# Task Efficiency and Signaling in the Age of GenAI: Effort Reallocation and Firm Value Effects [*]

Shiwei Ye[†]

This Version: September 3, 2025

Link to the Latest Version

## Abstract

In this paper, I study how Generative AI (GenAI) reshapes effort allocation and firm value by influencing the costs and signaling value of AI-assisted tasks. Using developer-level data from open-source projects linked to U.S. public firms and the launch of GitHub Copilot as a shock, I find that GenAI increases productivity in coding tasks but reduces the signaling value of such work for less-established developers. While senior developers benefit from increased efficiency in coding tasks, junior developers, whose contributions are less visible in an AI-assisted environment, create more-valuable projects as a more effective signal of ability. These changes in signaling incentives are reflected in selection of projects and languages, job mobility, promotion rates, and firm-level outcomes. Firms with more junior innovators exposed to AI see greater value creation from new projects, while non-innovative firms with senior teams capture efficiency gains. The findings shed new lights on the dual role of GenAI as both a productivity tool and a force reshaping labor market signaling.

Keywords: Generative AI, Productivity, Innovation, Signaling, Firm value
JEL Classification: G10, J24, O33, O36

[*]Acknowledgements: I am indebted to Thomas Lambert and Daniel Metzger for their mentorship and generous support. I am also grateful for Ashwini Agrawal, Dion Bongaerts, Jean-Edouard Colliard, Daniel Ferreira, and participants at ERIM PhD seminar and SGF Conference for helpful comments. All errors are my own. Part of this research was conducted during my visit to the London School of Economics. I am grateful for their hospitality and the financial support from the Erasmus Trustfonds.

[†]Rotterdam School of Management, Erasmus University. Email: ye@rsm.nl

## 1 Introduction

The public release of ChatGPT has led to a burst of discussion and debate over the potential implications of generative artificial intelligence (GenAI) on labor and business. Previous research has shown that labor-augmenting technologies, such as GenAI, provide greater benefits to less-experienced workers and help reduce inequality (Brynjolfsson, Li and Raymond, 2023; Kogan, Papanikolaou, Schmidt and Seegmiller, 2023; Cui, Demirer, Jaffe, Musolff, Peng and Salz, 2024; Hoffmann, Boysel, Nagle, Peng and Xu, 2024b). However, little is known about how GenAI might distort the signaling value of AI-assisted tasks, particularly for less-established employees, or how it could influence workers' incentives to engage in such tasks. For instance, Amazon recently banned AI usage in interviews, citing concerns that it prevents the company from accurately assessing candidates' skills and experience.[1] HackerRank's 2025 Developer Skills Report attributes hirers' uncertainty about coding ability without AI assistance to the recent decline in early-career developer hiring.[2]

In this paper, I fill this gap by investigating how different employees using GenAI contribute to firm value by allocating effort between AI-assisted tasks and creative activities. I use open-source software (OSS) projects made available by U.S. public firms on GitHub, the most popular open-source platform, as my empirical laboratory. This empirical setting is relevant because GenAI pronouncedly affects software development.[3] Furthermore, OSS represents an economically important context for studying value creation. As software made publicly available with no or little cost, OSS generates economic

[1]See https://www.businessinsider.com/amazon-stop-people-using-ai-cheat-job-interviews-2025-2.

[2]See https://www.hackerrank.com/reports/developer-skills-report-2025.

[3]For example, the Census Bureau conducted surveys showing that the share of firms adopting AI is highest in the information industry (See https://www.economist.com/business/2024/02/29/how-businesses-are-actually-using-generative-ai). A report published by the Burning Glass Institute and SHRM (https://shrm-res.cloudinary.com/image/upload/v1706729099/AI/CPR-230956_Research_Gen-AI-Workplace_FINAL_1.pdf) suggests Generative AI's biggest impact will be in banking and tech.

value for both releasing firms and the broader society through private value creation (Emery, Lim and Ye, 2024) and substantial externalities (Hoffmann, Nagle and Zhou, 2024a).[4]

I use a generalized difference-in-differences (DID) approach to study the causal effects of GenAI on coding efficiency and innovation outcomes of developers working for firms. I exploit the official launch of GitHub Copilot, a coding tool powered by OpenAI's large language models (LLMs) and widely adopted since then, on June 21st, 2022.[5] I construct a novel developer-level measure of GenAI exposure based on the programming languages developers use in their *ex ante* codebase portfolio. Because there is more training data in certain languages available for LLMs, some languages (such as Python) benefit more from Generative AI than others and therefore have higher GenAI exposure.[6] I then calculate AI exposure scores for each developer by taking the weighted average of their language-level AI exposure scores, and compare the top quartile of developers (treatment group) with the bottom three quartiles (control group) before and after GenAI's introduction.

I find that GenAI improves productivity in AI-assisted tasks for senior developers. Developers with high AI exposure are 1.16 percentage points more likely to contribute code to firm-owned projects each month, an effect concentrated on senior developers. These productivity gains cannot be purely explained by more working hours or lower quality of outputs. Dynamic effects observed from event-study analysis show that firms' developers immediately react to the introduction of GenAI, that the effects persist over

[4]In practice, firms increasingly choose to make their innovation open source. As of 2022, 90% of Fortune 100 companies use GitHub. See https://octoverse.github.com/2022/.

[5]The tool is integrated seamlessly into development environments and assists developers by suggesting code snippets in real time as they type code or natural language instructions. Since its introduction, developers have quickly adopted the tool, with over one million paid subscribers in 2023 and one third Fortune 500 adopters as of December 2022. See https://github.com/features/copilot (September 2024).

[6]For example, A GitHub post during the technical preview of GitHub Copilot says that "GitHub Copilot works with a broad set of frameworks and languages, but this technical preview works especially well for Python, JavaScript, TypeScript, Ruby and Go." See https://github.blog/2021-06-29-introducing-github-copilot-ai-pair-programmer/.

time, with no evidence of pre-treatment trend differences.

I further study how GenAI affects innovation outcomes. While GenAI does not affect the probability of innovation on average, it increases community interest in new projects, as measured by GitHub stars (a bookmarking system that serves as a widely-used proxy for popularity and quality) received by projects as of February 2024. I also find a positive stock market reaction to projects released by teams with high AI exposure. Exploiting variation in team seniority, I show that these effects are stronger for projects led by teams with more junior innovators.

Unlike many studies,[7] my findings suggest that less experienced developers do not engage more in AI-assisted tasks, even though they stand to benefit the most from GenAI. Instead, they generate higher-value innovation. One possible explanation is that AI-generated code reduces the signaling value of coding activities, especially for developers with shorter tenure. From my earlier example where Amazon banned the use of AI in interviews, they cite inability to accurately assess candidates' skills and experience. However, the signaling effect is less observable in lab settings, within internal firm activities, or among top-ranked open-source developers who typically have short tenure,[8] as studied in previous research.

I show that the signaling channel drives developer behavior after GenAI introduction in my sample. Senior developers target popular team projects for visibility benefits, while junior developers seek peer monitoring over popularity by avoiding solo work amid AI-introduced noise. Moreover, while senior developers increase activities for both new and

[7]Studies at the individual level consistently find that junior workers benefit more from generative AI than senior workers. They include, but are not limited to, Brynjolfsson et al. (2023); Dell'Acqua, McFowland, Mollick, Lifshitz-Assaf, Kellogg, Rajendran, Krayer, Candelon and Lakhani (2023); Noy and Zhang (2023), and Hoffmann et al. (2024b).

[8]For example, Hoffmann et al. (2024b) focus on developers contributing to the most popular public repositories for identification purposes. Tenure in their study has a sample mean of 706 days and a maximum of 1,420 days. In comparison, the sample used in this paper has a mean tenure of 2,690 days and a maximum of 5,274 days.

familiar languages, junior developers increase coding activities only for new languages, where signaling value is equal across seniority levels as neither group has established track records.

Signaling incentives further manifest in labor market outcomes. Junior *innovators* are more likely to exit the sample, move to other firms, and get promoted, suggesting that junior workers shift toward value-creating activities whose signaling value is less affected by GenAI. This effect extends to firms, where the incentive alignment between firms and their workforce composition plays a crucial role. For example, non-innovative firms, whose business is more exposed to GenAI, with a higher proportion of senior developers, who rely less on signaling through alternative tasks, are more likely to benefit from adopting GenAI tools to increase efficiency in AI-assisted tasks. Consistent with this prediction, I find that these firms experience higher cumulative abnormal returns following the introduction of GitHub Copilot.

**Literature.** This study makes several contributions to the literature. First, this paper speaks to the literature on the role of AI in firm value and growth. Several studies have examined how AI may affect firm value through labor productivity (Eisfeldt, Schubert and Zhang, 2023; Kogan et al., 2023), labor composition (Babina, Fedyk, He and Hodson, 2023; Berger, Cai, Qiu and Shen, 2024), product innovation (Babina, Fedyk, He and Hodson, 2024), and entrepreneur decision making (Otis, Clarke, Delecourt, Holtz and Koning, 2024). To my knowledge, this paper is among the first to directly show that GenAI can enable firms' innovators to create more novel products with higher value. More importantly, it suggests that employer-employee alignment in incentives for effort allocation between AI-assisted and less-affected tasks is important for value creation.

Secondly, I add to the growing body of research on the impact of Generative AI on labor outcomes. Previous research has studied the short-term impact of Generative AI on individual-level productivity and creativity across different types of discrete tasks, usually in experimental or single-firm settings (Brynjolfsson et al., 2023; Dell'Acqua et al., 2023; Noy and Zhang, 2023; Cui et al., 2024; Doshi and Hauser, 2024; Zheng, Wong, Zhou and Koh, 2024). Only a few studies have used large observational data, which allows researchers to examine longer-term impact at a larger scale, complementing experimental studies (Song, Agarwal and Wen, 2023; Zhou and Lee, 2023; Hoffmann et al., 2024b; Yeverechyahu, Mayya and Oestreicher-Singer, 2024). While my empirical setting is closely related to Song et al. (2023), Hoffmann et al. (2024b), and Yeverechyahu et al. (2024), who study the effects of GitHub Copilot on GitHub activities, their identification strategies restricted their sample to top maintainers or specific programming languages. Instead, the novel AI exposure score implemented at the developer level in this paper expands the sample to include general developers. More importantly, my paper focuses more on *firms* rather than on pure productivity outcomes. By using high-frequency, long-term observational data linked to firms, I am able to study detailed individual behavior in a collaborative work environment across all U.S. public firms that engage in open-source software development. The increased visibility allows me to provide more concrete evidence on the impact of GenAI on labor effort allocation, signaling incentives, career outcomes, and the interaction between labor and firm performance.

Methodologically, this study also contributes to the literature that uses Generative AI to generate new data and construct measurements to overcome various data challenges in academic research. For example, researchers have leveraged large language models (LLMs) to summarize or classify unstructured data (Cheng, Lee and Tambe, 2022; Beckmann, Beckmeyer, Filippou, Menze and Zhou, 2024; Chen and Wang, 2024; Kim, Muhn and Nikolaev, 2024) and generate synthetic data for variables that require less subjective evaluation, such as occupational AI exposure scores (Eisfeldt et al., 2023; Eloundou, Manning, Mishkin and Rock, 2023; Kogan et al., 2023). This paper uses novel LLM-based AI exposure scores for programming languages and applies LLM-inferred gender and task classification to overcome data challenges and improve research efficiency.

## 2 Conceptual Framework

In this section, I develop a parsimonious framework to conceptualize the trade-off between productivity gain and reduction in signal value of AI-assisted tasks when GenAI becomes available. In the spirit of Holmstrm (1999), I allow GenAI to both increase marginal return to effort and dilute the information that hiring markets extract from AI-assisted outputs. While the productivity channel always incentivize effort allocation to AI-assisted tasks, the model predicts that the signaling channel will drive career-concerned workers to shift towards alternative tasks that market weight more. Full model details and proofs are in Appendix A.

**Environment.** Consider developers work for their employers' open-source projects on GitHub. They perform two types of task in their daily workflow: task 1 that can be assisted by AI (such as coding) and task 2 that requires more human creativity (such as innovation). Suppose developers with ability $\theta$ make a one-time choice of efforts $e_i$ allocated to two types of tasks $i = 1, 2$ at personal cost $C(e_1, e_2) = \frac{1}{2}(e_1^2 + e_2^2)$. Developers' efforts produce outputs

$$y_1 = b_1 e_1 + \underbrace{(1-\lambda)\theta}_{\text{human}} + \underbrace{\lambda(\rho\theta + g)}_{\text{GenAI}} + \varepsilon_1, \quad \rho \in [0, 1],$$

$$y_2 = e_2 + \theta + \varepsilon_2,$$

$\theta$ is known to the current employer but the market only observes public signals

$(y_1, y_2)$. The observable coding output mixes human and AI components. Let $\lambda \in [0,1]$ captures the share of AI component. the mean AI share is $\mu_\lambda$, AI-share uncertainty is $\sigma_\lambda^2$, and the visibility of ability in AI content is $\rho \in [0,1]$. GenAI also introduces AI-quality uncertainty $\sigma_g^2$. Assume $\lambda \sim \text{Beta}(\alpha_\lambda, \beta_\lambda)$, $g \sim \mathcal{N}(0, \sigma_g^2)$, $\varepsilon_i \sim \mathcal{N}(0, \sigma_i^2)$, $\theta \sim \mathcal{N}(0, \sigma_\theta^2)$, and they are independent from each other.

These parameters shape two statistics that matter for signaling: the ability loading in the coding signal $\kappa_\theta = 1 - \mu_\lambda(1-\rho)$ and the coding signal's variance

$$\sigma_{\eta_1}^2 = \text{Var}(\eta_1) = \underbrace{(1-\rho)^2 \sigma_\lambda^2 \sigma_\theta^2}_{\text{AI-share uncertainty}} + \underbrace{\mathbb{E}[\lambda^2]\, \sigma_g^2}_{\text{AI-quality uncertainty}} + \underbrace{\sigma_1^2}_{\text{measurement noise}} .$$

**Effort allocation problem.** Risk-neutral, career-concerned developers choose efforts allocated to tasks $i = 1, 2$ to maximize their expected utility consisting of direct payoff from current employer, reputational benefits from signaling controlled by the degree of career concern $(\beta)$ , and personal cost of efforts:

$$\mathbb{E}[U(e_1, e_2; \theta)|\theta] = b_1 e_1 + e_2 + [2 - \mu_\lambda(1-\rho)]\theta + \beta\hat{\theta} - \frac{1}{2}(e_1^2 + e_2^2).$$

With career concerns $\beta > 0$, the developer's optimal efforts depend on how much weight the market puts on each (purged) signal $(x_1, x_2)$. In reduced form,

$$e_1^* = b_1\big(1 + \beta\, \alpha_1\big), \qquad e_2^* = 1 + \beta\, \alpha_2, \qquad \alpha_1 + \alpha_2 < 1.$$

In Appendix A, I show that under the best linear estimator framework, $\alpha_1$ is the linear weight the market assigns to $x_1$ when inferring $\theta$. $\alpha_1$ is increasing in $\kappa_\theta$ and decreasing in $\sigma_{\eta_1}^2$, and $\alpha_2$ increases when $\alpha_1$ decreases. Thus, higher productivity $(b_1)$

increases $e_1^*$, whereas signaling dilution (lower $\kappa_\theta$, higher $\sigma_{\eta_1}^2$) from AI-induced noise pushes shift efforts from coding tasks to innovation tasks. Environment where AI authorship and AI quality are better controlled, such as through team monitoring, helps to restore the informativeness of coding and pushes effort back.

The objective of this paper is to study whether the productivity gain effect on effort allocation from GenAI adoption will be partly offset by weight shifting on signals, especially for less-established workers, and what it implies for firm value. The signaling channel is often implicit and not directly observable in experimental or firm internal settings ($\beta \to 0$ or $\theta$ is known) used in prior studies. Instead, in the GitHub setting I use in this paper, signaling is a main driver for participation ($\beta > 0$ and $\theta$ is incompletely known).

## 3 Institutional Background

### 3.1 Open Source Software and Commercial Engagement

Systems granting excludability, such as patents, have been seen to be important to incentivize innovation (Arrow, 1962; Crouzet, Eberly, Eisfeldt and Papanikolaou, 2022). Yet, there has been an increasing trend in open-source innovations, particularly in the software industry. Based on the definition of the Open Source Initiative, "open source" means not only access to the source code but also allowing free redistribution and modification under terms defined by open-source licenses. Therefore, when an innovation is "open-sourced," it is made publicly available to all parties at little or no cost. Because of potential knowledge spillovers and the reduction of replacement costs for open-source software (OSS) adopters, OSS can generate large externalities and facilitate innovation in society as a whole (Fershtman and Gandal, 2011; Nagle, 2019; Hoffmann et al., 2024a; Chen, Shi and Srinivasan, 2024). The recent debates over open-source large language models further show the increasing importance and impact of open-source innovation.

While open-source innovations contribute to social welfare, they can also generate private value for firms.[9] Indeed, many firms choose to make their innovation open source. A recent survey finds that 90% of Fortune 100 companies use GitHub, the largest platform for developing open-source innovation.[10] Emery et al. (2024) document an increasing trend of open-source activity by U.S. public firms, with these firms representing 68% of the stock market by market capitalization by the end of 2023. They show open-source innovation can generate private value for firms, and this value is a predictor of future sales growth, profitability, employment growth, and patent innovation.

## 3.2 Software Development Activities on GitHub

GitHub operates on the Git system, which supports a distributed and collaborative framework for software development. Although not all open-source projects are developed on GitHub, it remains the largest platform for such efforts and is closely associated with the concept of open-source software. This section will outline key terms and activities related to software development on GitHub.

To share their innovations on GitHub, firms begin by setting up organization accounts. Within these accounts, they can establish repositories (projects), with administrators determining whether these will be publicly accessible or restricted to selected organization or project members with appropriate permissions. The creation and maintenance of public repositories incur minimal costs, whereas managing private repositories

[9]There is a broad literature studying the incentives for commercial firms to reveal their innovations in an open-source way, see Allen (1983), Lerner and Tirole (2002), Harhoff, Henkel and von Hippel (2003), Dahlander and Gann (2010), Henkel, Schöberl and Alexy (2014), Parker, Van Alstyne and Jiang (2017), Alexy, West, Klapper and Reitzig (2018), Nagle (2018), Teece (2018) and Lin and Maruping (2022). For reviews of the open-source literature, see von Hippel and von Krogh (2003), Goldfarb and Tucker (2019), and Dahlander, Gann and Wallin (2021).

[10]See https://octoverse.github.com/2022/.

may require GitHub Team or GitHub Enterprise subscriptions for additional support and features. Importantly, despite previous charges for private repositories before GitHub's 2015 shift from a repository-based to a user-based pricing model, public repository hosting has been free since GitHub's launch.

The development process starts with developers making modifications to the codebase, committing these changes locally with concise descriptions. These "commits" are then "pushed" to remote branches, making the updates accessible to other contributors and users.

Users who want to follow a repository's progress can "star" a repository, essentially bookmarking it for future reference. Those who have questions or suggestions can also "open issues," which are addressed by the development team and the broader community.

Additionally, users can contribute by "forking" the repository, creating a personal copy to work on independently. If the changes made in the fork are considered beneficial to the original project, users can submit "pull requests." These pull requests are formal proposals to merge their changes back into the original repository. These pull requests are reviewed, and if accepted, the modifications are integrated into the main codebase, further advancing the open-source project.

