El-Komboz, Lena Marie Abou

Research Report

Empirical essays on digital platforms

ifo Beiträge zur Wirtschaftsforschung, No. 107

Provided in Cooperation with:
Ifo Institute – Leibniz Institute for Economic Research at the University of Munich

Suggested Citation: El-Komboz, Lena Marie Abou (2025) : Empirical essays on digital platforms, ifo Beiträge zur Wirtschaftsforschung, No. 107, ISBN 978-3-95942-137-9, ifo Institut - Leibniz-Institut für Wirtschaftsforschung an der Universität München, München

This Version is available at:
https://hdl.handle.net/10419/313882

ifo
BEITRÄGE
zur Wirtschaftsforschung

107
2024

Empirical Essays on Digital Platforms

Lena Marie Abou El-Komboz

ifo INSTITUTE
Leibniz Institute for Economic Research
at the University of Munich

ifo
BEITRÄGE
zur Wirtschaftsforschung

107
2024

Empirical Essays on Digital Platforms

Lena Marie Abou El-Komboz

Herausgeber der Reihe: Clemens Fuest
Schriftleitung: Chang Woon Nam

ifo INSTITUTE
Leibniz Institute for Economic Research
at the University of Munich

Bibliografische Information der Deutschen Nationalbibliothek

Die Deutsche Nationalbibliothek verzeichnet diese Publikation
in der Deutschen Nationalbibliografie; detaillierte bibliografische
Daten sind im Internet über
https://dnb.d-nb.de
abrufbar.

ISBN Nr. 978-3-95942-137-9

Druck: Pinsker Druck und Medien GmbH, Mainburg

ifo Institut im Internet:
https://www.ifo.de

Empirical Essays on Digital Platforms

Inaugural-Dissertation
zur Erlangung des Grades
Doctor oeconomiae publicae (Dr. oec. publ.)

eingereicht an der
Ludwig-Maximilians-Universität München
2024

vorgelegt von

Lena Marie Abou El-Komboz

Referent: Prof. Dr. Oliver Falck
Korreferent: Prof. Dr. Florian Englmaier
Promotionsabschlussberatung: 29. Januar 2025

Datum der mündlichen Prüfung: 13.01.2025
Namen der Berichterstatter: Prof. Dr. Oliver Falck
Prof. Dr. Florian Englmaier
Prof. Dr. Lisandra Flach

Preface

Digital platforms alter almost every industry in the modern economy (De Reuver et al., 2018). For instance, in the media industry, there is an increasing shift from traditional media outlets such as newspapers and television to online newspapers, social media, or platforms such as Netflix or YouTube to watch content (Waldfogel, 2017; Wu and Zhu, 2022). In the software industry, more digital by its nature, it is nowadays the standard to use version control systems such as *git*, with code stored on online platforms. These platforms emerged and now dominate the supply of online media, for instance, software, music, news, and videos (Waldfogel, 2017; Wu and Zhu, 2022). Three of the top 10 of the global 2000 list by Forbes in 2023 are platform companies, namely Alphabet (Google), Microsoft, and Apple (Cusumano et al., 2019; Bonina et al., 2021; Forbes, 2023). These companies are at the global top concerning sales, profits, assets, and market value, and simultaneously shape our economies. Some firms have started as a digital firm, whereas others have moved their business model in the spirit of the digital transformation towards the online space (Nagle, 2022). For business models with high transaction costs, digital platforms are capable to lead to a reduction of its costs, whilst creating a more flexible and further reaching place for transactions than traditional offline platforms (Sutherland and Jarrahi, 2018).

Platforms in general are defined as a place or forum on which exchanges can occur. A digital platform builds on internet and communication technologies to enable transactions among two or more groups of users, thus, being technologically moderated (Hagiu, 2007; Demary and Rusche, 2018; Bonina et al., 2021). The content on digital platforms is to a large extent produced by the platforms' users (Lerner and Tirole, 2002; Lakhani and Wolf, 2005; Fershtman and Gandal, 2011). Individuals often voluntarily provide their knowledge and skills to contribute to the development of software or create content to watch (Loh and Kretschmer, 2023). Digital platforms, thus, act as intermediaries between suppliers, content creators, and consumers (Aguiar et al., 2024). They can help to overcome the excessive amount of information nowadays available by aggregating it. With reduced production costs due to technologies, and the simultaneously low-cost dispersion, digital innovations can emerge and lead to new product developments (Aguiar et al., 2024). Existing research was able to show that the increase of new products due to digital innovations is economically beneficial (Brynjolfsson et al., 2003; Aguiar and Waldfogel, 2018). Therefore, digital platforms

are important for economic growth by enabling actors to create and share knowledge leading to economically important innovations (Goldfarb and Tucker, 2019).

However, digital platforms face several challenges. They build on an often decentralized community with voluntary suppliers (Sutherland and Jarrahi, 2018; Nagle, 2022). Thus, platforms need to implement incentives for contributors to provide content, while simultaneously controlling for the platform's quality (Geva et al., 2019; Loh and Kretschmer, 2023). For that, it is important to understand the rationale of voluntary suppliers to be active on the platforms (Hars and Ou, 2002; Lerner and Tirole, 2002; Krishnamurthy, 2006). Additionally, hurdles in transactions common on traditional platforms can play a role on digital platforms as well. For instance, trust plays an important role in real-world transactions as it does on digital platforms (Luca, 2017). In online transactions, in which actors tend to have less information about the other side, platforms need to put features into place to create a space where participants can build confidence in each other (Sutherland and Jarrahi, 2018). Therefore, it is important to understand factors influencing and limiting the transactions on digital platforms, and as a result, their economic benefits for society.

* * *

In this dissertation, I empirically study the economics of digital platforms in four essays, organized in four self-contained chapters, that can be read independently. All essays focus on different aspects of digital platforms, either their design or how phenomenons of the offline world are mirrored in the online world. Physical proximity among knowledge workers on digital platforms is the focus of the *first essay*, and its importance for productivity. In the *second essay*, I provide evidence on labor market signaling as a motivation to contribute to the public good creation on digital platforms. In the *third essay*, I demonstrate the significance of peers on large digital platforms for user activity. Finally, the *fourth essay* explores platform designs and means to incentivize independent creators to supply content, while controlling the content quality. It shows, that even small changes in incentives can have important impacts on the content supply.

In the *first chapter* I focus on the geographic concentration of activity in the field of software engineering. The existing literature on the productivity effects of agglomerations measures innovation with patent data, capturing only a fraction of the industry's activity (Cohen and Lemley, 2001; Moretti, 2021). However, research suggests that physical proximity is, compared to other knowledge workers, even more important among open source software (OSS) contributors (Wachs et al., 2022), with Silicon Valley as the epitome of a tech cluster. This

in itself is surprising, because physical proximity should not matter for coding, as the 'death of distance' hypothesis claims (Cairncross, 1997; Baldwin, 2017; Baldwin and Dingel, 2022). It states that with the rise of internet and communication technologies, physical distance should lose importance. Though, for instance, videoconferencing hinders the generation of new ideas in collaboration. It cannot replace in-person interactions because the cognitive focus is limited by the focus on a screen (Brucks and Levav, 2022). Together with Thomas Fackler and Moritz Goldbeck, I use data from the largest online code repository platform *GitHub* to provide an alternative proxy for productivity, covering a broad range of software engineering, and present evidence on the importance of physical proximity in this field. I find a positive relationship between cluster size of other knowledge workers in the same field and productivity, which I validate as causal with instrumental variable and dynamic estimation approaches. By focusing on OSS contributors, I am able to provide an estimate for agglomeration effects on productivity in a high-tech sector with traditionally low patenting activity (Cohen and Lemley, 2001; Carlino and Kerr, 2015).

In the *second chapter* I turn to the puzzle of the voluntary contribution motivation by software developers on OSS platforms. OSS is a public good, which is increasingly used as input for modern products and services (Nagle, 2022). The motivation of software engineers to contribute to OSS, though, is unclear (Hars and Ou, 2002; Lerner and Tirole, 2002; Krishnamurthy, 2006). With Moritz Goldbeck, I focus on OSS developers on *GitHub* and provide evidence for career concerns as motivation to contribute by leveraging time differences in incentives for labor market signaling. For that, I compare users who move for a job to users who move for other reasons and find elevated OSS activity in the job search period for users who move for a job compared to users who change location for other reasons. The increase in activity is mainly driven by contributions to projects that increase external visibility of existing works and are written in programming languages that are highly valued in the labor market. Digital platforms, thus, offer a place for knowledge workers to signal their skills, whilst simultaneously contributing to the public good provision.

The *third chapter* analyzes the importance of social learning on digital platforms. Individuals learn and are influenced by the decisions of their peers, which is called social learning (Cai et al., 2009). Peers' decisions matter, for instance, for consumption choices, educational choices, or financial market choices, among others (Sacerdote, 2011; Anderson and Magruder, 2012; Ouimet and Tate, 2020; Bailey et al., 2022). The relationship between an individual's choice and their peers' decisions in the context of work is less clear. Further, on large digital platforms, observing others and their behavior may help to navigate through the extensive

information space. I focus again on the OSS platform *GitHub*, to study the importance of highly influential and skilled users, called rockstars, on project activity and project popularity. By observing the activity pattern of rockstars, individuals seem to learn about high-quality and promising projects and follow the rockstar's lead. After a single and minimal contribution by a rockstar, project activity, and popularity are elevated relative to similar projects without such a contribution. The relationship between rockstar contribution and project activity increases the more influential the rockstar is, and the less information about the project quality is available. This shows, that highly influential individuals are a way to navigate on large digital platforms, and their contribution is associated with elevated short-term project activity, and long-term attention.

The *fourth chapter* studies the design of digital platforms. Platforms, where the provided content stems from independent creators, want to implement incentives to generate high-quality content by the creators, while simultaneously taking actions to prevent "bad-faith" actors, which could do damage to the platform's reputation (Geva et al., 2019; Huang et al., 2022). One such instrument is sharing the platform's advertising revenues with the creators via partnership programs (Tang et al., 2012). With Anna Kerkhof and Johannes Loh, I investigate how alterations in access to partnership programs affect content supply. For that, I focus on creators who have lost access to the partnership program on YouTube because of the implementation of new and higher eligibility criteria. My results show that affected creators decreased their activity on the platform after the rule change by exhibiting lower video upload frequency, and reduced video quality and diversity. Further, my findings indicate that next to monetary incentives, also non-pecuniary motivation seems to be important for creating content on the platform. Especially more experienced creators reacted to a larger extent when being removed from the partnership program. They potentially built an identity-based attachment to the platform. Ad-based platforms should, thus, take both, content creators' monetary as well as non-pecuniary motivation, into account when using partnership programs to incentivize and control content supply.

In this dissertation, I report on important factors, such as physical proximity, labor market signaling, social learning, and partnership programs, on digital platforms that influence the performed transactions on them. Even if in the digital economy transaction costs across space are close to zero, the geographical distribution of platform contributors remains important. On the one hand, digital platforms allow previously too far-off actors to engage with each other. On the other hand, the benefits of physical proximity remain crucial on digital platforms as well. Digital platforms also resemble a new way of showcasing skills for career reasons.

Firms can learn from the activity of individuals on the platforms about their abilities, which can, thus, be a means of motivation for individuals to contribute when searching for a job. Further, identifying projects of high usability and quality on platforms can be costly, so learning through the activities of others about the quality of a project is a way to overcome this problem. Lastly, the platform design and its incentives for content suppliers greatly affect product production. To maximize the economic benefits, suitable incentives need to be implemented. This dissertation offers some insights into these considerations, but many other aspects of digital platforms are left to analyze. Platforms continue to evolve with new features created by technological advancements, which may reduce the impact of some of the reported findings in this dissertation, whilst introducing other facets to analyze. In sum, this dissertation points out several aspects of digital platforms which affect the behaviour of actors in the digital space.

* * *

The focus of this dissertation is on the dynamics of actors on digital platforms. By the nature of digital platforms, they provide immense data to be analyzed. This allows researchers to apply new methods such as Natural Language Processing (NLP) or other Machine Learning (ML) methods. Further, novel types of data emerge, such as video or audio data. Many platforms make their data accessible or offer the opportunity to run experiments on them. By that, new and more detailed insights can emerge, whilst due to the enormous data size, researchers can have higher confidence in the reliance of these insights. In the future, I hope that even more platforms make their data available for research to further increase the understanding of individuals' behaviour in the digital economy.

Keywords: high-skilled labor; geography; innovation; peer effects; collaboration; software; knowledge work; digital platforms; signaling; job search; social interactions; social multiplier; open source; platform governance; partnership programs; content supply; ad-based business models; access restrictions

JEL-No: D62; D83; H40; J24; J30; L17; L84; L86; O18; O30; O33; O36; R32

Acknowledgments

This dissertation would not have been possible without my dissertation supervisor Oliver Falck. I am grateful for Oliver's valuable advice, genuine encouragement, and unwavering guidance. He was always open to discussing my research and offered helpful comments to find new solutions, whilst allowing me to explore and follow my own research interests. I am deeply appreciative of the freedom in research I had because of him. I also thank my co-supervisor Florian Englmaier for his inspiring motivation and academic support from very early on. He provided me with new perspectives and ideas on my research while pointing out important pitfalls. Lastly, I want to thank Lisandra Flach for completing my dissertation committee, and her thoughtful and beneficial remarks on my research.

I benefited immensely from the collaborations with my co-authors and colleagues Moritz Goldbeck, Thomas Fackler, Anna Kerkhof, and Johannes Loh. The joint work made my PhD journey a lot more doable, enjoyable and contributed to a great part of my acquisition of knowledge and skills. To me, especially the conversations and the opportunity to philosophize with them about our work allowed me to get a deeper understanding of economic research. They were an important addition to the otherwise large part of work in solitude.

I am also thankful for my colleagues at the ifo Center of Industrial Organization and New Technologies – Christina, Fabian, Mo, Nikola, Sebastian, Simon, Valentin, and Victor for many inspiring discussions and comments throughout the years. Furthermore, I want to thank many other colleagues and friends from the ifo Institute and the LMU Munich for their support and fruitful discussions.

Finally, I want to thank my friends and family, who supported me through the highs and lows of my PhD journey and were always able to offer me new perspectives on my research. They were important for me to keep the balance between focusing on research and enjoying life outside academia.

Lena Marie Abou El-Komboz, September 2024

Contents

List of Figures

List of Tables

1 Productivity Spillovers among Knowledge Workers in Agglomerations: Evidence from GitHub

Software engineering is prototypical of knowledge work in the digital economy and exhibits strong geographic concentration, with Silicon Valley as the epitome of a tech cluster. We investigate productivity effects of knowledge worker agglomeration. To overcome existing measurement challenges, we track individual contributions in software engineering projects between 2015 and 2021 on *GitHub*, the by far largest online code repository platform. Our findings demonstrate individual productivity increases by 2.8 percent with a ten percent increase in cluster size, the share of the software engineering community in a technology field located in the same city. Instrumental variable and dynamic estimation results suggest these productivity effects are causal. Productivity gains from cluster size growth are strongest for clusters hosting between 0.67 and 13.5% of a community. We observe a disproportionate activity increase in high-quality, large, and leisure projects and for co-located teams. Overall, software engineers benefit from productivity spillovers due to physical proximity to a large number of peers in their field.[1]

Keywords: high-skilled labor; geography; innovation; peer effects; collaboration
JEL-No: D62; J24; O33; O36; R32

[1] This chapter is based on joint work with Moritz Goldbeck and Thomas Fackler. Versions of this chapter have been published as CRC Discussion Paper 399. We thank Florian Englmaier, Oliver Falck, Anna Kerkhof and Chris Stanton for valuable comments and suggestions. We also thank conference participants at EEA2021, EARIE2021, VfS2021, 16th North American Meeting of the Urban Economics Association; and participants at CRC Retreat Schwanenwerder 2021, CESifo/ifo Junior Workshop on Big Data 2021 and an ifo internal seminar. Support by the Deutsche Forschungsgemeinschaft through CRC TRR 190 (project number 280092119) and by the bidt Think Tank project ”Changing workplaces: Patterns and determinants of technology and skill adoption by firms and individuals” is gratefully acknowledged. Thomas Fackler thanks the Laboratory for Innovation Science at Harvard for their hospitality while writing parts of this paper.

1.1 Introduction

Urban density is associated with higher wages and productivity. One of the main reasons for this relationship is improved diffusion of knowledge through physical proximity (Jaffe et al., 1993; Glaeser, 1999; Atkin et al., 2022). Knowledge spillovers among workers occur when individuals benefit from the skills of their local peers and learn from each other, which increases productivity (Lucas, 1988; Cornelissen et al., 2017; De La Roca and Puga, 2017). Knowledge spillovers are especially important in innovative sectors (Audretsch and Feldmann, 1996), where collaboration and learning are crucial (Carlino et al., 2007; Jones, 2009; Azoulay et al., 2010; Combes et al., 2010; Andersson et al., 2014; Catalini, 2018). To exploit localized advantages related to collaboration and knowledge exchange, workers and firms tend to locate near each other, especially within a research field or industry (Alcácer and Chung, 2007; Carlino and Kerr, 2015; Moretti, 2021). This leads to geographical agglomeration of tech industries in few cities (Carlino et al., 2012; Atkinson et al., 2019). Surprisingly, software engineering, a key component of almost any high-tech endeavor today (Chattergoon and Kerr, 2022), is characterized by a particularly high spatial concentration of workers in a couple of large clusters (Kerr and Robert-Nicoud, 2020; Forman and Goldfarb, 2022; Wachs et al., 2022), even though it is highly digitized and codified.

In this paper, we investigate agglomeration effects in software engineering. Specifically, we examine the effect on software engineers' productivity of being located in cities with a larger share of other software engineers in their technology field. To this end, we exploit exogenous variation in cluster size resulting from software engineers moving across cities and joining or leaving a specific technology, an approach pioneered by Moretti (2021). This allows us to estimate the impact of changes in technology-specific cluster size on software engineers' productivity in the respective technology. We deploy a model that features a restrictive number of high-dimensional fixed effects to elicit productivity effects, considering both output quantity and quality as well as effect heterogeneity. Still, estimating agglomeration effects on productivity poses further challenges such as simultaneity and correlated unobserved productivity shocks (Combes et al., 2010). To address these challenges, we investigate effect dynamics and employ an instrumental variable approach by predicting variation in local cluster size from changes originating elsewhere. This shift-share approach ensures that the variation in cluster size is independent of technology-specific local productivity shocks, mitigating potential bias in estimates of the elasticity of productivity with respect to cluster size.

Data from *GitHub*, the by far largest online code repository platform, allows us to track software

engineers' productivity at unprecedented resolution. Our data has several crucial advantages over patent data, which the existing literature almost exclusively relies upon as a measure of productivity in the knowledge economy (see, e.g., Jaffe et al., 1993; Carlino et al., 2007; Carlino and Kerr, 2015; Guzman and Stern, 2020). While only a small share of knowledge workers files patents and there are large differences across fields and idea types (Cohen and Lemley, 2001; Carlino and Kerr, 2015), coding is a much more widespread activity and part of almost any high-tech project today (Andreessen, 2011; Tambe et al., 2020). *GitHub* data captures even smallest individual contributions to collaborative projects instantaneously with an exact timestamp. In contrast, for inventor teams, it is unclear who contributed what and when. Only one team outcome, the final patent application, is observable with a significant reporting lag. In addition, patents differ widely in market value and often are never used in production (Boldrin and Levine, 2013; Kogan et al., 2017). Code uploaded to *GitHub* is, by definition, more applied and always used in a software product or component. We, therefore, propose code changes by users on *GitHub*, called commits, as a novel measure of knowledge worker productivity and exploit the granularity and richness of the information from public projects in the *GHTorrent* database (Gousios, 2013), such as the integrated social features on the platform, to track the quantity and quality of software engineers' individual output over time.

Our findings indicate that cluster size, the share of other users in a field located in the same city, positively impacts software engineers' productivity. Specifically, a ten percent increase in technology-specific cluster size is associated with a 2.8 percent increase in user output in that technology. Non-parametric estimation shows the elasticity of productivity with respect to cluster size is largest for clusters hosting between 0.67 and 13.5% of a technology-specific community. Agglomeration effects are smaller for clusters with a community share below or above this range, indicating clusters need a critical mass of users to reap significant productivity benefits from agglomeration. An extensive set of fixed effects precludes that the productivity effect is driven by unobserved heterogeneity or trends. Additionally, contemporaneous effects and IV estimation mitigate potential remaining concerns regarding endogeneity due to sorting and simultaneity.

Heterogeneity analyses suggest that the effects are significantly larger for high-quality projects with increased use-value for the community as measured by stars and forks on the platform. Relative to the baseline estimates, activity increases disproportionately with cluster size in longer-running, larger, and co-located projects with more team members, indicating that especially collaborative projects are able to tap productivity spillovers from the wider local

community. Additionally, we observe a higher activity increase in leisure projects with a high share of commits out of business hours, which are typically not integrated in a formal structure of an organization. Additional analyses demonstrate robustness of our results with respect to measurement and modeling choices as well as sample construction.

This study contributes to three strands of literature. First, we add to the extensive literature exploring agglomeration effects. There is growing descriptive evidence documenting increasing geographic concentration of innovative activity (Verspagen and Schoenmakers, 2004; Bettencourt et al., 2007; Balland et al., 2020) where collaboration and teamwork are essential (Wuchty et al., 2007; Jones, 2009). Agglomeration is much less pronounced in manufacturing (e.g., Ellison and Glaeser, 1997), and recent evidence by Chattergoon and Kerr (2022) links growing concentration to the rise in software intensity. Rising concentration in knowledge-intensive sectors is remarkable as adoption of information and communication technology is high and tends to reduce geographic frictions (Agrawal and Goldfarb, 2008; Steinwender, 2018; Goldbeck, 2023). Presence of strong localized knowledge spillovers (e.g., Audretsch and Feldmann, 1996; Ganguli et al., 2020; Catalini, 2018; Rosenthal and Strange, 2020; Bikard and Marx, 2020) might explain rising geographic concentration. Notably, Moretti (2021) estimates aggregate effects on inventor productivity of geographic clustering. We are first to focus explicitly on software engineering and demonstrate that individual-level productivity effects of agglomeration in this field are significantly higher.

Second, we advance the measurement of innovative activity by introducing a novel proxy for knowledge worker productivity, the number of single code contributions to software engineering projects. This metric helps us overcome several shortcomings of existing measures based on patent data, which the literature almost exclusively relies upon (Acs et al., 2002; Lerner and Seru, 2022). With the rise of the service economy (Buera and Kaboski, 2012) software becomes ubiquitous in innovation (Andreessen, 2011; Chattergoon and Kerr, 2022). At the same time, software and information technology constitute an increasingly important blind spot of patent data (Acikalin et al., 2022; Lin and Rai, 2024). Our measure addresses this gap by proposing a more appropriate and reliable metric for innovative activity in software engineering. Furthermore, our measurement approach is more broad-based, capturing a less exclusive set of individuals compared to inventors, and granular both in terms of time resolution and assessment of individual output.

Third, our paper contributes to the understanding of peer effects. A large literature tries to quantify the extent to which individuals benefit from their peers (Angrist, 2014; Herbst

and Mas, 2015; Sacerdote, 2014). With a historically strong focus on learning in educational institutions (Manski, 1993a; Sacerdote, 2001; Jackson and Bruegmann, 2009) and science (Azoulay et al., 2010; Waldinger, 2012), this body of research extends to studies of the workplace and professional domain (Moretti, 2004; Mas and Moretti, 2009; Cornelissen et al., 2017). We add to this literature by using plausibly exogenous variation in the density of local peers to study their effect on individual-level productivity on a broad sample of knowledge workers in software engineering. Our technology field-specific definition of relevant communities of peers shows that even within software engineering, a fairly narrow domain according to traditional industry classifications, peer effects are confined to specific sub-fields.

The remainder of this paper is organized as follows. We discuss the setting and data in Section 1.2. Section 1.3 introduces our empirical strategy. In Section 1.4, we report the results and Section 1.5 concludes with a brief discussion.

1.2 Background and data

Today, software engineering is a crucial part of almost any scientific and innovative endeavor or high-tech product (Andreessen, 2011; Webb et al., 2018; Tambe et al., 2020; Chattergoon and Kerr, 2022; Aum and Shin, 2024), be it in artificial intelligence, engineering, app development, or the bio-pharmaceutical industry. For example, software engineers at the biotech company *Moderna* designed an artificial intelligence that greatly improved the speed of mRNA drug discovery and development, leading to one of the first vaccines against Covid-19 on the market (Bean, 2024). In practice, the vast majority of software engineering projects is hosted on the online code repository platform *GitHub*, which is based on the `git` version control system. The platform launched in 2008 and since then rapidly evolved as the main online platform for hosting code and collaborative software development (Fackler and Laurentsyeva, 2020). A free basic version and its ease of use due to seamless integration into software engineering tech stacks make *GitHub* attractive for over 100 million users (Dohmke, 2023). In addition, the platform exhibits features of a social network in line with its motto “social coding” (Lima et al., 2014).

On the platform, users can create and collaborate in projects (*repositories*) to which code can be *pushed*, i.e., uploaded. The smallest unit of user activity in projects is a *commit*, which captures the sum of code changes a user sends to the project during a session. We introduce commits as a novel measure of software developer productivity. Using commits has several advantages over patent data, the most commonly used measure in the literature. Coding is

essential in software development and therefore widespread, in contrast to patenting, which also differs widely across different fields and idea types (Cohen and Lemley, 2001; Carlino and Kerr, 2015). In addition, commits capture even small contributions by each individual with an exact timestamp. In patent data, only one team outcome is observed with a significant reporting lag and neither the nature nor the timing of individual members' contributions are observed. Patents also differ widely in use and value (Boldrin and Levine, 2013; Kogan et al., 2017); commits capture more applied activity and are used in software by definition. The *GitHub* platform contains further information. For example, users may *star* a project so that it is bookmarked for future reference. The number of stars per project measures popularity among other users and is a proxy for project quality (Lima et al., 2014). User profiles allow users to showcase their work and display public projects and activity as well as biographical information such as a name, location and organizational affiliation.

