here —

Table 3 shows the intent-to-treat estimates of the Copilot generative AI on work activities.[15] Panel A displays the reduced form effect of AI on activities that relate directly to programmer work. We find that coding activities increase by 2.6 pp (6% relative to the baseline) while project management drops by 11.7 pp (33% relative to the baseline). This indicates that overall coding activity is increasing due to the availability of Copilot. In line with Copilot being a teacher and problem solver tool well suited to address problems or inquiries that would typically arise in the repository's issues page, we observe a substantial drop in the average maintainers relative allocation towards activities related to project management. It implies that maintainers are less likely to seek out assistance from other humans. In this spirit, generative AI is helping the public good of open source programming since maintainers have to solve fewer issues and they can focus on the core work: writing software code.

Panel B displays the reduced form effect of AI on social interactions that are either low or high cost interactions. Comments are relatively high cost interactions since it takes a substantial amount of time to create a comment while emoticon interactions, labelled reactions, are substantially lower cost. We find that being eligible to Copilot reduces the likelihood to engage in commenting by

[15]The results are robust to a different bandwidth selection and functional form as shown in Figure C2 and Figure C3

3.1 pp (30% relative to the baseline) while it increases the likelihood of engaging in quick low-cost reactions by 3.9 pp (130% relative to the baseline). This is consistent with the technology substituting for a type of human need and consistent with many new technologies: the coding AI arguably reduces the level of high-quality social interactions on the platform while substituting for more low quality interactions.

Panel C displays the reduced form effect of AI on overall activities. This can be interpreted as a conservative measure of efficiency if Copilot is useful to increase the speed of coding and individuals are not using their new won time to substitute their time now on the platform. Indeed, we find a drop of nearly 5 activities (39% relative to the baseline) for individuals that are eligible for the coding AI relative to those who are not. If we are assuming that the estimate from the literature of a 55% drop in time required for coding using Copilot Peng et al. 2023 applies to our maintainer population, then this drop implies a back-of-the-envelope efficiency gain of 29% that is re-allocated on the platform while the rest is reallocated outside of the platform.

— Figure 2 about here —

Figure 2 shows the dynamic impact of a heightened propensity to use Copilot for free up to four quarters post general access. We find that while coefficients are qualitatively quite similar across each period, overall, the effect of Copilot on coding and project management are stronger in the quarter immediately following general access. It indicates that the benefits of accessing Copilot seem to be arising quite instantaneously and after some experimentation of the developers with it, the impacts are stable up to a year across the population of top maintainers.

6 Discussion

The benefit of this natural experiment is that we are able to observe activities that are driven by AI for a long period of time of over a year while using very granular level activty information. On the other hand, a limitation of this study is that we are not observing the exact code that individuals are programming and we are not observing how individuals are exactly using Copilot. While

randomized controlled trials can focus on this aspect, they often have to restrict themselves to a small set of individuals. In contrast, we are able to study the long-run effects of AI on many daily work activities for a large group of programmers that are extremely important contributors of the public good open source software.

Another limitation of this study is that we are only observing contributions to public repositories. Most of the learning with AI may happen in private repositories and as such, we are likely substantially underestimating the impact that generative AI has on coding behavior as well as our measure of efficiency. Based on our data, we find that 45.51% of Copilot usage happens in weeks where no public work activity is observed and GitHub has confirmed that the Copilot activity must be related to private repository activity. Further, there is likely an even higher fraction of private activity since this activity - while not measurable - is also likely occurring during weeks where public activities are occuring. As such, we expect that around 50 percent of the activity of highly active programmers happens on private and 50 percent on public repositories. Hence, a reasonable estimate for the overall private-public effort could be doubling our estimates conditional on private behavior being affected in similar ways to publicly observable behavior. A further interesting feature of the open source software platform is that private repositories can become public over time, which implies that not only will learning from private repositories spill over to public ones due to maintainers programming with the AI in public repositories but also AI generated code may appear at an increased rate over time in the public sphere.

7 Conclusion

This study seeks to shine light on the importance of AI, and in particular generative AI and it's consequences on distributed work. Going beyond the first level understanding of whether or not it increases productivity, we dig deeper to understand how it changes the digital work processes of adopters. We find that top developers of open source software are coding more and are engaging less in project management tasks. The decrease in project management activities is largely driven

by less time spent addressing problems arising in project issue boards. It implies that those problems might have been solved by generative AI instead of a human, reducing overall collaboration frictions and other transaction costs characteristic of distributed work. Similar to other novel technologies, we find that relatively high cost-high quality interactions drop while low-cost low quality interactions increase. It implies that we have to be cautious of the way that AI impacts interactions at the work place and that it has a lot of potential to increase efficiency but also might require additional policies that compensate for a loss of social interactions. Finally, we find that most of the efficiency gains from GitHub Copilot materialize themselves outside rather than on the platform.

Overall, our results are amongst the first that are able to illuminate the deeper level changes in decentralized work processes instigated by AI over a long time period with very granular level information using a natural experiment. Furthermore, our study yields early insight into how generative AI shapes the voluntary private provision of critical digital public goods infrastructure. Indeed the scope for positive externalities inherent to the public goods setting suggests the efficiency gains that arise from introducing generative AI to the OSS production process can generate far-reaching spillover benefits to downstream users. We believe they will help managers and policy makers better understand the nuances of this nascent yet transformational technology.