## 3.3 GitHub Copilot

GitHub Copilot is a cloud-based AI-powered code completion tool developed by GitHub in collaboration with OpenAI. Specifically, it is built on OpenAI's Codex model, a large language model trained on vast datasets of public code repositories. The tool integrates seamlessly into popular Integrated Development Environments (IDEs), and is designed to assist developers by suggesting code snippets and entire functions in real-time as they write code. Initially, it was launched in June 2021 in preview, available with a limited number of spots. It has later become generally available to all developers since June 21st, 2022. While GitHub Copilot is freely available for verified students and maintainers of popular open-source projects, for most individual developers it is priced at $10 per month. There is also an Enterprise option for business. The tool is widely adopted since then. There are over one million paid subscribers in 2023, and one third of Fortune 500 companies use GitHub Copilot as of December 2022.[11]

Developers use GitHub Copilot by installing it as an extension in supported IDEs. As they type, Copilot analyzes the code context and offers autocomplete suggestions. It can also generate code based on natural language descriptions, allowing users to input comments describing desired functions or algorithms, and Copilot will output the corresponding code. Therefore, it significantly enhances developer productivity by reducing the time spent on routine coding tasks, lowering the cognitive load, and minimizing common errors. In addition, by offering creative coding solutions and suggesting best practices, it enables developers to learn new coding techniques and languages. In March 2023, GitHub Copilot further offers GPT-4-powered chat feature, which allows developers to engage in a dialogue with the AI assistant to get feedback and suggestions.

## 4 Data and Methodology

### 4.1 Data

#### 4.1.1 GitHub Activity of U.S. Public Firms' Developers

To construct the dataset on GitHub activity of developers working for U.S. public firms, I begin by linking GitHub organization accounts with firms. Following the methodology

[11]See https://github.com/features/copilot (September 2024).

of Conti, Peukert and Roche (2021), I first collect websites of organization accounts via the GHTorrent project and the GitHub API. I then match these domains with the web URLs of U.S. public firms and their subsidiaries from Compustat or Orbis.[12] I then manually search for firms' open-source organization accounts to complement the domain-based matching.[13] Following this, I compile a comprehensive list of public repositories owned by the identified organization accounts through the GHArchive database, which records and archives timestamped public activity of GitHub repositories. In total, I match 1,281 firms with 3,314 organization accounts and 168,085 public repositories up to the year 2023.

Upon establishing a link between U.S. public firms and their respective GitHub organization accounts and public repositories, I use the GHArchive database to gather additional information on the public footprints of these repositories. Most importantly, I identify individuals who are internal contributors (i.e., those who push commits to firm-owned repositories) as firms' developers. Overall, my sample spans from January 1st, 2021 to December 31st, 2023, 18 months before and after the introduction of GitHub Copilot.

### 4.1.2 Developer and Repository Characteristics

I use the GitHub API to collect static characteristics of developers as of March 2024. In particular, I obtain the account create date and self-reported names. I use the user account create date to calculate tenure and proxy for seniority. Figure 2 illustrates the distribution of account create month in my data. For self-reported names, I use

[12]Domains that are indicative of hosting or social media services, such as "github.com" and "facebook.com.", are excluded.

[13]Specifically, I query the firm names together with the term "open source" via Google to locate official web pages that list their open source projects, and search the firm names on GitHub to identify associated organization accounts.

OpenAI's API to interact with the GPT-3.5 turbo model to exclude users with account name containing "bot" or with "bot" account type identified to ensure bot accounts will not contaminate my sample. This results in 26,026 GitHub individual accounts during the sample period. I then match GitHub developers in my sample to their LinkedIn profile from Revelio Labs using their names and employers. In total, 12,858 developers are matched.

Similarly, I collect static characteristics of repositories extant as of February 2024 via GitHub API. This includes an array of attributes from descriptive repository metadata, such as programming languages and their corresponding byte sizes, to quantitative measures of community engagement, including the number of stars, watchers, and forks.

#### 4.1.3 Repository Value and Firm Characteristics

I estimate the forward-looking value of repositories using a stock market-based approach. Specifically, the value is calculated based on the stock market reaction within three days after a project is made public. Our other paper (Emery et al., 2024) provides methodology details and validation of the value measure. Stock return data comes from CRSP and other firm financial characteristics are obtained from Compustat.

### 4.2 Generative AI Exposure Measure

To compare users with relatively higher *ex ante* exposure to Generative AI with users with lower exposure, I leverage the programming languages used by a user from June 2019 to June 2021, which ends right before the Copilot preview and one year before the introduction of GitHub Copilot to ensure that the AI exposure score does not reflect selection effects. The idea is that some languages (such as Python) benefit more from

Generative AI than others (such as Stata) because there are more training data in certain languages available for LLMs. For each language, I assign an exposure score (0-1) to Generative AI coding tools based on ChatGPT's suggestions. Section Internet Appendix A.3 provides prompt details. While this paper is among the first to use ChatGPT to assign AI exposure score to programming languages, LLM-based AI exposure score has been largely implemented for occupations (Eisfeldt et al., 2023; Eloundou et al., 2023; Kogan et al., 2023). Table 1 lists selected languages and their AI exposure scores, with Python ranked first with a score of 1 and Stata and TeX ranked among the lowest with a score of 0.5. Other languages irrelevant for coding, such as CSV, do not have an AI exposure score.

Because there is no directly available information on language usage over time at user-level, I take two steps to approximately measure user-level AI exposure. First, I calculate the total language byte size for each user ($b_i^l$) based on user activities in firm-owned repositories and the byte size of languages in each repository ($b_r^l$) between June 2019 and June 2021. For each repository, I calculate user's fraction of contribution of each language in terms of the user's share of "PushEvent" and then sum it up to user-language level. Specifically, I calculate:

$$b_i^l = \sum_r \frac{a_{i,r}}{\sum_j a_{j,r}} b_r^l,$$

where $b_i^l$ is the byte size of language $l$ contributed by user $i$, $a_{i,r}$ is the total number of PushEvent activity of user $i$ in repository $r$, and $b_r^l$ is byte size of language $l$ used in repository $r$.

Then for each user, I calculate the weighted AI exposure score, where the weight is the byte size of a given language to the byte size of all code contribution by user $i$ among the two-year period one year prior to the introduction of GitHub Copilot. Specifically, I construct the user-level AI exposure score as follows:

$$s_i = \sum_l \frac{b_i^l}{\sum_l b_i^l} s^l,$$

where $s_i$ is the weighted AI exposure score of user $i$, $b_i^l$ is the byte size of language $l$ contributed by user $i$, and $s^l$ is AI exposure score of language $l$ provided by ChatGPT. Lastly, I define users with $s_i$ in the 4th quartile as having high exposure to Generative AI.

One could argue that a higher usage share of AI-exposed language does not necessarily indicate greater AI exposure at the individual level, as developers with less experience in an AI-assisted language may benefit more. However, as Acemoglu (2024) points out, code autocompletion tools like GitHub Copilot perform subtasks, but the overall task requires completion of both AI and human subtasks. This subtask complementarity suggests that a developer who frequently uses an AI-assisted language is more likely to experience increased activity, particularly in languages where human coding costs are relatively high. See Internet Appendix B for the details of the economic model. Consistent with the model's prediction, Table IA4 shows that a higher usage share of AI-exposed language predicts greater coding activity, especially in languages other than a developer's primary language. Additionally, the main results remain consistent at the individual-language level.

## 4.3 Identification Strategy

I use a generalized difference-in-differences (DID) approach to study the reactions of labor productivity and innovation outcomes of firms' developers to the introduction of GitHub Copilot, a code autocompletion and chat tool powered by OpenAI's GPT models. Using the shock of GitHub Copilot's public release has several advantages. First, GitHub Copilot is designed for coding tasks and is seamlessly integrated with major IDEs (integrated development environments), making it particularly relevant and easy to use for software developers. Second, the tool was officially launched for individual developers on June 21st, 2022,[14] five months before the release of ChatGPT on November 30th, 2022. Therefore, any initial reaction observed is likely to be driven by Generative AI powering job-specific coding tasks of developers rather than changes in activities of other tasks unobservable in the software development context. Third, while there was a period of technical preview since June 29th, 2021[15], the preview was strictly limited to a number of spots with relatively poor performance. The general availability of the tool can therefore serve as an ideal shock for the main purpose of this paper.

For baseline regressions, I use the following specification:

$$Y_{i,t} = \beta_1 Post_t \times AI\ Exposure_i + \mu_i + \theta_t + \epsilon_{i,t}, \tag{1}$$

where $Post_t$ indicates periods after the introduction of GitHub Copilot. Specifically, it equals one since July 2022 for monthly analysis or the third quarter of 2022 for quarterly analysis. $AI\ Exposure_i$ equals one for the group with relatively high Generative

[14] For official announcement, see https://github.blog/news-insights/product-news/github-copilot-is-generally-available-to-all-developers/

[15] See https://github.blog/news-insights/product-news/introducing-github-copilot-ai-pair-programmer/.

AI exposure, i.e., the user's *ex ante* AI exposure score is in the fourth quartile. In addition, I include individual ($\mu_i$) and time ($\theta_t$) fixed effects to control for time-invariant individual characteristics and common time trends. The outcomes of interest $Y_{i,t}$, are individual-level outcomes, such as engagement in AI-assisted coding tasks or creativity tasks and job changes.

I further explore the heterogenous effects of Generative AI on employees along the gender and seniority dimensions. To do this, I conduct a triple difference-in-differences (DDD) analysis using the following specification:

$$\begin{aligned} Y_{i,t} =& \beta_1 Post_t \times AI\ Exposure_i + \beta_2 Post_t \times Char_i \\ &+ \beta_3 Post_t \times AI\ Exposure_i \times Char_i + \mu_i + \theta_t + \epsilon_{i,t}, \end{aligned} \tag{2}$$

where $Char_i$ is a dummy indicating the the characteristics of developer $i$. For example, the dummy for seniority equals one if the tenure of the developer on the GitHub platform, approximated based on the account's create date, is above median. The coefficient of interest is therefore $\beta_3$.

Additionally, I conduct an event-study analysis for individual-level reactions to the introduction of the Generative AI. While the generalized DID approach gives an estimate of the average impact over the time horizon after the AI shock, the event-study approach allows for examining dynamic effects and checking whether the parallel trend assumption is violated or not. The event-study specification is as follows:

$$Y_{i,t} = \sum_{l=\underline{l}+1}^{\bar{l}-1} \gamma_l D^l_{i,t} + \gamma_{\underline{l}} D^{\underline{l}}_{i,t} + \gamma_{\bar{l}} D^{\bar{l}}_{i,t} + \mu_i + \theta_t + \epsilon_{i,t}, \tag{3}$$

where $D_l$ are leads and lags of treatment for short-run effects, and $D^{\underline{l}}$ ($D^{\bar{l}}$) accounts for periods before $\underline{l}$ (after $\bar{l}$) periods relative to treatment for all longer-run effects. $D^{-1}$ is omitted for normalization, that is, one month or one quarter before the introduction of GitHub Copilot based on the panel frequency. For monthly analysis, I set $\underline{l} = -7$ and $\bar{l} = 13$, and for quarterly analysis, I set $\underline{l} = -6$ and $\bar{l} = 5$.

Lastly, I conduct DID analysis in repeated cross-sections for project-level innovation outcomes in terms of community interest and value. Specifically, I estimate the following:

$$\begin{aligned} Y_{r,f,t} = & \beta_1 Innovator\ AI\ Exposure_r + \beta_2 Post_t \times Innovator\ AI\ Exposure_r \\ & + \beta_3 Repo\ AI\ Exposure_r + \beta_2 Post_t \times Repo\ AI\ Exposure_r \\ & + Controls_{r,f,t-1} + \alpha_f + \theta_t + \epsilon_{r,f,t}, \end{aligned} \tag{4}$$

where $Y_{r,f,t}$ is the dependent variable for community interest (number of stars received) and repository value estimated based on stock market reaction. I include both repository-level AI exposure ($Repo\ AI\ Exposure_r$) based on the repository language compositionand team-level AI exposure ($Innovator\ AI\ Exposure_r$) if one of the initiators is with high AI exposure. I include lagged firm-year level controls, including the natural logarithms of one plus cumulative number of firm-owned repository, market capitalization, volatility, number of employees, and one plus value of patent portfolio. I also control for return on assets, R&D expenditure as a share of assets, whether R&D expenditure is missing, and innovator team size, and include firm and time fixed effects.[16] Similar to developer-level analysis described above, I further exploit the heterogeneity of team composition in terms of seniority.

[16]These controls have been shown to be significant determinants of repository value as documented in Emery et al. (2024).

## 5 Empirical Results

### 5.1 Summary Statistics

I provide an overview of monthly open-source activities of firm's developers before the introduction of GitHub Copilot in Table 2. First, Panel (a) shows that the average AI exposure score in my sample is around 0.81, with little difference between junior and senior developers. Coding activities account for the majority of activity records.[17] Specifically, 69% developer-month has code contribution, and an average developer contributes code around 33 times per month, showing that these developers are active contributors. Developers on average contribute code to 2.9 projects per month, although they show active public footprint in 4.2 projects. Firms' developers work mostly for firm-owned projects (1.8 projects per month), but they are also active for individual projects (1.5 projects per month) and projects owned by non-firm organizations (0.8 projects per month). Exploiting heterogeneity in developer's characteristics, I show that before the introduction of the GenAI coding tool, junior developers contribute less in terms of intensity and frequency than senior developers across all types of activities, and they work on less number of projects concurrently.

Panel (b) of Table 2 compares activities and characteristics between developers with high and low exposure to GenAI. High-exposure developers contribute less code and participate in fewer projects, yet they do not differ in gender or seniority. Given that these developers are more likely to work for larger firms (see below) but display otherwise similar individual attributes, the observed differences in GitHub activity likely reflect variation in firm-level engagement on the platform rather than fundamental differences across developers.[18]

[17]See Section Internet Appendix A.1 for activity classification.

[18]Emery et al. (2024) show that large firms represent 89% of repositories in their sample.

Table IA2 compares characteristics between firms with high and low exposure to GenAI, based on the average AI exposure scores of their developers active on GitHub prior to GitHub Copilot's launch. These firms show similarity across many financial dimensions, including revenue growth, profitability, foreign revenue share, leverage, interest expense to total assets ratio, and R&D expenditure as a share of total assets. High-exposure firms, however, have smaller market capitalization (significant at 10% level). Despite having fewer active developers on GitHub, high-exposure firms maintain nearly identical ratios of activities associated with senior developers (63% versus 61%). Thus, the two groups appear broadly comparable in fundamentals.

Before moving to empirical analysis, the raw changes of outcomes might already tell the impact of the GenAI shock. Figure 3 plots firm-related coding activities over time between developers with high and low exposure to GenAI. Similar to what has been shown in the summary statistics above, developers with lower AI exposure are more active in general. Both groups see declining trends of activity in the pre-treatment period, and the trends are generally parallel. This might be because developers in my sample code less as they become more senior over time, or it can be because teams grow larger over time that there is less work for a single developer. However, after the introduction of GenAI, the slope of the decrease becomes more flat for developers with low AI exposure, and developers with high exposure become increasingly more active. This shows that while the productivity of all developers are positively affected by GenAI, the effect is much stronger for developers with high AI exposure.

## 5.2 AI-Assisted Tasks

In this section, I examine the impact of GenAI on the productivity of AI-assisted tasks, specifically coding, and explore how the effects vary among developers with different tenure lengths. I start by investigating the extensive margin, i.e., whether developers have any open-source coding activity related to firm-owned projects within a given month. Columns (1) and (2) of Table 3 presents results estimated from equations 1 and 2. The findings indicate that the GenAI-powered coding tool significantly increases the likelihood of coding-related events. Developers with high AI exposure are 1.16 percentage points (more likely to contribute code to firm-owned projects.[19] Moreover, this effect is predominantly driven by senior developers with longer tenure on GitHub, for whom the total effect is 1.67 percentage points.

Next, I compare the quantity of coding activity between developers with high and low AI exposure before and after the introduction of Generative AI. For this analysis, I aggregate activities by quarter due to the frequent occurrence of zero values in monthly data. Columns (3) and (4) of Table 3 report the results, confirming that GenAI similarly boosts coding activity within firm-owned projects, with the effect again stronger among senior developers.

Figure 4 plots the event study results for coding activity engagement related to firm-owned projects, with coefficients estimated using equation 3. The pre-launch coefficients confirm that the parallel trend assumption is not violated. Moreover, following GenAI's introduction, an immediate increase in coding activity occurs. Specifically, in the short-term, developers with high exposure become 2-4 percentage points more likely to contribute code to firm-owned projects immediately after the launch, and this elevated activity persists for up to nine months (three quarters).

The increase in AI-assisted activities may not indicate higher task productivity as a decrease in quality could accompany the increase in quantity. To investigate this, I begin by examining two proxies for quality: the number of stars and the number of

[19]These results also hold at the individual-language level, as reported in Table IA5.

issues opened attributed to each developer. "Starring" indicates direct community interest, whereas users open issues to report bugs or provide suggestions. Since developers naturally accumulate more stars with increased contributions, I also compute the cumulative ratio of stars per code push. Additionally, I calculate the cumulative ratio of issues opened per star, considering that popular projects typically foster more active discussions.

Table 4 reports the regression results, while Figure IA1 visualizes the event study estimates. The findings indicate that GenAI usage increases both the number of stars and issues attributable to developers, and similarly, these effects are more pronounced for senior developers. However, the stars-per-push ratio remains largely unchanged. By contrast, the ratio of issues opened per star actually decreases. This suggests that GenAI-driven productivity enhancements primarily target maintenance (reducing bugs) rather than increasing product popularity.

Alternatively, the increase in coding activity among GenAI-exposed developers may be explained by longer working hours. Since only the timestamp of event completion is available, I examine three specific outcomes to assess changes in input and the output-to-input ratio: work completed outside common hours, work completed on weekends, and work completed per hour, where work is defined as coding activity associated with firm-owned projects.[20] Common hours are determined as hours during which a developer completes events that constitute more than 5% of all events on a given weekday, based on activity records from 2020 of developers with at least 100 coding events.[21] Table

[20]Specifically, I look at the cumulative ratio of coding events occurring outside common hours ($\frac{\text{cumulative number of coding events outside common hours}_{i,t}}{\text{cumulative total number of coding events}_{i,t}}$), the cumulative ratio of coding events occurring on weekends ($\frac{\text{cumulative number of coding events on weekends}_{i,t}}{\text{cumulative total number of coding events}_{i,t}}$), and the cumulative number of coding events per hour ($\frac{\text{cumulative total number of coding events}_{i,t}}{\text{cumulative total number of hours}_{i,t}}$).

[21]For example, if a developer completes 100 coding events on Mondays throughout 2020, with only 2 at 8 pm and 5 at 2 pm, then 2 pm on Monday qualifies as a common hour, whereas 8 pm on Monday is considered outside common hours. Additionally, any hour on weekends is deemed outside common

5 presents the results. It appears that GenAI does not affect input, as measured by overtime work (Columns (1)-(4)). Consistent with increased output and unchanged input, developers complete more events per hour, as shown in Columns (5)-(6). Similar to the output results above, the effects are stronger for senior developers. The findings show that GenAI leads to efficiency gains by enabling developers to complete more work within standard working hours without increasing overtime or weekend work.

Overall, I find that after the introduction of GitHub Copilot, a coding tool powered by GenAI models, developers with high AI exposure show higher productivity, an effect not explained by more working hours or lower quality of outputs. This productivity gain is in line with the idea that GeneAI tools like GitHub Copilot reduce the cost of subtasks that complement those performed by humans (Acemoglu, 2024).