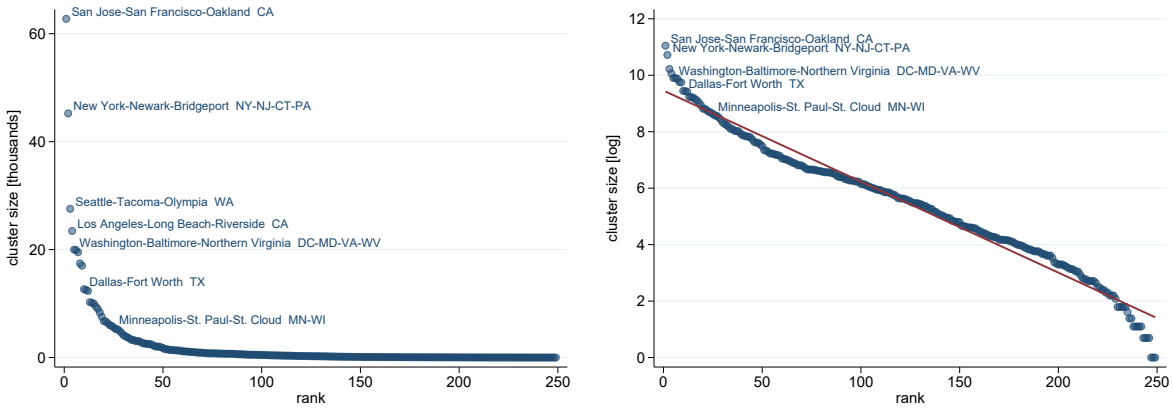
We tap *GHTorrent*, a relational database that mirrors the *GitHub* REST API and creates approximately biannual snapshots of public user profiles and activity on the platform. To obtain time-varying user information, we query ten snapshots dated between September 2015 and March 2021 for profiles of users with location in the US or Canada.[2] For these users, we extract the activity stream with timestamped information on commits and project activity from the latest available snapshot (March 2021). We then combine the activity stream and user profiles into a panel with ten time intervals arising from the snapshot dates.[3] Based on their self-reported location, we assign users to one of the 179 US economic areas defined by the *Bureau of Economic Analysis* or the Canadian equivalent, i.e., one of the 76 economic regions by *Statistics Canada* to city coordinates via exact name matching.[4] Economic areas delineate the "relevant regional markets surrounding metropolitan or micropolitan statistical areas" (Johnson and Kort, 2004). Generally, economic areas are similar to Metropolitan Statistical Areas (MSAs), but tend to be larger than corresponding MSAs for big cities to capture entire economic regions. Henceforth, we refer to this geographic definition as 'cities'.

[2] Specifically, snapshots dates in our data are 2015/09/25 (201509), 2016/01/08 (201601), 2016/06/01 (201606), 2017/01/19 (201701), 2017/06/01 (201706), 2018/01/01 (201801), 2018/11/01 (201811), 2019/06/01 (201906), 2020/07/01 (202007) and 2021/03/06 (202103). Goldbeck (2023) validates user locations in *GHTorrent*. For users with a reporting gap in the location information, we impute their location from the previous or next snapshot if possible.

[3] In *GHTorrent*, users are assigned a unique identifier. In principle, commits can be linked to users via `author_id` or `committer_id`. Since users may commit code authored by someone else, we link by `author_id`. This method ensures close connection to individual productivity, but is conservative as many users possess multiple accounts (Casalnuovo et al., 2015).

[4] US and Canadian city coordinates are sourced form `maps` (Becker and Wilks, 2018) and *SimpleMaps* (Simplemaps, 2021).

Figure 1.1: Agglomeration in software engineering



(a) rank-size distribution **(b)** logarithmic size

Sources: GHTorrent, own calculations.

Figure 1.1 displays the strong spatial concentration of software engineers (see, e.g., Kerr and Robert-Nicoud, 2020; Forman and Goldfarb, 2022; Wachs et al., 2022; Goldbeck, 2023) by plotting the number of users in each city as rank-size distribution. Silicon Valley (i.e., the economic area "San Jose–San Francisco–Oakland, CA") clearly stands out as the epitome of a tech cluster with more than 60 thousand users in our data. Cluster size rapidly decays with city rank, with the next largest cities being New York, Seattle, Los Angeles, and Washington, DC. About 50% of users are located in the ten largest cities. In contrast, the vast majority of cities host only few users. The right panel displays the rank distribution using logarithmic city size. Even here, geographic concentration in few large cities is prominently visible as the largest cities lie well above the linear power-law approximation of the distribution.

Technology clusters Since agglomeration benefits from localized knowledge spillovers are concentrated within related fields (see, e.g., Alcácer and Chung, 2007; Bloom et al., 2013; Carlino and Kerr, 2015; Moretti, 2021), we define cluster size on the city × technology level. For this purpose, we exploit that a programming language is recorded for each project and assign this programming language to every commit in that project.[5] We use the 18 most frequently occurring programming languages that cover about 90% of all commits.[6] Since different programming languages can be closely related, we group programming languages

[5] Programming languages are broadly defined and include databases and frameworks. Note that a project may contain files in several programming languages. *GHTorrent* assigns the programming language that makes up the largest number of bytes in the project.

[6] Limiting the total number of 404 programming languages to 18 avoids having a large number of cities with only one user in a particular programming language.

into five 'technologies' based on being frequently used together according to a developer survey (StackOverflow, 2020).[7] We determine the technology of a user in each time interval via her commit activity. For example, a user who commits to projects in technologies 1 and 3 in the second time interval and lives in Los Angeles is part of the clusters Los Angeles × Technology 1 and Los Angeles × Technology 3 in that time interval. Figure A.2 plots the rank-size distribution by technology, which shows a similar pattern within technologies as for all technologies together (Figure 1.1). The top ten clusters by technology and their respective user share are listed in Table A.2.

We hypothesize users benefit from being located in a city that hosts a larger share of the community in a specific technology. To robustly compute cluster size, we require a minimum user activity of committing in at least two time intervals.[8] There are 478,957 such users with a location in the US or Canada. Cluster size S for user i in time t in technology f in city c is computed as

$$S_{-ifct} = \frac{\sum_{j \neq i} N_{jfct}}{\sum N_{jft}}, \tag{1.1}$$

where the summation of users N across all users j in city c in technology f in time t, excluding user i, is divided by the total number of users N in technology f in time t. The accuracy of our measure of cluster size relies on users providing correct location information and maintaining up-to-date profiles. To maximize benefits of the social network functionality and increase visibility for local peers, users generally have an incentive to maintain correct profile information. Reassuringly, we exactly match 98.6% of locations. In addition, Goldbeck (2023) finds no bias in the location information compared to patent data and Abou El-Komboz and Goldbeck (2024) verify the timing of users' location changes on the platform.

Sample For our regression analyses, we select North American users active throughout the observation period, i.e., non-zero commits in all time intervals. This results in a sample of 21,116 users and 2,527,496 user-project-time observations. Summary statistics are reported in Table A.1. The median user makes 56 public code contributions per time interval, i.e., within about six months, and is active in two technologies. Like on most online platforms, activity is

[7] A visualization of the technology clusters can be found at `https://insights.stackoverflow.com/survey/2020#correlated-technologies`; last accessed on 03/17/2023. Technology 1 contains JavaScript, CSS, HTML, PHP, C# and TypeScript; Technology 2 Python, Shell, Go, Jupyter Notebook, and R; Technology 3 Ruby; Technology 4 Java, Objective-C, and Swift; and Technology 5 C++, C and Rust.

[8] Note that actual user activity likely is much higher as only public activity is observed. We include users whose account was created in the last time interval and who commit in that time interval.

heavily right-skewed. Only few projects receive stars and forks. The median city hosts users active in 17 programming languages and all five technologies. Overall, our sample captures a broad base of software engineers with constant activity on the platform that allows us to measure meaningful changes in output.

1.3 Estimation Strategy

We study the effect of cluster size on productivity by estimating the following fixed-effects panel data model via ordinary least squares:

$$\ln(y_{ijflct}) = \alpha + \beta \ln(S_{-ifct}) + d_i + d_j + d_{cf} + d_{cl} + d_{lt} + d_{ct} + \mu_{ijflct}, \tag{1.2}$$

where y_{ijflct} is the number of commits of user i in time interval t to project j located in city c in the technology f and programming language l and S_{-ifct} is the cluster size in city c of the technology f in time interval t, excluding user i. μ_{ijflct} is an error term. We cluster standard errors at the city × technology level to account for serial correlation. Importantly, this specification allows us to include a large amount of (high-dimensional) fixed effects d that address many potential concerns regarding identification and ensures that the identifying variation in cluster size originates from users moving between cities and starting or stopping to be active in a technology field.

In particular, user fixed effects d_i capture time-invariant differences in user activity, and project fixed effects d_j account for project-specific activity differences. In addition, we include city × technology d_{cf} and city × programming language d_{cl} fixed effects to control for city-specific productivity differences within technologies and programming languages. For example, if programmers in Toronto focused on artificial intelligence within projects, these fixed effects would account for the fact that such a specialization could systematically affect observed activity. Similarly, programming language × time fixed effects d_{lt} account for programming language-specific time trends and city × time fixed effects d_{ct} consider changes in average productivity over time for each city as well as changes in city size over time. These fixed effects would capture activity patterns over time, e.g., caused by new cohorts of students learning to program in a language in project-based courses at the start of the academic year.

Our coefficient of interest β captures the relationship between cluster size and user productivity conditional on fixed effects. The identifying variation net of fixed effects comes from users relocating to another city and starting or stopping to commit in a specific technology, similar to Moretti (2021). Thus, this relationship can be causally interpreted if the

included fixed effects eliminate endogeneity and the error term μ_{ijflct} is orthogonal to cluster size S_{-ifct}. Productivity spillovers from agglomeration are present if β is greater than zero and absent if β is zero. In particular, a positive β implies a user's productivity in a technology increases with cluster size, i.e., the share of other users in that technology being located in the same city.

An endogeneity concern when estimating agglomeration effects are unobserved determinants in the error term μ_{ijflct} simultaneously affecting productivity and cluster size (Combes and Gobillon, 2015). In particular, potential concerns are sorting and simultaneity. Equation 1.2 accounts for most forms of sorting into cities and technologies, e.g., due to (changes in) local amenities and infrastructure, by ability, or differences in technology-specific productivity differences across cities. Still, reverse causality might arise when users whose productivity would have increased anywhere sort into larger clusters. Note that sorting into large clusters on ability is not a concern, nor is sorting to the extent that it leads to an increase in cluster size affecting productivity. Only when users with expected future productivity increases select into growing clusters. A more salient potential concern is simultaneity due to unobserved time-varying productivity shocks that are technology-specific, such as policies at the city level that target a specific technology coinciding with cluster size growth.

We address potential bias due to unobserved time-varying productivity shocks at the city $\times$ technology level using an instrumental variable (IV) approach similar to Autor et al. (2013). The idea is to use only the part of variation in local cluster size that is arguably exogenous because it originates elsewhere. By that, unobserved local productivity shocks at the city $\times$ technology level that affect both productivity and cluster size simultaneously do not affect our estimate. To construct a valid instrument, we leverage a key feature of online code platforms, namely the possibility to commit to projects from anywhere. We instrument local cluster size by commits to local projects that originate elsewhere. Users on *GitHub* frequently contribute to non-local projects, which provides sufficient variation in the number of committers from different cities. At the same time, increases in activity originating elsewhere are unlikely to be an outcome of local productivity gains and are arguably unrelated to unobserved local productivity shocks at the city $\times$ technology level.

In particular, we predict cluster size by changes in the number of non-local users in all projects of a particular technology to which other local users commit, excluding the focal user's projects[9], relative to the change in the overall number of users in that technology. We denote

[9] We consider a user to be connected to a project if she ever committed to that project, not only in the current time interval.

the sum of users committing to project j in time interval t and technology f, excluding city c, as $N_{jf(-c)t}$ and its change between $t-1$ and t as $\Delta N_{jf(-c)t} = N_{jf(-c)t} - N_{jf(-c)(t-1)}$. We compute our instrument as

$$IV_{ifct} = \sum_{s \neq j_i} D_{sfc(t-1)} \frac{\Delta N_{sf(-c)t}}{\Delta N_{ft}}, \tag{1.3}$$

where $D_{sfc(t-1)}$ indicates if project s in technology f was present in city c at time $t-1$. $N_{sf(-c)t}$ is the logarithm of the sum of users committing to project s in technology f at time t in all cities but city c, and to which user i does not commit. Consequently, $\Delta N_{sf(-c)t}$ is the change in the logarithm of the number of users committing to project s in technology f at time t for all cities but city c and ΔN_{ft} is the change in the logarithms of the total number of users in technology f between time $t-1$ and t.

1.4 Results

1.4.1 Main results

Table 1.1 reports the results from our baseline model in Equation 1.2. The first column conditions on user, project, programming language, technology, city, and time fixed effects. The estimated elasticity of user productivity with respect to cluster size in this specification is 0.1144, suggesting a positive relationship of productivity and cluster size. Adding programming language × time fixed effects in the second column accounts for trends in programming languages and technologies as well as language-specific productivity shocks common to all users. The decrease in effect size hints that larger clusters experience higher productivity gains from increased popularity of programming languages most frequently used there. After including city × technology fixed effects in the third column, the elasticity of cluster size increases to 0.1966 and becomes statistically significant at the five percent level. This specification takes into account time-invariant technology-specific factors at the city level that affect user productivity. Higher task complexity in large clusters (Balland et al., 2020) causing users to take longer for each commit compared to equally productive workers elsewhere is a possible explanation for this increase in effect size. Accounting for city × language fixed effects in column four leaves estimates virtually unchanged, suggesting that our definition of technologies and clusters appropriately captures relevant software engineering communities.

Our preferred specification in column five adds city × time fixed effects to account for unobserved productivity shocks at the city level common to all technologies like policies improving local digital infrastructure or the establishment of a presence by a large tech

firm. This results in an estimated elasticity of productivity with respect to cluster size of 0.2777, which is statistically significant at the five percent level. The increase compared to column four suggests that city-specific productivity shocks or sorting on local amenities are especially pronounced in smaller clusters. Overall, these results consistently point to significant agglomeration effects in software development. Users are more productive when located in a city with a higher share of other users in their technology. Our preferred estimate implies users on average make 2.8% more commits in a given technology when the share of other users in that technology is ten percent higher. This finding suggests that, for example, a user's number of commits in Technology 1 is expected to increase by 19% if she moves from Chicago to Seattle due to the larger community of users in Technology 1 there.

Compared to the agglomeration effect for top inventors estimated by Moretti (2021), we thus find a four times larger elasticity for software engineers. Several factors might explain these stronger agglomeration effects in software engineering. First, software engineers tend to be younger than patenting inventors and, therefore, learning skills are more important to them in the human capital accumulation phase of their life cycle (Ben-Porath, 1967).[10] Second, the high degree of specialization in software development implies a higher probability that the activity of local peers is relevant to the focal user, leading to a larger potential for knowledge spillovers. Third, software is a particularly fast-moving field with a high rate of skill obsolescence (Deming and Noray, 2020a) even within STEM fields, requiring continuous learning to maintain and possibly increase productivity. Larger knowledge spillovers in software engineering compared to other fields are a strong incentive for agglomeration, which might, at least partly, explain the particularly high geographic clustering of programmers.

The elasticity of productivity with respect to cluster size might change depending on the position of cities in the size distribution. For example, productivity spillovers potentially require a certain minimum cluster size to occur as the benefits to individual productivity of only few other co-located users might be smaller. In contrast, similar increases in cluster size might result in smaller relative productivity gains in the largest clusters where already many users are co-located. The presence of both channels could give rise to an S-shaped relationship of the elasticity with respect to cluster size. Au and Henderson (2006), for example, estimate a bell-shaped relation between productivity and city size for Chinese cities and Cattaneo et al. (2021) demonstrate an S-shape pattern for the elasticity of US inventors in Moretti (2021).

[10] Survey results suggest that most software engineers in the US are aged 25-35 years (Patel, 2024; StackOverflow, 2020), whereas inventors are significantly older (Jones, 2010) with an average age of 45 years (Kaltenberg et al., 2023).

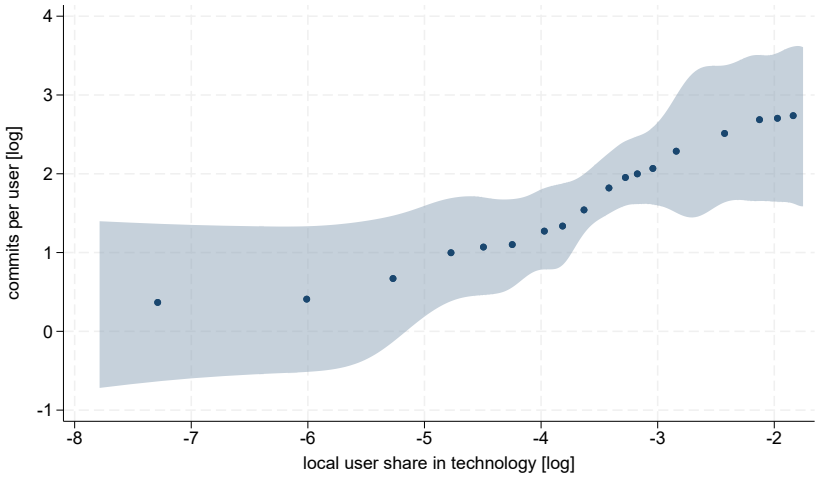
Table 1.1: Productivity and cluster size

Dep. var.: Commits [log]	(1)	(2)	(3)	(4)	(5)	(6)
Cluster size [log]	0.1144	0.1070	0.0929	0.1966**	0.1935**	0.2777**
	(0.1099)	(0.0785)	(0.0744)	(0.0949)	(0.0962)	(0.1253)
Fixed effects						
User	Yes	Yes	Yes	Yes	Yes	Yes
Project	Yes	Yes	Yes	Yes	Yes	Yes
Technology	Yes	Yes	Yes	Yes	Yes	Yes
Language	Yes	Yes	Yes	Yes	Yes	Yes
City	Yes	Yes	Yes	Yes	Yes	Yes
Time	Yes	Yes	Yes	Yes	Yes	Yes
Technology × time		Yes	Yes	Yes	Yes	Yes
Language × time			Yes	Yes	Yes	Yes
City × technology				Yes	Yes	Yes
City × language					Yes	Yes
City × time						Yes
Users	21,116	21,116	21,116	21,116	21,116	21,116
Adjusted R^2	0.287	0.289	0.290	0.291	0.291	0.292
Observations	2,527,496	2,527,496	2,527,496	2,527,496	2,527,496	2,527,496

Notes: Language refers to programming language. Robust standard errors clustered at the city × technology level in parenthesis. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Figure 1.2 depicts a binscatter plot to investigate monotonicity and potential non-linearity in the effect. Following the principled approach of Cattaneo et al. (2024), we obtain a suitable data-driven visualization of the conditional mean function. The relationship between productivity and cluster size is positively monotonous and follows a slight S-shape. The function increases only slightly for very small cluster sizes, while productivity increases are larger for cluster sizes between approximately 0.67 and 13.5 percent. Above this range, the increase is, again, less pronounced for the largest clusters. This suggests that significant agglomeration effects require a minimum cluster size of around 0.67 percent of the community in a technology being located in the same city. At the same time, when cluster size reaches a level of approximately 13.5 percent, there are little additional productivity gains from further growth in cluster size. This also suggests that our effect is not driven by few large clusters such as the Bay Area. Rather, the effect is present across the entire size distribution and features a slight S-shape with especially medium-sized clusters profiting from increases in cluster size.

Figure 1.2: Non-parametric estimation



Notes: Graph plots a binscatter representation of the relationship between software engineer productivity and cluster size using `binsreg` (Cattaneo et al., 2021). Specification includes fixed effects for time, technology, language, project, city, and user as well as for time × city, time × technology, and city × technology. *Sources:* GHTorrent, own calculations.

1.4.2 Heterogeneity

We explore potential heterogeneity of the effect with respect to user and project characteristics. To explore the relation between cluster size and quality of users' activity, we focus on commits to the top ten projects measured by the number of stars received from the community. Table 1.2 reports the main results for this subsample. Generally, the point estimates are significantly larger across all specifications compared to our baseline estimates. Effects are more precisely estimated, as well, even though the sample size is much smaller, pointing to a tighter relationship between cluster size and productivity in high-quality projects. For the preferred specification with the full set of fixed effects, the elasticity between cluster size and productivity of 0.3239 implies that a user commits about 3.2 percent more in a technology to projects with at least five stars with a ten percent increase in cluster size. This result indicates that the effect on high-quality activity is about 4.6 percentage points (or 16.6%) higher relative to the full sample in Table 1.1. Note that compared to specifications without city × time fixed effects, this difference is significantly smaller. This stresses accounting for time-varying unobservables at the city level like the opening of new large tech firm establishments is especially important for high-quality activity.

Table 1.3 explores heterogeneity with respect to further characteristics by estimating the relation of cluster size and productivity by quartiles. The first specification reports the effects

Table 1.2: Quality

Dep. var.: Commits [log]	(1)	(2)	(3)	(4)	(5)	(6)
Cluster size [log]	0.1451	0.1359	0.1229	0.2649***	0.2637***	0.3239**
	(0.1043)	(0.0866)	(0.0828)	(0.0860)	(0.0867)	(0.1462)
Fixed effects						
User	Yes	Yes	Yes	Yes	Yes	Yes
Project	Yes	Yes	Yes	Yes	Yes	Yes
Technology	Yes	Yes	Yes	Yes	Yes	Yes
Language	Yes	Yes	Yes	Yes	Yes	Yes
City	Yes	Yes	Yes	Yes	Yes	Yes
Time	Yes	Yes	Yes	Yes	Yes	Yes
Technology × time		Yes	Yes	Yes	Yes	Yes
Language × time			Yes	Yes	Yes	Yes
City × technology				Yes	Yes	Yes
City × language					Yes	Yes
City × time						Yes
Users	6,711	6,711	6,711	6,711	6,711	6,711
Adjusted R^2	0.407	0.408	0.409	0.410	0.412	0.413
Observations	392,984	392,984	392,984	392,984	392,984	392,984
$\Delta(\beta_{\text{top10}} - \beta_{\text{all}})$	0.0307	0.0289	0.0300	0.0683	0.0702	0.0462
$\Delta(\beta_{\text{top10}} - \beta_{\text{all}})/\beta_{\text{all}}$	0.2684	0.2701	0.3229	0.3474	0.3628	0.1664

Notes: Regressions based on the top decile of projects by stars. β_{top10} denotes the estimated coefficient on cluster size. β_{all} refers to the estimated coefficient of cluster size from the corresponding specification in Table 1.1. Robust standard errors clustered at the city × technology level in parenthesis. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

for each cluster size quartile. Similar to Figure 1.2, the results point to a slight S-shape of the elasticity of productivity with respect to cluster size. The differences are not pronounced as indicated by the Wald test, which yields a p-value of 0.170. The second specification investigates differences with respect to project age, measured in months since project creation. Theoretically, especially established projects might profit from cluster size as the initial set-up is typically trivial while in later phases external impulses are more beneficial to further improve the project (e.g., Ayoubi et al., 2017). Indeed, the elasticity increases with project age from 0.2639 (youngest quartile) to 0.2899 (oldest quartile). This variation is confirmed significant as a Wald test is rejected with a p-value of 0.046, suggesting that knowledge spillovers are

larger for older projects. Next, we study differences in the elasticity between business and leisure projects, which we elicit by the share of commits made during business hours. We find a significant variation in the elasticity (Wald test p-value of 0.006), with leisure projects benefiting more from increases in cluster size. Leisure projects typically exhibit less structure and are not embedded in a professional environment with a higher degree of knowledge organization and thus can profit more from spillovers from the wider local community. The fourth specification tests for differences in the elasticity with respect to user activity. Active users are often integrated more in local communities and therefore might experience larger productivity gains. We find sizable differences in point estimates, but the variation in the elasticity is not statistically significant (Wald test p-value 0.213). Thus, agglomeration effects are not significantly lower for less active users.

Table 1.3: Heterogeneity (by quartiles)

Dep. var.: Commits [log]	(1) cluster size	(2) project age	(3) business	(4) activity
1st Quartile (Smallest/Youngest/Leisure/Low)	0.2748**	0.2639**	0.2801**	0.2700**
	(0.1250)	(0.1228)	(0.1238)	(0.1234)
2nd Quartile	0.2688**	0.2661**	0.2806**	0.2716**
	(0.1258)	(0.1248)	(0.1261)	(0.1244)
3rd Quartile	0.2609**	0.2725**	0.2836**	0.2774**
	(0.1272)	(0.1268	(0.1254)	(0.1273)
4th Quartile (Largest/Oldest/Business/High)	0.2651**	0.2899**	0.2456**	0.3013**
	(0.1268)	(0.1263)	(0.1254)	(0.1287)
Full set of FE	Yes	Yes	Yes	Yes
Users	21,116	21,116	21,116	21,116
Adjusted R^2	0.292	0.292	0.292	0.292
Observations	2,527,496	2,527,496	2,527,496	2,527,496
Wald (joint nullity) [p-value]	0.170	0.046	0.006	0.213

Notes: Robust standard errors clustered at the city × technology level in parenthesis. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

In Table 1.4, we assess heterogeneity regarding binary characteristics by estimating the elasticity separately for subgroups. Specifications one and two distinguish between small and large teams, where projects with at least 5 team members are considered large. The estimated elasticity is almost twice as high for large team projects, indicating that projects with more

team members tend to benefit more from the wider local community. This is in line with evidence suggesting that sourcing knowledge from community networks is facilitated by larger team size (Lima et al., 2014). Specifications three and four confirm this notion by contrasting commits to distributed and fully co-located teams. Results show that the productivity increase within fully co-located teams is significantly larger, also pointing towards knowledge spillovers compounding in local teams. Furthermore, we use information on project ownership to separate full-fledged collaborative coding projects from single-person projects that might not require following guidelines with clear expectations on how a contribution should look like (Elazhary et al., 2019). We do so by separately considering commits to projects where the project owner is a different or the focal user in columns five and six. Results show that agglomeration benefits occur almost exclusively in projects owned by other users. This strongly points towards productivity gains in meaningful coding projects with a certain contribution standard.

Table 1.4: Heterogeneity (binary)

Dep. var.: Commits [log]	team size		geography		ownership	
	(1) small	(2) large	(3) distributed	(4) co-located	(5) others	(6) own
Cluster size [log]	0.1706	0.3243**	0.2736**	0.2932	0.3061**	0.0669
	(0.1217)	(0.1599)	(0.1303)	(0.2504)	(0.1494)	(0.1417)
Full set of FE	Yes	Yes	Yes	Yes	Yes	Yes
Users	21,116	16,061	19,295	21,098	20,644	20,917
Adjusted R^2	0.317	0.401	0.359	0.299	0.324	0.314
Observations	2,118,134	409,362	830,118	168,362	1,423,404	1,104,092

Notes: Robust standard errors clustered at the city × technology level in parenthesis. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

1.4.3 Endogeneity

Although our baseline fixed effects specification already precludes numerous endogeneity concerns, sorting of users with an expected future productivity increase or simultaneous unobserved time-varying productivity shocks on the city × technology level are remaining threats to identification. We address these concerns by estimating the instrumental variable model in Equation 1.3.

The instrumental variable approach in Equation 1.3 addresses potential simultaneity of cluster size changes and unobserved productivity shocks at the city × technology level. The first-stage results in Table 1.5 show that cluster size changes elsewhere are a strong instrument for local cluster size changes as indicated by an F-test of 1,480 in our preferred specification. The negative sign indicates cluster size growth outside the local cluster is associated with a decrease in the share of users in that cluster locally. Using only this plausibly exogenous variation in cluster size triggered by changes in cluster size elsewhere, the second-stage results present an estimate of the elasticity of productivity with respect to cluster size that is unaffected by potential technology-specific local simultaneity. Given the sample differences, the preferred specification in column 4 yields a significant and comparable effect size to our baseline results and suggests simultaneity does not drive our results.