References

Acemoglu, Daron. 2003. "Labor-and capital-augmenting technical change." *Journal of the European Economic Association* 1 (1):1–37.

Agrawal, Ajay, Joshua Gans, and Avi Goldfarb. 2019. "Economic policy for artificial intelligence." *Innovation policy and the economy* 19 (1):139–159.

Allcott, Hunt, Luca Braghieri, Sarah Eichmeyer, and Matthew Gentzkow. 2020. "The welfare effects of social media." *American Economic Review* 110 (3):629–676.

Blind, Knut, Mirko Böhm, Paula Grzegorzewska, Andrew Katz, Sachiko Muto, Sivan Pätsch, and Torben Schubert. 2021. "The impact of Open Source Software and Hardware on technological independence, competitiveness and innovation in the EU economy." *Final Study Report. European Commission, Brussels, doi* 10:430161.

Brynjolfsson, Erik, Danielle Li, and Lindsey R Raymond. 2023. "Generative AI at work." Tech. rep., National Bureau of Economic Research.

Brynjolfsson, Erik, Daniel Rock, and Chad Syverson. 2018. "Artificial intelligence and the modern productivity paradox: A clash of expectations and statistics." In *The economics of artificial intelligence: An agenda*. University of Chicago Press, 23–57.

Bughin, Jaques and J Manyika. 2018. "The promise and challenge of the age of artificial intelligence." *McKinsey Global Institute, May. https://www. mckinsey. it/idee/the-promise-and-challenge-of-the-age-ofartificial-intelligence* 30 (10).

Corrado, Carol, Jonathan Haskel, and Cecilia Jona-Lasinio. 2021. "Artificial intelligence and productivity: an intangible assets approach." *Oxford Review of Economic Policy* 37 (3):435–458.

Czarnitzki, Dirk, Gastón P Fernández, and Christian Rammer. 2023. "Artificial intelligence and firm-level productivity." *Journal of Economic Behavior & Organization* 211:188–205.

Dell'Acqua, Fabrizio, Edward McFowland, Ethan R Mollick, Hila Lifshitz-Assaf, Katherine Kellogg, Saran Rajendran, Lisa Krayer, François Candelon, and Karim R Lakhani. 2023. "Navigating the jagged technological frontier: Field experimental evidence of the effects of AI on knowledge worker productivity and quality." *Harvard Business School Technology & Operations Mgt. Unit Working Paper* 24 (013).

Dohmke, Thomas, Marco Iansiti, and Greg Richards. 2023. "Sea Change in Software Development: Economic and Productivity Analysis of the AI-Powered Developer Lifecycle." *arXiv preprint arXiv:2306.15033* .

Fügener, Andreas, Jörn Grahl, Alok Gupta, and Wolfgang Ketter. 2022. "Cognitive challenges in human–artificial intelligence collaboration: Investigating the path toward productive delegation." *Information Systems Research* 33 (2):678–696.

Furman, Jason and Robert Seamans. 2019. "AI and the Economy." *Innovation policy and the economy* 19 (1):161–191.

Hoffmann, Manuel, Frank Nagle, and Yanuo Zhou. 2024. "The Value of Open Source Software." *Harvard Business School Strategy Unit Working Paper* 24 (038).

Kraut, Robert, Michael Patterson, Vicki Lundmark, Sara Kiesler, Tridas Mukophadhyay, and William Scherlis. 1998. "Internet paradox: A social technology that reduces social involvement and psychological well-being?" *American psychologist* 53 (9):1017.

Lifshitz-Assaf, H and F Nagle. 2021. "The digital economy runs on open source. Here's how to protect it." *Harvard Business Review* :1–7.

Mosquera, Roberto, Mofioluwasademi Odunowo, Trent McNamara, Xiongfei Guo, and Ragan Petrie. 2020. "The economic effects of Facebook." *Experimental Economics* 23:575–602.

Nagle, Frank. 2018. "Learning by contributing: Gaining competitive advantage through contribution to crowdsourced public goods." *Organization Science* 29 (4):569–587.

———. 2019. "Open source software and firm productivity." *Management Science* 65 (3):1191–1215.

Nagle, Frank, David A Wheeler, H Lifshitz-Assaf, H Ham, and J Hoffman. 2020a. "Report on the 2020 foss contributor survey." *The Linux Foundation Core Infrastructure Initiative* .

Nagle, Frank, Jessica Wilkerson, James Dana, and Jennifer L Hoffman. 2020b. "Vulnerabilities in the Core: Preliminary Report and Census II of Open Source Software." *The Linux Foundation & The Laboratory for Innovation Science at Harvard* .

Noy, Shakked and Whitney Zhang. 2023. "Experimental evidence on the productivity effects of generative artificial intelligence." *Available at SSRN 4375283* .

Peng, Sida, Eirini Kalliamvakou, Peter Cihon, and Mert Demirer. 2023. "The impact of ai on developer productivity: Evidence from github copilot." *arXiv preprint arXiv:2302.06590* .