Yet, unlike many prior studies, where junior workers are more likely to adopt and benefit more from using GitHub Copilot or other Generative AI tools (Brynjolfsson et al., 2023; Dell'Acqua et al., 2023; Kogan et al., 2023; Cui et al., 2024; Gambacorta, Qiu, Shan and Rees, 2024; Hoffmann et al., 2024b), this paper finds stronger effects on AI-assisted tasks among senior developers. This counterintuitive result may reflect the declining signaling value of coding for junior workers, a key reason why they contribute to open-source projects on GitHub. If so, they may shift to other signals less influenced by GenAI, such as activities related to creativity and leadership. In the next section, I examine whether GenAI exposure leads to more product innovation, particularly among junior developers.

hours.

## 5.3 Product Innovation

In addition to improving labor productivity, GenAI may contribute to firm value and growth by stimulating new ideas and products (Babina et al., 2024). As noted above, junior workers with less established records may be more motivated to produce signals through tasks less affected by AI. This shift is feasible, as AI-augmented humans, freed from routine work, can redirect their time and cognitive capacity toward creative activities.

I study the impact of GenAI on firms' open-source innovation, focusing on the likelihood of developers becoming innovators, the community's interest in the innovation, and the value of the innovation. I define innovators as those who initiate new projects owned by the firm. Specifically, I identify innovators who publicly contribute code to newly created projects within two weeks of their creation dates.[22]

My analysis starts with the likelihood of producing innovation and the number of new projects initiated each quarter. Table 6 reports the results. Overall, the introduction of GenAI does not affect either the probability of innovation or the number of new projects. If anything, junior developers at firms with high AI exposure appear to be 0.6 percentage points less likely to create firm-owned new projects post-treatment. One potential explanation is that junior innovators, who produce more meaningful signals in the post-GenAI era, are more likely to leave public firms, resulting in their exclusion from the sample after the introduction of GenAI. Indeed, as shown in Table IA3, while *developers* with high AI exposure are 1.67 percentage points more likely to exit following the Copilot launch, there is no significant difference in exit probability between senior and junior developers. However, *innovators* with high AI exposure show a higher exit

[22]This definition is conservative as some projects are made public several months after creation, and, as a result, no innovators are identified for these projects.

rate (2.63 percentage points) than general developers, with junior innovators twice as likely to exit as senior innovators.

Next, I turn to project-level outcomes, focusing on community interest, proxied by the number of stars received as of February 2024, and repository value (in 2023 dollars), estimated based on the stock market reaction to the project's public release. In addition to the innovation team's AI exposure, I include a project-level AI exposure score derived from the project's language profile. This accounts for potential shifts in project composition following the AI breakthrough in this repeated cross-sectional analysis. For instance, there may be an increase in AI-related, Python-based projects that naturally are contributed by Python developers.[23]

Panel (a) of Table 7 reports the results. I show that GenAI does not necessarily help projects with higher AI exposure attract more community interest. However, projects initiated by innovators with high AI exposure receive significantly more stars, suggesting greater recognition. Exploiting team-level seniority, I find that the positive effects are primarily driven by projects led by teams with a higher ratio of junior members, though the difference is not statistically significant.

If GenAI has the potential to increase revenue by driving higher product innovation and demand, one would expect GenAI-exposed projects to create more firm value. I estimate the value of the new innovation based on the methodology documented in Emery et al. (2024), a stock-market based approach, and investigate the impact of Generative AI on the private value of open-source innovation for firms.[24] I control for innovator team

[23]In unreported analysis, I use alternative project-level AI exposure measure by using LLMs to infer whether the project's self-reported topics and description are related to core AI and machine learning algorithms or AI applications. The results remain consistent.

[24]For firms, open-source innovation has been shown to generate private value, and this value is a predictor of future sales growth, profitability, employment growth, and patent innovation (Emery et al., 2024). From a social welfare perspective, open-source innovation also largely benefits society by reducing replacement costs (Hoffmann et al., 2024a) and increasing overall patent values (Chen et al., 2024).

size and firm-year characteristics that have been shown to be determinants of repository value, along with firm and time fixed effects.

The results are shown in Panel (b) of Table 7. As with community interest, projects more exposed to GenAI do not generate significantly higher value. However, those contributed by innovators with high AI exposure are valued 8% higher. The effect is again stronger for projects led by teams of junior developers. In an AI-exposed team of five senior innovators, replacing one senior with a junior increases the effect by 8.5 percentage points, *ceteris paribus*.

In summary, I find no evidence that GenAI encourages developers to become innovators. However, projects led by teams with high AI exposure gain more community recognition and are valued more highly by the stock market. These effects are stronger for teams with more junior innovators. This contrasts with earlier findings on labor productivity, which show that GenAI benefits senior developers more. In the next section, I examine one channel that may reconcile these results: the generation of productivity information.

### 5.4 The Effects of Generative AI on Signaling

The desire for peer recognition and career advancement often motivates developers to contribute to open-source projects (Lerner and Tirole, 2005). As the largest platform for hosting open-source projects, GitHub is widely recognized by developers and employers as a source of productivity signals, particularly from firm-owned projects.[25] For example, Gupta, Nishesh and Simintzi (2024) show that high-skill developers from small firms who reveal their coding activities are more likely to be hired by large firms and promoted to

[25]According to GitHub, most first-time internal and external contributors to open source projects on GitHub chose bigger, company-run repositories. See https://octoverse.github.com/2022/state-of-open-source.

senior roles.

GenAI coding tools can have two opposing effects on productivity signal generation. On one hand, GenAI reduces the cost of AI-assisted tasks, particularly for less-experienced developers (Cui et al., 2024; Hoffmann et al., 2024b). On the other hand, because AI-generated code is harder to distinguish from human-written code, GenAI may introduce noise into the signal, especially for developers with limited track records. If this second effect dominates, developers with shorter tenure might choose to spend the time saved on AI-assisted tasks on areas less influenced by GenAI, such as creativity-focused work. This could explain why, despite GenAI tools proving more beneficial for low-skilled workers in lab experiments, the same effect is not observed in my setting, where signaling is a major incentive in participation.

Signaling incentives predict that developers exposed to GenAI will increase contributions to more popular projects because the visibility and reputational payoff from working on well-known repositories is greater. However, junior developers facing higher AI-introduced noise may prioritize working with larger teams where peer monitoring moderates the distortion, making project popularity secondary. Table IA6 confirms these predictions. All developers increase work on team projects after GenAI introduction, but the underlying mechanisms differ by seniority. Senior developers drive this increase partly through preference for more popular projects (Columns (4) and (8)), which also tend to be managed by larger teams. In contrast, junior developers show no responsiveness to popularity and actually decrease contributions to solo projects, where peer monitoring is absent.[26]

[26]One might wonder why, for junior developers with high AI exposure, contributions to solo projects decrease and contributions to team projects remain unchanged, yet total contributions across all projects are also unchanged (Table 3). This occurs because the sample includes only projects that existed before GenAI introduction, since team size classifications require pre-treatment activity data. Contributions to newly created projects are therefore excluded from these measures. This pattern further implies that junior developers reallocate effort toward new projects launched after GenAI's introduction, consistent with their increased investment in innovative activities.

Developer-language level analysis provides complementary evidence for this signaling mechanism. Table IA5 shows that senior developers increase coding activities for both familiar and new languages with high AI exposure after GenAI introduction. Junior developers, however, only increase coding activities for new languages with high AI exposure. This can be explained by the equal signaling value that new languages offer across seniority levels, as neither group has established track records. Overall, these divergent responses across project types and programming languages provide strong evidence that signaling motives drive developer behavior in my sample.

If signaling drives these patterns, it should be reflected in corresponding changes in job market success. I compare developers' job mobility and promotions before and after the official launch of GitHub Copilot. Empirically, I match GitHub developers to their LinkedIn profiles from Revelio Labs using their names and employers. In total, 12,858 developers are matched. Table IA7 presents summary statistics on job changes among firm developers with matched GitHub profiles at the developer-year-quarter level from January 2021 to December 2023.

As shown in Panel (a), the average probability of a job move is 7% per quarter, with 3% happening within the same firm. The probability of an across-firm promotion is only 1%, partly because job transitions with missing seniority or compensation data before or after the transition are excluded. Developers with longer GitHub tenure make up 65% of the sample, indicating they are more likely to create LinkedIn accounts early. The average job seniority is 3.12 on a 1-7 scale, and the total yearly compensation averages approximately $183,410.

Panel (b) compares developers based on their tenure and whether they are classified as innovators (i.e., those who initiated at least one GitHub project owned by their employers before the introduction of generative AI tools). On average, junior developers are 2 percentage points more likely to change job positions and 1 percentage point more likely to move to other firms. They also hold less senior positions with lower total compensation. The differences in job mobility between innovators and non-innovators are less pronounced, though innovators tend to hold more senior positions and receive higher total compensation.

I first present evidence on the impact of GenAI on employee mobility, controlling for previous job's characteristics, individual fixed effects, and firm-time fixed effects. As shown by Baird, Mar, Xu and Xu (2024), adopting GenAI tools like GitHub Copilot increases firms' labor demand for software engineers, particularly at the entry level. Panel (a) of Table 8 provides the estimates. Consistent with their findings, I find that developers with greater AI exposure are 0.64 percentage point more likely to change employers per quarter, with the effect mainly driven by junior developers (1.22 percentage points). This suggests that AI-exposed junior developers have stronger incentives to signal their abilities through GitHub activities before switching jobs, as potential employers may have limited information about their skills.

I now compare promotion probabilities across firms between senior and junior developers, as well as between innovators and non-innovators. The results are presented in Panel (b) of Table 8. Columns (1)-(2) show that among senior developers, GenAI does not affect the likelihood of demotion. However, senior innovators are less likely to be promoted when switching firms. This aligns with previous findings, where senior developers focus more on coding activities, and their new projects are perceived as having lower value by the stock market. In contrast, junior innovators with greater AI exposure are 1.27 percentage points more likely to be promoted across firms than junior developers. As shown in Columns (5)-(6), although the effect lacks statistical significance due to a smaller sample size, it is primarily driven by junior innovators working for less innovative firms, whose businesses are more affected by GenAI-powered coding tools. Overall, the results suggest that firms value creativity-related signals from AI-affected developers, particularly those with less established records.

If the signaling channel is important, the impact of GenAI on firm value will depend on employer-employee alignment around signaling incentives. Senior developers readily exploit GenAI's efficiency gains. Instead, junior developers need to establish credibility in their career profiles and might prioritize tasks where human contributions stand out. As a result, a firm focusing on AI-assisted coding benefit more if its workforce is primarily composed of senior developers than a junior-heavy firm. In contrast, innovative firms with more junior developers benefit from the aligned incentives, since junior developers can pursue innovation tasks that match both their signaling needs and the firm's objectives. However, this alignment matters less overall because innovative firms' core business models are less affected by GenAI.

To test this hypothesis, I use an event study approach, examining cumulative abnormal returns following the official launch of GitHub Copilot. To ensure a relevant and meaningful sample, I include only firms in the information technology industry (SIC code 737) and those with more than 100 code push events up to the event date. I calculate a firm's AI exposure score based on the language composition of its repositories and classify a firm as AI-exposed if its score falls in the fourth quartile. I assess firm-developer compatibility by considering a firm's innovativeness and workforce tenure. Specifically, incentives are aligned if an innovative firm (with R&D expenditure as a share of assets above the median) has an average developer tenure in the first quartile or if a non-innovative firm has an average tenure in the fourth quartile. Thus, if the hypothesis holds, incentive-aligned firms should experience higher abnormal returns after the introduction of the GenAI coding tool.

Table 9 report the event study results. Panel (a) examines all active GitHub firms in the information technology industry. On the day of Copilot's launch, there is little market reaction, suggesting that the market took time to process information about disruptive technologies like GenAI. In the event windows from 10 days to 30 days, consistent with the hypothesis, AI-affected firms with aligned incentives experience higher cumulative abnormal returns. Panel (b) further divides the sample into innovative and non-innovative firms. The positive effect of incentive alignment is concentrated in non-innovative firms. Consistent with previous prediction, these results suggest that investors perceive firms compatible with employees' signaling incentives benefit more from GenAI tools, particularly those whose businesses are more exposed to GenAI.

To summarize, the findings suggest that the signaling channel may help explain the surprisingly small change in coding activity among junior developers, who are supposed to benefit more from GenAI tools. At the firm level, GenAI's impact on firm value is not driven solely by technological advancement but also by how well a firm aligns with signaling incentives of its employees.

## 6 Conclusion

In this paper, I explore GenAI's impact on how employees reallocate effort between AI-assisted tasks and creative work in the context of open-source softwares released by U.S. public firms. Using a new developer-level measure of AI exposure and exploiting the official launch of GitHub Copilot, I show that Generative AI affects labor productivity in AI-assisted tasks and innovation differently depending on employee tenure. I also investigate the role of signaling in explaining these results and in shaping firm value.

For labor productivity in AI-assisted coding tasks, I find that GenAI generally boosts output. Developers at firms with high AI exposure are 1.16 percentage points more likely to contribute code to firm-owned projects each month. Event-study analysis shows that these productivity gains persist over time, with no evidence of violated parallel trends.

I then turn to creative activities, using project initiation as a measure of innovation. While GenAI does not significantly influence the likelihood of innovation at the developer level, it does increase community interest in new projects. In addition, the private value created for firms, measured by stock market reactions within three days of a repository's release, is 8% higher on average. This rise in interest and value is driven more by innovators' exposure to AI than by the projects' own AI exposure. In contrast to coding activities, AI's impact on the value of innovation is greater for teams with more junior developers. Specifically, replacing one senior with a junior in an AI-exposed team of five senior innovators further raises the project's value to the firm by 8.5 percentage points.

Unlike many studies in the literature, these findings indicate that less experienced developers in this sample do not engage more in AI-assisted tasks, despite potentially benefiting more from GenAI. One possible explanation is that AI-generated code weakens the signaling value of coding activities, particularly for developers with shorter tenure. Thus, junior developers who rely on open-source projects to demonstrate their ability may shift to other signals less influenced by GenAI, such as innovation.

The signaling channel drives developer behavior after GenAI introduction: senior developers increase contributions to popular team projects for visibility benefits, while junior developers prioritize peer monitoring over popularity, avoiding solo work due to AI-introduced noise. Meanwhile, the differences in coding activities between junior and senior developers disappear for new languages, for which neither group has established track records. These behavioral changes translate into job market outcomes, as junior innovators with high AI exposure are more likely to exit firms than either senior innovators with high exposure or junior innovators with low exposure. By linking GitHub developers to their LinkedIn profiles, I also find that junior developers more exposed to GenAI are more likely to change jobs and get promoted when moving between firms. Most importantly, these effects are driven primarily by junior innovators rather than pure developers, suggesting that creative tasks have become a stronger signal for junior workers following the introduction of GenAI.

This dynamic also affects firms differently, depending on the alignment between firm incentives and workforce composition. For example, non-innovative firms with more senior developers, who have already established their credibility, are less concerned about information asymmetry and therefore increase firms' output with the help of GenAI. Consistent with this, I show that these firms experience higher cumulative abnormal returns following the introduction of GitHub Copilot.

The study could have implications for both firms and policymakers. For firms, the findings highlight the importance of understanding how AI tools interact with employee experience and task type. Firms employing junior workers may need to reconsider how performance is evaluated and signaled, particularly as AI reduces the visibility of individual contributions to AI-assisted tasks. Encouraging innovation and providing alternative pathways for skill signaling could help retain and develop junior talent in the age of GenAI.

For policymakers, the results suggest that AI adoption may not uniformly benefit all workers and could exacerbate existing disparities tied to experience and role. Policies aimed at workforce development should account for these differences, supporting early-career workers in building distinctive skills that remain valuable in an AI-augmented environment. Additionally, as innovation becomes a key signal of ability, ensuring broad access to training and platforms that enable creative contributions will be essential for equitable participation in the digital economy.

## References

**Acemoglu, Daron**, “The Simple Macroeconomics of AI,” May 2024.

**Alexy, Oliver, Joel West, Helge Klapper, and Markus Reitzig**, “Surrendering control to gain advantage: Reconciling openness and the resource-based view of the firm,” *Strategic Management Journal*, 2018, *39* (6), 1704–1727.

**Allen, Robert C.**, “Collective invention,” *Journal of Economic Behavior & Organization*, 1983, *4* (1), 1–24.

**Arrow, Kenneth**, “Economic Welfare and the Allocation of Resources for Invention,” in “The Rate and Direction of Inventive Activity: Economic and Social Factors,” Princeton University Press, 1962, pp. 609–626.

**Babina, Tania, Anastassia Fedyk, Alex He, and James Hodson**, “Artificial intelligence, firm growth, and product innovation,” *Journal of Financial Economics*, January 2024, *151*, 103745.

**_ , _ , Alex X. He, and James Hodson**, “Firm Investments in Artificial Intelligence Technologies and Changes in Workforce Composition,” June 2023.

**Baird, Matthew, Carpanelli Mar, Brian Xu, and Kevin Xu**, “Early Evidence on the Impact of GitHub Copilot on Labor Market Outcomes for Software Engineers,” 2024.

**Beckmann, Lars, Heiner Beckmeyer, Ilias Filippou, Stefan Menze, and Guofu Zhou**, “Unusual Financial Communication: ChatGPT, Earnings Calls, and Financial Markets,” January 2024.

**Berger, Philip G., Wei Cai, Lin Qiu, and Cindy Xinyi Shen**, “Employer and Employee Responses to Generative AI: Early Evidence,” February 2024.

**Brynjolfsson, Erik, Danielle Li, and Lindsey R. Raymond**, “Generative AI at Work,” April 2023.

**Chen, Mark A. and Joanna (Xiaoyu) Wang**, “Displacement or Augmentation? The Effects of AI on Workforce Dynamics and Firm Value,” April 2024.

**Chen, Wilbur, Terrence Tianshuo Shi, and Suraj Srinivasan**, “The Value of AI Innovations,” May 2024.

**Cheng, Zhaoqi, Dokyun Lee, and Prasanna Tambe**, “InnoVAE: Generative AI for Mapping Patents and Firm Innovation,” March 2022.

**Conti, Annamaria, Christian Peukert, and Maria Roche**, “Beefing IT up for your Investor? Open Sourcing and Startup Funding: Evidence from GitHub,” 2021. Working paper, IE Business School, Harvard University, University of Lausanne.

**Crouzet, Nicolas, Janice C. Eberly, Andrea L. Eisfeldt, and Dimitris Papanikolaou**, "The Economics of Intangible Capital," *Journal of Economic Perspectives*, August 2022, *36* (3), 29–52.

**Cui, Zheyuan (Kevin), Mert Demirer, Sonia Jaffe, Leon Musolff, Sida Peng, and Tobias Salz**, "The Effects of Generative AI on High Skilled Work: Evidence from Three Field Experiments with Software Developers," September 2024.

**Dahlander, Linus and David M. Gann**, "How open is innovation?," *Research Policy*, 2010, *39* (6), 699–709.

**_ , _ , and Martin W. Wallin**, "How open is innovation? A retrospective and ideas forward," *Research Policy*, 2021, *50* (4), 104218.

**Dell'Acqua, Fabrizio, Edward McFowland, Ethan R. Mollick, Hila Lifshitz-Assaf, Katherine Kellogg, Saran Rajendran, Lisa Krayer, Franois Candelon, and Karim R. Lakhani**, "Navigating the Jagged Technological Frontier: Field Experimental Evidence of the Effects of AI on Knowledge Worker Productivity and Quality," September 2023.