Table 1.5: 2SLS estimates

Dep. var.: Δln(commit)	(1)	(2)	(3)	(4)
First Stage	-0.00001***	-0.00001***	-0.00001***	-0.00001***
	(0.00000)	(0.00000)	(0.00000)	(0.00000)
Δln(cluster size)	0.20336	0.29913***	0.29436***	0.19829**
	(0.19268)	(0.08786)	(0.08690)	(0.09711)
Fixed effects				
Time	Yes	Yes	Yes	Yes
City	Yes	Yes	Yes	Yes
User		Yes	Yes	Yes
Language			Yes	Yes
Language × time				Yes
Users	18,302	18,302	18,302	18,302
Observations	500,665	500,665	500,665	500,665
F-test (1st stage)	466.53	1,317.96	1,336.73	1,479.94

Notes: Robust standard errors clustered at the city × technology level in parenthesis. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

We further assess the plausibility of endogeneity arising from sorting on expected future productivity growth by investigating the dynamics of productivity changes. Sorting of users with future productivity growth independent of their location into larger clusters is unlikely to be tightly connected to the exact timing of changes in cluster size due to movers and entry

or exit into technologies of local users. In contrast, observing a strong contemporaneous reaction of productivity to cluster size growth would support agglomeration effects as driver of productivity growth. We estimate the contemporaneous effect in a three-period model with a lead, the contemporaneous period, and a lag in Table A.9. Results suggest a contemporaneous effect with a magnitude comparable to our main effect. In the preferred specification, we find a significant contemporaneous productivity increase of 0.2676 and insignificant reaction of productivity with a point estimate close to zero before. This mitigates potential concerns regarding sorting on unobserved future productivity shocks.

1.4.4 Robustness

We assess the robustness of our results via additional checks. Our main specification uses user density to measure cluster size, i.e., the share of users in a technology located in a given city. We generally prefer user density as it supports the notion of communities clustering geographically and users benefiting from being in such a hub. As some models rely on absolute cluster size, we test for differences in such a specification in Table A.4. Results vary only marginally, with an elasticity of 0.2777 in the specification with the full set of fixed effects. The distribution of cluster sizes using both measures depicted in Figure A.1 is similar, as well.

Our preferred non-parametric estimation for functional form assessment in Figure 1.2 uses 18 bins to elicit effect non-linearity with respect to cluster size. In Figure A.3, we extend the number of fixed effects used and use a smaller number of bins that are IMSE-optimally selected according to Cattaneo et al. (2021) for better representation of confidence bands. We generally observe a similar pattern across all binscatter representations with a slight S-shape of productivity with respect to cluster size. As the higher number of bins teases out the S-shape more clearly due to more narrowly spaced point estimates, we opt for this representation as our preferred specification.

On the GitHub platform, most registered users are inactive in most snapshots. Thus, we impose an activity requirement on our sample to study our population of interest, i.e., software engineers. In our main model, we require public activity in each of the ten time intervals in our sample to extract users with meaningful involvement in software engineering on the platform. Note that this improves upon the existing literature that is mainly focused on the top contributors (Vidoni, 2022). Nevertheless, in Table A.3 we relax this activity requirement to demonstrate our results generalize to broader samples. As productivity changes become less tractable for only occasional contributors, the estimated effect gets slightly smaller when reducing the minimum number of time intervals with activity. Still, the effect size

is generally similar in magnitude. Note that we capture activity in public projects. As a consequence, overall activity of users is likely higher since it also includes contributions in private repositories. Reassuringly, however, Goldbeck (2023) validates public contributions to be representative of overall regional software developer activity.

A potential concern with respect to our productivity measure would be automatic activity, e.g., by bots. Non-human activity due to bots is typically observed at high-frequency. We, therefore, exclude the top percentile of users by number of commits in Table A.6. Note that this approach risks losing the most active software engineers on the platform and therefore potentially underestimates our effect. Additionally, we estimate our baseline model excluding projects with more than 40 users or 100 commits in Table A.5 in order to ensure our results are not driven by large projects only. Similarly, in Table A.7 we present a specification without the ten largest cities. We find results comparable to our baseline specification across all these specifications, which indicates a broad-based effect that is not driven by automated contributions. An alternative measure for quality on the platform are forks, i.e., copies of repositories into other repositories. Like stars, this indicates use-value and community interest. Table A.8 reports the results, which show an even larger effect than for stars.

1.5 Conclusion

Software is ubiquitous, and understanding the economics of its production by knowledge workers is crucial. Yet, widely-used patent data has significant blind spots in software innovation that prevent comprehensive studies of this important sector. We introduce a novel measure of individual software engineer productivity based on granular data from the largest online code repository platform to overcome this challenge. We use our measure to show that higher agglomeration effects compared to other industries can explain the strong geographic concentration in the industry despite its high degree of digitization and, therefore, remote-work capability. Specifically, we estimate individual productivity increases by 2.8 percent for a ten percent increase in cluster size.

Our results have important policy and managerial implications. Most importantly, policymakers, firms, and workers should incorporate the significant effects of localized knowledge spillovers in software engineering into their decision making. The sizable heterogeneity in agglomeration effects on knowledge worker productivity has strong implications for regional policy. Results show effects are largest for cities hosting above 0.67% but below 13.5% of a technology-specific community. Subsidizing new establishments

such as Amazon's HQ2 could thus be a more beneficial strategy for regions within that range. For smaller cities, specialization in niche sub-fields where it is easier to attract a critical share of the community could be a more viable path. On the other end of the spectrum, the largest cities with cluster sizes above 13.5% reap smaller benefits from further community growth and might be better off with regional policies that deepen knowledge exchange between existing knowledge workers.

Firms that are too small to significantly affect cluster size themselves can benefit from knowledge spillovers from larger local communities, which is relevant for location decisions such as opening new or expanding existing establishments. At the same time, our findings suggest that firms may be able to avoid the very largest tech hubs – along with their high labor and real estate costs – while sacrificing little in terms of knowledge spillovers. Our results imply significant spillovers to individual productivity from the wider community that are higher for open innovation. A limitation of our data is that we do not observe activity in private projects and therefore are unable to assess spillovers within organizations if contributions are not made public. Further, the agglomeration effect is confined to specific sub-fields of software engineering suggesting that defining relevant peer groups is crucial for assessments of potential productivity benefits from agglomeration. For workers, our results highlight the importance of the location decision for individual productivity in this fast-moving field and the benefits arising from the local community.

2 Career Concerns as Public Good: The Role of Signaling for Open-Source Software Development

Much of today's software relies on programming code shared openly online. Yet, it is unclear why volunteer developers contribute to open-source software (OSS), a public good. We study OSS contributions of some 22,900 developers worldwide on the largest online code repository platform, *GitHub*, and find evidence in favor of career concerns as a motivating factor to contribute. Our difference-in-differences model leverages time differences in incentives for labor market signaling across users to causally identify OSS activity driven by career concerns. We observe OSS activity of users who move for a job to be elevated by about 12% in the job search period compared to users who relocate for other reasons. This increase is mainly driven by contributions to projects that increase external visibility of existing works, are written in programming languages that are highly valued in the labor market, but have a lower direct use-value for the community. A sizable extensive margin shows signaling incentives motivate first-time OSS contributions. Our findings suggest that signaling incentives on private labor markets have sizable positive externalities through public good creation in open-source communities, but these contributions are targeted less to community needs and more to their signal value.[1]

Keywords: software; knowledge work; digital platforms; signaling; open source; job search

JEL-No: L17; L86; H40; J24; J30

[1] This chapter is based on joint work with Moritz Goldbeck. Versions of this chapter have been published as ifo Working Paper No. 405 and CRC Discussion Paper 453. We thank Jean-Victor Alipour, Florian Englmaier, Thomas Fackler, Oliver Falck, Manuel Hoffmann, and Muhammed Yildirim as well as seminar participants at ifo Institute for valuable comments and suggestions. Further, we thank Raunak Mehrotra for excellent research assistance and gratefully acknowledge public funding through DFG grant number 280092119.

2.1 Introduction

Today's digital economy relies heavily on open-source software (OSS) (Lifshitz-Assaf and Nagle, 2021; Hoffmann et al., 2024). While the role of patents in IT decreases (see, e.g., Acikalin et al., 2022), OSS has long become an important mode of software production (Osterloh and Rota, 2007) with a 2019 investment equivalent of about USD 37 billion in the US alone (Korkmaz et al., 2024). Numerous modern products and services are built using OSS, including electronic devices, web applications, and AI algorithms. Estimates for 2022 suggest 96% of software codebases contain OSS (Synopsys, 2023). Yet, OSS is often created by a decentralized community of volunteer developers (Nagle, 2022). Because OSS is both non-rival in consumption and non-excludable due to open-source licensing (Lerner and Tirole, 2005b), OSS is a public good. This model of open community-based software development has always been "startling" to economists (Lerner and Tirole, 2002) as the motivation of individual contributors to exert private effort in order to create an openly available public good is hard to rationalize.

One potential rationale behind private contributions to OSS is it allows developers to signal valuable information and communication technology (ICT) skills to potential employers (Lerner and Tirole, 2002) since individual contributions are directly and transparently observable on online OSS platforms. Generally, ICT abilities are highly valued skills in the labor market (Bresnahan et al., 2002; Draca et al., 2007) that yield significant returns (Falck et al., 2021). At the same time, high skill obsolescence (Deming and Noray, 2020b) and the inability of formal education to certify job-relevant technical skills (Marlow and Dabbish, 2013; Fuller et al., 2022) lead to information asymmetries that make it difficult for employers to assess individuals' ability. Publicly visible OSS contributions could represent a valuable signal to potential employers (Long, 2009; Marlow and Dabbish, 2013) with respect to the most job-relevant skill in software development: practical programming ability (see, e.g., Surakka, 2007; Wagner and Ruhe, 2018). This implies that, besides private benefits from learning and improved labor market outcomes, signaling activity driven by developers' career concerns might directly generate considerable positive externalities (Leppämäki and Mustonen, 2009) in the form of a public good, open source software.

In this paper, we investigate whether career concerns are indeed a driver of OSS development. To this end, we exploit variation in individual incentives to signal over time. Specifically, because signaling is costly and its value quickly depreciates, individuals economize on the signal and dynamically allocate OSS activity to times of immediate job search in order to signal skill to employers. This allows us to test for the presence of the signaling motive empirically

by studying OSS contributions of software developers who move for a job on the largest online code repository platform, *GitHub*. We focus on movers as job changes are often associated with moving (Amior, 2019; Balgova, 2020), especially for the high-skilled (von Proff et al., 2017; Haussen and Uebelmesser, 2018), which might confound our results when not explicitly considered in the empirical model. We, therefore, compare developers relocating for a new job to developers moving to a new location for other reasons in a difference-in-differences design. We argue that while job movers face elevated signaling incentives driven by immediate career concerns in the period prior to moving, the "job search period," these incentives are absent for developers who relocate for other reasons. Consequently, OSS activity attributable to signaling is captured by the difference in OSS contributions between job movers and other movers during relative to outside of the job search period.

Our data comprises all *GitHub* users with changing location information from ten snapshots of the *GHTorrent* database dated between 2015 and 2021. Due to this sample selection approach, we are able to capture typical volunteer developers who occasionally contribute to OSS (Vidoni, 2022). In total, our sample contains some 22,900 movers worldwide, of which around a third simultaneously change their job. Besides location and organizational affiliation, we observe in detail each user's public activity on the platform such as the monthly number of commits in open-source projects, their collaborators, or quality metrics such as stars, followers, and forks. This allows us to investigate not only whether career concerns drive OSS activity, but also if there are systematic shifts in OSS activity when motivated by signaling incentives with respect to the types of projects, usefulness to the community and quality, or user groups.

We find significantly elevated OSS activity by about 12% of job movers in the job search period compared to developers moving for other reasons. Assuming an average job tenure of three years applies to OSS developers and constant (base) activity levels over time, this translates to 4.9% of overall OSS activity being caused by signaling incentives during job transitions. Within the job search period effect size steadily decreases, consistent with stronger incentives during the application preparation phase. Notably, our analysis points to the importance of the extensive margin, inducing first-time contribution to OSS. In general, the effect derives from a broad base of job movers rather than a specific group. But we observe a larger effect for users relocating internationally and for users moving to academia. The signaling effect tends to be smaller for users with new jobs at large firms and especially at big tech companies, where we do not see a signaling effect. Multiple classifications of projects based on programming languages indicate that the effect is mainly driven by contributions to web development and data engineering projects, and to projects using top-paying programming languages.

However, signaling projects are starred and forked less by other users, pointing to a lower direct use-value to the OSS community. In general, our results are in line with career concerns motivating significantly increased OSS contributions during the job search period as we observe activity shifts to projects that increase the visibility of existing works or necessitate skills highly valued in the labor market. Additional analyses with respect to model choice and other empirical decisions emphasize the robustness and conservativeness of our preferred specification.

This work makes several contributions. In contrast to most existing studies that follow a stated preferences approach, we deploy a quasi-experimental framework and are therefore able to achieve high internal validity of our results and causally link career concerns to OSS activity under reasonable assumptions. In addition, we improve on external validity by selecting our sample from the near-universe of OSS activity on *GitHub*, the by far largest online code repository platform. Therefore our data includes not only the most active OSS developers but also volunteer developers who only occasionally contribute to OSS, but together make up the vast majority of OSS contributors. We also add to the labor market literature by showing that employees indeed signal ability through OSS activity, which groups are especially likely to signal, and how this motivation impacts the type of projects users engage in. Importantly, we contribute to the literature on private public good provision by pointing out that there are significant positive externalities from private career concerns while, at the same time, the direction of public good creation changes when labor market considerations are prominent.

Our findings have multiple managerial and policy implications. Notably, they highlight an important but neglected channel of public good creation: the positive externalities from individual labor market signaling incentives. We show that these externalities are significant with respect to overall OSS activity and signaling incentives systematically induce first-time contributions of users previously inactive in the OSS community. To increase public good creation and platform growth, both management and policymakers should take these positive externalities of career concerns into account in platform design and public policy. For example, platform design that considers the signaling needs of their users explicitly could further boost growth at the extensive (user) and intensive (activity) margin. At the same time, decision-makers should be aware of the shift in focus towards labor market requirements and away from direct use-value for the OSS community in signaling projects. For labor market and education policy as well as HR professionals, our findings point out the continued shift away from formal (public) skill certification and emphasize greater importance of more fluid and practical skill signals that directly showcase work product. Lastly, innovation policy aiming

to foster public good creation in the knowledge economy may consider maximizing positive externalities from signaling incentives, e.g. via adopting open science policies that create synergies between funded and signaling activities.

The remainder of this paper is organized as follows. First, we discuss related literature in Section 2.2. Section 2.3 introduces the data. In Section 2.4, we present the empirical identification strategy. Results are provided in Section 2.5 and Section 2.6 concludes with a discussion.

2.2 Related literature

Economics of Open Source. This project is related to the economics of open source. Literature in this area examines the distinct innovation model of OSS, which is based on volunteer contributions of often decentralized teams and is governed by open licenses (Lerner and Tirole, 2005b; Osterloh and Rota, 2007). As such, open innovation contrasts sharply with traditional ("closed") innovation featuring exclusive intellectual property rights (Lerner and Tirole, 2002, 2005a). These unique properties, combined with the lasting success of OSS and the growing importance of software in general, spurred dedicated research (see, e.g., von Krogh et al., 2003; Lifshitz-Assaf and Nagle, 2021). Compared to volunteer developers, firms are of less significance as in traditional innovation models, but increasingly incorporate OSS in their business models (Lee and Cole, 2003; Butler et al., 2019), for example to increase visibility (Conti et al., 2021) or learn from community feedback (Nagle, 2018). OSS research addresses a wide variety of topics such as productivity effects (Nagle, 2019), team organization (Puranam et al., 2014; Raveendran et al., 2022), geography (Wachs et al., 2022), or innovation and entrepreneurship (Bitzer and Schröder, 2007; Colombo et al., 2014; Wen et al., 2016; Wright et al., 2023).

Naturally, a large literature revolves around the reasons volunteer developers contribute to OSS and broadly distinguishes between internal factors and external rewards (Hars and Ou, 2002; Krishnamurthy, 2006). von Krogh et al. (2012) cluster motivations into intrinsic (ideology, altruism, kinship, fun), internalized extrinsic (reputation, reciprocity, learning, own use), and extrinsic (career, pay). Empirically, researchers elicit the prevalence of different motivations to contribute predominantly through surveys. These works generally find evidence for mixed motivation, but internal factors tend to be most important (von Krogh et al., 2012). For example, a survey of *Linux* contributors by Hertel et al. (2003) emphasizes the role of group belonging, identification, and a feeling of indispensability while acknowledging own use-value

as another motivator. Likewise, Stewart and Gosain (2006) show that *SourceForge* contributors are more involved because of shared values. Hars and Ou (2002) conduct an e-mail survey among OSS developers, who state that self-determination, learning, and reputation are the main reasons to contribute. Community surveys by Lakhani and von Hippel (2003) and Nagle et al. (2020) explicitly stress that external and monetary factors are far less important than intrinsic motivation from creativity and intellectual stimulus. In a survey by Hann et al. (2004), *Apache* developers state own use-value, recreational value, and career impact most often as motivating factors. Gerosa et al. (2021) elicit from survey responses that reputation-building as a motive became more important in recent years, and that learning and career incentives are especially relevant for novice contributors. Shah (2006) finds motivational dynamics, where initial participation is typically driven by own use-value whereas maintainers of OSS are often intrinsically motivated. Roberts et al. (2006) note that motivations interact with each other in complex ways as, e.g., being paid increases status but at the same time is associated with a lower use-value. Indeed, Krishnamurthy et al. (2014) show that monetary reward can crowd out other motivations. Investigating behavioral changes of developer contribution after being sponsored, both Conti et al. (2023) and Wang et al. (2022) find evidence in favor of a net-positive effect of monetary incentives on activity. Projects with fast feedback and a non-commercial nature are associated with a higher probability of receiving contributions (Smirnova et al., 2022).

Our study adds to this literature in that it broadens the scope in terms of contributors being studied. While existing work mainly focuses on the most active OSS developers, often partly paid for their work, we investigate typical users on the platform, i.e., volunteer developers who sporadically contribute to open-source projects (Vidoni, 2022). The importance of economic benefits and motives for this group of OSS contributors is neglected in the literature, and this study is among the first to study the role of career concerns in a causal identification framework. As such, it sharply contrasts with the prevailing methodological approaches used in existing research on this topic. These works are largely based on surveys, which feature the important caveat of only eliciting stated preferences as opposed to the revealed-preference approach embodied in our causal framework. As a result, we are able to make quantifiable causal claims on the importance of career concerns motive for typical volunteer OSS developers under reasonable assumptions. Our findings suggest a sizable portion of OSS activity is driven by career concerns, and that motivations dynamically change over time, which in turn alters the content of contributions.

Labor market signaling. This article focuses on one specific motivating factor to contribute

to OSS, career concerns, and therefore adds to the vast literature on signaling originating from Spence (1973). Subsequent theoretical models explicitly relate career concerns to signaling via observable effort (Chevalier and Ellison, 1999; Holmström, 1999), even when beliefs on ability are precise (Miklós-Thal and Ullrich, 2015). While basic signaling models yield separation of skill types even if signaling has no real effects, Leppämäki and Mustonen (2009) provide a model where signaling activity generates (positive) product market externalities. Empirically, Miklós-Thal and Ullrich (2016) test the career concerns hypothesis in soccer and find confirmatory results for marginal individuals. Pallais (2014) shows detailed public performance records on the online marketplace *oDesk* improved workers' subsequent employment outcomes, especially for the inexperienced. Also on an online platform for contract labor, Agrawal et al. (2016) find standardized and verifiable information important for developing-country candidates' employment probability. For software developers, Xu et al. (2020) find career concerns increase reputation-generating activity in an online community forum. Experimental evidence by Piopiunik et al. (2020) reveals basic IT skills signals in CVs on the broader white-collar labor market significantly increase the probability of receiving a job interview invitation. Apart from this causal evidence, surveys show reputation-building, signaling, and career concerns are important motivations for developers to contribute to OSS (e.g., Hars and Ou, 2002; Hann et al., 2004; Marlow and Dabbish, 2013; Gerosa et al., 2021). Similarly, employers state they regard OSS contribution as a credible and valuable signal. For example, in a survey, Long (2009) finds tech companies value OSS experience of applicants. More specifically, Marlow and Dabbish (2013) surveys recruiting managers who state *GitHub* activity is used in hiring as a signal for technical abilities and motivation, and is regarded as a stronger signal than the applicants' resume with respect to these areas. A survey among developers by Hakim Orman (2008) shows OSS activity and traditional education are seen as complements and not substitutes. However, Bitzer and Geishecker (2010) finds formally educated individuals are underrepresented in the OSS community. For developing-country candidates, Hann et al. (2013) claim that valuable OSS activity is an effective and credible signal as it is associated with significant wage premiums for *Apache* project participants. Huang and Zhang (2016) associate improved outside options from OSS signaling with job-hopping, but also acknowledge retaining effects from learning.

The contribution of this research to this strand of literature is twofold. First, in contrast to most work in this area, we follow a quasi-experimental approach using observational data from the near-universe of OSS developers. This allows us to make causal claims under reasonable assumptions leading to a comparably high degree of internal validity. Furthermore, because we are able to study a large and diverse group of OSS contributors and do not limit our scope to

the most active users, the results also feature a higher level of external validity in comparison to the fairly specific and small groups typically studied in existing works thus far. Our second contribution, which received limited attention, is asking to what degree signaling activity is wasteful or productive from a content perspective. Our empirical evidence suggests lower but still positive direct use-value for the community of signaling activity, and therefore adds an empirical perspective to the notion of positive externalities of signaling, which has only been examined theoretically to date (Leppämäki and Mustonen, 2009).

Public good provision. The paper is also connected to the broader literature on private public good provision. In contrast to traditional innovation models that rely on private property, open innovation models like OSS largely depend on voluntary contributions by individual developers and thus can be framed as private public good provision (Lerner and Tirole, 2002). Traditional theory emphasizes group size as the main factor influencing the provision of the good (e.g., Chamberlin, 1974; Bliss and Nalebuff, 1984; Palfrey and Rosenthal, 1984; Bergstrom et al., 1986; Hendricks et al., 1988; Bilodeau and Slivinski, 1996). Explicitly modeling intrinsic motivation, Bitzer et al. (2007) show provision is more likely maintained when OSS programmers value gift benefits and the intellectual challenge, have a long time horizon (i.e., are younger), are patient, face low development cost, and derive a high own use-value. In a model of OSS development, Johnson (2002) shows how own use-value considerations drive the direction of software production. Incorporating own use-value considerations and provision costs, Myatt and Wallace (2002) model a public good provision game and show multiple equilibria can arise. Ignoring intrinsic motives, Bitzer and Schröder (2005) derive joining and exiting dynamics from signaling in a model of repeated contribution. Regarding the licensing regime, Fershtman and Gandal (2004) show that contributions are higher when OSS licensing is less restrictive. Athey and Ellison (2014) model a world where OSS projects can be successful when developers are motivated by reciprocal altruism if customer support is not needed. Zeitlyn (2003) emphasizes the gift economies motivation. Empirically, O'Neil et al. (2022) define contribution territories for firms and individuals in the space of possible innovation to rationalize why certain areas are neglected. Recently, del Rio-Chanona et al. (2023) find public good generation on *StackOverflow* is impacted negatively by large language models, a substitute to online forums.

Our empirical results are important to inform on the applicability of theoretical models depending on their presumptions. Our findings emphasize that external motives are relevant and that considering the dynamic evolution of motivation is important. At the same time, external motives such as career concerns likely do not explain OSS activity entirely. Hence,

theoretical models that aim to capture OSS contribution comprehensively should consider modeling multi-dimensional motivations to contribute that include both internal and external motivations and incorporate their dynamic evolution. In general, our study emphasizes the importance of labor market incentives of high-skilled professionals for the private provision of an important public good in the knowledge economy, which likely features considerable positive spillovers both on the private market and in the form of public follow-on innovation in the OSS community.

2.3 Data

We study software developers on *GitHub*, the by far largest online code repository platform. *GitHub* was founded in 2008, reached 10 million users by 2015, and in 2021 reported 73 million users worldwide (Startlin, 2016; GitHub, 2021). Around a fifth of all code contributions on the platform are made to public repositories, i.e., open-source projects (GitHub, 2021). Repositories are maintained using the integrated version control software *git*. Importantly, the nature of the *git* version control system allows us to track each user's contribution to open-source projects over time as it records and timestamps all activity in public repositories. *GitHub* provides access to public user profiles and repositories via API. Data analyzed in this paper originates from *GHTorrent*, a research project by Gousios (2013) that mirrors the data publicly available via the *GitHub* API and generates a queryable relational database in irregular time intervals.[2] The resulting snapshots contain data from public user profiles and repositories as well as a detailed activity stream capturing all contributions to and events in open-source repositories. This paper relies on ten *GHTorrent* snapshots dated between 09/2015 and 03/2021.[3]

On their *GitHub* profile, users can indicate their location. This self-reported indication is voluntary and is neither verified nor restricted to real-world places by *GitHub*. Goldbeck (2023) finds no systematic bias in the location information provided on the platform, even though only a fraction of users indicates their location. We assign users to cities via exact matching to city names in the *World Cities Database*.[4] Users can also provide an indication of their organizational affiliation, which we use to elicit job changes. Location and organization

[2] *GHTorrent* data contains potentially sensitive personal information. Information considered sensitive (e.g., e-mail address or user name) has been de-identified (i.e., recoded as numeric identifiers) by data center staff prior to data analysis by the authors. Data from the *GHTorrent* project is publicly available at `ghtorrent.org`.

[3] Snapshots are dated 2015/09/25, 2016/01/08, 2016/06/01, 2017/01/19, 2017/06/01, 2018/01/01, 2018/11/01, 2019/06/01, 2020/07/17, and 2021/03/06.

[4] A fraction of 0.25% of users (total: 58) are not matched to a city in the database but rather a state or a country. We do not geocode cities or states with a name that exists multiple times.

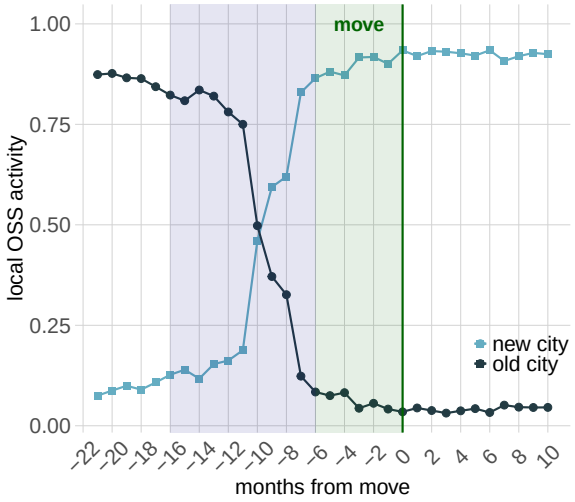
information is observed only on snapshot dates – i.e., roughly every six months – while user activity is timestamped. We aggregate users' timestamped activity to monthly data to obtain a panel structure. Since the data is highly skewed and most users are inactive (see, e.g., Vidoni, 2022; Luca, 2015), we restrict our sample to users with an observed minimum activity of three months with non-zero commits.