Raj, Manav and Robert Seamans. 2018. "Artificial intelligence, labor, productivity, and the need for firm-level data." In *The economics of artificial intelligence: An agenda*. University of Chicago Press, 553–565.

Raman, Naveen, Minxuan Cao, Yulia Tsvetkov, Christian Kästner, and Bogdan Vasilescu. 2020. "Stress and burnout in open source: Toward finding, understanding, and mitigating unhealthy interactions." In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: New Ideas and Emerging Results*. 57–60.

Robbins, Carol, Gizam Korkmaz, Ledia Guci, J Bayoán Santiago Calderón, and Brandon Kramer. 2021. "A First Look at Open-Source Software Investment in the United States and in Other Countries, 2009-2019." In *Paper presented the IARIW-ESCoE Conference, Measuring Intangible Capitals and their Contribution to Growth (November, RSA House, London)*. 1–30.

Sachs, Goldman. 2023. "Generative AI could raise global GDP by 7%." *GoldmanSachs.com* .

Synopsys. 2023. "023 OSSRA: A deep dive into open source trends." *Synopsys, May 1 2023. https://www.synopsys.com/blogs/software-security/open-source-trends-ossra-report/* 05 (1).

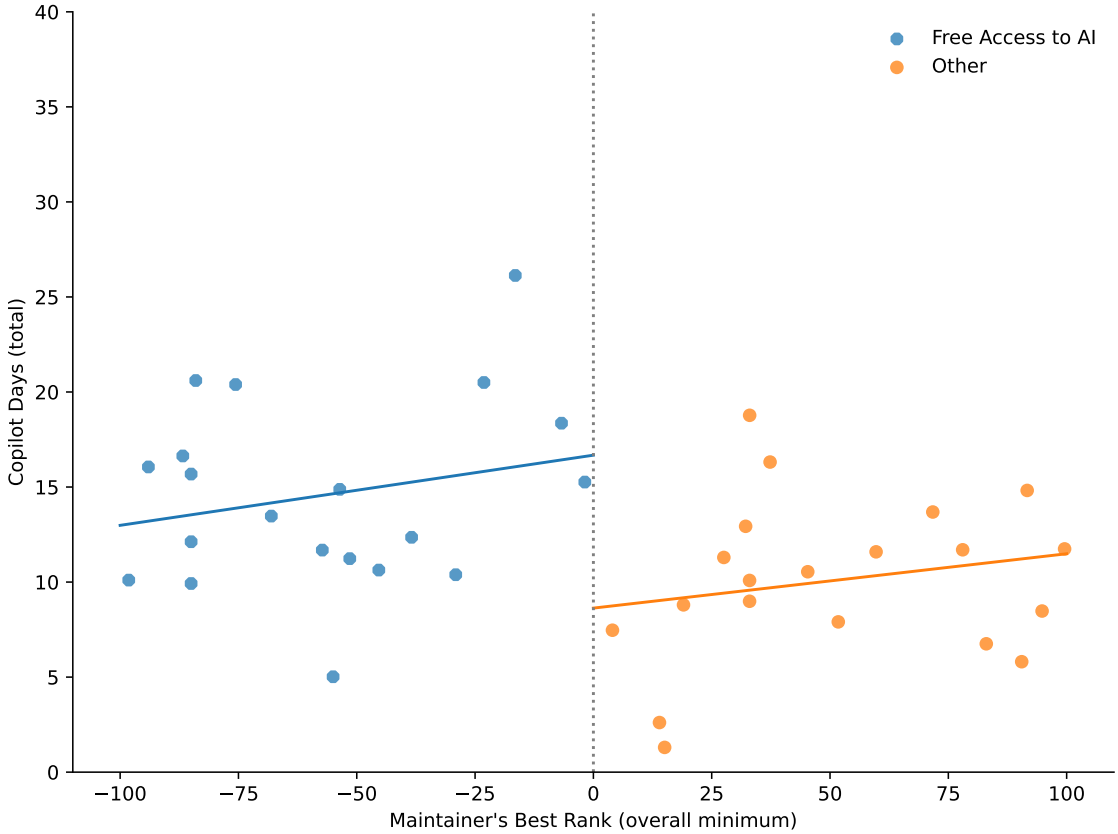
Tamayo, Jorge, L Doumi, S Goel, O Kovács-Ondrejkovic, and R Sadun. 2023. "Reskilling in the age of AI." *Harvard Business Review* 21.

Wright, Nataliya Langburd, Frank Nagle, and Shane Greenstein. 2023. "Open source software and global entrepreneurship." *Research Policy* 52 (9):104846.

Wright, Patrick M, Russell Coff, and Thomas P Moliterno. 2014. "Strategic human capital: Crossing the great divide." *Journal of Management* 40 (2):353–370.