**Doshi, Anil R. and Oliver P. Hauser**, "Generative AI enhances individual creativity but reduces the collective diversity of novel content," *Science Advances*, July 2024, *10* (28), eadn5290. Publisher: American Association for the Advancement of Science.

**Eisfeldt, Andrea L., Gregor Schubert, and Miao Ben Zhang**, "Generative AI and Firm Values," May 2023.

**Eloundou, Tyna, Sam Manning, Pamela Mishkin, and Daniel Rock**, "GPTs are GPTs: An Early Look at the Labor Market Impact Potential of Large Language Models," August 2023. arXiv:2303.10130 [cs, econ, q-fin].

**Emery, Logan P., Chan Lim, and Shiwei Ye**, "The Private Value of Open-Source Innovation," 2024.

**Fershtman, Chaim and Neil Gandal**, "Direct and indirect knowledge spillovers: the social network of open-source projects," *The RAND Journal of Economics*, March 2011, *42* (1), 70–91.

**Gambacorta, Leonardo, Han Qiu, Shuo Shan, and Daniel Rees**, "Generative AI and labour productivity: a field experiment on coding," September 2024.

**Goldfarb, Avi and Catherine Tucker**, "Digital Economics," *Journal of Economic Literature*, 2019, *57* (1), 3–43.

**Gupta, Abhinav, Naman Nishesh, and Elena Simintzi**, "Big Data and Bigger Firms: A Labor Market Channel," October 2024.

**Harhoff, Dietmar, Joachim Henkel, and Eric von Hippel**, "Profiting from voluntary information spillovers: how users benefit by freely revealing their innovations," *Research Policy*, 2003, *32* (10), 1753–1769.

**Henkel, Joachim, Simone Schöberl, and Oliver Alexy**, "The emergence of openness: How and why firms adopt selective revealing in open innovation," *Research Policy*, 2014, *43* (5), 879–890.

**Hoffmann, Manuel, Frank Nagle, and Yanuo Zhou**, "The Value of Open Source Software," January 2024.

**_ , Sam Boysel, Frank Nagle, Sida Peng, and Kevin Xu**, "Generative AI and Distributed Work: Evidence from Open Source Software," 2024.

**Holmstrm, Bengt**, "Managerial Incentive Problems: A Dynamic Perspective," *The Review of Economic Studies*, January 1999, *66* (1), 169–182.

**Kim, Alex, Maximilian Muhn, and Valeri V. Nikolaev**, "From Transcripts to Insights: Uncovering Corporate Risks Using Generative AI," July 2024.

**Kogan, Leonid, Dimitris Papanikolaou, Amit Seru, and Noah Stoffman**, "Technological Innovation, Resource Allocation, and Growth*," *The Quarterly Journal of Economics*, May 2017, *132* (2), 665–712.

**_ , _ , Lawrence D.W. Schmidt, and Bryan Seegmiller**, "Technology and Labor Displacement: Evidence from Linking Patents with Worker-Level Data," November 2023.

**Lerner, Josh and Jean Tirole**, "Some Simple Economics of Open Source," *Journal of Industrial Economics*, 2002, *50* (2), 197–234.

**_ and _** , "The Economics of Technology Sharing: Open Source and Beyond," *Journal of Economic Perspectives*, 2005, *19* (2), 99–120.

**Lin, Yu-Kai and Likoebe M. Maruping**, "Open Source Collaboration in Digital Entrepreneurship," *Organization Science*, 2022, *33* (1), 212–230.

**Nagle, Frank**, "Learning by Contributing: Gaining Competitive Advantage Through Contribution to Crowdsourced Public Goods," *Organization Science*, 2018, *29* (4), 569–587.

_ , "Open Source Software and Firm Productivity," *Management Science*, March 2019, *65* (3), 1191–1215.

**Noy, Shakked and Whitney Zhang**, "Experimental evidence on the productivity effects of generative artificial intelligence," *Science*, July 2023, *381* (6654), 187–192. Publisher: American Association for the Advancement of Science.

**Otis, Nicholas, Rowan Clarke, Solne Delecourt, David Holtz, and Rembrand Koning**, “The Uneven Impact of Generative AI on Entrepreneurial Performance,” February 2024.

**Parker, Geoffrey, Marshall Van Alstyne, and Xiaoyue Jiang**, “Platform Ecosystems: How Developers Invert the Firm,” *MIS Quarterly*, 2017, *41* (1), 255–266.

**Song, Fangchen, Ashish Agarwal, and Wen Wen**, “The Impact of Generative AI on Collaborative Open-Source Software Development: Evidence from GitHub Copilot,” October 2023.

**Teece, David J.**, “Profiting from innovation in the digital economy: Enabling technologies, standards, and licensing models in the wireless world,” *Research Policy*, 2018, *47* (8), 1367–1387.

**von Hippel, Eric and Georg von Krogh**, “Open Source Software and the “Private-Collective”' Innovation Model: Issues for Organization Science,” *Organization Science*, 2003, *14* (2), 107–225.

**Yeverechyahu, Doron, Raveesh Mayya, and Gal Oestreicher-Singer**, “The Impact of Large Language Models on Open-source Innovation: Evidence from GitHub Copilot,” July 2024.

**Zheng, Jiexin, Ka Chau Wong, Jiali Zhou, and Tat Koon Koh**, “Large Language Model in Ideation for Product Innovation: An Exploratory Comparative Study,” February 2024.

**Zhou, Eric and Dokyun Lee**, “Generative AI, Human Creativity, and Art,” October 2023.

## A Appendix

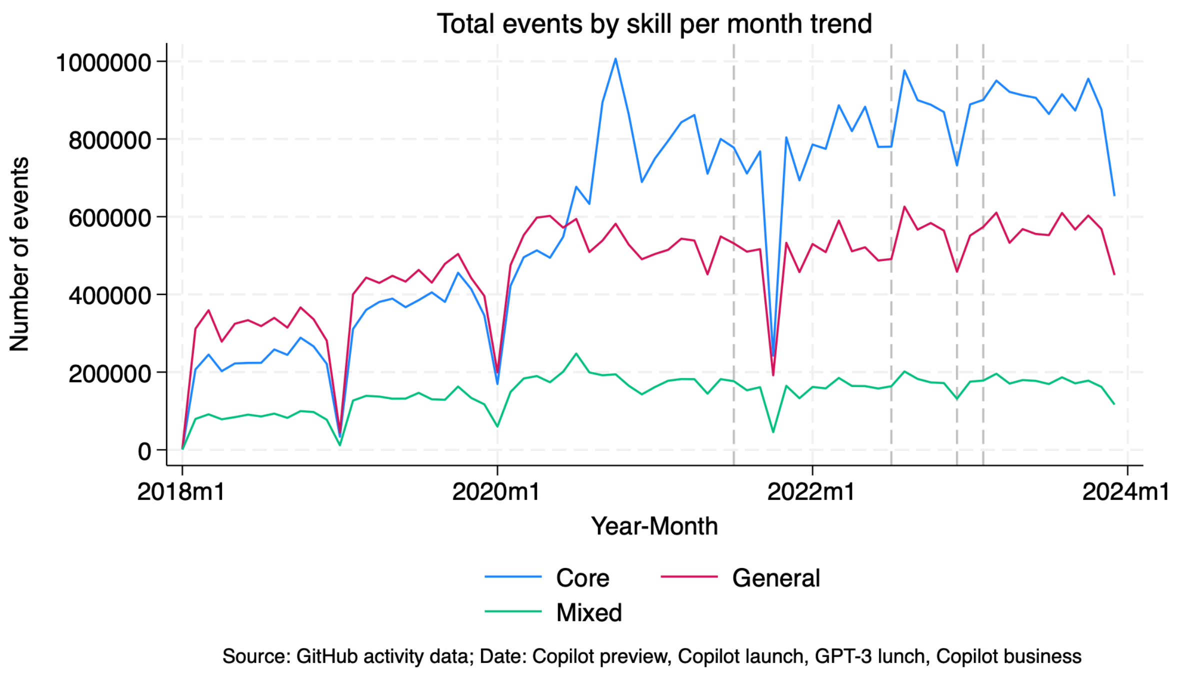

Figure 1. Monthly Aggregated Github Activities Over Time

This figure plots the monthly open-source activities within public firm-owned repositories on the GitHub platform from 2018 to 2023. Activities are grouped based on their related skill requirements. See Section Internet Appendix A.1 for classification details.

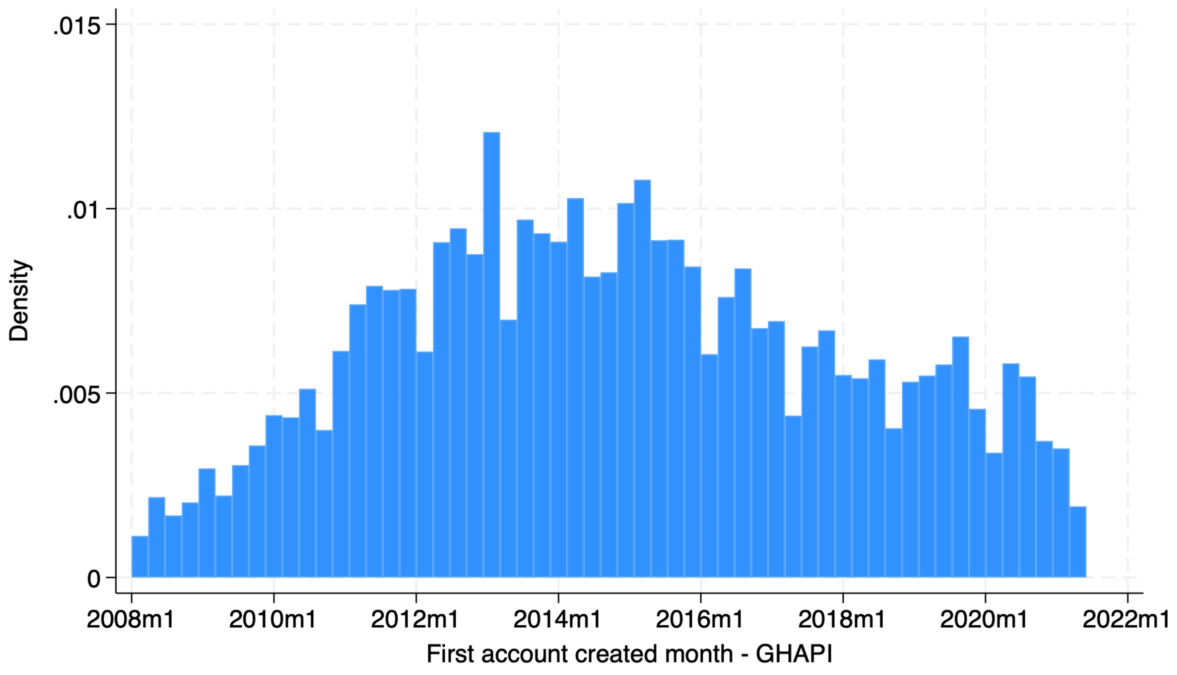

Figure 2. Density of Account Created Month

This figure plots the density of account create months of firms's developers, which is obtained via GitHub API.

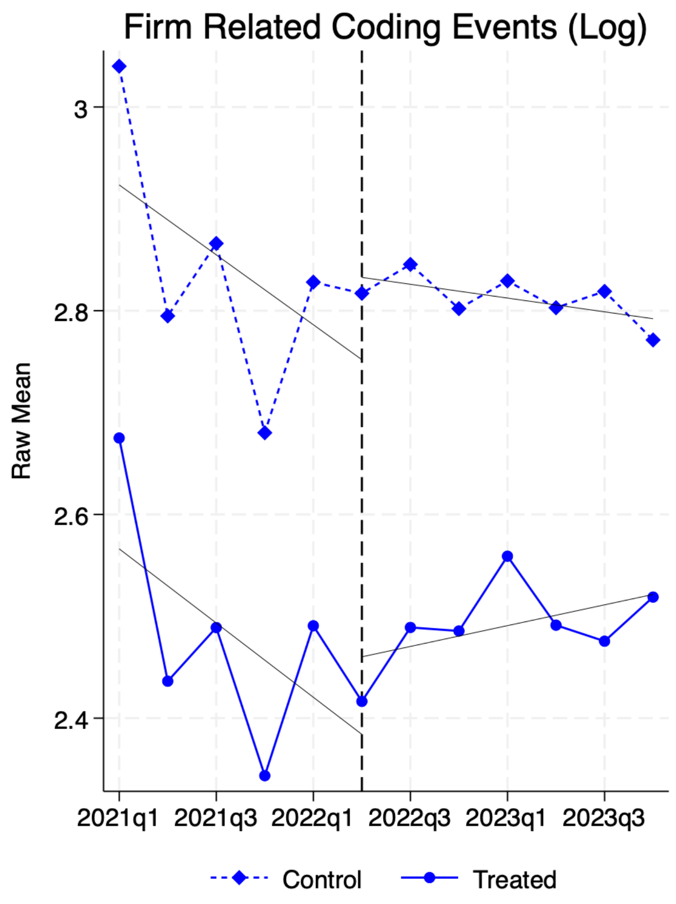

Figure 3. Raw Changes in Coding Productivity of Developers

This figure plots the raw mean of firms' developers coding related events in firm-owned repositories. Specifically, the graph shows the natural logarithms of activity counts per quarter of developers with high (treated) and low (control) AI exposure before and after the introduction of GitHub Copilot.

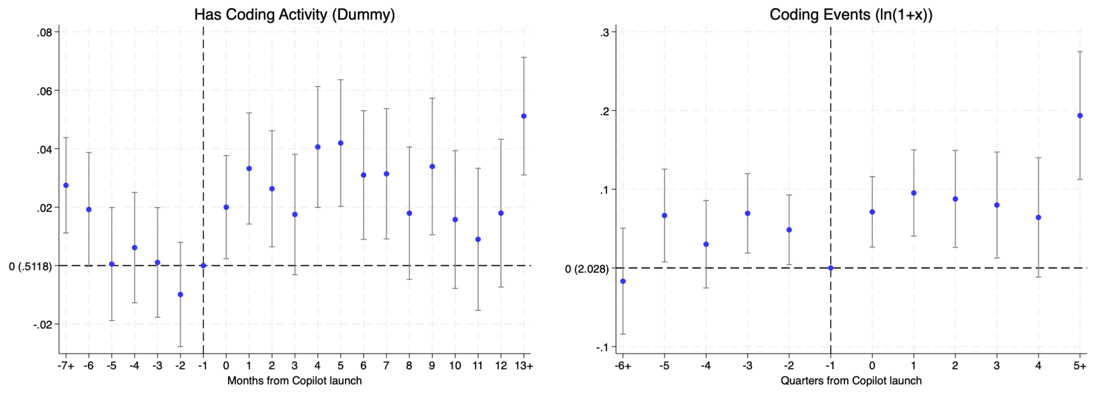

Figure 4. Firm-Related Github Coding Activity After Copilot Launch

This figure plots coefficients of the event study specification described in equation 3 with 95% confidence intervals. The outcome variables are a dummy variable that equals one if a developer has any public activity in firm-owned repositories in a given month (left) and the $log(1 + x)$ transformation of the number of coding activities in firm-owned repositories in a given quarter (right). Standard errors are clustered at developer level.

Table 1. GPT Generated AI Exposure Score of Selected Languages

This table lists the LLM-based Generative AI exposure scores of selected languages, which is later used to calculate developer-level exposure to Generative AI. The score ranges from 0 to 1. See Section Internet Appendix A.3 for the prompt used to obtain AI exposure scores for programming languages.

| High AI Exposure Languages | | Low AI Exposure Languages | | Random without AI Exposure |
|---|---|---|---|---|
| language | score | language | score | language |
| Python | 1.0 | BASIC | 0.4 | BrighterScript |
| C# | 0.9 | LiveScript | 0.4 | CSV |
| Java | 0.9 | Visual Basic 6.0 | 0.4 | Cadence |
| JavaScript | 0.9 | ASP | 0.5 | DTrace |
| Jupyter Notebook | 0.9 | Cython | 0.5 | Futhark |
| TypeScript | 0.9 | Markdown | 0.5 | Inno Setup |
| CSS | 0.8 | SAS | 0.5 | Lex |
| Go | 0.8 | Stata | 0.5 | Oxygene |
| HTML | 0.8 | TeX | 0.5 | Self |
| PHP | 0.8 | VBA | 0.5 | TOML |

Table 2. User-Month Github Activity Summary Statistics (Jan2021-Jun2022)

This table presents summary statistics of user-month GitHub activity before the official launch of GitHub Copilot, i.e., from January 2021 to June 2022. Panel (a) summarizes main outcome variables used in the regression analysis by seniority. Panel (b) summarizes main outcome variables and developer characteristics by AI-exposure level. A developer is considered as senior if the tenure of the developer on the GitHub platform, approximated based on the account's create date, is above median. *High AI Exposure* is a dummy that equals one if the developer's AI exposure score is in the fourth quartile. Gender is inferred based on developer name and LLM-based gender likelihood score. A developer is considered to be male/female when the likelihood score is above 0.5. See Internet Appendix A.2 for the methodology. Activities are grouped based on their related skill requirements. See Section Internet Appendix A.1 for classification details. Repository ownership can be firm or non-firm. The latter includes repositories owned by organization accounts (*org*) or individuals (*ind*). Count variables are winsorized at 95% level.