Movers. From the data, we select movers, i.e., users who change their city-level location once in the observation period. Our empirical strategy elicits signaling activity from time-varying incentives around a job change. When people change jobs, they often simultaneously move (Amior, 2019; Balgova, 2020), which is especially the case among high-skilled professionals (Greenwood, 1973, 1975; Machin et al., 2012; Amior, 2015). To attain a meaningful comparison and get rid of any confounding factors associated with moving we, therefore, compare users who move for a job to users who move for other reasons. We infer the reason for moving from changes in the organizational affiliation of users. Whenever there is no affiliation change around the move date we regard a user as moving for other reasons. Conversely, if a new affiliation appears around the move date we consider a user as job mover. To implement this, we extract users' move (and job change) dates from the data.

We infer the move date from user-level location information as the month of the first snapshot with a new city indication. There is some uncertainty regarding the actual move date for two main reasons. First, users manually enter (new) location information data on the platform themselves and do this not necessarily exactly at the time of moving. On the one hand, users might be busy during the time period of moving and enter their move late. On the other hand, it might be beneficial to communicate the future location early, maybe even before actually moving, to let peers know about their relevant location as soon as possible. We empirically investigate the plausibility of the move dates attained through the snapshots by looking at team member locations in the projects a user actively contributes to each month. To this end, we assign locations to projects depending on other members' locations. Specifically, we define a user's project as localized in a particular city if the current location of more than 60% of the team members is in that city. This is only possible for a subset of projects as few members share their location and team members can be distributed. Nevertheless, it allows us to get an impression of changes in the spatial collaboration pattern of users in our sample.

Figure 2.1 plots the share of users' activity in localized collaborative projects by origin and destination city. The dark blue line represents a users' activity share in projects where team members are predominantly located in her origin city while the light blue line represents

Figure 2.1: User collaboration around relocation date



Notes: Graph shows in-sample users' commits to new- and old-city repositories as a share in users' total commits to repositories with an assigned location. Location is assigned to repositories for which at least 60% of the team members indicate a common city as current location. *Sources:* GHTorrent, own calculations.

activity in projects with team members predominantly located in the destination city. The graph shows a clear pattern. Most localized activity is in old city projects up to ten months prior to the estimated move date. This starts to reverse afterward and most localized activity is measured in destination city projects from six months prior to moving until the end of the observation period. It is plausible that users start collaborating with teams in their destination city prior to moving and activity in old-city projects fades out. Importantly, this graph shows user-provided locations systematically and meaningfully relate to collaboration patterns, which validates our measurement of moving. Similarly to the move date, we elicit job changes from users' affiliation indication as the first month the new city location is observed in the data.

Summary statistics. The resulting sample of users comprises 22,896 movers, of which 7,211 (32%) simultaneously change their job.[5] Naturally, since most registered users are inactive, this sample is very different compared to the universe of users in the data and comprises more active users, which is confirmed by the summary statistics in Table B.1. More interestingly, Table 2.1 provides an overview of our sample and compares job movers and other movers. In general, job movers and movers are comparable in terms of activity, collaboration, and

[5] Figure B.2 reports the moves by data snapshot and shows a similar distribution for job movers and other movers.

quality metrics. At the same time, there are also some differences between the groups. The median mover has five followers, contributes around 170 commits to open-source projects in the observation period, and has 15 projects with on average 2 to 3 team members. Job movers contribute a bit less to team projects and the average team size is smaller compared to other movers, and their team projects also receive fewer stars and forks. Projects in our sample are very diverse both in terms of programming languages (cf. Table B.7) and topics covered and range from web development to data engineering (cf. Figure B.5).

Table 2.1: Summary statistics

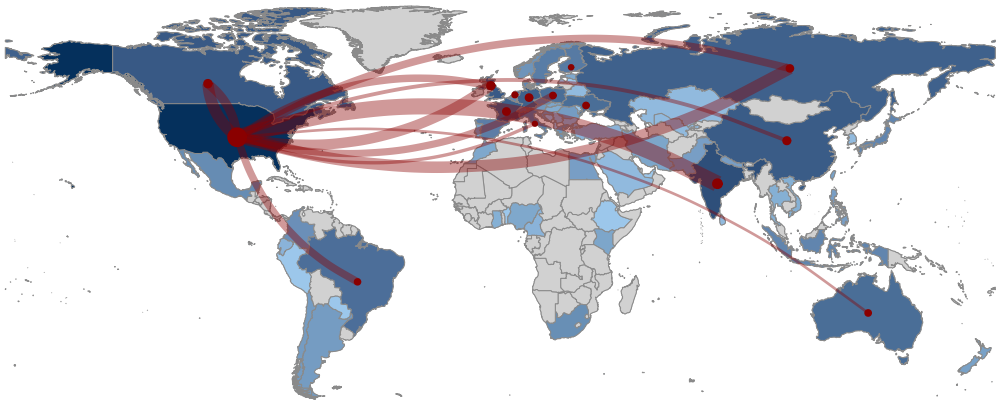
Median	**Movers**			
	job	other	Δ	%Δ
Activity				
Commits	163	188	−25	13.3%
commits single projects	*72*	*76*	*−4*	*5.3%*
commits team projects	*59*	*80*	*−21*	*26.3%*
Experience	37	42	−5	11.9%
Collaboration				
Projects	14	16	−2	12.5%
single projects	*9*	*9*	*0*	*0.0%*
team projects	*5*	*6*	*−1*	*16.7%*
Project members	2.21	2.82	−0.61	21.6%
Quality				
Followers	5	5	0	0.0%
Stars	1.10	1.88	−0.78	41.5%
stars single projects	*0.09*	*0.12*	*−0.03*	*25.0%*
Forks	0.62	1.11	−0.49	44.1%
forks single projects	*0*	*0*	*0*	*0.0%*

Notes: Experience is measured as tenure on the platform in months since the first commit at the move date. Column Δ reports the absolute difference in median between job movers and other movers. Column %Δ sets this difference in relation to other movers' median. *Sources:* GHTorrent, own calculations.

The differences between job movers and other movers regarding team project behavior is one reason why we look at single projects, i.e., projects in which only the focal user is active. But there is a more important reason derived from theoretical considerations and a practitioner's perspective with respect to labor market signaling through OSS activity. Not all contributions

to OSS communities constitute equally valuable signals of ability and thus generate reputation (Marlow and Dabbish, 2013; Xu et al., 2020). In particular, for potential employers, it is difficult and time-consuming to assess individual contributions to collaborative projects even if transparently available (Tubino et al., 2020). In contrast, single-authored projects can be assigned entirely to individual users. At the same time, quality metrics such as stars and forks make assessment effortless and enable non-software developers like HR professionals to perform such assessments. Consequently, using OSS activity in single projects as the main outcome metric ensures a close practical and theoretical relation to actual signaling potential.

Figure 2.2: Domestic and international user relocations



Notes: Blue country coloring shows the number of domestic movers after logarithmic transformation. There are 73 countries with domestic movers; grey indicates no domestic movers. The size of the red country centroids indicates the number of international moves a country is involved in. 14 countries are associated with international relocations. Red arcs represent edges in the directed country mover network, i.e., the number of international relocations from one country to another, and are scaled logarithmically. For clarity, only edges above 75 are shown. *Sources:* GHTorrent, own calculations.

Although we look at users moving worldwide, 71% are relocations to another city within the country. About 29% of relocations are international, and 19% of movers or two-thirds of international movers even move inter-continentally. This mirrors the fact that software developers are disproportionally mobile internationally (see, e.g., Solimano, 2006; D'Mello and Sahay, 2007; Adrian et al., 2017). The average relocation distance is 5,324km and there are no significant differences in these statistics between job movers and movers relocating for other reasons (cf. Figure B.1). Figure 2.2 maps the observed migration flows in our data in more detail. Countries are colored in darker blue the higher the number of domestic relocations and the width of the network edges represents the number of international relocations. The dominance of the USA as the central hub both in terms of domestic moves and as a receiving country is clearly visible even on the logarithmic scale. Domestic moves are observed most frequently in the USA (63.5%), India (7.5%), and the United Kingdom (3.9 %). Table B.4 shows

the ten countries with the most domestic moves, which account for over 90% of domestic moves and 65% of all relocations. The most important origin countries are shown in Table B.5. Table B.3 reports the ten largest origin and destination cities, which are predominantly the world's big software industry hubs, e.g., San Francisco and New York. Notably, for international relocations, we observe that users tend to move to richer countries as indicated by per capita GDP increasing on average by USD 9,780 (Figure B.3), with no systematic differences between job movers and other movers.

Users are affiliated with a diverse range of organizations. Most firms in the data are small, but the distribution is highly skewed to the right (Figure B.4). On average, each organization has four affiliated users and 23 users are affiliated with the median organization.[6] Table B.2 reports organizational affiliations and job transitions by organization type. As a consequence of the skewness, about 29% of users are affiliated with the 100 largest firms and 7.2% with the big technology firms (i.e., Google, Apple, Meta, Amazon, Microsoft; GAMAMs). Job transitions point out net movements towards larger, and especially big tech, firms and away from academic and small-firm affiliations. This is confirmed by Table B.6 depicting top origin and destination affiliations. While top origin affiliations include mostly students, universities, and freelancers the biggest destination shares almost exclusively are held by large software companies such as the GAMAMs or Red Hat, IBM, and LinkedIn.

2.4 Empirical strategy

The key idea behind our empirical model setup is to exploit temporary differences in signaling incentives across users. Specifically, we compare the activity of users who move for a job and movers who move for other reasons. The reasoning behind this is that users who move for a job experience increased incentives to signal their ability on the platform to potential employers prior to their move during the job search period, whereas movers who relocate for other reasons do not experience this temporary increase. As already discussed above, we focus on movers since job changers typically simultaneously relocate, which is widely acknowledged in the literature (Amior, 2015; Balgova, 2020) and especially the case for high-skilled professionals (see, e.g., Kodrzycki, 2001; Venhorst et al., 2011; Haapanen and Tervo, 2012; Ciriaci, 2014; Abreu et al., 2015; von Proff et al., 2017; Haussen and Uebelmesser, 2018). Thus, comparing movers leads to improved comparability as it accounts for confounding factors associated with moving.

[6] Note that these numbers are not to be confused with the number of employees since not all employees are active OSS contributors on *GitHub* and provide their affiliation.

Figure 2.3: Adapted difference-in-differences model

From a theoretical perspective, we structure signaling incentive dynamics into three phases, where each phase is governed by a distinct incentive regime. This is illustrated in Figure 2.3. In the first phase, which we call the pre-period, an eventual mover is still working in her previous arrangement and does not actively prepare to change jobs. In this phase signaling incentives are not entirely absent and are at a normal level as there is no immediate pressure to signal skill in the labor market. In the decisive second phase, the "job search period," the job mover then actively searches for a new employer and prepares to relocate while movers who relocate for other reasons only prepare to relocate. In this phase, job movers face elevated incentives to signal skill to potential employers. Finally, there is a third phase after the move, which we call post-period, in which movers have relocated and the job mover has started to work for her new employer. Movers who relocated for other reasons are still with their old affiliation. In this phase, as job movers just started a new job, signaling incentives vanish and are likely even lower than in the pre-period and compared to other movers because job movers have to settle in to their new job environment, and the especially low signaling incentives.

As a result of these theoretical considerations, we expect elevated OSS activity of users who move for a job compared to users who move for other reasons in the job search period if career concerns are an important factor for OSS contribution. Additionally, we expect to see lower OSS activity of job movers compared to other movers in the post-period. We empirically investigate the dynamics of OSS activity by estimating the following baseline event study model:

$$y_{it} = \beta_1 + \sum_{j=\underline{T}}^{\overline{T}} \left[\beta_j \left(t_j \times \text{JobChanger}_i \right) \right] + \delta_i + \delta_{s(t)} + \delta_{a(i)t} + e_{it}, \tag{2.1}$$

where y_{it} is the number of commits of user i in relative-to-move month t to single-authored repositories ("signaling projects"). Note that the event study panel is balanced in the job search and pre-period but unbalanced in the post-period as some moves happen during the end of our observation period. The variable JobChanger_i indicates if user i moves for the job, i.e., simultaneously changes her affiliation and location. The core element is the interaction term of JobChanger_i with relative months to the moving month t_j. Coefficients of interest are

$_j$ and reveal the difference in the temporal pattern of signaling activity around the move date between users who simultaneously change their job and users who do not. To control for time-constant unobserved user characteristics relevant to their level of OSS activity, we add user fixed effects δ_i. Calendar month fixed effects $\delta_{s(t)}$ account for unobserved factors affecting all users' activity in a given month. We include experience fixed effects $\delta_{a(i)t}$ to account for differences in platform tenure across users that impact OSS activity. Standard errors are clustered at the user level.

Starting from this flexible dynamic model, we adapt the standard difference-in-differences model to estimate the average treatment effect on the treated such that three phases around the move date are considered: a pre-period, a job search period, and a post-period. The reference period is the pre-period, and the temporary treatment of increased incentives to signal using OSS activity is present only during the job search period. In the post-period, signaling incentives for job changers are lower relative to the pre-period because of diminished career concerns and the new job crowding out OSS activity. The resulting model specification is

$$y_{is} = \beta_1 + \beta_2(\text{SearchPeriod}_{s(i)} \times \text{JobChanger}_i) + \beta_3(\text{PostMove}_{s(i)} \times \text{JobChanger}_i) + \delta_i + \delta_s + \delta_{a(i)s} + e_{is}, \tag{2.2}$$

where $\text{SearchPeriod}_{s(i)}$ is one if calendar month s falls in user i's job search period prior to the move. To account for generally reduced incentives of job switchers to make OSS contributions after the move relative to users who move for other reasons, we interact an indicator for the post-move period, $\text{PostMove}_{s(i)}$, with job changer status. The coefficient of interest β_2 captures the ATT of increased signaling incentives during the job search period, i.e., differences in OSS activity between job movers and other movers in the job search period relative to the period before. Similarly, β_3 represents the average difference in OSS activity between job movers and other movers in the post-move period relative to the pre-period.

Although the inclusion of the post-period is not formally needed for identification, we consider it explicitly in our model for two reasons. First, it adds credibility to the signaling effect estimated from the difference between the pre-period and the job search period if signaling activity declines when taking up a new job, which we assume reduces immediate signaling incentives. Second, validation of parallel trends between job movers and other movers in both the pre- and post-period helps to further assess the validity of our design. And third, although not the main goal of this analysis, estimating the effect of taking a new job on OSS activity is interesting in itself. The three-period specification with the pre-period as reference

is superior to alternatives. Taking the post-period as reference neglects the crowding-out of OSS via time constraints of formal work. Combining pre- and post-period as reference attenuates this issue, but leads to potential overestimation due to the same mechanism.

Empirical results from the event study specification guide the selection of appropriate time frames for the three phases in the ATT model. In addition, a priori theoretical and empirical considerations set our expectations. In his classical framework, Blau (1994) divides the job search period into three steps. The first step is the preparation phase, where applicants prepare their application package. Then there is the actual application step in which applicants undergo the formal application process. Finally, the third step is the decision step, in which employers and applicants decide on whether to enter an employment relationship or not. Signaling activity is expected to occur predominantly in the first step, i.e. preparation (Chamberlain, 2015). Recent statistics for the US show hiring time for complex jobs such as software development averages around four months prior to applying (Firaz, 2022), and people start thinking about and preparing for job search likely much earlier. Additionally, there is some fuzziness in our measurement of the move date due to only observing locations about every six months. Therefore, we expect to see most OSS signaling activity in the preparation phase of the job search period somewhere between six and 15 months prior to our estimated move date.

Note that our model specification provides a conservative and incomplete estimation of the role of career concerns for individual OSS activity for multiple reasons. First, signaling incentives are not entirely absent in the pre-period. Career concerns are not binary and we exploit time variation in their strength rather than presence or absence. Second, our estimates are downward biased due to measurement error when some control group movers in fact move for the job, as well, but do not change their affiliation. Third, our focus on movers implies we study a group of users who face significant additional time constraints relative to users who are not relocating and therefore trade-off their time allocation between more activities, potentially leading to less time spent on signaling activity in this group. Finally, the dynamics within the job search period as well as the fact that toward the end of our signaling period, the share of users who already found a job increases biases the ATT downward. Consequently, our estimates should be interpreted as a lower bound to the importance of career concerns for OSS activity.

Our key identifying assumption is that in the absence of signaling incentives for job changers, their activity would have evolved similarly to movers not changing jobs simultaneously,

conditional on controls. Although we cannot test this assumption directly we assess it by showing parallel trends in periods when signaling incentives are absent, i.e., both the pre- and post-period. The main remaining threats to our identification strategy are factors unrelated to signaling incentives that affect the user activity of job changers in the job search period prior to the move but not the user activity of movers that do not change jobs or vice versa. One such concern could be due to potentially reduced work ethic of job movers in their old job as it comes to an end and, as a consequence, more time for side projects. However, one could also expect the old job claims more time towards the end as, e.g., projects have to be handed over. Another potential concern is an increased prevalence of learning motives or decreased opportunity cost of contributing during periods of unemployment between two jobs. This is, however, not only unlikely due to generally short unemployment spells for IT professionals; the median duration of unemployment in the US, for example, is only eight weeks.[7] It is especially unlikely given that our design focuses on movers, and relocating to another city or even country is generally time-consuming and stressful. Nevertheless, in Sections 2.5.2 and 2.5.3 we address these concerns and assess related channels by investigating the kind of OSS activity of job movers and how it differs from other movers to validate if the observed activity can likely be attributed to signaling or not.

2.5 Results

2.5.1 Main effect

Figure 2.4 plots the event study coefficients for user activity around the relocation date resulting from the model in Equation 2.1. The dynamics are consistent with signaling as a driver of OSS activity and the hypotheses derived from our theoretical considerations. In the pre-period, there are no statistically significant differences in OSS activity between users who move for a job and users who move for other reasons. Similarly, after moving we observe a lower activity level for job movers compared to other movers but the dynamic development is, again, parallel to each other. This absence of differential trends between treatment and control group users is reassuring of the validity of our empirical design as it provides confidence that our key identifying assumption holds. Importantly, during the period prior to moving, OSS activity of job movers is significantly elevated relative to other movers conditional on controlling for time, user, and experience fixed effects. We claim this increase is driven by immediate career concerns in the period of job search which incentivizes signaling activity.

[7] Statistic retrieved from BLS based on the Current Population Survey 2018: `https://www.bls.gov/web/empsit/cpseea37.htm`. Last accessed on 11/10/2023.

Figure 2.4: Event study estimates

Notes: Estimates for $t_j \times \text{JobChanger}_i$ based on Equation 2.1 with user, experience and calendar month fixed effects. The outcome is IHS-transformed commits to single-authored projects. The reference month is $t = -16$. Bars show 95% confidence intervals. Standard errors are clustered at the user level. *Sources:* GHTorrent, own calculations.

The dynamic activity pattern during the job search period is consistent with signaling behavior, too. Signaling activity is strongest at the beginning of the job search period 10 to 14 months before the move month with activity in signaling projects being elevated by up to 24.5% for job movers. The effect then declines steadily to substantially lower levels before the move date around 6-10% before returning to a permanently lower, stable, activity level from the move month onward, with estimates centering around -7 to -13%. Model (3) in Table B.13 provides estimates for each period. This pattern is in line with our theoretical considerations predicting more intense signaling in the preparation step of the job search period as users generally have an incentive to have their signal ready by the time of application which is likely earlier in the job search period. In addition, more and more users finding a job during the job search period or moving earlier than the observed move month, both leading to reduced incentives to signal.

Because of sparsity, we transform the dependent variable using the inverse hyperbolic sine transformation in order to retain zero-valued observations (Bellemare and Wichman, 2020). At the same time, this transformation approximates the natural logarithm and is commonly interpreted in a similar way (Burbidge et al., 1988; MacKinnon and Magee, 1990). As our data typically features right-skewed but low numbers of commits, we do not rescale the dependent variable prior to transformation. Estimates are generally sensitive to scaling and

as there is no overarching guideline, scaling choice is described as a data fitting problem in the econometric literature (Aihounton and Henningsen, 2021). As rescaling typically leads to larger estimates our choice with respect to dependent variable scaling is conservative (Chen and Roth, 2024).[8] The effect size of the resulting coefficient estimates thus is not only statistically highly significant but also economically sizable as we estimate between 5 and 25% higher OSS activity of job movers compared to other movers in the job search period, depending on the month relative to move date.

Table 2.2: Difference-in-differences model

IHS(single commits)	(1)	(2)	(3)
Job mover × job search	0.3621***	0.2962***	0.1646***
	(0.0137)	(0.0144)	(0.0141)
Job mover × post move	-0.2608***	-0.2208***	-0.1036***
	(0.0189)	(0.0203)	(0.0190)
User FE	×	×	×
Month FE		×	×
Experience FE			×
Adjusted R^2	0.289	0.308	0.359
Observations	1,946,413	1,946,413	1,946,413
Users	22,896	22,896	22,896

Notes: Results from estimation of Equation 2.2. experience is measured as months since the first commit at move month. Robust standard errors are clustered at the user level. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

The dynamic event study specification validated by theoretical and empirical evidence from the literature informs our definition of the job search period. We identify the period of distinctly elevated OSS activity in the 15 months prior to the month of moving as job search period. Using this definition of the job search period allows us to estimate the average treatment effect on the treated (ATT) per Equation 2.2. Table 2.2 provides the ATT estimates of our adapted three-period difference-in-differences specification. As expected, job movers' OSS activity is elevated during the job search period relative to other movers and is lower in the post period. The inclusion of calendar month and experience fixed effects considerably improves model fit as described by adjusted R^2. The coefficient(s) of interest are attenuated as a result. Our preferred specification in Model (3) estimates that job movers contribute about 11.8% more

[8] We discuss model specification in more depth in Section 2.5.3.

on average in the job search period compared to other movers.

While the ATT effect size as such is suitable in assessing the importance of signaling incentives for individuals' OSS contributions during a job transition, we are further interested in the broader relevance of this motivation for the OSS community. Because our definition of the job search period is broad and includes periods with only moderately elevated signaling incentives, the ATT is best interpreted relative to the length of the job search period by performing a back-of-the-envelope calculation. Recent statistics state average job tenure in the US is around four years and only two years for software developers (Firaz, 2022). Assuming an average job tenure of three years applies to OSS developers, constant (base) activity levels across users and over time, and using our estimates ATT coefficient implies 4.9% of overall OSS activity is caused by signaling incentives during job transitions.[9] This suggests career concerns are a significant motivation for software developers and cause a sizable portion of contributions to OSS.

2.5.2 Heterogeneity

A natural question that arises from our main finding is whether there are systematic shifts in job movers' OSS activity during the job search period. This not only improves our understanding of how the signaling motive impacts users and activities differently but provides further validation of the signaling as the motive behind increased OSS activity. In particular, we explore two main dimensions of heterogeneity. First, we ask if job movers systematically focus their OSS activity during the job search period on certain types of projects, e.g., projects that are especially valuable as signal in the labor market. Second, we investigate if particular groups of job movers exhibit significant differences in effect size or if the effect size derives from all job movers equally.

We investigate effect heterogeneity with respect to the type of projects users contribute to during the job search period in Table 2.3. For this purpose, we use information on the main programming languages of projects and classify them into categories to distinguish broad project types. Our classification is documented in Table B.7 in the Appendix. This project-level approach requires using the number of contributions to each project type as outcome variable in user-level regressions. Thus, we run separate regressions of the model in Equation 2.2 for each project type. Results show significant differences in the ATT effects.[10] Notably, we

[9] Calculated as: $\hat{\beta}_2 * \frac{\#months_{JobSearch}}{\#months_{JobTenure}} = 11.77\% * \frac{15}{36}$.

[10] Note that increased sparsity leads to a loss of quantitative comparability to the main results in favor of comparability between project-type regression estimates.

Table 2.3: Heterogeneity by project type

IHS(single commits)	(1) low-level	(2) data eng.	(3) app dev.	(4) web dev.	(5) routine	(6) other
Job mover × job search	0.0136**	0.0426***	0.0256***	0.0607***	0.0277***	0.0353***
	(0.0061)	(0.0082)	(0.0051)	(0.0109)	(0.0073)	(0.0072)
Job mover × post move	-0.0047	-0.0177*	-0.0068	-0.0852***	-0.0145	0.0015
	(0.0077)	(0.0107)	(0.0077)	(0.0144)	(0.0098)	(0.0089)
User FE	×	×	×	×	×	×
Month FE	×	×	×	×	×	×
Experience FE	×	×	×	×	×	×
Adjusted R^2	0.26051	0.26955	0.29500	0.28444	0.28765	0.33629
Observations	1,946,413	1,946,413	1,946,413	1,946,413	1,946,413	1,946,413
Users	22,896	22,896	22,896	22,896	22,896	22,896

Notes: Results from estimation of Equation 2.2 with IHS-transformed number of commits to single-authored projects featuring main programming language of the respective class. Classification of programming languages according to Table B.7. Experience is measured as months since the first commit at move month. Robust standard errors are clustered at the user level. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

obtain the largest effects for web development and data engineering projects. Low-level programming, program routine, and app development projects experience much smaller increases in the job search period. These results are consistent with, first, job movers focusing on web development because such projects are a way to showcase their work product and thus skill in existing works. Secondly, job movers might signal more through data engineering projects as skills related to such projects are especially valuable in the labor market.

To investigate the second channel in more detail, we classify programming languages directly by their valuation in the labor market as stated in the *StackOverflow* list of top-paying technologies.[11] Using the same method as above, we compare the ATT for programming languages listed as top-paying technologies compared to non-listed programming languages. Among top-paying programming languages, we further separate the top 30 best-paying from other listed programming languages. Which languages are in each category is shown by Table B.8 in the Appendix. According to survey evidence by *StackOverflow*, programming languages in the best-paying category are associated with about USD 16,500 higher total annual compensation compared to other listed languages, a 24% premium. Table 2.4 displays

[11] The list is available at `https://survey.stackoverflow.co/2023/#technology-top-paying-technologies`. Last accessed on 11/03/2023.

the estimation results. While job movers significantly increase OSS activity during the job search period in all groups, the increase is by far the largest for the best-paying programming languages. Compared to the increase in languages lower on the list, the increase in OSS activity in projects using best-paying programming languages is about twice as large. The effects in the other two categories are not statistically distinguishable. This provides further indication that job movers focus their signaling activity on projects requiring skills especially valuable in the labor market.