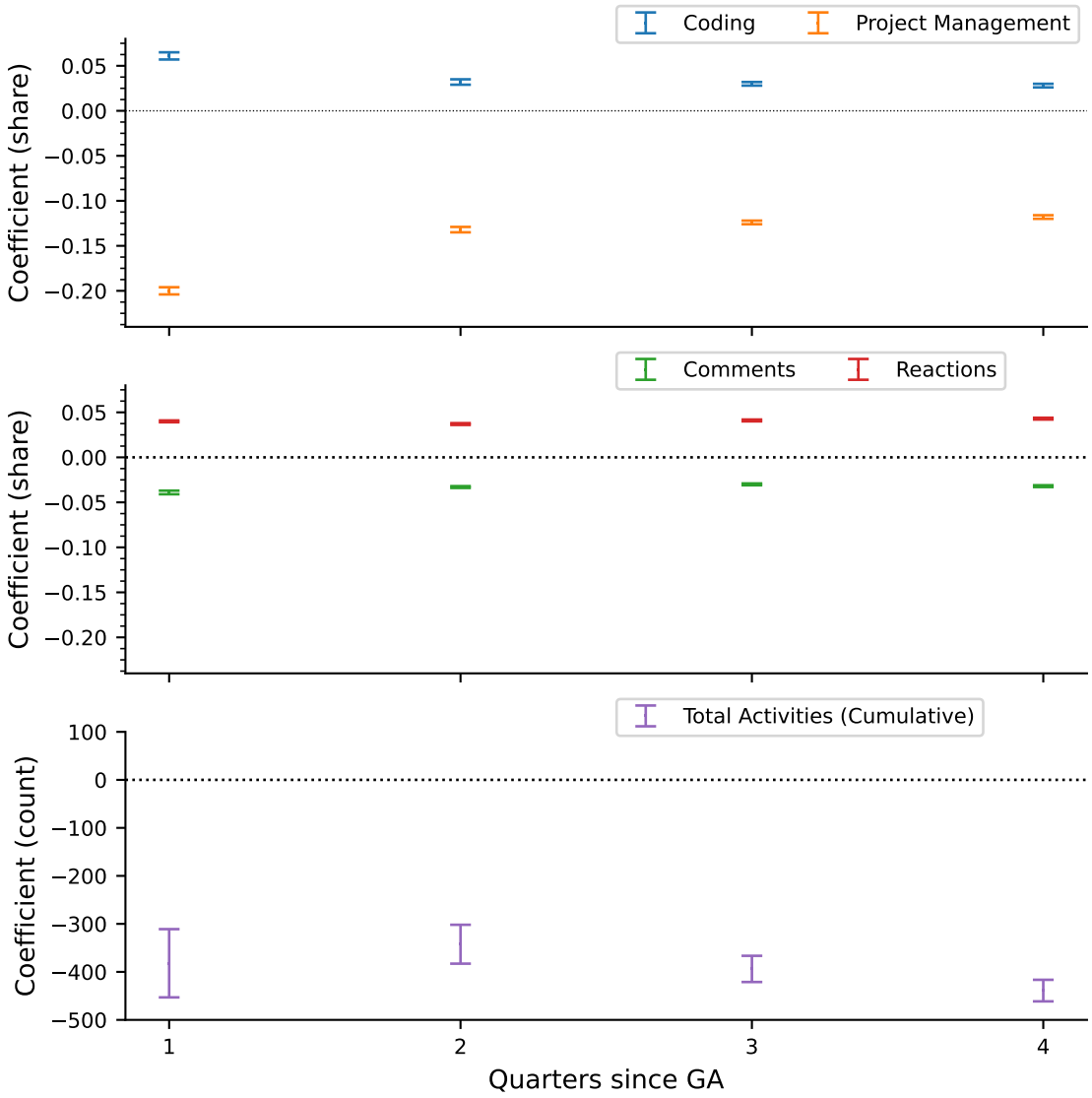
Figures and Tables

Figure 1: COPILOT AI ADOPTION ACROSS RANKS



Note: The figure shows the total number of days the Artificial Intelligence (AI) GitHub Copilot was used across the overall minimum ranking on GitHub using a linear fit on either side of the threshold. Maintainers with rankings below 0 receive free access to the AI through the top maintainer channel while those above do not. Time frame: July, 2022 to July, 2023.

Figure 2: INTENT-TO-TREAT EFFECTS OF COPILOT OVER TIME



Note: The figures plots intent-to-treat (ITT) coefficient estimates for our core activity measures of interest by pooling observations in different windows (i.e. quarters) since the General Access (GA) period.

Table 1: CORRELATION OF COPILOT AND WORK ACTIVITIES

Panel A	Coding	Project Management
$\mathbb{1}(Eligible)$	0.000	0.000***
	(0.000)	(0.000)
Panel B	**Comments**	**Reactions**
$\mathbb{1}(Eligible)$	0.000**	0.000
	(0.001)	(0.000)
Panel C	**Total Activities**	
$\mathbb{1}(Eligible)$	1.348***	
	(0.058)	

Note: This table shows the association of the generative AI tool Copilot with work activities on the GitHub platform. We display the correlation of the daily copilot usage within a week with the work activities of (1) coding, (2) project management, (3) comments, (4) reactions, and (5) total activities. Our full balanced panel is observed within a time period from July 2022 - July 2023.All standard errors are robust standard errors. *** $p < 0.001$, ** $p < 0.01$, * $p < 0.05$