(a) By Seniority

| | All | | | Senior | | | Junior | | |
|---|---|---|---|---|---|---|---|---|---|
| | Mean | Median | SD | Mean | Median | SD | Mean | Median | SD |
| Coding events | 32.93 | 5.00 | 82.88 | 36.30 | 6.00 | 86.35 | 27.51 | 4.00 | 76.66 |
| General skill events | 10.82 | 1.00 | 35.29 | 13.07 | 1.00 | 38.74 | 7.19 | 0.00 | 28.48 |
| Mixed events | 6.54 | 0.00 | 19.52 | 7.25 | 0.00 | 20.80 | 5.42 | 0.00 | 17.20 |
| Has coding | 0.69 | 1.00 | 0.46 | 0.71 | 1.00 | 0.45 | 0.66 | 1.00 | 0.47 |
| Has general | 0.50 | 1.00 | 0.50 | 0.55 | 1.00 | 0.50 | 0.43 | 0.00 | 0.50 |
| Has mixed | 0.35 | 0.00 | 0.48 | 0.38 | 0.00 | 0.48 | 0.31 | 0.00 | 0.46 |
| AI exposure from push events | 0.81 | 0.84 | 0.13 | 0.82 | 0.84 | 0.13 | 0.81 | 0.84 | 0.14 |
| Active repositories (total) | 4.20 | 2.00 | 7.95 | 4.92 | 2.00 | 8.89 | 3.03 | 1.00 | 5.96 |
| Active repositories (coding events) | 2.88 | 1.00 | 5.67 | 3.27 | 2.00 | 6.20 | 2.25 | 1.00 | 4.64 |
| Active repositories (general events) | 1.36 | 1.00 | 2.67 | 1.64 | 1.00 | 3.07 | 0.90 | 0.00 | 1.78 |
| Active repositories (mixed events) | 0.64 | 0.00 | 1.26 | 0.73 | 0.00 | 1.38 | 0.51 | 0.00 | 1.02 |
| Active repositories (total) (firm) | 1.76 | 1.00 | 3.23 | 1.87 | 1.00 | 3.39 | 1.59 | 1.00 | 2.95 |
| Active repositories (total) (org) | 0.84 | 0.00 | 2.85 | 1.11 | 0.00 | 3.33 | 0.42 | 0.00 | 1.76 |
| Active repositories (total) (ind) | 1.52 | 0.00 | 3.36 | 1.86 | 1.00 | 3.79 | 0.96 | 0.00 | 2.39 |

(b) By Generative AI Exposure

| | High AI Exposure | | | Low AI Exposure | | |
|---|---|---|---|---|---|---|
| | Mean | Median | SD | Mean | Median | SD |
| Female (inferred) | 0.13 | 0.00 | 0.34 | 0.13 | 0.00 | 0.34 |
| Senior developers (GHAPI) | 0.60 | 1.00 | 0.49 | 0.62 | 1.00 | 0.48 |
| Coding events | 24.81 | 3.00 | 72.47 | 35.37 | 6.00 | 85.61 |
| General skill events | 7.17 | 0.00 | 28.04 | 11.91 | 1.00 | 37.12 |
| Mixed events | 4.85 | 0.00 | 16.54 | 7.05 | 0.00 | 20.30 |
| Has coding | 0.64 | 1.00 | 0.48 | 0.71 | 1.00 | 0.45 |
| Has general | 0.44 | 0.00 | 0.50 | 0.52 | 1.00 | 0.50 |
| Has mixed | 0.28 | 0.00 | 0.45 | 0.37 | 0.00 | 0.48 |
| Active repositories (total) (firm) | 1.39 | 1.00 | 2.76 | 1.87 | 1.00 | 3.35 |
| Active repositories (total) (org) | 0.67 | 0.00 | 2.48 | 0.90 | 0.00 | 2.96 |
| Active repositories (total) (ind) | 1.29 | 0.00 | 3.11 | 1.59 | 0.00 | 3.42 |

Table 3. Firm-Related Github Coding Activities After Copilot Launch

This table reports regression results of equation 1 and equation 2. In Columns (1)-(2), the outcome variables are dummy variables that equal one if a developer has any public coding activity in firm-owned repositories in a given month. In Columns (3)-(4), the outcome variables are logarithm transformations of one plus the number of coding activities in firm-owned repositories in a given quarter. *Post* is a dummy that equals one if the time period is after July 2022 (or the third quarter of 2022). *AI Exposure* or *AI* are dummy variables that equal one if the developer's AI exposure score is in the fourth quartile. *Senior* is a dummy that equals one if the developer's tenure is above median. Standard errors are clustered at developer level. Significance: *, $p < 0.1$; **, $p < 0.05$; ***, $p < 0.01$.

| | Has Coding Events | | Ln(1+Coding Events) | |
|---|---|---|---|---|
| | (1) | (2) | (3) | (4) |
| Post×AI Exposure | 0.0116** | 0.0028 | 0.0574*** | -0.0018 |
| | (2.39) | (0.35) | (2.63) | (-0.05) |
| Post×Senior | | -0.0157*** | | -0.0713*** |
| | | (-3.37) | | (-3.31) |
| Post×AI×Senior | | 0.0139 | | 0.0942** |
| | | (1.39) | | (2.09) |
| Total Effect (Senior) | | 0.0167*** | | 0.0924*** |
| | | (2.71) | | (3.35) |
| N | 563,656 | 563,582 | 194,973 | 194,948 |
| Adj. R2 | 0.4040 | 0.4040 | 0.6868 | 0.6868 |
| Individual FE | Y | Y | Y | Y |
| Time FE | Y | Y | Y | Y |

Table 4. Quality Change After Copilot Launch

This table reports regression results of equation 1 and equation 2. In Panel (a), the outcome variables are the $ln(1 + x)$ transformations of the number of stars and issues opened that are associated with developers' work each month. In Panel (b), the outcome variables are the cumulative number of stars, scaled by the number of pushes, and the cumulative number of issues opened, scaled by the number of stars. *Post* is a dummy that equals one if the time period is after July 2022. *AI Exposure* or *AI* are dummy variables that equal one if the developer's AI exposure score is in the fourth quartile. *Senior* is a dummy that equals one if the developer's tenure is above median. The total effects for senior developers (sum of the coeffcients of the post treatment indicator and the interaction term) are reported underneath. Standard errors are clustered at developer level. Significance: *, p < 0.1; **, p < 0.05; ***, p < 0.01.

(a) Log Number of Stars and Issues Opened

| | Ln(1+Stars) | | Ln(1+Issues Opened) | |
|---|---|---|---|---|
| | (1) | (2) | (3) | (4) |
| Post×AI Exposure | 0.0473*** | 0.0164 | 0.0211*** | -0.0080 |
| | (4.91) | (1.14) | (2.89) | (-0.70) |
| Post×Senior | | -0.0477*** | | -0.0351*** |
| | | (-5.24) | | (-4.59) |
| Post×AI×Senior | | 0.0488** | | 0.0461*** |
| | | (2.54) | | (3.11) |
| Total Effect (Senior) | | 0.0652*** | | 0.0381*** |
| | | (5.08) | | (4.02) |
| N | 563,877 | 563,803 | 563,877 | 563,803 |
| Adj. R2 | 0.5883 | 0.5885 | 0.6215 | 0.6216 |
| Individual FE | Y | Y | Y | Y |
| Year-Month FE | Y | Y | Y | Y |

(b) Scaled Number of Stars and Issues Opened

| | Stars Per Push | | Issues Opened Per Star | |
|---|---|---|---|---|
| | (1) | (2) | (3) | (4) |
| Post×AI Exposure | 0.0141 | 0.0001 | -0.0287*** | -0.0112 |
| | (1.46) | (0.00) | (-2.81) | (-0.65) |
| Post×Senior | | -0.0167* | | 0.0244** |
| | | (-1.74) | | (2.43) |
| Post×AI×Senior | | 0.0222 | | -0.0268 |
| | | (1.07) | | (-1.26) |
| Total Effect (Senior) | | 0.0222* | | -0.0381*** |
| | | (1.92) | | (-2.98) |
| N | 452,142 | 452,075 | 404,663 | 404,630 |
| Adj. R2 | 0.9736 | 0.9736 | 0.9599 | 0.9599 |
| Individual FE | Y | Y | Y | Y |
| Year-Month FE | Y | Y | Y | Y |

Table 5. Hours Spent on Core Activity of Firm-Owned Projects After Copilot Launch

This table reports regression results of equation 1 and equation 2. The outcome variables include the cumulative ratio of core events occurring outside common hours, the cumulative ratio of core events occurring on weekends, and the cumulative number of core events per hour. Core events are defined in Section Internet Appendix A.1. Common hours are defined as hours during which a developer completes events that constitute more than 5% of all events on a given weekday, based on 2020 activity records (with at least 100 events). Only activities related to firm-owned repositories are considered. *Post* is a dummy that equals one if the time period is after July 2022. *AI Exposure* or *AI* are dummy variables that equal one if the developer's AI exposure score is in the fourth quartile. *Senior* is a dummy that equals one if the developer's tenure is above median. The total effects for senior developers (sum of the coeffcients of the post treatment indicator and the interaction term) are reported underneath. Standard errors are clustered at developer level. Significance: *, p < 0.1; **, p < 0.05; ***, p < 0.01.

| | Outside Common Hours | | Weekends | | Events Per Hour | |
|---|---|---|---|---|---|---|
| | (1) | (2) | (3) | (4) | (5) | (6) |
| Post×AI Exposure | 0.0041 | 0.0003 | 0.0012 | -0.0006 | 0.0109** | -0.0015 |
| | (1.58) | (0.08) | (1.06) | (-0.36) | (2.02) | (-0.17) |
| Post×Senior | | 0.0002 | | 0.0001 | | -0.0307*** |
| | | (0.11) | | (0.07) | | (-5.54) |
| Post×AI×Senior | | 0.0059 | | 0.0031 | | 0.0190* |
| | | (1.12) | | (1.30) | | (1.70) |
| Total Effect (Senior) | | 0.0062* | | 0.0024 | | 0.0176** |
| | | (1.91) | | (1.57) | | (2.55) |
| N | 165,364 | 165,344 | 509,468 | 509,401 | 509,468 | 509,401 |
| Adj. R2 | 0.8946 | 0.8946 | 0.8067 | 0.8067 | 0.9101 | 0.9101 |
| Individual FE | Y | Y | Y | Y | Y | Y |
| Year-Month FE | Y | Y | Y | Y | Y | Y |

Table 6. Firm-Owned Open-Source Innovation Activity After Copilot Launch

This table reports regression results of equation 1 and equation 2. In Columns (1)-(2), the outcome variables are a dummy that equals one if a developer initiated at least one new firm-owned repository (project) in a given quarter. In Column (3)-(4), the outcome variables are number of newly initiated projects of a developer in a given quarter. *Post* is a dummy that equals one if the time period is after the third quarter of 2022. *AI Exposure* or *AI* are dummy variables that equal one if the developer's AI exposure score is in the fourth quartile. *Senior* is a dummy that equals one if the developer's tenure is above median. The total effects for senior developers (sum of the coeffcients of the post treatment indicator and the interaction term) are reported underneath. Standard errors are clustered at developer level. Significance: *, p < 0.1; **, p < 0.05; ***, p < 0.01.

| | Initiated Project | | # of Initiated Project | |
|---|---|---|---|---|
| | (1) | (2) | (3) | (4) |
| Post×AI Exposure | -0.0021 | -0.0065$^{***}$ | -0.0012 | 0.0002 |
| | (-1.42) | (-2.66) | (-0.39) | (0.03) |
| Post×Senior | | -0.0010 | | 0.0059 |
| | | (-0.75) | | (1.29) |
| Post×AI×Senior | | 0.0071$^{**}$ | | -0.0021 |
| | | (2.30) | | (-0.29) |
| Total Effect (Senior) | | 0.0006 | | -0.0019 |
| | | (0.32) | | (-0.61) |
| N | 194,973 | 194,948 | 194,973 | 194,948 |
| Adj. R2 | 0.0594 | 0.0594 | 0.1430 | 0.1430 |
| Individual FE | Y | Y | Y | Y |
| Year-Quarter FE | Y | Y | Y | Y |

Table 7. Value of Firm's Open-Source Innovation After Copilot Launch

This table reports regression results of equation 4. In Panel (a), the outcome variables are the natural logarithm of one plus the number of stars received as of February 2024. In Panel (b), the outcome variables are the natural logarithm of repository value estimated based on stock-market reaction within three days of the project's public release. See Emery et al. (2024) for methodology details. *Post* is a dummy that equals one if the time period is after July 2022. *Innovator AI Expo* or *AI* are dummy variables that equals one if the AI exposure score of at least one member of the innovator team is in the fourth quartile. *Repo AI Expo* is a dummy that equals one of the repository's AI exposure score, based on repository's language composition, is in the fourth quartile. *Senior* is the share of developers with tenure above median. Control variables include the natural logarithms of one plus cumulative number of firm-owned repository, market capitalization, volatility, number of employees, and one plus value of patent portfolio. I also control for return on assets, R&D expenditure as a share of assets, whether R&D expenditure is missing, and innovator team size. All firm-year control variables are one-year lagged and winsorized at 1% and 99% levels. See Table IA1 for variable descriptions and sources. Standard errors are clustered at firm level. Significance: *, p < 0.1; **, p < 0.05; ***, p < 0.01.

(a) Dependent Variable: Ln(1+Stars)

| | (1) | (2) | (3) | (4) |
|---|---|---|---|---|
| Innovator AI Expo | 0.0636 | | -0.0942 | -0.3471 |
| | (0.24) | | (-0.29) | (-0.79) |
| Post×Innovator AI Expo | 0.4382** | | 0.5481** | 0.7181 |
| | (2.21) | | (2.18) | (1.09) |
| Repo AI Expo | | 0.4729 | 0.4968 | 0.3606 |
| | | (1.41) | (1.28) | (1.12) |
| Post×Repo AI Expo | | 0.0357 | -0.0028 | 0.0864 |
| | | (0.16) | (-0.01) | (0.36) |
| Senior | | | | 0.4660** |
| | | | | (2.36) |
| Post×Senior | | | | 0.0060 |
| | | | | (0.03) |
| Innovator AI×Senior | | | | 0.5964 |
| | | | | (1.37) |
| Post×AI×Senior | | | | -0.3603 |
| | | | | (-0.39) |
| N | 1,995 | 1,995 | 1,995 | 1,666 |
| Adj. R2 | 0.2642 | 0.2767 | 0.2795 | 0.2901 |
| Firm FE | Y | Y | Y | Y |
| Year-Month FE | Y | Y | Y | Y |

(b) Dependent Variable: Ln(Repo Value)

| | (1) | (2) | (3) | (4) |
|---|---|---|---|---|
| Innovator AI Expo | 0.0523** | | 0.0557 | 0.0304 |
| | (2.06) | | (1.40) | (0.31) |
| Post×Innovator AI Expo | 0.0796* | | 0.0747 | 0.3012** |
| | (1.68) | | (1.64) | (2.30) |
| Repo AI Expo | | 0.0032 | -0.0105 | -0.0484 |
| | | (0.06) | (-0.17) | (-0.94) |
| Post×Repo AI Expo | | 0.0251 | 0.0345 | 0.0679 |
| | | (0.40) | (0.50) | (1.16) |
| Senior | | | | 0.0363 |
| | | | | (0.44) |
| Post×Senior | | | | 0.0342 |
| | | | | (0.31) |
| Innovator AI×Senior | | | | 0.0715 |
| | | | | (0.46) |
| Post×AI×Senior | | | | -0.4256** |
| | | | | (-2.27) |
| N | 1,995 | 1,995 | 1,995 | 1,666 |
| Adj. R2 | 0.8477 | 0.8473 | 0.8476 | 0.8437 |
| Firm FE | Y | Y | Y | Y |
| Year-Month FE | Y | Y | Y | Y |

Table 8. Job Changes of Firm Developers on GitHub After Copilot Launch

This table reports regression results of equation 1 and 2 at the individual-year-quarter level. Panel (a) examines the impact of generative AI on developers' job changes. In Columns (1)-(2), the outcome variable is a binary indicator equal to one if a developer starts a new job in a given quarter. In Columns (3)-(4), the outcome variable is limited to new job positions outside previous employers. Panel (b) explores heterogeneous effects on promotion and demotion among innovators (i.e., GitHub project initiators) and non-innovators based on developer characteristics. In this panel, Columns (1)-(2) focus on senior developers with above-median tenure on GitHub, while Columns (3)-(6) present results for junior developers. Further, Columns (5) and (6) divide the junior subsample into those with non-innovative previous employers (i.e., firms with below-median R&D expenditure as a share of total assets) and those with innovative employers. The outcome variables are either *Promotion*, which equals one if the new job position offers higher total compensation or higher seniority, or *Demotion* if the new job position has lower total compensation or a lower seniority rank. *Post* is a dummy that equals one if the time period is after 2022Q3. *AI Exposure* is a dummy variable that equals one if the language's AI exposure score is in the fourth quartile. *Senior* is a dummy that equals one if the developer's tenure is above median. *Innovator* is a dummy that equals one if the developer has initiated at least one project prior to 2022Q3. Control variables include seniority and the natural log of total compensation of the developer's previous position. Standard errors are clustered at the developer level. Significance: *, $p < 0.1$; **, $p < 0.05$; ***, $p < 0.01$.

(a) Job Change After Copilot Launch

| | Job Change | | Across-Firm Job Change | |
|---|---|---|---|---|
| | (1) | (2) | (3) | (4) |
| Post×AI Exposure | 0.0025 | 0.0121** | 0.0064** | 0.0122*** |
| | (0.74) | (2.16) | (2.45) | (2.79) |
| Post×Senior | | 0.0032 | | 0.0055** |
| | | (0.89) | | (2.02) |
| Post×AI×Senior | | -0.0153** | | -0.0091* |
| | | (-2.24) | | (-1.72) |
| N | 115,864 | 115,864 | 115,864 | 115,864 |
| Adj. R2 | 0.1111 | 0.1112 | 0.1468 | 0.1468 |
| Individual FE | Y | Y | Y | Y |
| Firm-Time FE | Y | Y | Y | Y |
| Controls | Y | Y | Y | Y |

(b) Across-Firm Promotion and Demotion By Developer Characteristics

| | Senior Developers | | Junior Developers | | | |
|---|---|---|---|---|---|---|
| From Firms | All | | All | | Non-Innovative | Innovative |
| | Promotion | Demotion | Promotion | Demotion | Promotion | Promotion |
| | (1) | (2) | (3) | (4) | (5) | (6) |
| Post×AI Exposure | 0.0012 | -0.0001 | 0.0026 | 0.0026 | 0.0027 | 0.0063 |
| | (0.57) | (-0.03) | (0.86) | (1.07) | (0.74) | (1.31) |
| Post×Innovator | 0.0009 | -0.0007 | -0.0025 | -0.0041** | -0.0039 | 0.0019 |
| | (0.38) | (-0.39) | (-0.79) | (-2.14) | (-1.28) | (0.39) |
| Post×AI×Innovator | -0.0090** | -0.0001 | 0.0127* | 0.0075 | 0.0132 | 0.0045 |
| | (-2.04) | (-0.02) | (1.71) | (1.25) | (1.20) | (0.42) |
| N | 73,283 | 73,283 | 40,288 | 40,288 | 19,778 | 18,737 |
| Adj. R2 | 0.1225 | 0.0955 | 0.1300 | 0.1095 | 0.2520 | 0.1193 |
| Individual FE | Y | Y | Y | Y | Y | Y |
| Firm-Time FE | Y | Y | Y | Y | Y | Y |
| Controls | Y | Y | Y | Y | Y | Y |

Table 9. Cumulative Abnormal Returns After Copilot Launch

This table reports event study results after Copilot launch. Panel (a) uses the sample of firms in the information technology industry (with 3-digit SIC code as 737) that are active on the GitHub platform before the event day (with cumulative number of pushes higher than 100). Panel (b) splits the sample based on firms' innovativeness. Specifically, firms with R&D expenditure as a share of total assets higher than the median are classified as innovative firms. *AI Exposure* is calculated based on languages used in firm-owned projects. *Aligned* is a dummy that equals to one if an innovative firm has an average employees' tenure in the first quartile or or if a non-innovative firm has an average employees' tenure in the fourth quartile. Control variables include the natural logarithms of market capitalization and cumulative number of pushes, revenue growth, profitability, R&D expenditure as a share of assets, and whether R&D expenditure is missing. All firm-year control variables are one-year lagged and winsorized at 1% and 99% levels. See Table IA1 for variable descriptions and sources. Robust standard errors are used. Significance: *, $p < 0.1$; **, $p < 0.05$; ***, $p < 0.01$.