Table 2.4: Heterogeneity by labor market value

IHS(single commits)	listed		
	(1) top 30	(2) other	(3) not listed
Job mover × job search	0.0842***	0.0456***	0.0396***
	(0.0095)	(0.0104)	(0.0076)
Job mover × post move	-0.0181	-0.0703***	-0.0165*
	(0.0126)	(0.0132)	(0.0094)
User FE	×	×	×
Month FE	×	×	×
Experience FE	×	×	×
Adjusted R^2	0.23914	0.24635	0.27395
Observations	1,946,413	1,946,413	1,946,413
Users	22,896	22,896	22,896

Notes: Results from estimation of Equation 2.2 with IHS-transformed number of commits to single-authored projects featuring main programming language of the respective class. Classification of programming languages according to Table B.8. Experience is measured as months since the first commit at move month. Robust standard errors are clustered at the user level. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

As an alternative method to classify projects, we tap project descriptions and deploy a natural language processing approach (Gentzkow et al., 2019). Only about one fourth of projects in our sample have descriptions and descriptions are typically brief. Therefore, we use a bag-of-words representation of all project descriptions and create a list of keywords and two-word phrases associated with four project categories (education, data science, website, and code) from analyzing the most frequently appearing uni- and bigrams.[12] We then assign

[12] We remove stop words and use word stems for this purpose. The respective n-grams for each category are

projects to a cluster when their description contains at least one associated n-gram.[13] This approach naturally results in a smaller sample due to few project with description and strict requirements from the keyword and phrase list. Yet, using appropriate keywords is a targeted approach and increases the confidence in our classification. Estimating our baseline model for commits to the project types generated with this method yields similar results, reported in Table B.12. We obtain the by far largest effect for coding projects and only small effects for websites and education. These findings are generally in line with the programming language-based approach. Notably, we find largest effects for coding projects, consistent with signaling of coding skills.

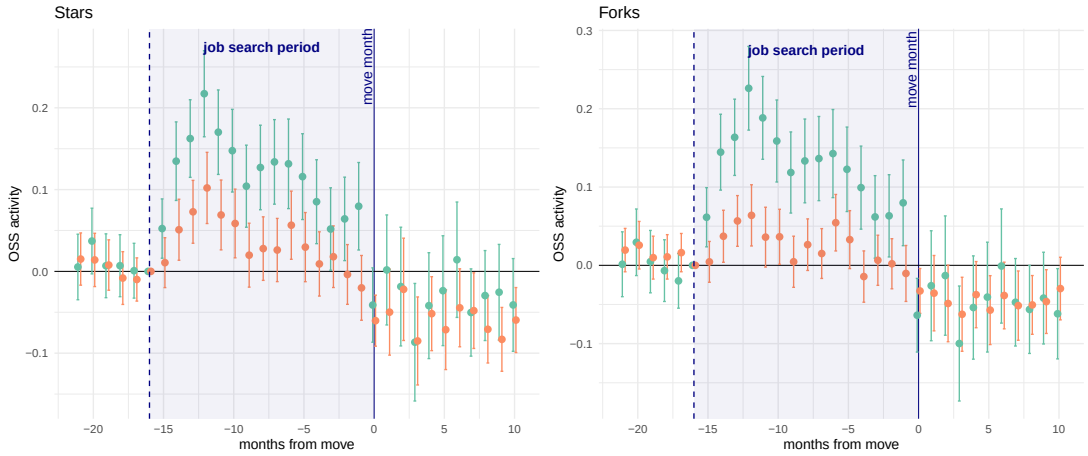
To distinguish whether career concerns induce job movers to start contributing to OSS, we formulate the model as a linear probability model (LPM) with an indicator for contribution rather than the number of contributions as recommended in Chen and Roth (2024). Estimation results are shown in Table B.10 and suggest a 5% higher probability of job movers contributing during the job search period relative to other movers. To investigate the extensive margin further, we run our baseline event study model using contributions to new projects, defined as projects initiated (i.e., first commit date) during the month under consideration, and compare new single projects to new team projects. Results in Figure B.7 show that job movers especially start working on new single projects during the job search period. Together, these findings suggest the extensive margin plays a significant role, and job movers specifically engage in OSS activity that is unambiguously attributable to themselves, which is advantageous in order to signal personal ability.

When thinking about the relevance of OSS contributions spurred by career concerns as a public good, quality is an important factor. On *GitHub*, projects may receive stars and can be forked by other users on the platform. Stars are a way for other users to indicate they find the project useful and to bookmark them for future reference. Forking refers to a process that copies a project into a new repository of the forking user so that she can use and alter the code in her own projects. Forking thus indicates other users' interest. We use both quality indicators and estimate the event study model, differentiating between OSS activity in projects with and without stars or forks, respectively. Figure 2.5 depicts the results and shows most OSS contributions of job movers during the job search period are in low-quality projects. This implies other users do not find signaling projects immediately useful. However, we found

reported in Table B.9.

[13] If a project description contains n-grams from multiple categories, we assign the project to the category with most n-grams in the description. In case of multiple categories with equal number of associated n-grams, we assign the project to each of these categories.

Figure 2.5: Heterogeneity by community use-value



Notes: Estimates for $t_j \times \text{JobChanger}_i$ based on Equation 2.1 with user, experience and calendar month fixed effects. The outcome is IHS-transformed commits to single-authored projects with (orange) or without (green) stars (left) or forks (right), respectively. The reference month is $t = -16$. Bars show 95% confidence intervals. Standard errors are clustered at the user level. *Sources:* GHTorrent, own calculations.

before that many signaling projects are websites that likely do not contain new code but rather showcase existing work more clearly. Such repositories are rarely starred or forked since usage is mostly off-platform. This might explain why the selected quality indicators suggest low quality and it does not necessarily mean that projects are perceived as not valuable. Rather, the value could lie in making existing works more visible and accessible to the community. Nevertheless, these findings do suggest a lower direct use-value of signaling projects for the OSS community regarding the usefulness of code in other projects on the platform.

Labor market signaling via OSS activity might be valuable to a different extent for job movers. We, therefore, investigate whether the effect is broad-based among all users or driven by a group of users with a particularly large increase in OSS activity during the job search period. For this purpose, we first explore heterogeneity with respect to followers comparing quartiles and find no significant differences (cf. Figure B.6). Second, we investigate whether signaling activity differs for users moving internationally by interacting dummy variables for types of moves to our baseline model. The results are reported in Table 2.5. Model (1) indicates that users moving internationally engage in 55% more labor market signaling via OSS compared to domestic movers. Likewise, inter-continental job movers signal even more and feature a 71% higher effect compared to non-intercontinental movers as shown by Model (2). Models (3) and (4) suggest that the effect differences are especially driven by international movers relocating to higher-income countries, though the coefficients lack statistical significance. These results are in line with existing evidence (e.g., Hann et al., 2013; Agrawal et al., 2016)

Table 2.5: International relocations

IHS(single commits)	international		upward moves	
	(1) international	(2) inter-continental	(3) income group	(4) GDP p. c.
Job mover × job search	0.1461***	0.1472***	0.1620***	0.1625***
	(0.0158)	(0.0150)	(0.0146)	(0.0144)
Job mover × job search × indicator	0.0619**	0.0923***	0.0295	0.0450
	(0.0260)	(0.0313)	(0.0393)	(0.0452)
Job mover × post move	-0.1040***	-0.1038***	-0.1038***	-0.1038***
	(0.0190)	(0.0190)	(0.0190)	(0.0190)
User FE	×	×	×	×
Month FE	×	×	×	×
Experience FE	×	×	×	×
Adjusted R^2	0.35948	0.35949	0.35945	0.35945
Observations	1,946,413	1,946,413	1,946,413	1,946,413
Users	22,896	22,896	22,896	22,896

Notes: Results from estimation of Equation 2.2 adding a triple interaction which features an indicator variable to separate heterogeneous effects of interest. Upward income group moves are defined as moves from developing to developed countries. Upward moves in GDP per capita are based on current 2021 PPP USD. Experience is measured as months since the first commit at move month. Robust standard errors are clustered at the user level. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, World Development Indicators, own calculations.

suggesting that OSS signals could substitute formal certification, which is less transferrable and accepted internationally, particularly for developing countries.

Table 2.6 shows that there is some heterogeneity in signaling activity depending on users' origin (old) and destination (new) affiliation. Importantly, users who obtain new jobs at big tech firms do not engage in labor market signaling through OSS activity to a significant extent. In contrast, users changing jobs to academic affiliations signal significantly more. There is no statistically significant difference in signaling activity depending on the old affiliation, but an economically significant point estimate for above-median firm size points towards more signaling activity by users coming from larger firms. These results, though weak, are consistent with an arguably generally greater role of open source in academia while large corporations like the big tech firms emphasize proprietary software more, and users qualified for a job at the big tech firms typically do not need (additional) ability signals from OSS activity

Table 2.6: Heterogeneity by affiliation

IHS(single commits)	destination			origin	
	(1) median	(2) big tech	(3) academia	(4) median	(5) academia
Job mover × job search	0.1784***	0.1753***	0.1578***	0.1631***	0.1601***
	(0.0198)	(0.0144)	(0.0145)	(0.0142)	(0.0502)
Job mover × job search	-0.0219	-0.1460***	0.0930**	0.0843	-0.0114
× indicator	(0.0234)	(0.0480)	(0.0457)	(0.0999)	(0.0652)
Job mover × post move	-0.1038***	-0.1042***	-0.1032***	-0.1040***	-0.1693***
	(0.0190)	(0.0190)	(0.0190)	(0.0190)	(0.0528)
User FE	×	×	×	×	×
Month FE	×	×	×	×	×
Experience FE	×	×	×	×	×
Adjusted R^2	0.35946	0.35950	0.35947	0.35946	0.36126
Observations	1,946,413	1,946,413	1,946,413	1,946,413	1,406,169
Users	22,896	22,896	22,896	22,896	22,896

Notes: Results from estimation of Equation 2.2 adding a triple interaction which features an indicator variable to separate heterogeneous effects of interest. Median split refers to median size of affiliation in terms of users in the full *GHTorrent* sample. Big tech refers to Google, Amazon, Meta, Apple and Microsoft. Academia refers to students and university affiliations. Specifically, users stating *university, college, institute, universiteit, universidad, universität* or *student* in their affiliation are assigned to academia. Destination (origin) refers to users' affiliation before (after) the affiliation change. Experience is measured as months since the first commit at move month. Robust standard errors are clustered at the user level. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

as they tend to have the highest credentials anyways.

2.5.3 Robustness

We choose a model that uses the inverse hyperbolic sine (IHS) transformation of the outcome variable as the preferred specification, which has the mentioned advantages of retaining zeros while approximating the logarithmic transformation (see, e.g., Burbidge et al., 1988; MacKinnon and Magee, 1990; Bellemare and Wichman, 2020). A related and widely-used transformation is the logarithmic transformation and specifically $\log(y + 1)$ (Bellégo et al., 2022). The challenge with these transformations is that they are scale-dependent, but this problem is more severe for high-valued and sometimes-zero outcomes (Mullahy and Norton,

2022; Chen and Roth, 2024). Aihounton and Henningsen (2021) frame scaling as a data fitting exercise. Since our data is low-valued and sparse, we opt for a conservative quantitative interpretation arising from IHS transformation of the unscaled dependent variable. Another class of alternative models are Poisson models such as the PPML estimator. These models are the established go-to choice in trade (Larch et al., 2019) and other applications with high-valued count data featuring zeros such as investment, profit, or revenue data (Cohn et al., 2022). However, these models perform poorly in practice on low-valued sparse panel data such as ours and there is no standard econometric approach yet. Additionally, our data features sparsity not only across units but also within. For such applications, IHS or logarithmic transformations are the preferred choice in practice, e.g. in Xu et al. (2020) or Bahar et al. (2022).

Apart from being conservative in our preferred model specification, we assess the robustness of our results by estimating several alternative models. Results are reported in Table B.10 in the Appendix. First, we show that the most widely-used alternative way to transform the dependent variable in similar applications (e.g., Xu et al., 2020), a logarithmic transformation, yields similar coefficient estimates. Second, we run two types of frequently used count data models: a negative binomial and a Poisson fixed effects model. Both models are known to frequently exhibit performance issues with fixed effects and convergence issues (Correia et al., 2020; Bellégo et al., 2022). The PPML model results in similar coefficient estimates for the job search period and an increased estimate for the post-period. The negative binomial model estimates are significantly inflated by a factor of four to five compared to our preferred specification. These findings indicate the robustness of our results with respect to model specification and confirm that our estimated effect size is conservative. Furthermore, we follow state-of-the-art best practices (Chen and Roth, 2024) in that we explicitly consider intensive and extensive margin effects. The formulation of our model as LPM suggests reasonably high importance of the extensive margin (see Model (3) in Table B.10). Note that through our sample selection of active OSS contributors only, extensive margin effects are likely downward biased. At the same time, this implicit conditioning decreases potential bias of the intensive margin in our main specification (Hersche and Moor, 2020).

Measurement error in the move date possibly introduces bias in our estimates due to observing location data only every six months and users entering their new location after relocation. The event study results in Figure 2.4 partly alleviate this concern as there is a discontinuous drop in OSS activity of job movers at the proxied move date. Nevertheless, it is unclear whether the downward trend during the job search period is due to already-moved job movers still in the

treatment group or, e.g., due to decreased signaling incentives of users who already found a job. We address this by varying the job search period definition and separately estimating a coefficient for the period for which we are unsure if the user actually already moved. This adjustment generally increases the estimated effect by up to three percentage points to about 14.2%. Note that although this introduces upward bias in our estimates it simultaneously alters the length of the job search period and, as a result, leads to a mechanic downward adjustment in the interpretation when thinking about overall OSS activity attributable to career concerns.

Our approach exploits the specific timing of elevated career concerns during the job search period. Still, coinciding increases in other motives are a potential concern. Specifically, if people disproportionately learn new skills in between jobs and this activity is conducted in public repositories on *GitHub*, our model would wrongly attribute such activity to career concerns. One of our project types in the keyword-based classification are educational projects. This category captures repositories associated with coursework, assignments, or online education (e.g., *Coursera*). Table B.12 shows no effect on the activity in educational projects, suggesting that activity driven by learning motives does not drive our effect. In addition, we investigate projects not owned by the mover, such as company projects, or projects consisting of initial forks (a copy of existing repositories). We find no evidence for a significant relevance of these channels (see Table B.11).[14]

Decisions to relocate and change jobs are endogenous. Therefore, unobserved differences between treatment and control group that affect job movers' but not other movers' OSS activity during but not outside the job search period could potentially create the observed activity pattern. We argue this is unlikely for four main reasons. First, the observed activity pattern outside the job search period shows a strikingly similar evolution as shown in Figure 2.4. Second, the observed activity pattern is highly specific to the exploited time variation and, thus, likely not due to general differences between treated and control users. Third, job and other movers are already quite similar in their observable characteristics (cf. Table 2.1). If anything, job movers are less active than other movers, which contradicts potential reverse causality concerns that job movers are more active and therefore succeed in getting a new job. Fourth, the generally sparse activity of developers predominantly in small projects not widely known in the community does mitigate potential concerns that job moves were initiated by

[14] Note that project ownership is prone to measurement error, as it might wrongly capture the same individual as distinct persons, e.g., when committing to projects using two different e-mail addresses as identification or using multiple devices. Thus, it is not surprising that there is a small significant effect for non-own projects in Table B.11.

employers that detected users' OSS activity. All these circumstances point to genuine plans to change jobs that drive signaling motives during the job search period.

For completeness, we report estimation results for the event study specification in Table B.13 and, similarly as in Table 2.2 for the ATT, show the results for the models without experience and calendar month fixed effects, as well. Figure B.8 plots event study coefficients for variations of the baseline model. Further, we establish the robustness of our results to alternative sample definitions with respect to geocoding and job changes in Models (3) and (4) of Table B.10. For user-level heterogeneity analyses using interaction terms, alternative model specifications based on separate regressions with redefined outcome variables similar to the project-derived heterogeneity analyses (Tables B.15, B.16, and B.17) show qualitatively similar results.

2.6 Conclusion

We show private career concerns of software developers induce significant contributions to open-source software, a public good. By exploiting temporal variation in signaling incentives in a quasi-experimental design, we establish a causal increase of OSS activity of job movers compared to users relocating for other reasons in the job search period by about 12%. These positive externalities of labor market signaling are sizable from both the individual and the community perspective but often neglected in existing works that predominantly emphasize other motives to contribute to OSS development. A broad base of users on the largest online code repository platform, *GitHub*, engages in labor market signaling during the job search period and signaling opportunity even attracts first-time contributors. OSS activity driven by signaling motives is disproportionately directed to projects that increase external visibility of existing works or are written in programming languages highly valued in the labor market. At the same time, signaling projects are starred and forked less by other users on the platform. This suggests OSS activity induced by career concerns is targeted less to the direct use-value of the OSS community and more to their value as a labor market signal.

Our study has limitations. Data does not contain information on users besides activity on the platform, location, and affiliation and cannot be linked to other data on the individual level, which constrains the number of possible heterogeneity analyses. Furthermore, location and affiliation changes are only observed at snapshot frequency, i.e., roughly every six months. This leads to blurriness in the proxied move (and affiliation) change months and likely biases our estimates downwards. In general, we opt for a conservative model specification as a

quantitative interpretation of our effect size depends on econometric choices regarding model class and outcome scaling and transformation. It should also be noted that although our empirical strategy identifies the causal effect of temporarily elevated signaling incentives under reasonable assumptions, it by no means captures all OSS activity attributable to labor market signaling and therefore should be interpreted as a lower bound estimate. Similarly beyond the scope of this work is to assess the extent to which OSS signals improve individual-level labor market outcomes.

Despite these limitations, our findings have several managerial implications. Importantly, decision-makers aiming to increase OSS activity should take into account career concerns as a significant motivating factor for developers. Platform design addressing the signaling needs of users explicitly might grow the platform at both the intensive (activity) and the extensive (users) margin. Measures that foster public visibility, transparency as well as accessibility for non-experts might contribute to this goal, e.g., through easily understandable activity metrics, skill badges, or lists of spoken programming languages on user profiles. At the same time, platform managers should be aware that signaling motives might steer OSS activity towards projects with lower direct use-value for the community whenever there is a gap between signaling value and community value of projects. For hiring managers, our results emphasize that OSS is a commonplace and potentially valuable signal of skill for developer talent. Consequently, it should receive attention in employee search and assessment.

Finally, our study provides several insights for public policy. In general, the positive externalities of career concerns on public good creation merit attention due to likely significant positive spillovers of OSS on the private sector and innovation. Innovation policy that enables and encourages publicly funded software development to be hosted and shared on online open-source platforms may increase the motivation of the funded developer teams while at the same time generating OSS, a public good that potentially spurs further innovative activity. With respect to labor market and educational policy, our results point to the continued shift away from (public) skill certification in occupations related to software development and emphasize a greater role of more fluid and practical skill signals directly showcasing work product. Educational institutions should acknowledge both the labor market value of OSS activity for their students and the positive societal externalities from such activity and consider encouraging students to engage in OSS development or even explicitly integrate OSS projects into curricula.

3 Social learning on digital platforms: Evidence from GitHub

Open-source software (OSS) is a cornerstone of the modern digital economy, but little is known about how developers choose the projects to which they contribute their valuable time out of thousands available. Focusing on the OSS platform *GitHub*, I provide evidence on a social learning mechanism, that is how a contribution by a highly influential and followed user, called rockstar, to a project is beneficial to project developments. For that, I compare projects that receive a one-time rockstar contribution to similar projects that do not, using several difference-in-differences approaches as well as NLP methods. I find a sizable increase in project contributions and popularity in the month of rockstar contribution. Overall, contributions by highly influential users correspond to sizable increases in project developments, mainly originating from rockstar followers.[1]

Keywords: peer effects; social interactions; social multiplier; open source; online collaboration
JEL-No: D83; L17; O36

[1] I thank Annalí Casanueva Artís, Florian Englmaier, Thomas Fackler, Oliver Falck, Lisandara Flach, Moritz Goldbeck, Yuchen Mo Guo, Emma Harrington, Anna Kerkhof, Arianna Ornaghi, and Sebastian Wichert for valuable comments and suggestions. I also thank participants at the 6th Doctoral Workshop on the Economics of Digitization, INT, and ORG internal seminars. Support by the Deutsche Forschungsgemeinschaft through CRC TRR 190 (project number 280092119) is gratefully acknowledged.

3.1 Introduction

Open source software (OSS) is nowadays an important input in products and services by companies. Some estimates suggest that 96% of software codebases entail OSS parts (Synopsys, 2023) and the global value of widely-used OSS should range between 4.15 billion USD and 8.8 trillion USD (Hoffmann et al., 2024). Artificial intelligence (AI), increasingly used in all aspects of life, to a large extent builds on machine learning open source software (Langenkamp and Yue, 2022) with, among others, Meta-CEO Mark Zuckerberg also recently publishing its open-source AI model.[2] The provision of OSS, a public good, relies on a decentralized community of high-skilled developers on large platforms (Nagle, 2022), and software evolves with the voluntary contribution of developers (McDonald and Goggins, 2013). They need to allocate their limited and valuable time, however little is known about how they choose which projects to work on. Understanding how developers decide which projects to join is of critical importance, as their contribution decisions are crucial for the digital economy and its software production. Especially on large platforms, hosting millions of projects, it is costly to identify which project is promising to work on. By observing highly influential developers and their activity on OSS platforms developers can potentially overcome this problem (Dabbish et al., 2012). If this is the case, do contributions by highly influential users correspond with increased project developments by other users learning, and, in turn, contributing to these projects?

The concept of acquiring information by observing peers and their actions and their influence on an individual's decisions is called social learning. The literature on peer effects and social interactions ranges from consumption choices to financial or even education decisions (Sacerdote, 2011; Anderson and Magruder, 2012; Ouimet and Tate, 2020; Bailey et al., 2022). When less or no information is available, it is especially helpful to observe others and their actions (Anderson and Magruder, 2012). On OSS platforms with millions of projects to potentially use or join, assessing and identifying which project is promising and worth working on, is costly. Survey evidence suggests that developers on OSS platforms learn from the actions of others about interesting or useful projects (Dabbish et al., 2012). Thus, by observing the activity of peers on the platform and which projects they work on, an individual can learn about the project and its quality.

In this paper, I focus on one of the largest OSS platforms, *GitHub*, to study social learning, and by that try to understand the behaviour of software engineers. On the platform, there exists a

[2] See `https://about.fb.com/news/2024/07/open-source-ai-is-the-path-forward/`, accessed 26 August 2024.

subset of highly influential and skilled users, called rockstars, with a large number of followers, which is often used as a proxy for their work quality and expertise (Lee et al., 2013; Huang and Chung, 2019; Grisold et al., 2021).[3] I analyze the change in project activity and popularity after a one-time contribution by one rockstar to a project. Rockstars have been studied for their social influence on their followers, drawing their attention to interesting projects the rockstars contribute to (Lee et al., 2013; Badashian et al., 2014). First, I analyze the quantitative and qualitative OSS project characteristics and rockstar contributions. Therefore I use natural language processing (NLP) approaches to analyze project descriptions and machine learning (ML) algorithms to predict rockstar contributions. Second, I compare projects and their activity and popularity dynamics that experience a minimal one-time rockstar contribution, to similar projects that do not in a difference-in-differences setting. For that, I match no rockstar projects to rockstar projects on project activity and popularity characteristics. This allows me to obtain a control group of comparable no rockstar projects, and by that I can account for project dynamics over time, and study how project activity and popularity are related to a rockstar contribution. By focusing on a minimal rockstar contribution I can mitigate any concerns that my results are driven by knowledge spillovers. For projects where a rockstar contribution occurs, this signal may help the rockstar's followers to learn about the existence and usefulness of the project and lead their attention towards the project. This, in turn, potentially increases their likelihood to contribute and use the project themselves. I focus on a minimal contribution by a rockstar, where I expect increases in activity are most likely related to social learning, and I clearly can identify changes in project dynamics before and after the rockstar contribution.

The prediction analysis reveals that more mature and more active, and, thus, potentially high-quality, projects are more likely to receive a rockstar contribution. In the difference-in-differences analysis, I find a sizable increase of 30% in code contributions in the rockstar contribution month for projects receiving such a contribution relative to projects without a contribution. In the post-period, I observe a short-term decrease in activity, however statistically insignificant in total. Including the activity of forks of the project, i.e. project copies, the decrease in the short-term post-period stays statistically insignificant. The increase in OSS activity seems to be mainly driven by the rockstar's followers but also non-followers show a positive relation with rockstar contribution. I find evidence for a social multiplier effect, as the followers of the rockstar followers show a lagged increase in activity as well. Additionally, the rockstar contribution is associated with an increase in the number of new users contributing to

[3] For instance, the most followed user as of March 2021 is Linus Torvalds, the founder of the Linux kernel system and the distributed version control system *git*.

the project. New users tend to add or fix code, whereas old users mainly merge code changes to the project. Next to activity, the rockstar contribution is also related to an increase in project popularity, proxied by the number of stars, i.e. bookmarks, or forks, i.e. project copies. A rockstar contribution, thus, is linked in the short term to elevated project activity, and in the long term to an increase in attention by stars and forks. The findings provide evidence that social learning potentially matters for working decisions on *GitHub*, as contributions by highly influential users correspond to sizeable increases in project development.

Focusing on a small and single contribution by such a highly influential peer to a project, I can differentiate social learning from knowledge spillovers and relate the single contribution to the overall project activity and popularity and analyze changes in activity relative to projects that do not experience such a rockstar contribution. Project activity captured by code contributions are direct improvements to the project and resemble costly and valuable invested time. Project popularity, such as stars or forks, do not require much effort and may not lead to direct changes in the project, but they reflect attention towards the project, which can, later on, turn into project improvements. Thus, both are important measures of project development.

There is a continuously growing literature on social learning. In the past, most research studied the concept of social learning from a theoretical point of view (Granovetter, 1973; Banerjee, 1992; Bikhchandani et al., 1992; Young, 2009). In recent times there is increasing empirical evidence on social learning focusing on consumption decisions in various settings (Moretti, 2011; Anderson and Magruder, 2012; Acemoglu et al., 2022; Anenberg et al., 2022; Bailey et al., 2022), financial decisions (Ouimet and Tate, 2020), housing market decisions (Bailey et al., 2018) or education decisions (Sacerdote, 2011). The theory of social learning suggests, that individuals learn and update their beliefs about the quality of a good by observing others (Bikhchandani et al., 1992, 2024). Empirically identifying social learning faces two main challenges. It requires testing if, for instance, an individual's consumption decision depends on the consumption and/or satisfaction of other close-by individuals. First, this requires detailed consumption data, and, second, similar choices among peers may reflect common preferences, not necessarily informational spillovers (Moretti, 2011). Regarding the role of individuals on digital platforms leading the attention towards content, there is some evidence on the relationship between sharing scientific papers on Twitter (now called X) and the papers' numbers of citations (Finch et al., 2017; Chan et al., 2023; Branch et al., 2024). Findings suggest mixed results on the effect of sharing scientific papers and subsequent citations (Tonia et al., 2020; Branch et al., 2024).