Table 2: AI ADOPTION ACROSS MAINTAINER RANKING

	AI Adoption			
	Cross Section		**Panel**	
	Total Days	Ever Adoption	Days/Week	Exposure Share
$\mathbb{1}(Eligible)$	5.709***	0.121***	0.080***	0.011***
	(1.012)	(0.015)	(0.008)	(0.001)
Baseline	3.475***	0.083***	0.145***	0.022***
	(0.610)	(0.010)	(0.005)	(0.001)
Rel. TE (%)	164.3	145.8	55.2	50.0
F	31.82	65.07	100.00	121
N	10,363	10,363	330,388	330,388

Note: This table shows the first stage uptake of the generative AI tool Copilot for top maintainers that are eligible based on the internal GitHub ranking relative to those who are not eligible through this pathway using a bandwidth of ranks $h \in [-100, 100]$. The first two column show cross-sectional estimates on the total days of AI adoption and whether someone ever adopted AI. The last two columns show the panel estimates of the days per week of AI adoption and exposure shares, i.e. the cumulative days of AI adoption over the cumulative total days from the General Access period onward. All standard errors are robust standard errors. $***p < 0.001, **p < 0.01, *p < 0.05$

Table 3: INTENT-TO-TREAT EFFECT OF COPILOT AI ON WORK ACTIVITIES

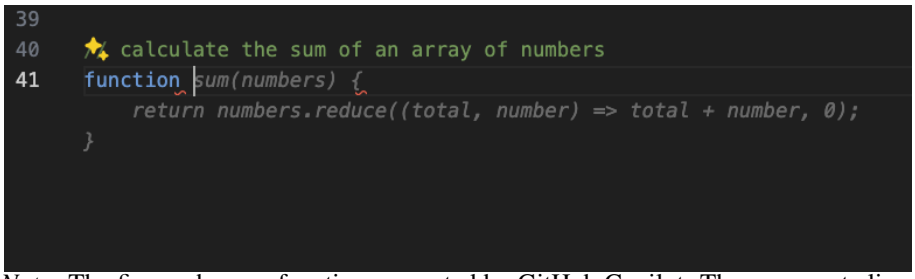
Panel A	Coding	Project Management
$\mathbb{1}(Eligible)$	0.026***	-0.117***
	(0.002)	(0.002)
N	330,388	330,388
Panel B	**Comments**	**Reactions**
$\mathbb{1}(Eligible)$	-0.031***	0.039***
	(0.001)	(0.001)
N	330,388	330,388
Panel C	**Total Activities**	
$\mathbb{1}(Eligible)$	-4.865***	
	(0.276)	
N	330,388	

Note: This table shows the coefficient estimates from the intent-to-treat specification across each activity classification for top maintainers that are eligible based on the internal GitHub ranking relative to those who are not eligible through this pathway using a bandwidth of ranks $h \in [-100, +100]$. All standard errors are robust standard errors. $***p < 0.001, **p < 0.01, *p < 0.05$

Appendices

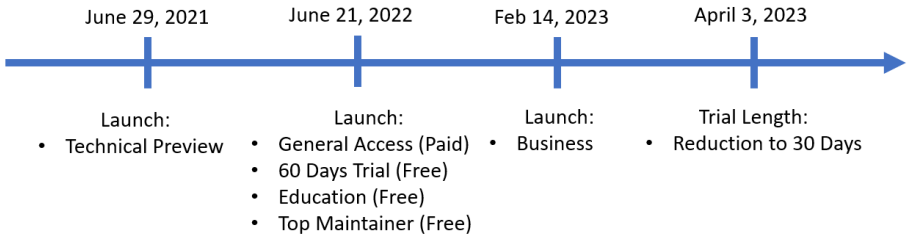
Appendix A More Tables and Figures

Figure A1: GitHub Copilot: Artificial Intelligence in Action



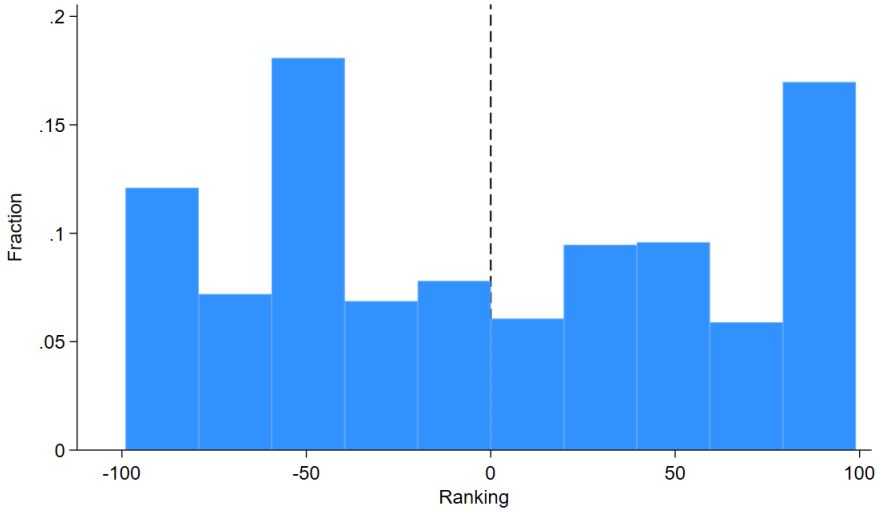
Note: The figure shows a function generated by GitHub Copilot. The user wrote line 40 asking in a comment to sum an array of numbers and Copilot suggested lines 41 and beyond.
Source: GitHub 2022

Figure A2: GitHub Copilot AI Deployment Timeline



Note: The figure shows the timeline for the introduction of the generative AI GitHub Copilot.