(a) Information Technology Firms Active on GitHub

| | Day 0 AR | 10-Day CAR | 20-Day CAR | 30-Day CAR |
|---|---|---|---|---|
| | (1) | (2) | (3) | (4) |
| AI Exposure | 0.0064 | -4.9531* | -8.9922*** | -8.5442* |
| | (0.01) | (-1.94) | (-2.98) | (-1.94) |
| Aligned | 1.2194 | -7.2286** | -9.0036** | -8.7373* |
| | (1.38) | (-2.53) | (-2.28) | (-1.82) |
| AI Exposure×Aligned | -2.6533* | 11.3800** | 16.3546** | 16.2955** |
| | (-1.67) | (2.27) | (2.50) | (1.97) |
| N | 191 | 191 | 191 | 191 |
| Adj. R2 | 0.0540 | 0.0646 | 0.1655 | 0.1522 |

(b) Non-Innovative vs. Innovative Tech Firms

| | Non-Innovative Firms | | | Innovative Firms | | |
|---|---|---|---|---|---|---|
| | 10-Day CAR | 20-Day CAR | 30-Day CAR | 10-Day CAR | 20-Day CAR | 30-Day CAR |
| | (1) | (2) | (3) | (4) | (5) | (6) |
| AI Exposure | -5.8487 | -9.4358** | -8.2365 | -4.3185 | -8.4402 | -8.1433 |
| | (-1.58) | (-2.62) | (-1.24) | (-1.10) | (-1.64) | (-1.30) |
| Aligned | -10.1184** | -13.3066** | -11.4851 | -3.5804 | -2.5088 | -4.5615 |
| | (-2.09) | (-2.22) | (-1.64) | (-1.03) | (-0.47) | (-0.58) |
| AI Exposure×Aligned | 16.6087** | 21.9545** | 19.9154 | 5.3011 | 7.8709 | 9.9377 |
| | (2.06) | (2.17) | (1.58) | (0.87) | (0.94) | (0.80) |
| N | 96 | 96 | 96 | 95 | 95 | 95 |
| Adj. R2 | 0.0869 | 0.1948 | 0.1596 | 0.0288 | 0.0613 | 0.0463 |

## Appendix A Theoretical Appendix

In the spirit of Holmstrm (1999), I develop a stylized model to characterize the effects of GenAI on effort allocation between AI-assisted tasks (such as coding) and tasks unaffected by AI (such as innovation) through the the productivity and signaling channels.

### Appendix A.1 Model Setup

Consider developers work for their employers' open-source projects on GitHub. They perform two tasks in their daily workflow: task that can be assisted by AI (such as coding) and task that requires more human creativity (such as innovation). Suppose developers with ability $\theta$ make a one-time choice of efforts $e_i$ allocated to two types of tasks $i = 1, 2$ at personal cost $C(e_1, e_2) = \frac{1}{2}(e_1^2 + e_2^2)$.

Absent GenAI, developers' efforts produce outputs:

$$\begin{aligned} y_1 &= b_1 e_1 + \theta + \varepsilon_1, \\ y_2 &= e_2 + \theta + \varepsilon_2. \end{aligned}$$

**Production with GenAI** I adapt the benchmark outputs by allowing GenAI to blur the informational content of code. Specifically, the observable output from the AI-assisted task is a combination of developers' own contribution and AI-generated content. Let $\lambda \in [0, 1]$ captures the share of AI component and $\lambda \sim \text{Beta}(\alpha_\lambda, \beta_\lambda)$ with mean $\mu_\lambda$ and variance $\sigma_\lambda^2$. $\lambda$ is independently distributed and incompletely known to both developers and firms.

Developers allocate efforts $(e_1, e_2)$ across two tasks ($i = 1$: AI-assisted coding, $i = 2$: innovation). Effort costs remain quadratic with $C(e_1, e_2) = \frac{1}{2}(e_1^2 + e_2^2)$. With GenAI, developers' efforts produce outputs

$$\begin{aligned} y_1 &= b_1 e_1 + \underbrace{(1 - \lambda)\theta}_{\text{human}} + \underbrace{\lambda(\rho\theta + g)}_{\text{GenAI}} + \varepsilon_1, \quad \rho \in [0, 1], \\ y_2 &= e_2 + \theta + \varepsilon_2, \end{aligned}$$

where $g \sim \mathcal{N}(0, \sigma_g^2)$, $\varepsilon_i \sim \mathcal{N}(0, \sigma_i^2)$, and $\theta \sim \mathcal{N}(0, \sigma_\theta^2)$. I take $g$, $\varepsilon_i$, and $\theta$ as uncorrelated.

Note that the output process of task 1 captures three elements of GenAI's influence. First, GenAI may directly improve productivity by allowing higher marginal return to effort (higher $b_1$). Second, GenAI could reduce the visibility of ability from the observable output (lower $[1-\lambda(1-\rho)]$) and increase uncertainty unless the AI-generated content can fully reflect ability ($\rho = 1$). Third, GenAI may introduce additional noise in production ($\sigma_g^2$) due to factors like model temperature and hallucination.

**Information structure** Current risk-neutral employers observe $(\theta, e_1, e_2)$, so the competitive wage is

$$\begin{aligned} B(e_1, e_2; \theta) &= \mathbb{E}[y_1 + y_2 | \theta, e_1, e_2] \\ &= b_1 e_1 + e_2 + [2 - \mu_\lambda(1-\rho)]\theta. \end{aligned}$$

In contrast, the market (future employers) observes only the signals $(y_1, y_2)$.

**Purged signals** While efforts are unobservable, the market can infer $(e_1^*, e_2^*)$ by solving the maximization problem in equilibrium. Thus, observing $(y_1, y_2)$ is equivalent to observing the purged signals below:

$$\begin{aligned} x_1 &:= y_1 - b_1 e_1^* = \kappa_\theta \theta + \eta_1, \\ x_2 &:= y_2 - e_2^* = \theta + \varepsilon_2, \end{aligned}$$

where

$$\kappa_\theta = 1 - \mu_\lambda(1-\rho),$$

$$\eta_1 = (\lambda - \mu_\lambda)(\rho - 1)\theta + \lambda g + \varepsilon_1, \quad \mathbb{E}(\eta_1) = 0, \ \ \mathrm{Cov}(\eta_1, \theta) = 0.$$

$\kappa_\theta$ reflects signal attenuation caused by AI-generated content, unless AI content can also fully reflect developers' ability ($\rho = 1$). GenAI also introduces higher noise in the signal, including uncertainties about AI share ($\sigma_\lambda^2$) and AI quality ($\sigma_g^2$). To see this, the

AI-induced noise variance in $x_1$ is

$$\sigma_{\eta_1}^2 = \mathrm{Var}(\eta_1) = \underbrace{(1-\rho)^2 \sigma_\lambda^2 \sigma_\theta^2}_{\text{AI-share uncertainty}} + \underbrace{\mathbb{E}[\lambda^2]\,\sigma_g^2}_{\text{AI-quality uncertainty}} + \underbrace{\sigma_1^2}_{\text{measurement noise}},$$

with

$$\mathbb{E}[\lambda^2] = \mu_\lambda^2 + \sigma_\lambda^2 = \frac{\alpha_\lambda(\alpha_\lambda+1)}{(\alpha_\lambda+\beta_\lambda)(\alpha_\lambda+\beta_\lambda+1)}.$$

**Posterior belief** Define precisions

$$\tau_1 = \frac{\kappa_\theta^2}{\sigma_{\eta_1}^2}, \quad \tau_2 = \frac{1}{\sigma_2^2}, \quad \tau_\theta = \frac{1}{\sigma_\theta^2},$$

and $T = \tau_1 + \tau_2 + \tau_\theta$.

For tractability I use the best linear estimator that minimizes mean squared error to approximate the posterior mean (LMMSE)[A1]

$$\hat{\theta} = \mathbb{E}[\theta \mid x_1, x_2] \approx \alpha_1 x_1 + \alpha_2 x_2,$$

where the weights are

$$\alpha_1 = \frac{\tau_1}{\kappa_\theta T}, \qquad \alpha_2 = \frac{\tau_2}{T}.$$

**Developer's utility** the risk-neutral, career-concerned developer's expected utility is

$$\begin{aligned}\mathbb{E}[U(e_1,e_2;\theta)|\theta] &= b_1 e_1 + e_2 + [2 - \mu_\lambda(1-\rho)]\theta + \beta[\alpha_1 x_1 + \alpha_2 x_2] - \frac{1}{2}(e_1^2 + e_2^2) \\ &= b_1 e_1 + e_2 + [2 - \mu_\lambda(1-\rho)]\theta \\ &\quad + \beta[\alpha_1(b_1 e_1 + \kappa_\theta \theta - b_1 e_1^*) + \alpha_2(e_2 + \theta - e_2^*)] \\ &\quad - \frac{1}{2}(e_1^2 + e_2^2).\end{aligned}$$

[A1]LMMSE is suboptimal here because $(x_1, x_2, \theta)$ are not jointly normally distributed. However, the model is linear-Gaussian conditional on $\lambda$. Integrating over the Beta prior for $\lambda$ induces a mean-zero heteroskedastic error in $x_1$, and the approximation is accurate when $(1-\rho)^2\sigma_\lambda^2$ is small. Moreover, the linear equilibrium incentives, where the second-order relationships matter (how pay covaries with outputs), are preserved.

The developer's utility function is composed of three parts: direct payoff from the current employer, reputational benefits from signaling $W(\hat{\theta}) = \hat{\theta}$ with parameter $\beta$ capturing the degree of career concern, and personal cost of efforts.

**Equilibrium effort choice** Conditional on the conjectured equilibrium efforts $(e_1^*, e_2^*)$, the developer maximizes $\mathbb{E}[U \mid \theta]$. The first-order conditions are

$$\begin{aligned}\frac{\partial \mathbb{E}[U \mid \theta]}{\partial e_1} &= b_1 + \beta\alpha_1 b_1 - e_1 \;=\; 0,\\ \frac{\partial \mathbb{E}[U \mid \theta]}{\partial e_2} &= 1 + \beta\,\alpha_2 \;-\; e_2 \;=\; 0,\end{aligned}$$

yielding optimal efforts

$$\begin{aligned}e_1^* &= b_1 + \beta\alpha_1 b_1,\\ e_2^* &= 1 + \beta\,\alpha_2.\end{aligned}$$

## Appendix A.2 Comparative Statics

In this section, I focus on the partial equilibrium effects of GenAI on effort allocation from three perspectives: productivity improvement, visibility of ability (AI-share level and visibility in AI-generated content), and uncertainty (AI-share uncertainty and AI-quality uncertainty).

Recall

$$e_1^* = b_1 + \beta\alpha_1 b_1, \quad e_2^* = 1 + \beta\,\alpha_2,$$

with $\alpha_1 = \dfrac{\tau_1}{\kappa_\theta T}$, $\alpha_2 = \dfrac{\tau_2}{T}$, $\tau_1 = \dfrac{\kappa_\theta^2}{\sigma_{\eta_1}^2}$, $\tau_2 = \dfrac{1}{\sigma_2^2}$, $\tau_\theta = \dfrac{1}{\sigma_\theta^2}$, $T = \tau_1 + \tau_2 + \tau_\theta$, and $\sigma_{\eta_1}^2 = (1-\rho)^2\,\sigma_\lambda^2\,\sigma_\theta^2 + (\mu_\lambda^2 + \sigma_\lambda^2)\sigma_g^2 + \sigma_1^2$.

### Appendix A.2.1 Productivity Gains ($b_1$)

GenAI raises the marginal return to task 1's effort in two ways: directly in current output and via the career term. The effects are:

$$\frac{\partial e_1^*}{\partial b_1} = 1 + \beta\alpha_1 > 0, \qquad \frac{\partial e_2^*}{\partial b_1} = 0.$$

The term $\beta\alpha_1$ shows that stronger career concerns (larger $\beta$) and a more informative signal from task 1 (larger $\alpha_1$) increase the responsiveness of $e_1^*$ to productivity improvements. Instead, effort allocation to task 2 is not affected.

### Appendix A.2.2 Higher AI-Share Level ($\mu_\lambda$)

Note that $\alpha_1 = \dfrac{\tau_1}{\kappa_\theta T} = \dfrac{\kappa_\theta}{\kappa_\theta^2 + \sigma_{\eta_1}^2(\tau_2 + \tau_\theta)}$, $\kappa_\theta = 1 - \mu_\lambda(1-\rho)$.

Since $\mu_\lambda$ affects both $\kappa_\theta$ and $\sigma_{\eta_1}^2$,

$$\begin{aligned}
\frac{\partial \alpha_1}{\partial \kappa_\theta} &= \frac{\sigma_{\eta_1}^2(\tau_2+\tau_\theta) - \kappa_\theta^2}{[\kappa_\theta^2 + \sigma_{\eta_1}^2(\tau_2+\tau_\theta)]^2},\\
\frac{\partial \alpha_1}{\partial \sigma_{\eta_1}^2} &= -\frac{\kappa_\theta(\tau_2+\tau_\theta)}{[\kappa_\theta^2 + \sigma_{\eta_1}^2(\tau_2+\tau_\theta)]^2},\\
\frac{\partial \kappa_\theta}{\partial \mu_\lambda} &= -(1-\rho),\\
\frac{\partial \sigma_{\eta_1}^2}{\partial \mu_\lambda} &= 2\mu_\lambda\sigma_g^2.
\end{aligned}$$

Applying chain rule yields the total derivative

$$\frac{de_1^*}{d\mu_\lambda} = -\frac{\beta b_1}{[\kappa_\theta^2 + \sigma_{\eta_1}^2(\tau_2+\tau_\theta)]^2}\{(1-\rho)[\sigma_{\eta_1}^2(\tau_2+\tau_\theta) - \kappa_\theta^2] + 2\kappa_\theta\mu_\lambda(\tau_2+\tau_\theta)\sigma_g^2\} \gtrless 0.$$

When $\sigma_{\eta_1}^2(\tau_2+\tau_\theta) < \kappa_\theta^2$, the sign of the derivative $\dfrac{de_1^*}{d\mu_\lambda}$ can be positive. However, it means task 1 is almost precise ($\sigma_{\eta_1}^2$ is small) while alternative signals are very noisy

($(\tau_2+\tau_\theta)$ is small), which is empirically unlikely. Thus, in reality an increase in prior mean of AI share will likely result in less effort in the AI-assisted task, as it both attenuates ability loading ($\kappa_\theta$) and raises residual variance ($\sigma^2_{\eta_1}$).

Similarly, since $\alpha_2 = \tau_2/T$,

$$\frac{d\alpha_2}{d\mu_\lambda} = -\frac{\tau_2}{T^2}\frac{d\tau_1}{d\mu_\lambda} = \frac{\tau_2}{T^2}\frac{2\big[\kappa_\theta(1-\rho)\sigma^2_{\eta_1} + \kappa_\theta^2\,\mu_\lambda\,\sigma_g^2\big]}{(\sigma^2_{\eta_1})^2} \;>\; 0,$$

whenever $\rho < 1$ or $\sigma_g^2 > 0$. Therefore,

$$\frac{de_2^*}{d\mu_\lambda} = \beta\,\frac{d\alpha_2}{d\mu_\lambda} \;>\; 0.$$

Hence, when career concern exists ($\beta > 0$), a higher mean AI share reduces $\tau_1$ and shifts weight toward $x_2$, so $e_2^*$ rises. The effect vanishes only when $\rho = 1$ and $\sigma_g^2 = 0$.

### Appendix A.2.3 Visibility of Ability in AI Component ($\rho$)

First start with

$$\frac{d\kappa_\theta}{d\rho} = \mu_\lambda, \qquad \frac{\partial\sigma^2_{\eta_1}}{\partial\rho} = -2(1-\rho)\,\sigma_\lambda^2\,\sigma_\theta^2.$$

Use the chain rule:

$$\frac{d\alpha_1}{d\rho} = \frac{\mu_\lambda\big[(\tau_2+\tau_\theta)\sigma^2_{\eta_1} - \kappa_\theta^2\big] + 2\kappa_\theta(\tau_2+\tau_\theta)(1-\rho)\sigma_\lambda^2\sigma_\theta^2}{\big[\kappa_\theta^2 + (\tau_2+\tau_\theta)\sigma^2_{\eta_1}\big]^2},$$

$$\frac{d\alpha_2}{d\rho} = \frac{\tau_2\big[-2\mu_\lambda\,\kappa_\theta\,\sigma^2_{\eta_1} - 2\kappa_\theta^2(1-\rho)\sigma_\lambda^2\sigma_\theta^2\big]}{\big[\kappa_\theta^2 + (\tau_2+\tau_\theta)\sigma^2_{\eta_1}\big]^2}.$$

Therefore,

$$\frac{de_1^*}{d\rho} = \beta\,b_1\,\frac{d\alpha_1}{d\rho} \;\gtrless\; 0, \qquad \frac{de_2^*}{d\rho} = \beta\,\frac{d\alpha_2}{d\rho} \;\le 0.$$

Similar to the partial equilibrium effect of prior mean of AI share, the effect on effort allocated to task 1 is ambiguous but tends to be positive in reality. Higher $\rho$ makes sure that ability can also be reflected in AI-generated content (larger $\kappa_\theta$) and reduces the share-opacity noise (smaller $\sigma^2_{\eta_1}$), so it helps to restore the distortion introduced by AI-induced noise, and $e_1^*$ increases while $e_2^*$ decreases when career concern exists ($\beta > 0$).

### Appendix A.2.4 Higher AI-Share Uncertainty ($\sigma^2_\lambda$)

First, for task 1,

$$\frac{\partial e_1^*}{\partial \sigma^2_\lambda} = \beta\, b_1 \frac{\partial \alpha_1}{\partial \sigma^2_\lambda} = -\beta\, b_1 \frac{\kappa_\theta(\tau_2+\tau_\theta)}{(\kappa_\theta^2 + (\tau_2+\tau_\theta)\sigma^2_{\eta_1})^2}\Big[(1-\rho)^2\sigma^2_\theta + \sigma^2_g\Big] \;\leq\; 0.$$

Next we move to task 2:

$$\alpha_2 = \frac{\tau_2}{\tau_1+\tau_2+\tau_\theta} = \frac{\tau_2\sigma^2_{\eta_1}}{\kappa_\theta^2 + \sigma^2_{\eta_1}(\tau_2+\tau_\theta)}, \qquad \kappa_\theta = 1-\mu_\lambda(1-\rho).$$

Hence

$$\frac{\partial \alpha_2}{\partial \sigma^2_{\eta_1}} = \frac{\tau_2\,\kappa_\theta^2}{(\kappa_\theta^2 + (\tau_2+\tau_\theta)\sigma^2_{\eta_1})^2} > 0, \qquad \frac{\partial \sigma^2_{\eta_1}}{\partial \sigma^2_\lambda} = (1-\rho)^2\sigma^2_\theta + \sigma^2_g \geq 0,$$

and

$$\frac{\partial e_2^*}{\partial \sigma^2_\lambda} = \beta\,\frac{\partial \alpha_2}{\partial \sigma^2_{\eta_1}}\frac{\partial \sigma^2_{\eta_1}}{\partial \sigma^2_\lambda} \;\geq\; 0.$$

When career concern exists ($\beta > 0$), more uncertainty about the share of AI component increases the residual variance of the task 1's signal, so $e_1^*$ decreases and $e_2^*$ rises.