This paper contributes in two distinct ways to the current literature on social learning. First, in contrast to most research, I focus in this setting on working or contribution decisions. The past literature was able to show that consumption, educational, or financial choices of peers matter for an individual's choices, e.g. restaurant or movie consumption (Moretti, 2011; Anderson and Magruder, 2012). However, it is also important to understand how individuals choose where or what to work on. OSS developers have limited time to work on projects, thus, it is important to understand how they identify and choose where to contribute to the public good provision. To my knowledge, this has not been empirically studied for OSS development yet. Second, my setting and the fine-grained data enable me to clearly identify links between peers. In contrast to most studies on social learning, this allows me to study how information disseminates across links. Further, that way I can provide a quantitative measure of the influence of highly followed users on their followers' contribution patterns. It is unclear if influential individuals can help in spreading academic knowledge (Branch et al., 2024). For OSS development my findings suggest a sizable and positive relationship between highly followed users and their followers' contribution activity.

The remainder of the paper is organized as follows. First, I describe the data and the setting in Section 3.2. Thereafter, I present the empirical strategy in Section 3.3. In Section 3.4 I discuss the results and conclude in Section 3.5.

3.2 Context and Data

GitHub is the world's biggest code hosting site and is based on the *git* version control system (GIT, 2021). Since its launch in 2008, the platform experienced an increase in popularity across software developers and reports 73 million users worldwide and hosting about 189 million projects in 2021 (GitHub, 2021). Figure C.1 shows the increase in new projects created on *GitHub* over time .[4] The OSS activity data used for the analysis is the publicly observable activity stream on *GitHub* provided by *GitHub* Torrent (*GHTorrent*) (Gousios, 2013) between 2008 and 2021.[5] Additionally, I tap the *GitHub* REST API to obtain more detailed information

[4] Interestingly, after the Microsoft acquisition of *GitHub* in 2018, there is a drop in new projects created on the platform. It presumably stems from some developers leaving the platform due to the acquisition. See `https://www.heise.de/news/GitHub-Entwicklergemeinde-in-Sorge-ueber-Ausverkauf-an-Microsoft-4068008.html`, accessed on 27 August 2024.

[5] *GHTorrent* creates snapshots of the public activity stream on *GitHub*, e.g. user registration, projects, and commits, and makes it accessible to everyone in a relational database. The snapshots were taken on 2015/09/25 (201509), 2016/01/08 (201601), 2016/06/01 (201606), 2017/01/19 (201701), 2017/06/01 (201706), 2018/01/01 (201801), 2018/11/01 (201811), 2019/06/01 (201906), 2020/07/01 (202007) and 2021/03/06 (202103). Potentially sensitive personal information (e.g. user name or e-mail address) was de-identified by data center staff prior to data analysis by the author.

on the code contributions.

Rockstars I identify rockstars by their number of followers, which is a measure of a user's popularity. The higher interest in the user's action on *GitHub*, reflected in a large number of followers, can be seen as a proxy for work quality and they are therefore perceived as especially skilled (Lee et al., 2013; Huang and Chung, 2019). Users tend to become rockstars by exhibiting a large number of significant contributions to projects or being the owner of very popular projects (Badashian et al., 2014; Blincoe et al., 2016). When following a user, actions performed by the followed user lead to a news feed event. Most *GitHub* users, i.e. 95% of all users, do not have any followers. Of those 5% of all users with at least one follower, I take the upper 10% as my rockstar sample. These are 433 users with between 2,742 and 95,258 followers.[6]

Highly followed users are more experienced users, and compared to the average registered user, more active. Table C.1 compares the universe of users to the rockstar sample. Most registered users on *GitHub* have no followers and are inactive, noticeable by having a median of six total commits, zero followers, and 34 months of platform duration. The rockstar sample, on the other hand, consists of more active contributors with a median of 4,188 commits and a median platform duration of 92 months, or about 7.5 years. Lastly, by the definition of rockstars, they have a high number of followers, with a median of 4,446 followers. Rockstars tend to work more on collaborative projects than single projects with a median of team projects of 96 compared to a median of single projects of 70 projects. In those team projects, though, they mostly are the only rockstar and contribute merely a few times to a project with an average of three months of active contributions per project (cf. Figure C.2). In comparison, other users work similarly on team projects as on single projects, displayed by the same median number of two projects for each project class.

Sample selection For the analysis I focus on projects consisting of at least three users with either only one rockstar committing up to three commits to the project in one month over the whole project life cycle, where the contributing rockstar is not the project owner, or no rockstar contribution at all. A concern could be that any increase in activity after the rockstar contribution is not a result of social learning but rather driven by knowledge spillovers. Focusing on a minimal contribution of three commits in one month should mitigate this concern. The rockstar's contributions in these cases are likely minimal changes, and the rockstar is not the main developer in the project. The rockstar's followers, though, will be

[6] In Section 3.4.3, I vary my definition of rockstars for robustness and use the upper 25% or upper 5% as rockstars. The results are similar, and increasing in size the higher the threshold.

notified about the rockstar's contributions, and their attention is directed towards this project. Additionally, the projects must exist for at least 12 months to have a sufficient number of observations and have a minimum activity of six months with non-zero commits.[7] My activity requirements ensure that projects in the analysis are actively worked on.

Survey evidence suggests that rockstars are able to identify interesting projects with high usability (Dabbish et al., 2012). Therefore, projects receiving a rockstar contribution are in some respects inherently different from projects without a rockstar contribution, otherwise a rockstar contribution would not occur. For the analysis, however, it is crucial to have a comparison group of no rockstar projects that are similar in activity and popularity before the rockstar contribution. Therefore, I match the 5,737 rockstar projects to the 972,285 no rockstar projects using coarsened exact matching on project activity and popularity characteristics. Specifically, I use as project activity characteristics the average monthly number of project commits, the number of months with non-zero commits, and project age. I consider the number of stars, i.e. bookmarking a project, and project copies, called forks, per project as project popularity characteristics. This results in 1,913 (33.34%) rockstar projects matched to 2,481 no rockstar projects and 204 (47.11%) rockstars. The matching algorithm leaves me with a sample of rockstar and no rockstar projects with a similar probability of receiving a rockstar contribution conditional on the covariates used for matching. No rockstar projects should now resemble an appropriate comparison group for rockstar projects and their activity patterns if a rockstar contribution would not have occurred.[8]

Table 3.1 provides a summary statistic on all rockstar projects, the subsample of matched single rockstar projects, and no rockstar projects, and reports the median due to the skewness of the data. Single rockstar projects are projects with high activity concerning the number of commits and contributors, also before the rockstar joins, and exist for a longer time compared to all rockstar projects. Project activity, size, and age are larger, partly as a mechanical result of my activity requirements. The contributors in single rockstar projects are more experienced regarding platform duration than in all rockstar projects. The share of projects that have the rockstar's main programming language is similar among single rockstar projects and all

[7] I define a project's start and end date by the first and last time any user committed to the project.

[8] My results may be driven by the matching results, or, alternatively, no rockstar projects, in general, do not resemble an appropriate comparison group for rockstar projects. Therefore, in Table C.16, I compare either all rockstar projects to all no rockstar projects, i.e. without matching the projects, or limit the sample to only rockstar projects, i.e. a comparison of earlier vs. later treated. The results do not differ much from the baseline estimates. Additionally, I account for staggered adoption in Table 3.8 by estimating cohort-specific treatment effects and aggregating the coefficients using the Sun and Abraham (2021) approach. Again, the results remain qualitatively the same.

Table 3.1: Summary statistics: projects

Median	Rockstar projects		No rockstar
	all	single rockstar	
Activity			
Commits	13	230	156
at rockstar contribution	-	*122*	-
av. monthly commits	4	9	7
non-zero commit months	2	12	16
Project age	5	62.5	16
age at rockstar contribution	-	*16*	-
age quarter at rockstar contribution	-	*third*	-
Project members	2	34	3
at rockstar contribution	-	*20*	-
No. rockstars	*1*	*1*	-
User follower	*0*	*18*	0
User experience	*8*	*58*	*19*
Rockstars main language	0.62	0.58	-
Quality			
Forks	0	100	0
at rockstar contribution	-	*0*	*0*
Stars	0	0	0
at rockstar contribution	-	*0*	*0*
Forked	0.58	0.40	0.10

Notes: Project age is measured as months since the first commit. Rockstar main programming language is the programming language the rockstar is active in the most over the observation period. *Sources:* GHTorrent, own calculations.

rockstar projects, with 58% and 62%, respectively. Both types have a median of zero stars, whereas single rockstar projects have a median of 100 forks compared to a median of zero for all rockstar projects, with the latter being more often a fork itself.

Projects, that don't experience a rockstar contribution, have similar levels of activity and popularity as single rockstar projects before the rockstar contribution occurs after implementing the matching algorithm. Their median number of monthly commits is 7, their median project age is 16 and they have a median of zero stars and forks, compared to 9 commits, 16 months, and zero stars and forks for single rockstar projects at the time of rockstar contribution, respectively. No rockstar projects are smaller with a median of three contributors

than single rockstar projects. The projects rockstars contribute to tend to be low-level programming projects, whereas no rockstar projects are rather web development projects, reflected by the projects' programming languages. Table C.2 reports the five most frequently used programming languages per single and no rockstar projects and their respective shares. The projects, thus, moderately differ in their usage. In the regressions, I include project fixed effects and programming language time trends to take these differences into account. Overall, no rockstar projects are similar to rockstar projects at the contribution time regarding activity and popularity characteristics, though somewhat smaller in team size. They potentially are a good comparison group to capture a project life cycle on *GitHub* if a rockstar contribution would not have occurred.

Going back to the universe of all single rockstar and no rockstar projects, I analyze project characteristics and descriptions to understand the differences that may lead to a rockstar contribution or not. First I run a LASSO regression, single tree, random forest, and gradient boosting tree and compute the variable importance predicting a rockstar contribution.[9] Figure C.3 presents the variable importance calculated for the different models.[10] For single rockstar projects, I take the variable values in the calendar month before the rockstar contribution occurs. Project age and the number of contributors are the most predictive variables among the models if a project receives a rockstar contribution or not. It seems, that rockstars rather point towards more developed and larger projects, as most single rockstar projects are more mature and have a large contributor base at the rockstar contribution time. The rockstar may learn about the project herself by observing others who work on the project. For LASSO, the single regression tree, and the gradient boosting tree, project stars also show high variable importance, which is likely related to the total number of contributors. If more developers work on the project, they may simultaneously also give more stars to the project. Overall, the models suggest that more mature and more active projects have a higher chance of receiving a rockstar contribution.

Turning to the project description, I apply a natural language processing approach to analyze the similarities for the subset of all single and no rockstar projects for which a project

[9] For potential predictors I include in the models total project commits, total project contributors, project age, project forks, project stars, project programming language, project contributor experience, and an indicator variable if the project is a fork.

[10] The models vary in their way of calculating the variable importance. LASSO coefficients resemble variable effect sizes. For regression trees, variable importance depends on how much a variable adds to the tree's ability to correctly forecast the outcome variable. In ensemble methods, the variable importance is combined across the several trees, and, thus, is more robust. In sum, LASSO reveals which variables affect the outcome, whereas regression trees identify variables improving the prediction accuracy.

description is available. After cleaning the data by removing stopwords and stemming, Figure C.4 displays word clouds of the most frequent words used in single rockstar and no rockstar projects. Words used in the project descriptions moderately differ between single rockstar and no rockstar projects. Interestingly, the by far most frequent word in single rockstar projects is *kernel*. The kernel is the heart of the operating system and is important for the security and integrity of the whole operating system. Due to its high complexity, especially for the Linux kernel, it requires great knowledge of the refinements of software and hardware (Tan et al., 2020). Single rockstar projects, thus, tend to be projects working on the kernel with high skill requirements for the contributors. Again, project fixed effects control for the different projects' usage in the regressions.

3.3 Empirical Strategy

I analyze the relationship between project activity and project popularity in a given month on the one hand and how it varies before and after a rockstar contribution, on the other hand using a difference-in-differences model. To do so, I compare projects with a single rockstar contribution to similar projects with no rockstar contribution to account for project dynamics over time. The reasoning behind that is that other users may use the rockstar contribution as a signal for project quality, and, thus, learn about project quality by a rockstar contribution. Projects receiving a rockstar contribution are exposed to the potential learning of by other users. This may lead other users to be inclined to contribute themselves, as the projects are now potentially perceived as more useful and interesting. For projects with no rockstar contribution, this increase in project activity and popularity should not occur. By the activity requirements and matching, as explained in detail in Section 3.2, rockstar projects[11] and no rockstar projects resemble projects, that are actively worked on, and exhibit similar trends in activity before the rockstar contribution. After the rockstar contribution and the associated increase in attention, I assume increased project activity for projects receiving a rockstar contribution relative to projects with no rockstar contribution if social learning increases the contribution probability of other users.

As a result of these considerations, I empirically investigate the dynamics of OSS activity by estimating the following event study model:

$$y_{it} = \beta_1 + \sum_{j=\underline{T}}^{\overline{T}} \left[\beta_j \left(t_j \right) \right] + \delta_{s(t)} + \delta_{ls(t)} + \delta_{a(i)} + \delta_i + e_{it}, \tag{3.1}$$

[11] For simplicity, I refer to single rockstar projects now as rockstar projects if otherwise not mentioned.

where y_{it} is a measure of project activity or project popularity of project i in relative-to-rockstar contribution month t, excluding the rockstar's contribution. The event study panel is balanced in the pre-period but unbalanced in the post-period, because some rockstar contributions occur at the end of the observation period. Coefficients of interest are β_j and reflect the difference in the temporal pattern of project activity around the rockstar contribution date between projects that receive a rockstar contribution to projects that do not. Calendar month fixed effects $\delta_{s(t)}$ from January 2008 to March 2021 account for month-specific shocks that affect all projects simultaneously, and also take into account the increasing popularity of *GitHub*. Programming language × calendar month fixed effects $\delta_{ls(t)}$ control for programming language trends, and thereby the possible change in popularity of projects. Project age at treatment $\delta_{a(i)}$, as an important predictor for rockstar contribution and related to different activity patterns, are controlled for. Lastly, project fixed effects δ_i account for time-invariant project characteristics. Robust standard errors are reported on the project level.

This way, I implement the standard difference-in-differences model to estimate the average treatment effect on the treated based on a pre-, contribution- and post-period. The reference period is the pre-period, the time a project has no rockstar contributions. In the contribution period, the rockstar commits to the project, and in the post-period, after projects experienced a one-time rockstar contribution, the treated projects encounter a potential increase in other users' attention due to social learning. The resulting model specification is

$$y_{is} = \beta_1 + \beta_2 \text{rockstar contribution}_{s(i)} + \beta_3 \text{post}_{s(i)} + \delta_s + \delta_{ls} + \delta_{a(i)} + \delta_i + e_{is}, \qquad (3.2)$$

where $\text{rockstar contribution}_{s(i)}$ is one if calendar month s is equal to the month the rockstar contributes to project i. $\text{post}_{s(i)}$ becomes one if calendar month s is larger than the calendar month a project i experienced a rockstar contribution. The coefficients of interest are β_2 and β_3. β_2 captures the difference in the change in OSS activity between projects that experience a rockstar contribution in a given month, and the change in OSS activity for other projects that do not experience a rockstar contribution at the same point in time relative to the pre-period. β_3 represents the average difference in changed OSS activity after experiencing a rockstar contribution, i.e. differences in OSS activity between rockstar projects and no rockstar projects in the post-period relative to the pre-period.

The key identifying assumption is that in the absence of a rockstar contribution, OSS activity in rockstar projects would have evolved similarly to no rockstar projects, conditional on controls. Although I cannot test this assumption directly, I assess it by showing parallel trends in the

pre-period when increased attention towards rockstar projects due to social learning is absent. The main remaining threats to my empirical design are factors unrelated to social learning that affect project activity for projects that experienced a rockstar contribution but not projects that did not experience a rockstar contribution or vice versa. One possible concern could be that the timing of the rockstar contribution is driven by the project activity. The rockstar may contribute to the project because it shows increasing activity. Then the change in activity with the rockstar contribution may be because of other reasons than the rockstar contribution. Statistically insignificant parallel trends in the pre-period mitigate this concern, but can not completely rule out that the contribution timing is not random. Therefore, I conduct a placebo test in Section 3.4.3, in which I do not find a statistically significant effect for a randomly assigned treatment to rockstar projects in the pre-period. Another concern could be, that the choice of the rockstar to contribute to a project or not is likely not exogenous to project characteristics, often referred to as the selection problem stated by Manski (1993b). I include project fixed effects to account for project-specific characteristics that may simultaneously affect project activity and rockstar contribution. I further control for project age, which is an important predictor for a rockstar contribution as shown in the variable importance analysis. Including the controls potentially limits this concern but cannot completely remove it. It could also be the case that because of the rockstar contribution, users move from no rockstar projects to rockstar projects. This would lead me to overestimate the effect of a rockstar contribution on project activity. However, I also study project popularity, where this shift should not happen because there is no limitation in giving stars as there is with contribution time.

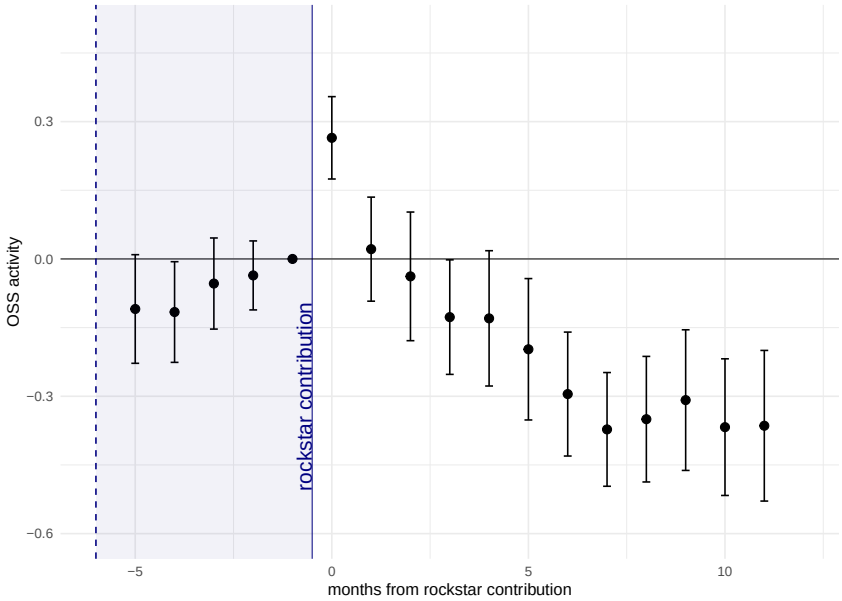
I am interested in the percentage change in project activity with a rockstar contribution, however, the dependent variables, while being count data, contain occasionally zero-valued observations. Therefore, I follow the recent literature on log-like transformations (Chen and Roth, 2024) and implement a Poisson Pseudo Maximum Likelihood (PPML) estimator such that I obtain estimates in levels as a percentage of the baseline mean. This allows me to obtain a treatment effect interpretable in percentage change, though I am not able to distinguish between intensive and extensive margin (Chen and Roth, 2024). Therefore, in Section 3.4.3 I estimate separately the treatment effect on the extensive and intensive margin, as well as account for cohort-specific treatment effects by implementing the Sun and Abraham (2021) estimator.

3.4 Results

3.4.1 Main Effect

Figure 3.1 shows the event study coefficients for project activity before and after the rockstar contribution, excluding the rockstar contribution, specified by the model in Equation 3.1. In the pre-period, besides four months before treatment, there is no statistically significant difference in project activity between projects that receive a rockstar contribution and projects that do not, which is not surprising given the matching algorithm. In the month of the rockstar contribution, there is a statistically significant increase in project activity in projects that receive a rockstar contribution relative to no rockstar projects of 30% conditional on calendar month, project, project age, and programming language × calendar month fixed effects. This is followed by a statistically insignificant effect of 2.16%. In the next two to twelve months the relationship switches sign and effect sizes range between -3.74% to -30.54%, becoming statistically significant. Table C.3 presents estimates for each period. The change in sign is

Figure 3.1: Event study estimates



Notes: Estimates for t_j based on Equation 3.1 with calendar month, project, project age, and programming language x calendar month fixed effects. The outcome is monthly commits to a project, excluding the rockstar commits. The reference month is $t = -1$. Bars show 95% confidence intervals. Robust standard errors in parentheses are reported. *Sources:* GHTorrent, own calculations.

surprising at first but possibly explained by the software development process on *GitHub*. Often changes in project code occur not in the main project directly. Rather, users work on a project copy, i.e. a fork, and, later, when being sure the changes work, they are merged to

the main project (Peterson, 2013; Vasilescu et al., 2014). In Figure C.5 I estimate Equation 3.1 with the combined number of monthly project commits to the main project and its forks. For rockstar projects, I consider only forks created in the rockstar contribution month or up to twelve months later. Including the activity of forks leads to event study coefficients close to zero three months post rockstar contribution. The estimated change in the month of the rockstar contribution increases to a statistically significant 44.82%, likely because the additional activity in forks is included.[12] In the subsequent two months there is now a statistically significant elevated activity for rockstar projects relative to no rockstar projects. Therefore, the decrease in activity in the main project in the post period is plausibly a result of the software development process on *GitHub*, where often users continue to work in the forks of a project and not in the project directly.[13] The positive relationship between rockstar contribution and project activity in the contribution month is consistent with the theory of social learning (Anderson and Magruder, 2012). Other users may learn about the project and project quality after the rockstar contributes to the project, and then start working on the project themselves. The short-lived increase in project activity with the rockstar contribution is similar to sharing patterns on social media platforms, where increases in sharing activity are temporarily clustered around 24 hours (Bakshy et al., 2012). Further, it may reflect the fact, that developers are always on the hunt for new technologies, and thus, potentially quickly move on to the next project (Dabbish et al., 2012).

Motivated by the dynamic pattern in the event study, I estimate the average treatment effect on the treated (ATT) using Equation 3.2. Table 3.2 presents the ATT estimates of this three-period difference-in-differences model. The coefficients now capture the difference in the change in project activity for projects receiving a rockstar contribution and projects without a rockstar contribution relative to the pre-period. Similar to the event study, the effect size for rockstar contribution in column 1 for the total number of monthly project commits is positive and statistically significant with 88.68%, conditional on calendar month and project fixed effects.[14] After including controls for programming language trends by programming language $\times$ calendar month and project age, the coefficient increases to 103.27%. At the start of a project, a project is in the development phase and receives more commits, whereas later on it moves to the maintaining stage, where activity slightly decreases. Therefore, it

[12] See Table C.4 for the event study estimates of the regression with fork activity included.
[13] Alternatively, the decrease may stem from losing some projects, because the rockstar contribution occurred at the end of the observation period. Figure C.6 shows the event study coefficients based on a balanced sample. The estimates are very similar to the main results, suggesting the decrease is not driven by not observing some projects anymore in the post period.
[14] Calculated as $(\exp(\hat{\beta}_2) - 1) \times 100 = (\exp(0.6349) - 1) \times 100$.

Table 3.2: Difference-in-differences model

commits	(1)	(2)	(3)
rockstar contribution	0.6349***	0.6349***	0.7094***
	(0.0738)	(0.0738)	(0.0706)
post	0.0104	0.0104	0.0731
	(0.0732)	(0.0733)	(0.0710)
Month FE	Yes	Yes	Yes
Project FE	Yes	Yes	Yes
Project Age FE		Yes	Yes
Language FE-Month FE			Yes
Projects	4,394	4,394	4,394
Adjusted Pseudo R^2	0.662	0.662	0.694
Observations	154,067	154,067	154,067
Avg. commits	17.96	17.96	17.96

Notes: Results from PPML estimation of Equation 3.2. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

is important to account for this pattern by controlling for the project age at treatment time. Project popularity also depends on the programming language, where trends in programming languages may change the activity in projects of a certain programming language (Borges et al., 2016). Including programming language $\times$ calendar month fixed effects in the regression takes this consideration into account.

The coefficient in column 3 implies that a rockstar contribution to a project is associated with an average increase of about 19 monthly commits in the contribution month relative to projects that do not receive a rockstar contribution. This reflects a sizable increase in activity given the average number of monthly commits is 17.96. In the months after the rockstar contribution, the coefficient is statistically insignificant, implying the short-term decrease observable in the event study analysis returns to the activity level before the rockstar contribution in the long run. The results provide evidence of social learning on *GitHub*. The contribution of a highly followed developer seems a means for other users to learn about new projects and assess project quality. It is associated with a considerable increase in activity,

though short-lived.[15]

Table 3.3: Project popularity

	forks		stars	
	non-follower	follower	non-follower	follower
	(1)	(2)	(3)	(4)
rockstar contribution	0.1587**	0.6318***	0.3008***	1.059***
	(0.0722)	(0.1055)	(0.0723)	(0.0777)
post	0.0102	-0.0300	0.1259*	-0.1449*
	(0.0783)	(0.0865)	(0.0680)	(0.0779)
Month FE	Yes	Yes	Yes	Yes
Project FE	Yes	Yes	Yes	Yes
Project Age FE	Yes	Yes	Yes	Yes
Language FE-Month FE	Yes	Yes	Yes	Yes
Projects	4,394	4,394	4,394	4,394
Adjusted Pseudo R^2	0.909	0.508	0.905	0.760
Observations	154,067	154,067	154,067	154,067

Notes: Results from PPML estimation of Equation 3.2 with the number of stars or forks given to a project. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

By leading the attention of their followers to a project, next to the project activity, also project popularity might increase. To investigate the change in project popularity I estimate Equation 3.2 with the number of stars or forks as the dependent variable. Table 3.3 presents the estimates for project stars or forks given by followers or non-followers. The coefficients for both followers and non-followers in the rockstar contribution month are statistically significant and positive. The estimates for followers are more than twice as large as for the non-followers. For the post period, non-followers continue to give significantly more stars, whereas followers significantly less. The latter potentially shift their attention towards the created project copies. The increase in stars or forks by followers suggests that especially following others on the platform is a way of learning about interesting projects. Forks and

[15] The results are potentially driven by some rockstars, that have a high influence on their followers, whereas other rockstars do not. Therefore, I perform a leave-one-out analysis excluding one rockstar at a time based on Equation 3.2. Figure C.7 plots the resulting coefficients. The estimates are fairly similar among the sample splits, suggesting the results are not driven by one highly influential rockstar.

watchers reflect attention towards the project which potentially turns into activity at a later point in time. Focusing on commits by watchers or fork owners in Table C.5, I find a statistically significant increase in activity at the rockstar contribution time and thereafter. This shows, that a high number of project copies and project bookmarking leads to increased activity after the rockstar contribution as well. For the platform Twitter, there are mixed results on the relationship between sharing content by influential individuals and the subsequent use of the content (Finch et al., 2017; Chan et al., 2023; Branch et al., 2024). On OSS platforms, though, influential individuals' contributions to a project correspond to a sizable increase in attention towards the project, to a large extent driven by their followers.