Figure A3: DISTRIBUTION OF MINIMUM RANKINGS



Note: The figure shows the distribution of the minimum rankings for the top maintainer based on the internal scoring system from GitHub. The dashed line is at the eligibility cutoff of the rank 0.

Table A1: DESCRIPTIVE STATISTICS OF TOP MAINTAINERS

	Mean	SD	Min	Max
AI Treatment				
AI Total Days	9.95	32.06	0	365
AI Ever Used	0.19	0.39	0	1
AI Exposure Share	0.03	0.10	0	1
AI Days Used / Week	0.21	0.95	0	7
Work Activities				
Coding	0.44	0.29	0	1
Trouble Shooting	0.24	0.22	0	1
Comments	0.09	0.12	0	1
Reactions	0.05	0.12	0	1
All Activities	13.47	251.75	0	280,214
Ranking				
Minimium Ranking	737.41	827.82	-1000	7,156

Note: The table shows univariate descriptive statistics with the arithmetic mean (mean) in the first column, followed by the standard deviation (SD), the lowest value of a variable (min), the highest value of a variable (max) and the number of observations (N). Our full balanced panel contains 5,995,924 observations within a time period from July 2022 - July 2023.

Table A2: DESCRIPTIVE STATISTICS OF TOP MAINTAINER FOR ALL WORK ACTIVITIES

	Mean	SD	Min	Max
Coding				
Create Repository	0.03	1.06	0	1,387
Fork Repository	0.05	1.03	0	1,077
Pull Request	0.54	10.09	0	6,026
Push	5.51	237.51	0	280,214
Trouble Shooting				
Issue: Assigned	0.24	2.5	0	1,616
Issue: Closed	1.02	13.81	0	16,022
Issue: Created	0.17	8.22	0	16,018
Issue: Comment Deleted	0.01	0.35	0	225
Issue: Labeled	0.90	27.07	0	10,074
Issue: Merged	0.62	6.00	0	4,338
Issue: Reopened	0.02	0.73	0	460
Issue: Review Requested	0.55	7.713	0	2,647
Issue: Subscribed	0.72	6.25	0	1,935
Comments				
Comment on Commit,	0.01	0.69	0	794
Comment on Pull Request of an Issue	1.42	19.81	0	5,360
Reactions				
Reacted to: Comment of a Commit	0.00	0.04	0	21
Reacted to: Pull Request of an Issue	0.06	0.47	0	90
Reacted to: Comment on a Pull Request of an Issue	0.15	1.11	0	616
Reacted to: Pull Request Review	0.10	0.84	0 164	

Note: The table shows univariate descriptive statistics with the arithmetic mean (mean) in the first column, followed by the standard deviation (SD), the lowest value of a variable (min), the highest value of a variable (max) across all individual work activities that are included in the categories from Table A1. Our full balanced panel contains 5,995,924 observations within a time period from July 2022 - July 2023.

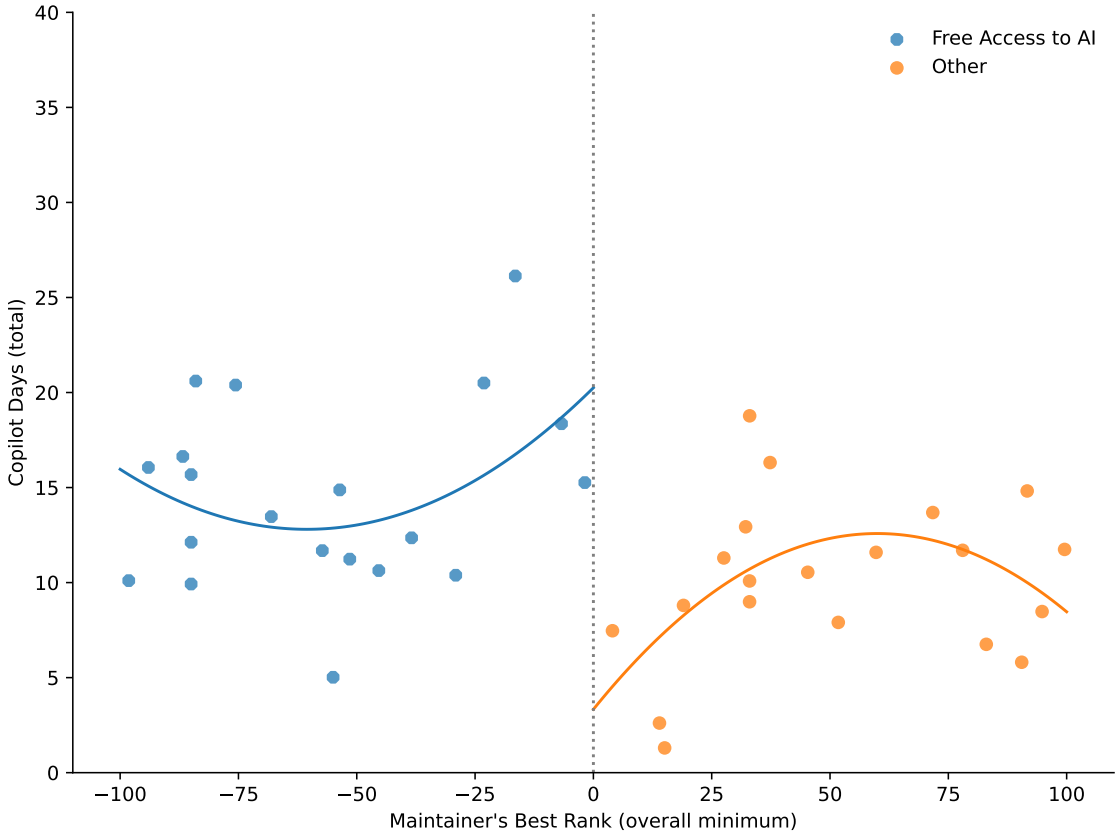
Appendix B Classification of Work Activities