### Appendix A.2.5 Higher AI-Quality Uncertainty ($\sigma^2_g$)

With

$$\frac{\partial \sigma^2_{\eta_1}}{\partial \sigma^2_g} = \mu_\lambda^2 + \sigma^2_\lambda \;>\; 0,$$

and by the chain rule

$$\frac{\partial \alpha_1}{\partial \sigma^2_g} = \frac{\partial \alpha_1}{\partial \sigma^2_{\eta_1}}\frac{\partial \sigma^2_{\eta_1}}{\partial \sigma^2_g} = -\,\frac{\kappa_\theta\,(\tau_2+\tau_\theta)}{\big[\kappa_\theta^2 + (\tau_2+\tau_\theta)\sigma^2_{\eta_1}\big]^2}\,(\mu_\lambda^2+\sigma^2_\lambda) \;\leq\; 0,$$

$$\frac{\partial \alpha_2}{\partial \sigma^2_g} = \frac{\partial \alpha_2}{\partial \sigma^2_{\eta_1}}\frac{\partial \sigma^2_{\eta_1}}{\partial \sigma^2_g} = \frac{\tau_2\,\kappa_\theta^2}{\big[\kappa_\theta^2 + (\tau_2+\tau_\theta)\sigma^2_{\eta_1}\big]^2}\,(\mu_\lambda^2+\sigma^2_\lambda) \;\geq\; 0.$$

Therefore,

$$\frac{\partial e_1^*}{\partial \sigma_g^2} = \beta\, b_1 \frac{\partial \alpha_1}{\partial \sigma_g^2} = -\beta\, b_1 \frac{\kappa_\theta\,(\tau_2+\tau_\theta)}{\left[\kappa_\theta^2+(\tau_2+\tau_\theta)\sigma_{\eta 1}^2\right]^2}\,(\mu_\lambda^2+\sigma_\lambda^2) \;\le\; 0,$$

$$\frac{\partial e_2^*}{\partial \sigma_g^2} = \beta \frac{\partial \alpha_2}{\partial \sigma_g^2} = \beta \frac{\tau_2\,\kappa_\theta^2}{\left[\kappa_\theta^2+(\tau_2+\tau_\theta)\sigma_{\eta 1}^2\right]^2}\,(\mu_\lambda^2+\sigma_\lambda^2) \;\ge\; 0.$$

Similar to uncertainty about AI share, when career concern exists ($\beta > 0$), a more volatile AI output (larger $\sigma_g^2$) increases the noise the task 1's signal, so the market lowers weight on $x_1$ and rises weight on $x_2$, shifting effort from task 1 to task 2.

### Appendix A.2.6 Summary of Comparative Statics

In summary, GenAI can affect effort allocation through both productivity and signaling channels, as summarized in Table A1. On the productivity side, a higher marginal return in the AI-assisted task raises $e_1^*$ one-for-one and is further amplified by career concerns, while $e_2^*$ is unchanged. On the signaling side, GenAI typically weakens the information content of output from task 1. A higher mean AI share limits visibility of ability from AI-assisted signals and introduce additional noise. Similarly, uncertainties about AI share and AI quality reduce AI-assisted task's precision. Therefore, when career concerns are strong and signaling incentives matter, developers may shift efforts to task 2 (innovation). Greater visibility of ability in AI content ($\rho \uparrow$) strengthens the signal and shift efforts back to task 1. Importantly, all signaling effects disappear without career concerns ($\beta = 0$) or with complete information on $\theta$, and exceptions occur only when AI adds no informational distortion (e.g., $\rho = 1$ and $\sigma_g^2 = 0$).

Table A1. Summary of Partial-Equilibrium Effects of GenAI on Effort Allocation

| **Channel** | **Parameter** | **Effort response** | **Mechanism** |
|---|---|---|---|
| Productivity | Marginal return to effort ($b_1 \uparrow$) | $\dfrac{\partial e_1^*}{\partial b_1} = 1 + \beta\alpha_1 > 0$, $\dfrac{\partial e_2^*}{\partial b_1} = 0$. | Higher productivity in task 1 raises the current marginal return and the career return; $\alpha_1$ itself does not depend on $b_1$. |
| Signaling: visibility of ability | Mean AI share ($\mu_\lambda \uparrow$) | $\dfrac{de_1^*}{d\mu_\lambda} \gtrless 0$ (typically $< 0$), $\dfrac{de_2^*}{d\mu_\lambda} > 0$. | Higher $\mu_\lambda$ reduces ability loading ($\kappa_\theta$) and increases residual variance $\sigma^2_{\eta_1}$, so signal precision of task 1 drops and weight shifts to $x_2$. |
| Signaling: visibility of ability | Ability visible in AI ($\rho \uparrow$) | $\dfrac{de_1^*}{d\rho} \gtrless 0$ (typically $> 0$), $\dfrac{de_2^*}{d\rho} < 0$. | Higher $\rho$ increases ability loading ($\kappa_\theta$) and decreases residual variance $\sigma^2_{\eta_1}$, so signal precision of task 1 is restored and weight shifts back to $x_1$. |
| Signaling: uncertainty | AI-share uncertainty ($\sigma^2_\lambda \uparrow$) | $\dfrac{\partial e_1^*}{\partial \sigma^2_\lambda} \leq 0$, $\dfrac{\partial e_2^*}{\partial \sigma^2_\lambda} \geq 0$. | Higher $\sigma^2_\lambda$ increases residual variance $\sigma^2_{\eta_1}$, so signal precision of task 1 drops and weight shifts to $x_2$. |
| Signaling: uncertainty | AI-quality uncertainty ($\sigma^2_g \uparrow$) | $\dfrac{\partial e_1^*}{\partial \sigma^2_g} \leq 0$, $\dfrac{\partial e_2^*}{\partial \sigma^2_g} \geq 0$. | Higher $\sigma^2_g$ increases residual variance $\sigma^2_{\eta_1}$, so signal precision of task 1 drops and weight shifts to $x_2$. |

# Internet Appendix A

## Internet Appendix A.1 Skill-Based GitHub Activity Classification

### Internet Appendix A.1.1 Prompt Used With GPT-4 Model in April 2024

Suppose you are a programmer who is active on Github platform. Define what may be job-specific core skills and what may be transferable general skills.

For the following Github events, classify them into three categories: job-specific core skills, transferable general skills, mixture of core and general skills, and others. Each event should be uniquely assigned to only one category that is the most relevant.

List of GitHub events: CommitCommentEvent, CreateEvent, DeleteEvent, ForkEvent, GollumEvent, IssueCommentEvent, IssuesEvent, MemberEvent, PublicEvent, PullRequestEvent, PullRequestReviewEvent, PullRequestReviewCommentEvent, PullRequestReviewThreadEvent, PushEvent, ReleaseEvent, SponsorshipEvent, WatchEvent

### Internet Appendix A.1.2 Classification Details

**Job-specific core skills**

- PushEvent: Relates to pushing code to a repository, a basic GitHub operation.
- PullRequestEvent: Central to managing code contributions and integrations.
- PullRequestReviewEvent: Linked to the code review process within pull requests.

**General skills**

- IssueCommentEvent: Involves communication and discussion over issues.
- IssuesEvent: Engages problem-solving, managing bug reports, and feature requests

**Mixture of core and general skills**

- CommitCommentEvent: Tied to code reviews, requiring technical insights as well as communication skills.

- PullRequestReviewCommentEvent: Specific to commenting on code reviews in pull requests, requiring technical understanding and collaborative feedback.
- PullRequestReviewThreadEvent: Involves discussions around specific parts of a pull request, blending code-specific knowledge with teamwork and communication.

**Nonskill related**

- ForkEvent: Represents a user's engagement with and branching off from an existing repository to potentially contribute or alter separately.
- GollumEvent: Pertains to the management of Wiki pages on a GitHub repository.
- SponsorshipEvent: Linked to the GitHub Sponsors program, reflecting community support and funding mechanisms.
- WatchEvent: Involves starring a repository, indicating interest or following updates, more about user engagement than a direct skill.

**Others**

There are other related events I define as core/general in a broader sense. But they are not used in the analysis.

- Broader core activities
  - CreateEvent: Involves creating branches or tags, fundamental to version control.
  - DeleteEvent: Involves deleting branches or tags, another version control aspect.
  - ReleaseEvent: Pertains to the release of new software versions, important in software lifecycle management.
- Broader general activities
  - PublicEvent: While more of an administrative function, it also involves decision-making and policy setting regarding project visibility. (Initiate project)

  – MemberEvent: Related to teamwork and the management of repository collaborators.

## Internet Appendix A.2 Name-Based Gender Inference

### Internet Appendix A.2.1 Parameters for GPT Model Interaction via OpenAI's API

- `model`: gpt-3.5-turbo
- `temperature`: 0
- `system_text`: Process a list of names, extracting identifiable components and infer demographic information. Return the findings in JSON format with fields for original_str, first_name, last_name, company, type (with an inf_type among "user", "organization" and "bot", and score), gender (with an inf_gender either "female" or "male", and score), race (with an inf_race and score), ethnicity (with an inf_ethnicity and score), and country_of_origin (with an inf_origin and score). Put 'NA' for string subfields with no findings, and 0 for scores with no findings. Scores are for the confidence level of the inference and range from 0 to 1 rounded to two decimals. Score closer to 1 means the inference is certain while score closer to 0 means the inference is uncertain. The output is with 'results' as the key.
- `user_text`: ['name1', 'name2', 'name3',...]

### Internet Appendix A.2.2 Example: Name-Based Inference Response

The JSON response example for a person named Bob Chen is:

```
{
  "results": [
    {
      "original_str": "Bob Chen",
      "first_name": "Bob",
      "last_name": "Chen",
      "company": "NA",
      "type": {
        "inf_type": "user",
        "score": 0.95
      },
      "gender": {
        "inf_gender": "male",
        "score": 0.85
      },
      "race": {
        "inf_race": "Asian",
        "score": 0.80
      },
      "ethnicity": {
        "inf_ethnicity": "NA",
        "score": 0
      },
      "country_of_origin": {
        "inf_origin": "United States",
        "score": 0.75
      }
    }
  ]
}
```

## Internet Appendix A.3 Prompt for Language AI Exposure Score With GPT-4 in April 2024

For the following programming languages, assign a score between 0 and 1 for its exposure to LLMs such as Github Copilot. Exposure is defined as to what extent are the Generative AI tools helpful for programmers using these languages to complete their daily tasks. If it is not a programming language, return 'NA' for the score. Return your result in JSON format (language:score).

Language list: ['language1', 'language2', ...]

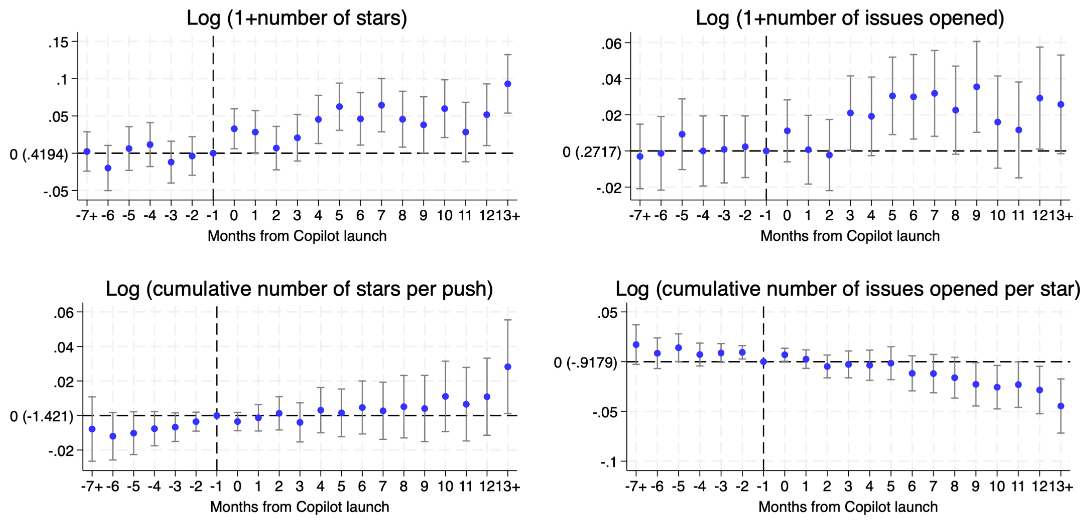

Figure IA1. Quality Change After Copilot Launch

This figure plots coefficients of the event study specification described in equation 3 with 95% confidence intervals. The outcome variables are: the $ln(1 + x)$ transformations of the number of stars and issues opened that are associated with developers' work each month; the cumulative number of stars, scaled by the number of pushes, and the cumulative number of issues opened, scaled by the number of stars. Standard errors are clustered at developer level.

Table IA1. Variable Definitions

| Variable | Description | Source |
|---|---|---|
| 10/20/30-day Cumulative Abnormal Return | Cumulative abnormal return over 10/20/30 days | CRSP |
| AI Exposure | Dummy that equals one if the AI exposure score of the developer (0-1) is in the fourth quartile | GPT-4 and author's calculation |
| AI Exposure Score | The AI exposure score of a programming language | GPT-4 |
| Abnormal Return | The difference between the actual return and the expected return, as estimated by the Fama-French 3-factor model using the Nasdaq 100 index for market return | CRSP |
| Across-Firm Demotion | Dummy that equals one if a developer is demoted, either by compensation or by seniority, across firms in a given quarter | Revelio |
| Across-Firm Job Change | Dummy that equals one if a developer changes jobs across firms in a given quarter | Revelio |
| Across-Firm Promotion | Dummy that equals one if a developer is promoted, either by compensation or by seniority, across firms in a given quarter | Revelio |
| Aligned | Dummy that equals to one if an innovative rm has an average employees' tenure in the rst quartile or or if a non-innovative rm has an average employees' tenure in the fourth quartile | Compustat |
| Core Event | Number of core-skill related events in a given month/quarter | GHArchive |
| Cumulative Nrepo | Cumulative number of repositories released by a firm prior to month t | GHArchive |
| Employees | Number of employees in the firm | Compustat |
| Exit | Dummy that equals one if a developer becomes inactive in all firm-owned projects and subsequently exits the sample | GHArchive |
| Female | Dummy that equals one if the developer is inferred as female | GitHub API, GPT-3.5 |
| Firm AI Exposure | Dummy that equals one if the AI exposure score of the firm is in the fourth quartile | GPT-4 and author's calculation |
| General Event | Number of general-skill related events in a given month/quarter | GHArchive |
| Has Core Event | Dummy that equals one if a developer has at least one core-skill related event in a given month | GHArchive |
| Has General Event | Dummy that equals one if a developer has at least one general-skill related event in a given month | GHArchive |
| Has Mixed Event | Dummy that equals one if a developer has at least one mixed-skill related event in a given month | GHArchive |
| Initiated Project | Indicator if a developer is among the innovator team of a new project in a given quarter | GHArchive |
| Initiator N | Number of developers in the innovator team of a new project | GHArchive |
| Innovative Firm | Dummy that equals one if the firm's R&D expenditure as a share of total assets is higher than the median | Compustat |

Continued

| Variable | Description | Source |
|---|---|---|
| Innovator | Dummy that equals one if the developer has initiated at least one project prior to 2022Q3 | GHArchive |
| Innovator AI Exposure | Dummy that equals to one if the AI exposure score of at least one member of the innovator team is in the fourth quartile | GPT-4 and author's calculation |
| Issues Opened | Number of issues opened that are associated with a developer's work in a given month | GHArchive |
| Issues Opened Per Star | Cumulative number of issues opened divided by the cumulative number of stars received | GHArchive |
| Job Change | Dummy that equals one if a developer changes jobs in a given quarter | Revelio |
| Language Share | The share of a given programming language used by a developer before July 2022 | GHArchive, GitHub API |
| Main Language | The main programming language used by a develope before July 2022 | GHArchive, GitHub API |
| Market Capitalization | Share price times the number of shares outstanding | CRSP |
| Mixed Event | Number of mixed-skill related events in a given month/quarter | GHArchive |
| Novelty | A LLM-based novelty score of the repository between 0 and 1 inferred from repository information. The score measures how novel or groundbreaking a repository is compared to existing solutions, focusing on whether it introduces new ideas, techniques, or approaches | GPT-4o |
| Number of Cumulative Pushes | Cumulative number of pushes by a firm before July 2022 | GHArchive |
| Number of Initiated Projects | Number of projects initiated by a developer in a given quarter | GHArchive |
| Patent Portfolio Value | The total estimated economic value of the patents owned by the firm using stock market returns around the patent grant date | Kogan et al. (2017) |
| Post | Dummy that equals one if the time period is or after July 2022 or the third quarter of 2022 | |
| Profitability | Pre-tax income divided by total assets | Compustat |
| R&D Expenditure as a Share of Assets | R&D expenses divided by lagged total assets | Compustat |
| R&D Missing | Dummy that equals one if R&D expense is missing | Compustat |
| Repo AI Exposure | Dummy that equals one if the AI exposure score of the repository (0-1) is in the fourth quartile | GPT-4 and author's calculation |
| Repo Value | The estimated private value of the repository in 2023 USD estimated by using stock market returns around the release date of the repository | Author's calculation based on Emery et al. (2024) |
| Repos With Core Event | Number of repositories with core-skill related events in a given month | GHArchive |
| Repos With General Event | Number of repositories with general-skill related events in a given month | GHArchive |
| Repos With Mixed Event | Number of repositories with mixed-skill related events in a given month | GHArchive |

Continued

| Variable | Description | Source |
|---|---|---|
| Return on Assets | Net income divided by lagged total assets | Compustat |
| Revenue Growth | The growth rate of revenue | Compustat |
| Senior | Dummy that equals one if the developer's tenure is in the fourth quartile | GitHub API |
| Seniority | Seniority rank of the job position assigned by Revelio (1-7) | Revelio |
| Stars | Number of stars received by a repository as of Feburary 2024 | GitHub API |
| Stars Per Push | Cumulative number of stars received by a repository divided by the cumulative number of pushes | GHArchive |
| Total Compensation | Total yearly compensation in USD of a job position predicted by Revelio | Revelio |
| Volatility | Standard deviation of daily returns over one month | CRSP |
| Work Completed During Weekends | Cumulative ratio of core events occurring during weekends $(\frac{\text{cumulative number of core events during weekends}_{i,t}}{\text{cumulative total number of core events}_{i,t}})$ | GHArchive |
| Work Completed Outside Common Hours | Cumulative ratio of core events occurring outside common hours $(\frac{\text{cumulative number of core events outside common hours}_{i,t}}{\text{cumulative total number of core events}_{i,t}})$ | GHArchive |
| Work Completed Per Hour | Cumulative number of core events per hour $(\frac{\text{cumulative total number of core events}_{i,t}}{\text{cumulative total number of hours}_{i,t}})$ | GHArchive |

## Table IA2. Summary Statistics of GitHub Firms by GenAI Exposure (Jan2021-Jun2022)

This table reports summary statistics of GitHub firms' characteristics before GitHub Copilot's official launch, from January 2021 to June 2022. The analysis includes only firms with more than 10 developer-months during this period. *High AI Exposure* is a dummy variable that equals one if the average AI exposure score of developers affiliated with a given firm falls in the fourth quartile. A developer has high AI exposure if their AI exposure score ranks in the fourth quartile among all developers. Senior developers are individuals whose GitHub platform tenure is above the median. See Table IA1 for variable descriptions.

| | High AI Exposure | | | | Low AI Exposure | | | |
|---|---|---|---|---|---|---|---|---|
| | Mean | Median | SD | N | Mean | Median | SD | N |
| Number of developer-months | 100.58 | 36.00 | 209.96 | 168 | 691.60 | 79.00 | 3,771.73 | 504 |
| AI exposure score | 0.91 | 0.90 | 0.04 | 168 | 0.77 | 0.80 | 0.09 | 504 |
| Developer with high AI exposure | 0.62 | 0.66 | 0.36 | 168 | 0.14 | 0.07 | 0.18 | 504 |
| Senior developer | 0.63 | 0.70 | 0.36 | 168 | 0.61 | 0.65 | 0.30 | 504 |
| Market capitalization | 24985.43 | 5137.73 | 57,771.04 | 136 | 61955.95 | 6559.70 | 222426.87 | 393 |
| Percentage revenue growth | 19.86 | 16.86 | 20.27 | 137 | 18.58 | 17.87 | 21.05 | 410 |
| Profitability | -2.42 | 0.91 | 18.44 | 144 | -2.22 | 1.17 | 23.24 | 434 |
| Foreign revenue share | 0.41 | 0.37 | 0.32 | 124 | 0.45 | 0.43 | 0.31 | 379 |
| Leverage | 0.25 | 0.21 | 0.21 | 144 | 0.26 | 0.23 | 0.22 | 432 |
| Interest expense / total assets | 0.01 | 0.01 | 0.01 | 138 | 0.01 | 0.01 | 0.01 | 419 |
| R&D / total assets | 0.06 | 0.04 | 0.08 | 147 | 0.07 | 0.06 | 0.09 | 442 |