3.4.2 Heterogeneity and Mechanism

User heterogeneity Motivated by the findings on project popularity, I investigate heterogeneity concerning user characteristics. First, I assess if the increase in project activity stems from the rockstar's followers' change in activity. They get notified about the rockstar's activity, and thus, their attention may be especially directed towards the project. Therefore I split the monthly project commits either created by non-followers or by followers of the contributing rockstar in columns 1 and 2 in Table 3.4, respectively. Again, the coefficient for follower is almost twice as large as for non-follower, both statistically significant as well as the post coefficient for follower.

This implies that both followers and non-followers increase their activity with the rockstar's contribution and not only followers learn about the project by the rockstar. However, it could be the case that the non-followers of the rockstars might be followers of the rockstar's followers. Then a chain reaction, or social multiplier effect (Moretti, 2011), might occur. Therefore, in column 3 I consider the monthly commits by followers of the rockstar's followers, which do not follow the rockstar themselves. The coefficient is slightly smaller than for the rockstar's followers in column 2, however, the post estimate now doubles in size. Combining the commits by rockstar's followers and followers of rockstar's followers, which could simultaneously follow the contributing rockstar as well, the associated increase in the contribution month and thereafter is further amplified and statistically significant in column 4. Presumably, the rockstar's followers learn about the project by the rockstar's contribution and are inclined to work on the project themselves. Then, the followers of the rockstar's followers are notified about their activity in the project, and their attention is additionally drawn towards the project. This supports the hypothesis of a social multiplier effect amplifying the rockstar's attention-leading role. The results further suggest that the signal of contributing to a project by a rockstar is not necessarily limited to the month of the contribution, after taking into

Table 3.4: Heterogeneity by user

commits	(1) non-follower	(2) follower (1st)	(3) follower (2nd)	(4) follower (3rd)
rockstar contribution	0.6935***	1.044***	1.011***	1.176***
	(0.0755)	(0.1104)	(0.1406)	(0.1170)
post	0.0798	0.2340**	0.5979***	0.6147***
	(0.0746)	(0.1165)	(0.1761)	(0.1486)
Month FE	Yes	Yes	Yes	Yes
Project FE	Yes	Yes	Yes	Yes
Project Age FE	Yes	Yes	Yes	Yes
Language FE-Month FE	Yes	Yes	Yes	Yes
Adjusted Pseudo R^2	0.704	0.775	0.769	0.750
Observations	154,067	154,067	154,067	154,067

Notes: Results from PPML estimation of Equation 3.2 with the monthly number of commits to projects by the respective user type. Non-follower refers to commits by users not following the contributing rockstar. Follower (1st) refers to commits by users following the contributing rockstar. Follower (2nd) are followers of the rockstar's followers, that do not follow the rockstar. Followers (3rd) are followers of the rockstar's followers and/or of the rockstar. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

account the links between rockstar followers and other users.

Focusing on this pattern on the daily level, Figure C.8 plots the event study coefficients for the number of daily commits by rockstar followers and followers of rockstar followers, which do not follow the rockstar.[16] The largest increase in daily commits on the rockstar contribution day is for rockstar followers, whereas for followers of rockstar followers, the coefficient is the largest on the next day. This again hints at a social multiplier effect. In comparison to other studies on the social multiplier effect (Glaeser et al., 2003; Moretti, 2011) or the spread of academic knowledge on Twitter (Chan et al., 2023), my setting allows me to clearly identify the links between users and that way study how information spreads among links.

Rockstar heterogeneity Moving on to study heterogeneity with respect to rockstar

[16] In the analysis on the daily level, I limit the sample of rockstar projects to projects with only one rockstar contribution at one day. See Table C.6 for the event study estimates of a regression on the daily number of project commits by rockstar followers and followers of rockstar followers.

characteristics, a first aspect to analyze is the rockstar's number of followers. The most followed rockstars can lead the attention of a much higher number of users towards a project, which might be linked to a larger increase in activity, if followers use the rockstar to assess project quality. In Table 3.5 I include interaction terms between treatment indicators and an indicator if the contributing rockstar has above median followers in column 1. Contributions by an above median followed rockstar are associated with an additional and statistically significant increase in activity in the contribution month. The increase in attention towards a project seems to be multiplied by a highly followed rockstar contributing. To further show evidence of the social multiplier effect in my setting, I include interactions of the treatment with an indicator if the contributing rockstar's followers committing to a project have a large follower base themselves in column 2 of Table 3.5. In the contribution month and thereafter, I find an associated increase in activity, again hinting at a chain reaction of rockstar's followers contributing after the rockstar's activity, which then draws the attention of their followers towards the project.

There may also be heterogeneity by the type of rockstar contribution. After applying a natural language processing approach (Gentzkow et al., 2019), I identify *fix*, *use* or *add* as the most frequently used words in the commit message.[17] The interaction terms for the rockstar commit message containing *use* or *add* in Table 3.5 in columns 3 and 5 are statistically insignificant, suggesting no heterogeneity with respect to these types of rockstar contribution. If a rockstar fixes something, this corresponds to a statistically significant increase in the months thereafter. The rockstar may helped to solve an important problem, that allowed other users to continue to work on the project, which might was not possible before because of a problem. Turning to the contribution quantity, i.e. lines of code changes in Table C.7, I find a statistically significant increase in project activity in the post period for a rockstar deleting code lines, whereas for adding lines, the corresponding increase in the contribution month is smaller. Potentially when adding code, fewer changes can be done by other users, and the opposite applies if the rockstar deletes code. In Table C.8 I study heterogeneity with respect to rockstar affiliation by including interactions for a rockstar affiliation of freelance, academia, or big tech.[18] There is no statistically significant difference in rockstar contribution depending on a rockstar's

[17] Due to changes on the platform, only for 864 out of the 1,913 rockstar projects, I can retrieve the rockstar commit message. Figure C.9 plots a word cloud of the most frequent words used in the rockstar commit message to single rockstar projects. For projects with retrieved rockstar commit messages, the commit message includes in 27.78% projects *fix*, in 27.66% projects *use*, and 27.2% projects *add*.

[18] Freelance refers to stating *freelance* as the affiliation. Academia refers to university affiliations. Specifically, users stating *university, college, institute, universiteit, universidad, or universitat* in their affiliation are assigned to academia. Big tech refers to Google, Amazon, Meta, Apple and Microsoft. For 49.18% of rockstars, no affiliation is available. For the remaining rockstars, I assign 36.42% to big tech, 7.28% to academia, and 2% to freelance.

Table 3.5: Heterogeneity by rockstar

commits	(1) median followers	(2) social multiplier	(3) add commit	(4) fix commit	(5) use commit
rockstar contribution	0.6180***	0.6111***	0.7111***	0.7375***	0.7212***
	(0.0737)	(0.0953)	(0.0674)	(0.0720)	(0.0709)
rock. contr. × indicator	0.5055**	0.2162*	-0.0053	-0.1746	-0.0693
	(0.2107)	(0.1206)	(0.1791)	(0.1484)	(0.1736)
post	0.0066	-0.0916	0.0394	-0.0044	0.0542
	(0.0875)	(0.1165)	(0.0802)	(0.0878)	(0.0814)
post × indicator	0.3667	0.3658**	0.1777	0.2818*	0.0966
	(0.2929)	(0.1754)	(0.1885)	(0.1573)	(0.1882)
Month FE	Yes	Yes	Yes	Yes	Yes
Project FE	Yes	Yes	Yes	Yes	Yes
Project Age FE	Yes	Yes	Yes	Yes	Yes
Language FE-Month FE	Yes	Yes	Yes	Yes	Yes
Adjusted Pseudo R^2	0.694	0.695	0.694	0.694	0.694
Observations	154,067	154,067	154,067	154,067	154,067

Notes: Results from PPML estimation of Equation 3.2 adding an interaction with an indicator variable of interest to study heterogeneous effects. Median followers refer to a rockstar contribution by a rockstar with above median followers, i.e. above 34,813 followers. Social multiplier refers to at least two rockstar followers contributing to the project having more than 100 followers. Add commit refers to a rockstar contribution with a commit message containing *add*. Similarly, fix and use commit refers to a rockstar contribution with a commit message containing *fix* or *use*, respectively. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

affiliation of freelance or big tech. Contributions by a rockstar with an academic affiliation are linked to a statistically significant decrease in activity. These contributions may be to projects on statistical packages and, thus, are of lower interest to the majority of *GitHub* users. Lastly, to explore the relationship between rockstar contribution and attracting new users, I focus on the monthly number of new contributors and the total monthly number of contributors in Table C.9. As expected, both increase with the rockstar contribution.

Project heterogeneity Social learning should matter most in contexts where less information about a good is available (Moretti, 2011; Anderson and Magruder, 2012; Finch et al., 2017). Then, individuals especially learn about the quality of a good through their peers. For projects with ex-ante unclear project quality, the rockstar contribution should, thus, be more important as a signal for other users to learn about the projects' usability. In Table 3.6, I include interactions if a project has no stars and forks, or if the majority of contributions prior to

Table 3.6: Heterogeneity by project

commits	(1) no stars/forks	(2) less skilled users	(3) rockstar's language
rockstar contribution	0.5615***	0.6249***	0.6280***
	(0.0709)	(0.0755)	(0.0934)
rockstar contribution × indicator	0.5563***	0.2191*	0.1411
	(0.1324)	(0.1253)	(0.1196)
post	-0.0242	0.0808	-0.0776
	(0.0827)	(0.0858)	(0.1141)
post × indicator	0.2670**	-0.2008*	0.1887
	(0.1347)	(0.1220)	(0.1534)
Month FE	Yes	Yes	Yes
Language FE-Month FE	Yes	Yes	Yes
Project FE	Yes	Yes	Yes
Project Age FE	Yes	Yes	Yes
Adjusted Pseudo R^2	0.692	0.692	0.692
Observations	156,235	156,235	156,235

Notes: Results from PPML estimation of Equation 3.2 adding an interaction with an indicator variable of interest to study heterogeneous effects. No stars/forks refers to a project having zero stars and zero forks. Less skilled users refers to projects with more than 50% of contributions done by users with less than 100 followers. Rockstar's language refers to project programming language equals rockstar's main programming language. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

the rockstar contribution stems from less followed users in columns 1 and 2, respectively. In both cases, I find statistically significant elevated activity in the contribution month. For projects with no stars and forks, there is also an associated increase in activity after the rockstar contribution. Overall, the results suggest that the rockstar contribution is a stronger quality signal for projects with ex-ante unclear quality. This is in line with the findings of others about content sharing by influential individuals being more important if quality is perceived lower ex-ante (Finch et al., 2017). Lastly, in column 3 in Table 3.5 I study if the rockstar's attention leading role is domain-specific by including an interaction between treatment indicators and an indicator if the project is in the rockstar's main programming language. The interactions are statistically insignificant, suggesting the rockstar's attention-leading role is not limited to one domain.

The variable importance analysis in Section 3.2 revealed that project age is one of the most

predictive variables for a project receiving a rockstar contribution or not. In Table C.10 I split the sample of rockstar projects by the project age quarter in which the rockstar contribution takes place to study how the relationship between rockstar contribution and activity varies between projects in different development stages. For projects in the first age quarter, there is a statistically significant decrease in activity in the post period, which however becomes statistically insignificant after including fork activity.[19] For more mature projects, there is elevated activity in the contribution month and thereafter, however, no clear pattern is observable.

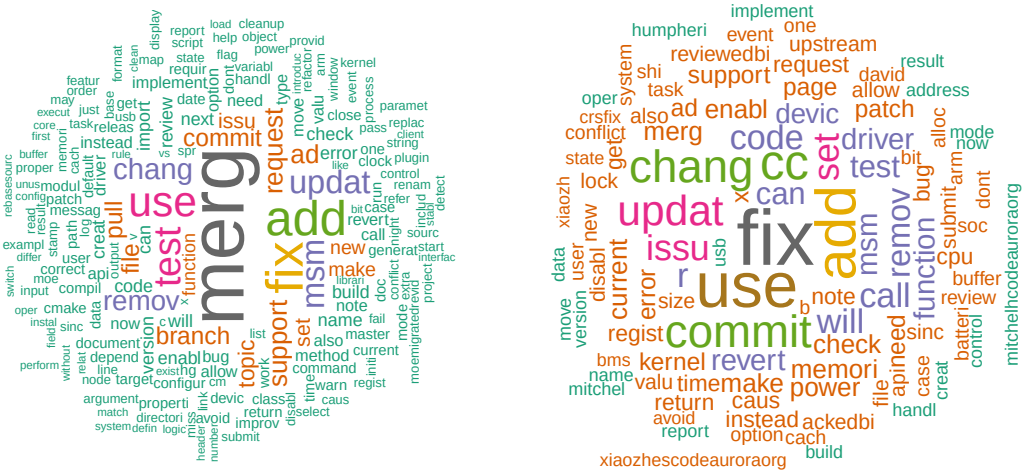
Finally, projects on *GitHub* evolve with the number of commits, as well as with the issues that are opened, updated, and closed. Issues are a way of reporting problems in the software code by other users (Bissyandé et al., 2013), which helps to keep the project up-to-date and work without problems. The rockstar contribution and the presumable increase in attention towards the project may increase the number of reported or changed issues. In the rockstar contribution month, there is a statistically significant increase in the number of updated issues, while simultaneously fewer issues are closed, shown in Table C.12. In the periods thereafter, more issues are opened. When working on the project copies created in the rockstar contribution month, users might find errors in the code, which may result in an increase in new issues reported. The rockstar contribution is, thus, not limited to direct changes in project activity, but also other types of project developments increase with a rockstar contribution. Further, it seems that users do not purely imitate the rockstars but are more likely to learn about the project's usability, and therefore invest time in working, for instance, on the project's issues.

Mechanism The results, for now, suggest that a rockstar contribution corresponds to an increase in project activity and popularity, however, limited to a short period. One possible explanation for this short-lived increase may be, that the rockstar, and by attracting other users, solves all open tasks and brings the project to an end. Column 1 of Table C.13, though, shows that a rockstar contribution is positively related to a longer project duration. Simultaneously, the rockstar contribution also corresponds with a larger number of total commits as well as number of months with active contribution, presented in columns 2 and 3 of Table C.13 respectively. For OSS projects it is actually a bad signal if a project is not actively worked on, i.e. it may not be further maintained and kept up-to-date with technology developments (Coelho et al., 2018). Thus, a project is never really finished, only becoming unmaintained. One indicator for a project possibly being unmaintained is a period of at least one year without

[19] See Table C.11 for the regression by project age quarter including the fork activity.

commits (Khondhu et al., 2013; Mens et al., 2014). In 11.19% of single rockstar projects, this is the case, in no rockstar projects never. Therefore, some rockstar projects may become unmaintained, though the longer project duration for rockstar projects indicates that the rockstar may help sustain an ongoing activity flow.

Figure 3.2: Frequent words in user commit messages



Notes: Word clouds show frequently occurring words in user commit messages to single rockstar projects in the rockstar contribution month. Word size and color represent word frequency in old users (left) and new users (right) commit messages. Frequency limit is set at 150. I remove stop words, the words *signedoffbi* and *changeid*, white space, and use word stems before creating unigrams. *Sources:* GHTorrent, own calculations.

To understand the types of commits users, excluding the rockstar, contribute in the rockstar contribution month, I retrieve for 75% of commits the commit message via the *GitHub* REST API.[20] Figure 3.2 displays word clouds of the frequently occurring word stems in the commit messages of old users, i.e. contributing to the project already prior to the rockstar, and new users, after removing stop words, and white space. Old users slightly merge code changes more often, whereas new users rather fix or add something. This is also reflected by about 95% of new users' commits being code lines added, whereas old users add code lines in 63% of commits.[21] Potentially after the rockstar's code changes, old users to a larger extent merge the changes to the project, and, then continue to work as they did before the rockstar contribution. New users, on the other hand, may shift their activity away, potentially towards

[20] There are limitations on the extent and frequency of queries via the *GitHub* REST API, which leads me to obtain only 75% of commit messages.

[21] In a back-of-the-envelope calculation, following the approach by Hoffmann et al. (2024), the created value in monetary terms of the contributed code lines by users in the rockstar contribution month, excluding the rockstar contribution, would range between 0.59 million USD and 1.69 million USD. This reflects the labor costs for a software developer in India, a low-income country, or in the USA, a high-income country, writing the same amount of code created in the rockstar contribution month.

forks. The short-lived increase, thus, presumably stems from new users fixing and adding code, and code merges of old users.

3.4.3 Robustness

For now, I used the PPML estimator following the recent literature on log-like transformations (Chen and Roth, 2024). In this case, there is no distinction between extensive and intensive margin possible. Additionally, a possible concern could be that the results are driven by the model specification. Therefore I implement alternative models to study extensive and intensive margin separately and test the robustness of my results with respect to model specification.

Table 3.7: Model specification

Model class:	Poisson	OLS	OLS	Logit	Probit
Dependant variable:	count	log	dummy	dummy	dummy
	(1)	(2)	(3)	(4)	(5)
rockstar contribution	0.7094***	0.5770***	0.3125***	22.07***	8.533***
	(0.0706)	(0.0309)	(0.0075)	(0.1074)	(0.0632)
post	0.0731	0.0948***	-0.0584***	-0.3885***	-0.2216***
	(0.0710)	(0.0296)	(0.0104)	(0.0603)	(0.0355)
			6		
Month FE	Yes	Yes	Yes	Yes	Yes
Project FE	Yes	Yes	Yes	Yes	Yes
Project Age FE	Yes	Yes	Yes	Yes	Yes
Language FE-Month FE	Yes	Yes	Yes	Yes	Yes
Adjusted (Pseudo) R^2	0.694	0.482	0.283	0.181	0.18
Observations	154,067	102,377	154,067	139,684	139,684

Notes: Results from estimation of Equation 3.2 for different model classes, outcome transformations, and sample definition. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

To assess the effects on the intensive margin I run an OLS model with log commits as the independent variable excluding zero commits in Table 3.7 in column 2. The coefficients for rockstar contribution and post are both positive and statistically significant. In size the coefficient for rockstar contribution is smaller than in the baseline model, suggesting that the effect on the intensive margin is smaller than the combined effect in the Poisson model.

The positive coefficient for post indicates that the positive relationship between rockstar contribution and project activity remains in the following months on the intensive margin. Turning to the extensive margin, I estimate a linear probability model, as well as a logit and probit model for the probability a project receives non-zero monthly commits in Table 3.7 in columns 3 to 5. In all models, the coefficient for rockstar contribution is positive and statistically significant. The post coefficient is negative and also statistically significant in all specifications. This suggests, that in the contribution month, the rockstar contribution is associated with a higher probability to commit, whereas for the months thereafter, only given a user commits, there is elevated activity observable. Overall, these findings suggest robustness of my results with respect to model specification.

Another concern might be that my results are driven by my rockstar definition. Rockstars are users with a large number of followers (Badashian et al., 2014), but there is no clear cut-off in the literature at which a user becomes a rockstar. In the main analysis, I consider users rockstars that are in the upper 10% of the user-follower distribution given a user has at least one follower. For robustness, I vary the threshold and present estimates based on rockstars in the upper 25% or upper 5% of the user-follower distribution given a user has followers in Table C.14. The estimate for rockstar contribution increases in size the higher the rockstar threshold is, suggesting the more influential the rockstar, the more other users use the rockstar to assess project quality. In sum, the results confirm the positive relationship between rockstar contribution and project activity.

Alternatively, the timing of rockstar contribution may not be random. Potentially, because projects show elevated activity, the rockstar contributes to the project. In that case, the associated increase with the rockstar contribution cannot be related to the rockstar contribution. A randomly assigned placebo treatment in the periods before the rockstar contribution should be statistically insignificant if rockstar projects do not show increased activity prior to the rockstar contribution relative to no rockstar projects. By my activity requirements projects should exist for at least twelve months during which a rockstar contribution could occur. However, this leads me to have a low number of months prior to rockstar contribution to conduct a placebo analysis. Therefore, I implement the placebo analysis on the daily level to have a sufficient number of observations. Figure C.10 shows the event study coefficients before and after a randomly assigned treatment to the rockstar projects prior to the rockstar contribution based on the model in Equation 3.1 with the number of daily project commits as the dependant variable.[22] The coefficients are mostly statistically

[22] See Table C.15 for the respective ATT.

insignificant, suggesting that the rockstar contribution timing is random and not driven by elevated activity prior to treatment.

By applying coarsened exact matching I attempt to have a comparable sample of rockstar and no rockstar projects with respect to project activity and quality characteristics. Still, the projects may not be comparable, or, my results are driven by the matching. In Table C.16 I vary my comparison groups to test if my results are robust with respect to matching. In column 1, I compare all rockstar projects with all no rockstar projects. In column 2, I compare only single rockstar projects with each other, i.e. earlier vs. later treated. In both specifications, the positive relationship between rockstar contribution and project activity is similar to my baseline results, and statistically significant. The post coefficient is now also negative and statistically significant. It seems that the matching leads to an improved comparison group by also capturing the shift in activity towards forks, which these specifications cannot account for.

Following the literature on staggered adoption, I implement the Sun and Abraham (2021) estimator for the analysis of my matched baseline sample, and for only treated rockstar projects in Table 3.8. The coefficients tend to increase in size after accounting for cohort-specific treatment effects, whilst qualitatively remaining the same. My results are, thus, not driven by cohort-specific treatment effects.

In the main analysis, I compare projects that received a single rockstar contribution to projects that did not receive any rockstar contribution. An alternative comparison group to study the dynamics in project activity after a single rockstar contribution may be projects that receive several contributions by a rockstar. It could be the case that if a project receives just one rockstar contribution, it is actually a bad signal for the project. It may be of less interest and therefore the rockstar does not work more often on it. Alternatively, users may contribute for signaling reasons to rockstar projects. They want to work on a project with a rockstar, and by that be noticed by the rockstar, or perceived as more skilled due to working with a rockstar. Table C.17 shows the estimates of a difference-in-differences model where single rockstar projects are compared to projects with several contributions by a single rockstar using Equation 3.2.[23] In the contribution month I find a statistically significant increase in activity for rockstar projects receiving a single contribution relative to several rockstar contribution

[23] Several rockstar contribution projects have a minimum activity of twelve months, more than one month with rockstar contributions and the period between rockstar contributions does not exceed 6 months. This leads to a sample of 633 projects with several rockstar contributions. The median number of rockstar contributions is three.

Table 3.8: Sun & Abraham (2021) estimator

commits	(1) main sample	(2) only treated
rockstar contribution	0.7658***	0.6990***
	(0.0607)	(0.0642)
post	-0.0159	-0.1555
	(0.0877)	(0.0999)
Month FE	Yes	Yes
Project FE	Yes	Yes
Project Age FE	Yes	Yes
Language FE-Month FE	Yes	Yes
Projects	4,394	1,913
Adjusted Pseudo R^2	0.704	0.734
Observations	153,535	109,195

Notes: Results from PPML estimation of Equation 3.2 for every treatment cohort and aggregating the coefficients à la Sun and Abraham (2021). Main sample refers to the matched sample of rockstar and no rockstar projects. Only treated refers to limiting the sample to the single rockstar projects, i.e. comparing earlier vs. later treated. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

projects.[24] This suggests, that for projects a single rockstar contribution occurs, the related increase in activity in the contribution month is larger than for projects that receive several rockstar contributions. The results, thus, contradict with a single rockstar contribution being a bad project quality signal or users contributing due to signaling. If users contribute to rockstar projects for signaling, they should be even more likely to do so in projects a rockstar contributes more often, not less likely.

3.5 Conclusion

In this study, I provide evidence on the importance of social learning for working decisions among software developers on one of the largest OSS platforms, *GitHub*. By comparing

[24] For the projects receiving several rockstar contributions, the indicator of rockstar contribution is one when calendar month is equal to the month the first rockstar contribution occurs.

projects that receive a minimal one-time contribution by a highly followed user, called rockstar, to similar projects that do not receive such a contribution, I find an associated increase in project activity with the rockstar contribution in the contribution month of 30%. The related increase in activity with a rockstar contribution is larger for projects where quality is ex ante less clear, and mainly driven by rockstar followers as well as followers of rockstar followers, hinting at a social multiplier effect. This is also reflected by an increase in the project's popularity with a rockstar contribution, proxied by the number of new stars or forks of a project, to a large extent stemming from rockstar followers. Overall the corresponding increases in activity are short-lived, which relates to some rockstar projects becoming inactive over time. Still, the push in activity and popularity with the rockstar contribution corresponds with a longer project duration, and, thus, an ongoing activity flow.

In sum, my analysis suggests that highly followed peers play an important role in open-source software development. They seem to lead the attention towards projects and may help to identify the high-quality and promising projects to work on. Open-source software is an important input source for firms and, thus, where software developers decide to contribute to, impacts the benefits from integrating open-source software (Hoffmann et al., 2024). OSS platforms build on a decentralized community (Nagle, 2022). However, if there are some individuals on the platforms directing contribution efforts, this could also be problematic. Highly followed users may have interests going against what could be best for the community, e.g. are paid by a firm, and therefore leading the attention mainly towards projects of the respective firm, which do not necessarily have to be the most promising projects. Therefore, on OSS platforms, additional metrics on project quality could be implemented to decrease the influence of a small number of individuals. My results further suggest that in contrast to the mixed findings on sharing content and subsequent paper citations (Branch et al., 2024), influential users can be related to sizable increases in project outcomes.

My findings are limited by several factors. The decision of the rockstar to contribute to a project is likely not random but endogenous to the characteristics of the project or its previous contributors. It should also be noted, that there is no clear definition of rockstars. I provide alternative rockstar definitions, but the results build on my chosen follower cutoffs. Further, I am not able to identify if the projects a rockstar contributes to, are really high-quality projects, even though common metrics like project stars and the number of contributors suggest they are. Lastly, it seems that users tend to shift their activities from the main project towards project copies, i.e. forks, in the short term. This is a result of different development practices (Hindle et al., 2008; Vasilescu et al., 2014). However, it leads me to not observe all activity

changes associated with the rockstar contribution. The analysis suggests, that my results, thus, are lower bound effects of a rockstar contribution on project activity. Future research could address these issues by randomly assigning rockstar contributions to projects, which could help in solving the selection problem (Manski, 1993b). Such settings would allow a clear causal interpretation of the analysis.