Each classification is defined as the sum of its more disaggregated constituent activities.

- **Coding**: Create Repository, Push, Fork, Pull Request
- **Project Management**: Issue Created, Issue Closed, Issue Assigned, Issue Comment Deleted, Issue Labeled, Issue Merged, Issue Reopened, Issue Review Requested, Issue Subscribed, Created Project Board, Reviewed Pull Request
- **Comments**: Comment on a Commit, Comment on a Pull Request of an Issue
- **Reactions**: Reacted to a Comment from a Commit, Reacted to a Pull Request of an Issue, Reacted to a Comment for a Pull Request of an Issue, Reacted to a Pull Request Review

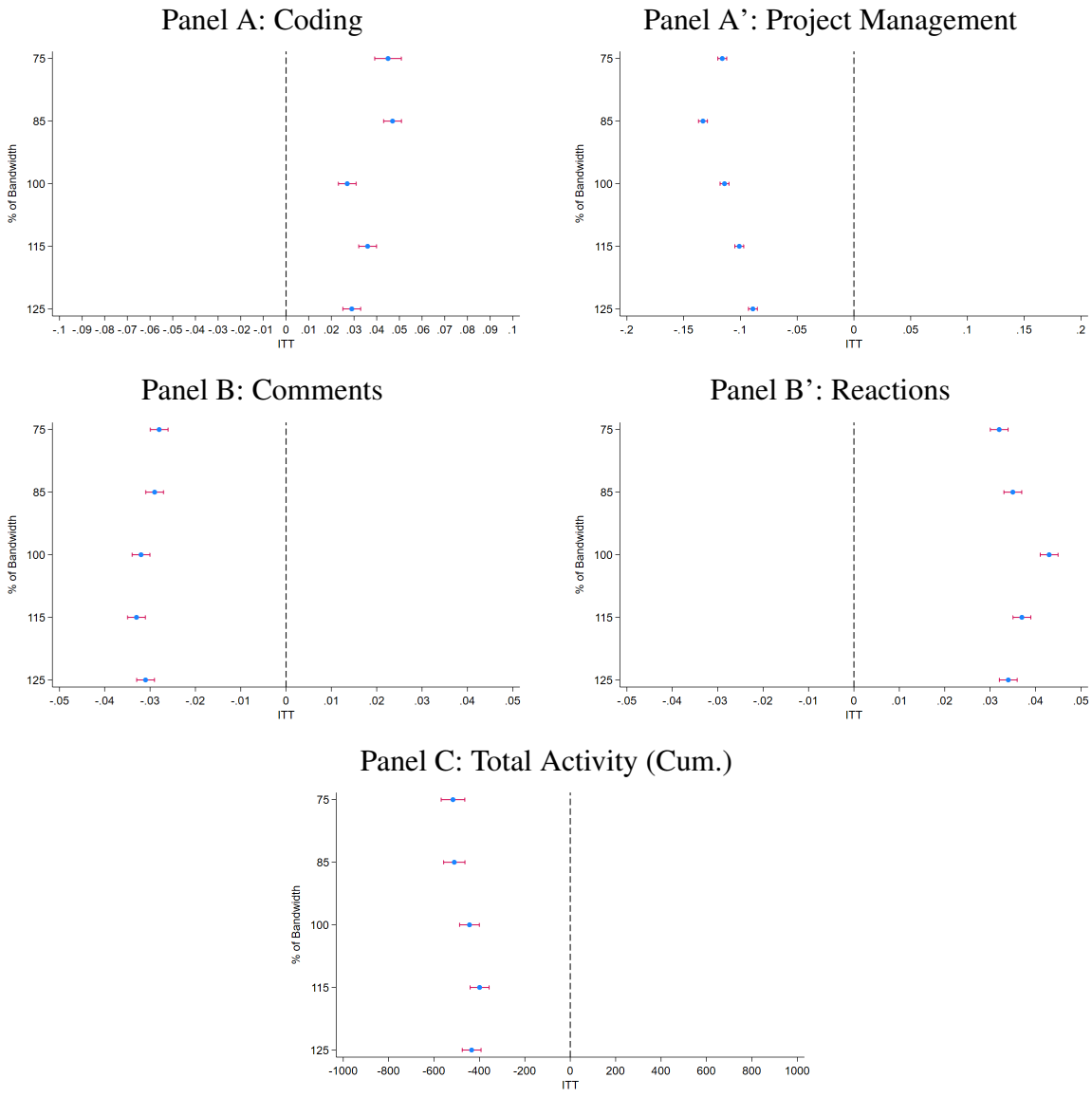
Appendix C Robustness Checks

Figure C1: COPILOT AI ADOPTION ACROSS RANKS



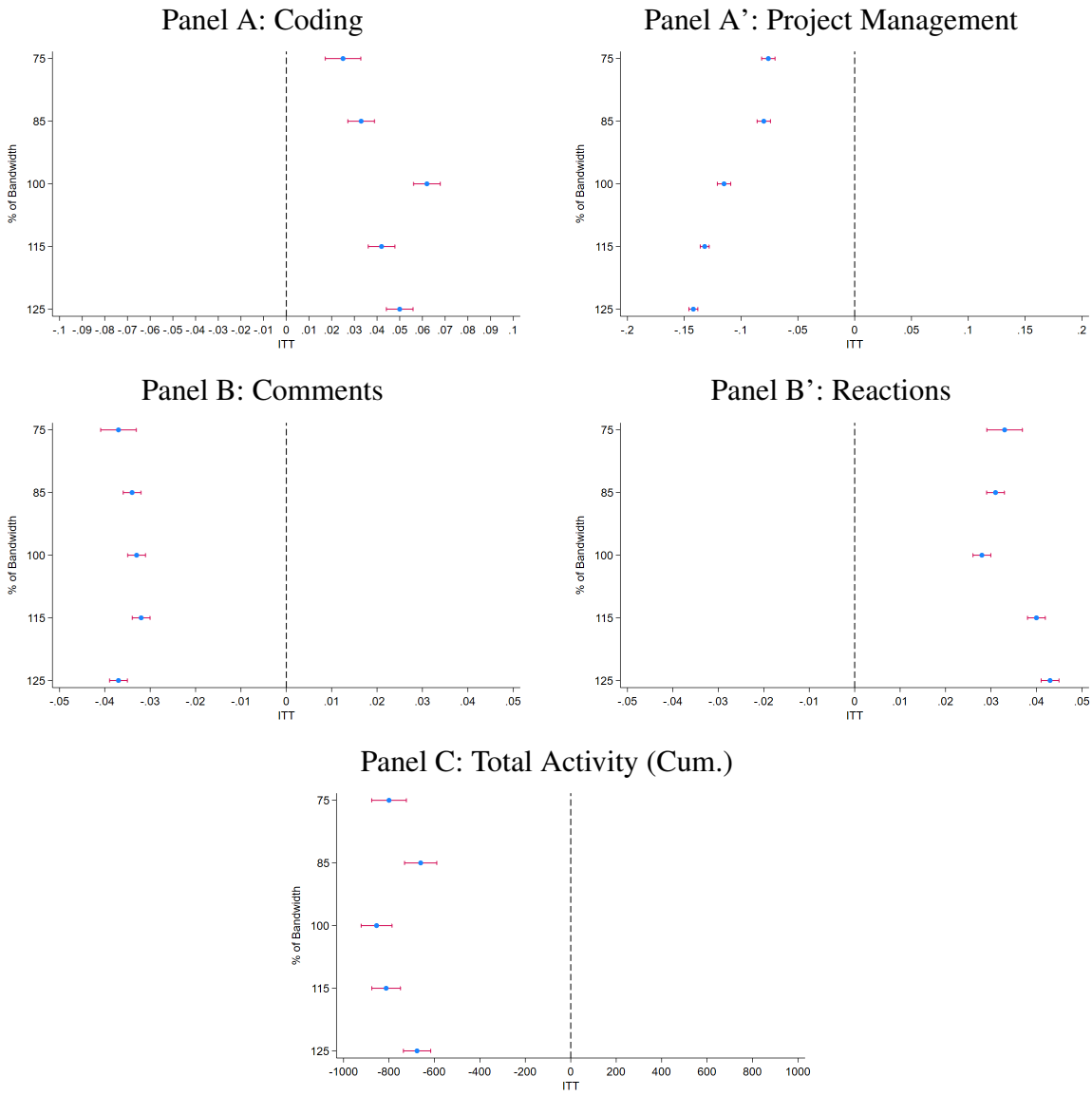
Note: The figure shows the total number of days the Artificial Intelligence (AI) GitHub Copilot was used across the overall minimum ranking based on the six top languages on GitHub with a quadratic fit on either side of the threshold. Maintainers with rankings below 0 receive free access to the AI through the top maintainer channel while those above do not. Time frame: July, 2022 to July, 2023.

Figure C2: ITT OF VARYING BANDWIDTHS, POLYNOMIAL OF DEGREE ONE



Note: The figure shows the ITT effects when crossing the ranking threshold from the right to the left. Panel A (B) displays linear ITT effects. The bandwidth of 100 is our baseline estimation using $h \in [-100, 100]$.

Figure C3: ITT OF VARYING BANDWIDTHS, POLYNOMIAL OF DEGREE TWO



Note: The figure shows the ITT effects when crossing the ranking threshold from the right to the left. Panel A (B) displays quadratic ITT effects. The bandwidth of 100 is our baseline estimation using $h \in [-100, 100]$.