Table IA3. Exit Probability After Copilot Launch

This table reports regression results of equation 1 and equation 2. The outcome variable is a dummy that equals one if a developer becomes inactive in all firm-owned projects and subsequently exits the sample. Columns (1)-(2) report results for the full sample, while Columns (3)-(4) restrict the sample to innovators only. Innovators are defined as developers who contribute to firm-owned projects within two weeks of their creation. *Post* is a dummy that equals one if the time period is after July 2022. *AI Exposure* or *AI* are dummy variables that equal one if the developer's AI exposure score is in the fourth quartile. *Senior* is a dummy that equals one if the developer's tenure is above median. The total effects for senior developers (sum of the coefficients of the post treatment indicator and the interaction term) are reported underneath. Standard errors are clustered at developer level. Significance: *, $p < 0.1$; **, $p < 0.05$; ***, $p < 0.01$.

| | All | | Innovators | |
|---|---|---|---|---|
| | (1) | (2) | (3) | (4) |
| Post×AI Exposure | 0.0167*** | 0.0124** | 0.0263*** | 0.0374*** |
| | (4.60) | (2.08) | (3.03) | (2.61) |
| Post×Senior | | -0.0115*** | | -0.0076 |
| | | (-3.32) | | (-0.94) |
| Post×AI×Senior | | 0.0067 | | -0.0198 |
| | | (0.89) | | (-1.10) |
| Total Effect (Senior) | | 0.0191*** | | 0.0176 |
| | | (4.17) | | (1.63) |
| N | 194,973 | 194,948 | 19,917 | 19,917 |
| Adj. R2 | 0.3768 | 0.3769 | 0.5399 | 0.5400 |
| Individual FE | Y | Y | Y | Y |
| Year-Quarter FE | Y | Y | Y | Y |

## Table IA4. Coding Activity And Pre-Treatment Language Share Exposure

This table reports regression results of equation 2 at the individual-language level. The outcome variable is the $ln(1 + x)$ transformations of the number of pushes that are associated with developers' work in a given language each quarter. Columns (1) and (3) represent results for the main language a developer uses, while columns (2) and (4) include all other languages a developer uses prior to GenAI. *Post* is a dummy that equals one if the time period is after July 2022. *AI Exposure* is a dummy variable that equals one if the language's AI exposure score is in the fourth quartile. *AI Score* is the raw AI exposure score of a language. *Share* is the pre-treatment share of a language used within a developer. Main effects and other cross-interactions are included. Standard errors are clustered at developer level. Significance: *, p < 0.1; **, p < 0.05; ***, p < 0.01.

| | Main Language | Other Languages | Main Language | Other Languages |
|---|---|---|---|---|
| | (1) | (2) | (3) | (4) |
| Post×AI Exposure | 0.0066 | 0.0136*** | | |
| | (0.24) | (5.50) | | |
| Post×AI Exposure×Share | -0.0137 | 0.0448 | | |
| | (-0.39) | (1.20) | | |
| Post×AI Score | | | -0.0882 | 0.0295*** |
| | | | (-0.84) | (4.60) |
| Post×AI Score×Share | | | 0.1442 | 0.3011** |
| | | | (1.06) | (2.33) |
| N | 211,402 | 1,343,431 | 211,402 | 1,343,431 |
| Adj. R2 | 0.4648 | 0.4493 | 0.4648 | 0.4495 |
| Language FE | Y | Y | Y | Y |
| Individual-Year-Quarter FE | N | Y | N | Y |
| Individual FE | Y | N | Y | N |
| Year-Quarter FE | Y | N | Y | N |

## Table IA5. Coding Activity After Copilot Launch at the Individual-Language Level

This table reports regression results of equation 1 and equation 2 at the individual-language level. In Columns (1)-(4), the outcome variable is a dummy that equals one if a developer contributes code in a given language in a given quarter. In Columns (5)-(8), the outcome variable is the $ln(1 + x)$ transformations of the number of coding events that are associated with developers' work in a given language each quarter. Columns (3) and (7) restrict the sample to languages a developer worked with prior to GenAI introduction, while Columns (4) and (8) include only languages new to the developer. For computational efficiency, the analysis includes only languages with over 10,000 coding events across the sample period. These 42 languages represent 98.8% of total coding activities. *Post* is a dummy that equals one if the time period is after July 2022. *AI Score* is the raw AI exposure score of a language. Standard errors are clustered at the developer level. Significance: *, p < 0.1; **, p < 0.05; ***, p < 0.01.

| | Has Coding Events | | | | Ln(1 + Coding Events) | | | |
|---|---|---|---|---|---|---|---|---|
| Language Type | All<br>(1) | All<br>(2) | Familiar<br>(3) | New<br>(4) | All<br>(5) | All<br>(6) | Familiar<br>(7) | New<br>(8) |
| Post×AI Score | 0.0555$^{***}$<br>(25.29) | 0.0462$^{***}$<br>(13.52) | 0.0030<br>(0.44) | 0.0510$^{***}$<br>(28.58) | 0.0439$^{***}$<br>(22.15) | 0.0374$^{***}$<br>(11.93) | -0.0272$^{**}$<br>(-2.38) | 0.0318$^{***}$<br>(24.50) |
| Post×AI×Senior | | 0.0158$^{***}$<br>(3.55) | 0.0284$^{***}$<br>(3.19) | 0.0050$^{**}$<br>(2.07) | | 0.0113$^{***}$<br>(2.79) | 0.0233<br>(1.64) | 0.0019<br>(1.12) |
| N | 9,745,218 | 9,731,274 | 1,560,209 | 8,154,030 | 9,745,218 | 9,731,274 | 1,560,209 | 8,154,030 |
| Adj. R2 | 0.2793 | 0.2793 | 0.6303 | 0.1431 | 0.1199 | 0.1199 | 0.2924 | 0.0428 |
| Language FE | Y | Y | Y | Y | Y | Y | Y | Y |
| Individual-Year-Quarter FE | Y | Y | Y | Y | Y | Y | Y | Y |

Table IA6. Coding Activity by Pre-Treatment Project Team Size and Popularity

This table reports regression results of equation equation 2. Outcome variables in Columns (1)-(4) are monthly indicators for any public coding activity in firm-owned repositories by project type. Columns (5)-(8) use the log of one plus quarterly coding activities by project type. *Team* projects have on average at least two contributors monthly from January 2021 to June 2022; *Solo* projects average fewer. *High Popularity* projects have cumulative stars above the 90th percentile during January 2021 to June 2022. *Post* is a dummy that equals one if the time period is after July 2022 (or the third quarter of 2022). *AI Exposure* or *AI* are dummy variables that equal one if the developer's AI exposure score is in the fourth quartile. *Senior* is a dummy that equals one if the developer's tenure is above median. The total effects for senior developers (sum of the coefficients of the post treatment indicator and the interaction term) are reported underneath. Standard errors are clustered at developer level. Significance: *, $p < 0.1$; **, $p < 0.05$; ***, $p < 0.01$.

| | Has Coding Events | | | | Ln(1+Coding Events) | | | |
|---|---|---|---|---|---|---|---|---|
| Project Type | Team Size | | Popularity | | Team Size | | Popularity | |
| | Solo | Team | Low | High | Solo | Team | Low | High |
| | (1) | (2) | (3) | (4) | (5) | (6) | (7) | (8) |
| Post×AI Exposure | -0.0096 | -0.0067 | 0.0044 | -0.0064 | -0.0553** | -0.0069 | 0.0040 | -0.0170 |
| | (-1.47) | (-0.91) | (0.57) | (-0.91) | (-2.33) | (-0.21) | (0.13) | (-0.55) |
| Post×Senior | -0.0167*** | -0.0138*** | -0.0081* | -0.0173*** | -0.0609*** | -0.0611*** | -0.0407** | -0.0696*** |
| | (-4.20) | (-3.15) | (-1.73) | (-4.10) | (-4.22) | (-3.00) | (-2.17) | (-3.63) |
| Post×AI×Senior | 0.0043 | 0.0157* | 0.0021 | 0.0205** | 0.0443 | 0.0648 | 0.0364 | 0.1002** |
| | (0.52) | (1.71) | (0.21) | (2.27) | (1.47) | (1.57) | (0.93) | (2.53) |
| Total Effect (Senior) | -0.0053 | 0.0090 | 0.0065 | 0.0141** | -0.0111 | 0.0578** | 0.0404* | 0.0833*** |
| | (-1.04) | (1.62) | (1.12) | (2.50) | (-0.60) | (2.33) | (1.74) | (3.37) |
| N | 563,582 | 563,582 | 563,582 | 563,582 | 194,948 | 194,948 | 194,948 | 194,948 |
| Adj. R2 | 0.3714 | 0.6025 | 0.4454 | 0.6097 | 0.5695 | 0.7865 | 0.6454 | 0.7963 |
| Individual FE | Y | Y | Y | Y | Y | Y | Y | Y |
| Time FE | Y | Y | Y | Y | Y | Y | Y | Y |
| Time Frequency | Monthly | Monthly | Monthly | Monthly | Quarterly | Quarterly | Quarterly | Quarterly |

## Table IA7. Summary Statistics of Job Changes of Firm Developers on GitHub

This table presents summary statistics on employee mobility at the individual-year-quarter level from January 2021 to December 2023. Panel (a) summarizes job changes and position characteristics of firms' developers active on GitHub. Panel (b) compares developers based on whether their tenure is above or below the median and whether they initiated a GitHub project owned by their employers before the official launch of GitHub Copilot. *Promotion* is a binary variable equal to one if the new job position offers higher compensation or a higher seniority rank.

### (a) Full Sample at the Individual-Quarter Level

| | Mean | SD | Min | P25 | Median | P75 | Max | Obs |
|---|---|---|---|---|---|---|---|---|
| Job Change | 0.07 | 0.25 | 0.00 | 0.00 | 0.00 | 0.00 | 1.00 | 151,568 |
| Across-Firm Job Change | 0.04 | 0.18 | 0.00 | 0.00 | 0.00 | 0.00 | 1.00 | 151,568 |
| Across-Firm Promotion | 0.01 | 0.09 | 0.00 | 0.00 | 0.00 | 0.00 | 1.00 | 151,568 |
| Senior GitHub Developer | 0.65 | 0.48 | 0.00 | 0.00 | 1.00 | 1.00 | 1.00 | 151,568 |
| GitHub Project Initiator | 0.17 | 0.37 | 0.00 | 0.00 | 0.00 | 0.00 | 1.00 | 151,568 |
| Job Position Seniority | 3.12 | 1.25 | 1.00 | 2.00 | 3.00 | 4.00 | 7.00 | 118,095 |
| Total Compensation USD (000) | 183.41 | 100.42 | 5.71 | 110.49 | 175.24 | 242.55 | 1,705.44 | 118,095 |

### (b) By Developer Characteristics

| | Senior Developer | | | | Project Initiator | | | |
|---|---|---|---|---|---|---|---|---|
| | Yes | | No | | Yes | | No | |
| | Mean | SD | Mean | SD | Mean | SD | Mean | SD |
| Job Change | 0.06 | 0.24 | 0.08 | 0.27 | 0.07 | 0.25 | 0.07 | 0.26 |
| Across-Firm Job Change | 0.03 | 0.18 | 0.04 | 0.19 | 0.03 | 0.18 | 0.04 | 0.18 |
| Across-Firm Promotion | 0.01 | 0.09 | 0.01 | 0.10 | 0.01 | 0.09 | 0.01 | 0.09 |
| Job Position Seniority | 3.16 | 1.26 | 3.04 | 1.22 | 3.30 | 1.32 | 3.08 | 1.23 |
| Total Compensation USD (000) | 192.87 | 103.90 | 166.71 | 91.59 | 191.45 | 107.90 | 181.79 | 98.76 |

## Internet Appendix B Economic Model on Programming Language Skills And Productivity Gain from AI Exposure

This appendix section provides a simple economic model that explains the intuition behind the relation between programming language-based AI exposure and productivity gain of programmers.

In the absence of GenAI coding tool, a programmer uses two programming languages $l$ and $l_2$, with human labor costs $c_h$ and $c_2$. Assume the programmer has a Cobb-Douglas utility function and she faces the following optimization problem:

$$\begin{aligned} \max_{l,l_2} \quad & u(l,l_2) = l^{\rho} {l_2}^{(1-\rho)} \\ \text{s.t.} \quad & c_h l + c_2 l_2 \le w \end{aligned} \tag{5}$$

The Marshall demand functions of the programmer before AI introduction are:

$$\begin{aligned} l^0 =& \frac{\rho w}{c_h} \\ l_2^0 =& \frac{(1-\rho)w}{c_2} \end{aligned} \tag{6}$$

The GenAI coding tool only supports language $l$ with a cost of $c_{AI}$ which is assumed to be lower than human coding ($c_{AI} < c_h$). If the programmer uses the GenAI tool, her utility function becomes:

$$u(l_{AI}, l, l_2) = [(a_{AI}^{\frac{1}{\sigma}} l_{AI}^{\frac{\sigma-1}{\sigma}} + a_h^{\frac{1}{\sigma}} l_h^{\frac{\sigma-1}{\sigma}})^{\frac{\sigma}{\sigma-1}}]^{\rho} l_2^{1-\rho}$$

Where the nested utility function associated with language $l$ is a CES function. Thus, her optimization problem becomes:

$$\begin{aligned} \max_{l_{AI},l_h,l_2} \quad & u(l_{AI}, l_h, l_2) = [(a_{AI}^{\frac{1}{\sigma}} l_{AI}^{\frac{\sigma-1}{\sigma}} + a_h^{\frac{1}{\sigma}} l_h^{\frac{\sigma-1}{\sigma}})^{\frac{\sigma}{\sigma-1}}]^{\rho} l_2^{1-\rho} \\ \text{s.t.} \quad & l_{AI} + c_h l_h + c_2 l_2 \le w \end{aligned} \tag{7}$$

Without losing generalizability the cost of AI is normalized to 1 ($c_{AI} = 1$).

Solve the optimization problem and we have:

$$\begin{aligned} l_{AI} + c_h l_h &= \rho w \\ c_2 l_2 &= (1-\rho) w \\ \frac{l_{AI}}{l_h} &= \frac{a_{AI}}{a_h} c_h^{\sigma} \end{aligned}$$

The Marshall demand functions are then:

$$l_2 = \frac{(1-\rho)w}{c_2}$$
$$l_{AI} = \frac{\rho w}{\frac{a_h}{a_{AI}} c_h^{1-\sigma} + 1}$$
$$l_h = \frac{\rho w c_h^{-\sigma}}{c_h^{1-\sigma} + \frac{a_{AI}}{a_h}}$$

The total observed activities related to language $l$ is:

$$l = l_{AI} + l_h = \rho w \frac{\widetilde{a} + c_h^{-\sigma}}{c_h^{1-\sigma} + \widetilde{a}}$$

Where $\widetilde{a} = \frac{a_{AI}}{a_h}$.

Next, we compare the activities in language $l$ before and after the introduction of GenAI. The activity in $l_2$ is unaffected and therefore the change is zero ($\Delta l_2 = 0$). For language $l$, the change is written as:

$$\Delta l = l_{AI} + l_h - l^0 = \rho w \frac{\widetilde{a}[c_h^{\sigma} - c_h^{\sigma-1}]}{\widetilde{a} c_h^{\sigma} + c_h}$$

Since we assume $c_h > c_{AI} = 1$, $\Delta l > 0$. However, it is not immediately clear whether such increase is more for programmers with lower $c_h$ or not. To see this, take the partial derivative of $\Delta l$:

$$\frac{\partial \Delta l}{\partial c_h} = \frac{\rho w \widetilde{a} c_h^{\sigma-1}}{(\widetilde{a} c_h^{\sigma} + c_h)^2}[(\sigma-1)c_h + \widetilde{a} c_h^{\sigma-1} + 2 - \sigma]$$

Let $f(c_h) = (\sigma-1)c_h + \widetilde{a} c_h^{\sigma-1} + 2 - \sigma, c_h > 1$. When $\sigma = 1$, $f(c_h) = \widetilde{a} + 1 > 0$. Similarly, $f(c_h) > 0$ when $\sigma > 1$. In both cases, $\Delta l$ increases as $c_h$ increases. Intuitively, when AI coding and human coding are substitutes, programmers with relatively higher cost in language $l$ ($c_h$) benefit more from GenAI than programmers with lower cost.

However, when AI coding and human coding are complements ($0 < \sigma < 1$), the relationship depends on the model's parameters. Because $f(1) = \widetilde{a} + 1 > 0$, $f(\infty) < 0$ and $f'(c_h) = (\sigma-1)(1 + \widetilde{a} c_h^{\sigma-2}) < 0$, there exists a unique root of $f(c_h)$ in its domain. Let $c_h^*$ denotes the root such that $f(c_h^*) = 0$.

We have,

$$\begin{cases} \dfrac{\partial \Delta l}{\partial c_h} > 0, & c_h < c_h^* \\ \dfrac{\partial \Delta l}{\partial c_h} = 0, & c_h = c_h^* \\ \dfrac{\partial \Delta l}{\partial c_h} < 0, & c_h > c_h^* \end{cases}$$

That is, when the human coding cost $c_h$ is relatively low, programmers with lower ability (higher $c_h$) benefit more from GenAI tool. However, when the programmer has a relatively high human coding cost for language $l$ such that $c_h > c_h^*$, programmers with lower ability benefit less.

The next question is to decide how the model's parameters affect $c_h^*$. From the Implicit Function Theorem,

$$\frac{\partial c_h^*}{\partial \sigma} = -\frac{c_h + \widetilde{a} ln(c_h) c_h^{\sigma-1} - 1}{(\sigma-1)(1+\widetilde{a} c_h^{\sigma-2})} > 0$$
$$\frac{\partial c_h^*}{\partial \widetilde{a}} = -\frac{c_h^{\sigma-1}}{(\sigma-1)(1+\widetilde{a} c_h^{\sigma-2})} > 0$$

Therefore, the higher the complementarity between AI and human coding (smaller $\sigma$), and the lower the weight is assigned to AI coding relative to human coding (smaller $\widetilde{a}$), the smaller the $c_h^*$. In these cases, it's likely that for most programmers who are unfamiliar with language $l$ (thus with higher $c_h$) do not benefit as much as those who are skilled at language $l$ before the introduction of GenAI.

The reality seems to be close to the scenario where $0 < \sigma < 1$ and $c_h > c_h^*$. Autocompletion tools like GitHub Copilot are largely complementary. These tools generate codes line by line as programmers code themselves instead of "Autostart" or generating a whole script from scratch. Moreover, several large technology firms disclose the share of their AI-generated codes with an average about 25%. Thus, the weight assigned to AI relative to human coding does not seem to be large either. These indicate that $c_h^*$ could be relatively low. Therefore, under these assumptions, the model predicts that programmers more skilled in the languages exposed to GenAI, on average, arguably benefit more from GenAI tools than their peers, and it is particularly true for non-primary languages, where $c_h$ is more likely to be higher than $c_h^*$.