4 Platform Partnership Programs and Content Supply: Evidence from the YouTube "Adpocalypse"

Many digital platforms host content produced by independent creators and rely on advertising as their primary source of revenues. To incentivize the supply of high-quality content, platforms often share their advertising revenue through partnership programs, which may also prevent the presence of "bad-faith" actors who could otherwise harm the platforms' integrity. Changes in the eligibility criteria to such partnership programs are likely to affect content supply, yet they are poorly understood. We exploit a rule change on YouTube that made access to its partnership program more restrictive and disabled previous revenue sharing for all creators who did not meet the new requirements. Using a sharp regression discontinuity design, we provide causal evidence that affected creators reduced the frequency of video uploads and provided content of lower quality and diversity. We also investigate and discuss effect heterogeneity between mainstream and niche as well as between more and less experienced creators to learn about their financial and non-pecuniary motivations. Our findings provide novel insights into the effective governance of ad-based platforms using partnership programs.[1]

Keywords: platform governance; partnership programs; content supply; ad-based business models; access restrictions

JEL-No: L84; O18; O30; R32

[1] This chapter is based on joint work with Anna Kerkhof and Johannes Loh. A version of this chapter has been published as CESifo Working Paper No. 10363. We thank Sam Cao, Verena Dorner, Luise Eisfeld, Oliver Falck, Philipp Hukal, Tobias Kretschmer, Tim Meyer, Christian Peukert, and participants of the 6th Digital Economics Workshop in Lausanne, for useful comments and suggestions. The paper has benefited from feedback from conference participants at the 6th Digital Economics Workshop in Lausanne and the SMS annual meeting in London, as well as seminar participants at Ludwig-Maximilians-University, the ifo institute, and BI Norwegian Business School. Anna Kerkhof gratefully acknowledges financial support by the Joachim-Herz-Stiftung. This project is funded by the Bavarian State Ministry of Science and the Arts in the framework of the bidt Graduate Center for Postdocs. Lena Abou El-Komboz acknowledges support by the Deutsche Forschungsgemeinschaft through CRC TRR 190 (project number 280092119).

4.1 Introduction

Recent years saw the emergence of large digital platforms that dominate the delivery of various types of online media, such as software, music, news, and videos (Tiwana, 2013; Waldfogel, 2017; Foerderer et al., 2021). These platforms commonly rely on independent creators who produce the content, which then attracts an audience of end consumers (Claussen et al., 2013; Jain and Qian, 2021). Popular examples include YouTube, which has more than 2 billion monthly users, with creators uploading more than 500 hours of video content per minute.[2] Likewise, on the social media platform Twitter, more than 350 million monthly active users themselves generate content to be consumed by their peers.[3] At the same time, many consumers have come to expect the offered content to be free, which is why ad-based business models are a prevalent source of revenues (Sun and Zhu, 2013). Hence, platforms in this "creator economy" reside over multi-sided ecosystems that connect content creators, consumers, and advertisers (Bhargava, 2022).

Digital platforms must attract and engage consumers to ensure a steady inflow of ad revenue. This means that they must create incentives for creators to produce a steady stream of new content of high quality (Huang et al., 2022). Commonly they therefore share part of their ad revenue with creators (Tang et al., 2012). In addition, platforms can face an inflow of low-quality or "bad faith" actors, which can threaten the health and integrity of their ecosystems (Geva et al., 2019). For instance, the crash of the video game industry in the early 1980s has been attributed to the high number of games with low quality or obscene content (Pursey, 2022), and YouTube has faced intense public backlash and advertiser boycotts in 2017 due to the presence of hate-speech and other problematic content (Statt, 2017). To address such challenges, many content-based platforms have created so-called partnership programs to regulate which creators are able to monetize their content. For example, the streaming platform Twitch requires a minimum amount of content supply, consumer engagement, and conformity with guidelines for creators to become eligible for their basic "Affiliate"[4] and more prestigious "Partner"[5] programs. YouTube applies similar criteria but paired with a degree of manual curation to govern participation in their partner program, which lets creators earn part of the ad revenue.[6] These programs are useful for platforms: They incentivize the creation of

[2] See `https://blog.youtube/press`, accessed on 13 February 2023
[3] See `https://www.statista.com/statistics/303681/twitter-users-worldwide/`, accessed on 13 February 2023
[4] See `https://help.twitch.tv/s/article/joining-the-affiliate-program`, accessed on 13 February 2023
[5] See `https://help.twitch.tv/s/article/partner-program-overview`, accessed on 13 February 2023
[6] See `https://support.google.com/youtube/answer/72851`, accessed on 13 February 2023

high quality content while regulating access to them, which protects advertisers from being associated with "bad faith" actors.

However, regulating access to partnership programs is a non-trivial governance challenge for platform owners: On the one hand, if the eligibility criteria for creators are too restrictive, their effectiveness as an incentive device will be limited. On the other hand, if they are too open, then their control function will be compromised. Therefore, the decision about how selective access should be is a balancing act (Boudreau, 2010). In addition, as a platform evolves over time, the criteria for access may have to be adjusted as well (Rietveld et al., 2020). This, however, can be a disruptive event (Wareham et al., 2014; Jacobides et al., 2018), which can create confusion and uncertainty (Jhaver et al., 2018) and ultimately lead to unanticipated and undesirable reactions by creators (Gawer and Henderson, 2007; Tiwana, 2015b). This issue is exacerbated in the context of the creator economy, in which the supply of content is determined by a heterogeneous mix of financial and non-pecuniary (such as status- and identity-based) sources of motivation (Ma and Agarwal, 2007), which makes reactions to governance attempts hard to predict (Boudreau and Hagiu, 2009).

In this paper, we study how creators on YouTube reacted to a change to the access requirements to its partnership program. After facing intensive public backlash and advertiser boycotts as part of the so-called YouTube "Adpocalypse" (Alexander, 2019), the platform significantly increased the eligibility criteria in an effort to exert more control over who participates. This made it not only harder for new creators to become "YouTube Partners", but it also removed all former program participants who did not meet the new criteria at the time of the rule change. Hence, these creators were not completely shut out of the platform, but they lost their partner status as well as the possibility to monetize their content. To study how this had an effect on their subsequent motivations to create content on the platform, we ask the following research questions: *How did YouTube creators react to losing access to the partnership program in terms of content supply? And how did this reaction vary across different creator types?* In particular, we investigate the impact of the rule change on the amount, quality, and diversity of creator's subsequent content production.

Prior research has studied how regulating access to a platform as such is related to the supply of complementary products (Boudreau, 2010), highlighting an important trade-off in more open systems: They benefit from increased network effects due to a larger number of participants (Eisenmann et al., 2006), which however comes at the expense of the quality and innovativeness of the products on offer (Boudreau, 2012; Parker and Van Alstyne, 2018).

However, we know less about how regulating access to partnership programs affects the *supply* of content. This is distinct because the rules governing access to such programs do not prevent creators' participation in the platform ecosystem altogether. In addition, the questions of how existing creators react to a change in the degree of access control, and what the role of complementor heterogeneity is, are hitherto unanswered. However, these aspects are important determinants of partnership programs' effectiveness in incentivizing the creation of content while protecting a platform's integrity. Moreover, prior research cautions that creator heterogeneity can make it hard to predict their reactions to governance attempts, which can therefore drive unintended reactions to them (Boudreau and Hagiu, 2009; Tiwana, 2015a; Gawer and Henderson, 2007). Still, we know little about the potential sources of heterogeneity, and how they matter for the successful application of ecosystem governance.

Empirically, we leverage a unique dataset about YouTube creators' content supply and estimate the causal effect of losing access to the partnership program within a sharp regression discontinuity design. The new eligibility criteria for program participation constitute a clear threshold in creators' subscriber count at the time of the rule change. We therefore compare the subsequent creation of videos between those who are just below (lost access) and just above (remained in the program) that threshold. In our analysis of German creators, we find deteriorating effects of losing access: affected YouTubers decreased their frequency of video uploads, and their content was both of lower quality and diversity. We also provide evidence for and discuss heterogeneity in these effects between mainstream and niche creators, as well as along their pre-change experience to learn about how the rule change had an impact on different financial and non-pecuniary motivational sources. In particular, our results suggest that the loss of financial incentives is not sufficient in explaining heterogeneity in reactions across these creator types. Instead, we attribute this to non-pecuniary motivations arising from the loss of status (Toubia and Stephen, 2013) or their identity-based attachment (Ren et al., 2007) to the platform.

Our findings have important implications for the successful governance of platform ecosystems when using partnership programs. While primarily aimed at creating financial incentives for creators, our results imply that the acceptance to such programs also comes with additional, non-pecuniary incentives. This, in turn, drives heterogeneity in adverse reactions when program access is denied. Our contribution to this stream is therefore two-fold: First, we highlight partnership programs as a control device for platform owners, as well as some of the factors that can determine their (un-)successful implementation. And second, our results show the risks associated with "one-size-fits-all" governance attempts. Instead, platform

owners need to heed the limitations of governance practices that are applied indiscriminately across the entire ecosystem, and may therefore be unsuccessful in eliciting desirable reactions by heterogenous creators that draw on diverse sources of motivation.

4.2 Related Literature

Regulating access to revenue-sharing systems via partner programs is an aspect of platform governance. This literature investigates the set of rules and design features put into place by platform owners to coordinate and facilitate complementors' (in our case: creators') value creation processes (Boudreau and Hagiu, 2009; Ceccagnoli et al., 2012; Wareham et al., 2014). Specifically, researchers have studied different types of platform governance, such as boundary resources or interface features (such as APIs for app developers) (Ghazawneh and Henfridsson, 2013; Tae et al., 2020), selective promotion of complements (Rietveld et al., 2019), and algorithmic or ranking-based recommendation systems to guide and facilitate interactions between consumers and complementors (e.g. Oestreicher-Singer and Sundararajan, 2012; Kapoor and Agarwal, 2017; Dinerstein et al., 2018). One prevailing point of discussion is the extent to which and how platform owners attempt to exert control over complementors' activities. Prior studies have investigated trade-offs related to how restrictive access to the platform should be (e.g. Boudreau, 2010, 2012; Parker and Van Alstyne, 2018) and have shown that too few restrictions can compromise the quality and integrity of the products and services on offer (Eaton et al., 2015; Geva et al., 2019). Others have investigated how "softer" control measures affect complementor activity, such as certification systems or awards (Huang et al., 2013; Foerderer et al., 2021; Rietveld et al., 2021), sending signals about desirable content (Hukal et al., 2020), or regulating access to boundary resources (Constantinides et al., 2018) or users (Claussen et al., 2013). In the context of the creator economy, ad-based revenue sharing is prevalent to incentivize the supply of high-quality content (Jain and Qian, 2021; Bhargava, 2022).

On the one hand, such governance attempts can be successful in aligning complementors' and platform owners' goals. On the other hand, however, designing and implementing them appropriately is challenging. Because platform governance greatly affects complementors' ability to create and capture value from their activity on a platform, changes to these practices can be highly disruptive (Jhaver et al., 2018; Koo and Eesley, 2020) and entail unintended and detrimental reactions from them (Gawer and Henderson, 2007; Tiwana, 2015b). In addition, two factors can aggravate this challenge. First, complementor heterogeneity complicates the design of governance practices that address a variety in needs and characteristics (Boudreau

and Hagiu, 2009). Second, in particular in the context of creator economy platforms such as YouTube, content production often follows a mix of financial and non-pecuniary sources of motivation (Ma and Agarwal, 2007). This adds complexity to the design of appropriate incentive mechanisms.

Related to the latter, how to stimulate user-generated content (UGC) has received some attention. First, the effectiveness of financial incentives has been studied in the context of online reviews, providing nuanced insights. Cabral and Li (2015) find only a small positive influence on creator activity. Similarly, Burtch et al. (2018) find that they increase activity, but are most effective in combination with other nudges, such as social norms. Others find that the effectiveness of financial rewards is subject to heterogeneity across different creator types (Sun et al., 2017), and that they entail the crowding out of reviews from more intrinsically motivated creators (Khern-am-nuai et al., 2018). In the context of the creator economy, studies find that sharing ad revenue with creators can incentivize content supply on YouTube (Tang et al., 2012) and lead to the production of higher quality and more mainstream content on a Chinese blogging website (Sun and Zhu, 2013). Second, a range of studies shows the importance of intrinsic or reputation-based motivations (e.g. Ma and Agarwal, 2007; Tang et al., 2012; Toubia and Stephen, 2013). Many UGC-based platforms therefore use awards or community badges to motivate activity. Prior research largely found positive effects, with reputation-based nudges stimulating activity on StackOverflow (Anderson et al., 2013) and increasing newcomer retention on Wikipedia (Gallus, 2017). At the same time, Burtch et al. (2022) – while finding a positive effect on activity – also document that (peer) awards tend to decrease content novelty, thus demonstrating a potential downside in the context of the generation of creative content. Goes et al. (2016) highlight an additional limitation in the context of milestone-based incentive hierarchies: Consistent with motivation stemming from the pursuit of a goal (Locke and Latham, 2002), they find that milestones initially stimulate activity, but that motivations decrease immediately after the successful accomplishment.

These studies document both a range of attempts at governing platform ecosystems as well as stimulating user-generated content via financial and reputation-based nudges. Still, our understanding of these phenomena remains limited, with two particular areas lacking investigation: First, we know little about how partnership programs incentivize the creation of content. While they primarily aim at creating financial incentives, it is unclear if this is how they are perceived by creators. This is a non-trivial issue, given their multi-layered mix of financial and non-pecuniary sources of motivation. Second, we have a limited understanding of the (potentially adverse) reactions to governance attempts from heterogeneous complementors.

However, especially in the context of UGC, creators exhibit a high degree of heterogeneity in terms of their experience, type of content they produce (for example, niche or mainstream), and what motivates their activities (financial vs. non-pecuniary sources). Both areas are important for the effective regulation of access to platform partnership programs as well as the implementation of changing governance practices in the creator economy more generally.

4.3 Background and research framework

4.3.1 Empirical background

We study how video creators on YouTube reacted to a change to the eligibility criteria for its partnership program, which removed the ability to earn ad revenue for some of them. YouTube is the world's largest video sharing platform, and – as of February 2023 – the second-largest website in terms of overall traffic behind Google.[7] YouTube has more than 2 billion monthly users, and more than 500 hours of user-generated videos are uploaded every minute.[8] These videos constitute YouTube's supply of complements; they are created by registered users which are commonly referred to and self-identify as "YouTubers" (Kerkhof, 2024). Creators upload videos to their own "channels", which can be subscribed to by viewers, who will subsequently become informed about the release of new content. The videos as such, however, can be viewed by anyone for free. While also offering paid premium memberships for viewers[9], YouTube's main source of revenue is advertisements that are played before and during videos.

YouTube partner program

Creators have the option to earn money with their content by participating in the YouTube partner program (YPP).[10] This program mainly serves two purposes. For the platform, it provides a means of quality control. Its ads only run with videos uploaded by creators who are part of this program[11] to ensure that they are not shown alongside inappropriate content. In addition, creators cannot freely join the program, but they have to fulfill certain criteria – which are the subject of this study – before they are eligible to apply for membership. As part of the application process, they then undergo a (partly automated, partly manual) review process to

[7] See `https://www.statista.com/statistics/1201880/most-visited-websites-worldwide/`, accessed on 13 February 2023

[8] See `https://blog.youtube/press`, accessed on 13 February 2023

[9] See `https://www.youtube.com/premium`, accessed on 13 February 2023

[10] See `https://creatoracademy.youtube.com/page/lesson/ypp_what-is-ypp_video`, accessed on 13 February 2023

[11] Following a change in the platform's policy in late 2020, YouTube now also holds the option to run ads with videos outside the program as well (Koetsier, 2020). This, however, has no implications for the present study as it falls outside our sample period.

ensure that their videos follow the platform's guidelines. In turn, for the creator, it provides a means to monetize their videos as YouTube shares part of the generated revenue. However, while creators have agency about *whether or not* and *how many* ads may be shown during a video, the platform determines *which* ads are actually shown via an algorithm (Kerkhof, 2024). Accordingly, advertisers and creators have no way of directly interacting with one another. In terms of the attractiveness of the YPP, anecdotal evidence – official statistics do not exist – suggests that creators can earn about three to five USD per 1,000 video views.[12] As a result, YouTube relies on two tiers of creators: First, creators in the YPP are vetted and they can earn money by allowing the platform to run ads before and during their videos. Second, creators outside the program cannot earn money, and no advertisements are shown with their videos. In addition, through the eligibility criteria and application process the platform limits the access to the program and actively controls who can move from the latter to the former tier.

The eligibility criteria for the YPP changed several times since its launch in 2007. While having been quite selective in the beginning, YouTube opened it up in 2012 by removing virtually all access barriers (YouTube Official Blog, 2012). However, this entailed an inflow of "bad actors", threatening the platform's integrity. In response, in early 2017, it put into place the restriction that creators have to have accumulated a minimum of 10,000 lifetime views before being able to apply (Popper, 2017). In this study, we analyze the subsequent rule change put into place in early 2018, which instituted an additional and much more significant increase in the eligibility criteria.

YouTube "Adpocalypse" and the Rule Change in 2018

We study a change to the eligibility criteria for the YPP that occurred in February 2018. This change was preceded by YouTube facing considerable backlash and, ultimately, a large-scale boycott by its advertisers (Nicas, 2017) – commonly referred to as the YouTube "Adpocalypse" (Alexander, 2018). Despite existing access requirements, advertisements routinely appeared alongside hate speech as well as racist and anti-Semitic content. In addition, YouTube faced criticism more broadly for its lack of restrictions as to what type of content is allowed on the platform (Statt, 2017). The situation was then further exacerbated by scandals surrounding two of the platform's most prominent creators (Gillespie, 2018), entailing further scrutiny. As part of this "Adpocalypse" and in reaction, the platform announced more manual curation of creators allowed to the YPP in December 2017 as an effort to ensure that advertisements are only shown alongside unproblematic content (Wojcicki, 2017). In January 2018, the platform

[12] See `https://influencermarketinghub.com/how-much-do-Youtubers-make/`, accessed on 13 February 2023

went on to reveal new criteria determining the eligibility to apply to the program, which would take effect one month later, in February 2018 (Mohan and Kyncl, 2018).

This rule change contained two important elements: First, it updated the eligibility criteria. Now, to be able to apply for the YPP, creators had to have accumulated a minimum of 1,000 subscribers and 4,000 hours of "watchtime" over the preceding twelve months. The latter is calculated by multiplying the times videos are viewed with the amount of time each viewer actually spends with the videos. Together, this made access considerably more restrictive compared to the previous requirement of 10,000 lifetime views.[13] Second, those creators who had been part of the YPP, but did not meet the new criteria would be excluded from the program, effectively making it impossible for them to earn money on the platform. In the blog post announcing these changes, it was also noted that – while affecting a "significant number of channels" – the financial ramifications of this "demonetization" would be mild as "99 % of those affected were making less than 100 [USD] per year in the last year, with 90 % earning less than 2.50 [USD] in the last month" (Mohan and Kyncl, 2018).[14] In addition, it was announced that those creators who lost access would get the possibility to reapply once they met the new criteria. Importantly, YouTubers did not undergo permanent surveillance. That is, once they met the new criteria and became (re-)eligible for the YPP, they did not lose access to the program again even if their number of subscribers or their watchtime dropped below the respective thresholds again.

Still, YouTube faced negative reactions following this announcement, primarily from creators who were affected by the rule change (Alexander, 2018). Points of criticism are diverse. Some creators worried about the loss of earnings, while others felt treated unfairly, that is, they were not primarily concerned about the financial ramifications, but rather felt disappointed to lose the platform's endorsement after having been part of it for a long time. Others still considered it a sign that YouTube generally shifted its focus from smaller creators to larger, more prominent ones, regarding it as an indication that "the golden age of YouTube is over" (Alexander, 2019).

The rule change presents a well-suited opportunity to study how a platform's increased use of control mechanisms affects the supply of content. First, the primary reason for the platform to increase access requirements has been to make more manual curation feasible. In light

[13] For reference, a view is counted when a video is watched for at least 30 seconds. Accordingly, before the change, the requirement of 10,000 views translates to a minimum of 83,33 hours of watchtime ($\frac{10{,}000}{60\times60}$), without any further restriction on the time span across which a creator may attain this goal.
[14] However, it is important to note that the median number of subscribers on YouTube is approximately 1,000 (Kerkhof, 2024). Therefore, the new policy affected about half of the creator population.

of the sheer amount of videos that are uploaded to the platform, limiting the number of participating creators has arguably been necessary. Second, the rule change occurred in reaction to advertiser boycotts, which in turn had been sparked by problematic content, such as hate-speech by many small channels, but also controversies surrounding popular creators. However, *all* creators who did not meet the new requirements lost access to the platform. In other words, the majority did not become demonetized due to their content being problematic, but simply because they did not meet the new criteria. We therefore study creators who were not the *cause* of the rule change, but who were very much *affected* by it.

4.3.2 Research framework

In our study, we are interested in how those creators who lost access to the partnership program reacted to the rule change in terms of their activity levels as well as the quality of the content they produce. Both are the result of the effort they put into their activities on the platform. Not only is this relevant for the overall appeal of the platform as a whole, but is also indicative of how creators' motivation to produce content changed due to the rule change. In addition, to provide more insights into the nature of the underlying motivations, we explore heterogeneity in their reaction along with their experience and the type of content (mainstream or niche) they produced before the rule change. In this section, we lay out the mechanisms tying the rule change to the outcomes of our study. We provide details about our empirical strategy in section 4.4.2.

Creator activity and content quality

When the rule change was communicated via the YouTube blog, it was not only announced that ineligible participants will be removed from the partner program, but they were also explicitly encouraged to reapply once they met the new requirements. This bears opposing implications for the expected reactions from affected creators. On the one hand, the encouragement to reapply could have acted as a motivator. The new access requirements present a clear goal for creators to attain, which can spark an increased provision of effort to grow their channel to the necessary size and engagement. Consistent with this idea, existing research provides evidence that the pursuit of such goals can induce activity (Locke and Latham, 2002). Similar to our empirical context, Goes et al. (2016) have shown that volunteer contributors in a knowledge-exchange platform increased their effort to reach a higher "rank" in the community's status hierarchy, which they attribute to goal attainment incentives. Hence, it is possible that former participants of the YouTube partner program were subject to similar motivations after losing access, which implies an *increase* in effort provision afterward.

On the other hand, however, there are several reasons to expect that affected creators were less motivated after losing access to the program. First, they lost the ability to monetize their content on the platform. Even if their earnings were low before the rule change, this removed any financial incentives that may have been a driving force behind their content production. Second, many creators are not only, or even mainly, driven by financial concerns. Rather, they follow non-pecuniary motivations, such as building a reputation or attaining status among their peers and audience (Toubia and Stephen, 2013), or their own "intrinsic" enjoyment of creating content (Shah, 2006). Losing access to the partner program deteriorates these motivational sources as well: Attaining the status of "YouTube partner" benefits their reputation and lets them form an attachment with the community and the platform, both of which are subsequently lost due to the rule change. In addition, the actions of the platform owner can send a signal about desirable content (Hukal et al., 2020), and the loss of the partner status may then be perceived as the withdrawal of its endorsement (Ho and Rai, 2017). Together, the rule change, therefore, deteriorated both financial and these non-pecuniary incentives, which implies a *decrease* in effort provision afterward.[15]

In the end, it is an empirical question of which of the two mechanisms dominates in our empirical setting. If the ability to reapply after losing access is a sufficient motivator, we would expect a net increase in both activity and content quality following the rule change. However, if the deterioration of financial and non-pecuniary incentives is too strong, we would expect a negative net effect.

Creator heterogeneity: Content positioning and experience

The reaction to the rule change is unlikely to be uniform across all creators. They are subject to heterogeneous needs and characteristics (Boudreau and Hagiu, 2009) and a complex mix of financial and non-pecuniary sources of motivation (Ma and Agarwal, 2007). To add more insights into how the rule change affected their motivation to exert effort, we, therefore, investigate differences in reactions between different types of creators.

Content positioning First, we explore creator differences in the type of content they produced prior to the rule change. In particular, we differentiate between mainstream and niche creators. That is, before the rule change, some had mainly produced content that

[15] In principle, a third possibility exists: If creators are purely intrinsically motivated – that is not driven by, say, financial or reputational concerns – they may not react to the rule change at all. In our empirical analysis, we would then not find significant effects.

covered highly popular topics and themes on the platform, while others positioned themselves in narrower content niches. This is a meaningful distinction because the ex-ante positioning on the platform provides an indication of creators' predominant sources of motivation. If they are primarily financially motivated, this creates the incentive to position themselves within popular segments in an effort to reach a larger audience, which maximizes the ad revenue they can attain (Wilbur, 2008). In contrast, covering a narrow niche with lower audience appeal likely indicates that creators are more intrinsically motivated – they may draw enjoyment from producing content they themselves find appealing. As a result, we expect the effect of the rule change to be stronger for mainstream than niche creators. Because the former is more reliant on financial incentives (and potentially other, related reputational benefits), they are more severely affected by the loss of the partnership status.

Creator experience Second, we explore the role of creators' experience on the platform. They vary both in terms of their tenure on the platform as well as the amount of content they had produced prior to the rule change. This can impact how their motivation is affected through two mechanisms. First, creators often form a bond with the communities of their peers and viewers, which creates a common identity attached to the platform (Ren et al., 2007). As a case in point, many creators in our setting explicitly self-identify as "YouTubers". More experienced creators are more likely to have developed such a bond, and they may place a higher value on their status as "YouTube partner". Second, as they have produced more content on the platform, receiving a negative signal about its desirability and support from the platform should have a stronger impact on their subsequent motivation. Therefore, while the direction of the effect is a priori unclear, the reactions to the rule change should be stronger for more than less experienced creators.

4.4 Data and Methods

4.4.1 Data Set

To analyze how the supply of videos changed for creators who lost access to the YPP after the rule change, we combine information from two waves of data collection via the YouTube Data API. First, we use the same snapshot as Kerkhof (2024) who has obtained information about all active German YouTube channels as of December 2017 – that is just before the rule change –, including whether or not they have participated in the YPP at that point in time. This piece of information is unique and crucial to our analysis, as it is impossible to assess historical

information on a creator's program participation otherwise. From that snapshot, we select all creators who were part of the YPP and had between 500 and 5,000 subscribers by the end of 2017[16], that is whom we consider "at risk" of losing access to the program.

For the second wave, we accessed the YouTube Data API from September to November 2020 to obtain a snapshot containing updated information for the selected sample of creators, which lets us track their upload history since January 2018. Combining the two snapshots provides us with crucial information for the construction of our regression samples and key variables. Specifically, we obtained cross-sectional information at the creator level, such as their subscriber count in December 2017 (first snapshot) and November 2020 (second snapshot) and their total number of videos. In addition, we collected information at the video level, such as the number of views, likes, dislikes, duration, date of upload, keywords, and the video category. This lets us track each creator's video uploads over time and provides us with information about the extent of their activity (e.g. upload frequency), as well as if and how their content strategy has changed over time.

One shortcoming is that a creator's watchtime over the past twelve months is not directly provided via the API. However, this measure is crucial for our analysis as it is part of the new eligibility criteria instituted by the rule change under study. Since the measure is not publicly available, we compute it ourselves using the length of all videos that a creator has uploaded in the twelve months before the rule change and the number of views that these videos have accumulated. However, viewers often do not watch the entire video. For example, Maggi et al. (2018) find that the least popular videos are watched to about 50% on average, and the most popular ones are watched to about 75% on average. Since our sample consists of relatively unknown creators with small channels, we conservatively presume in our main analysis that 50% of each video is watched. We then simply multiply each video's duration by 0.5 and subsequently take the sum over all video views in the twelve months before the rule change in February 2018. Note that not all selected creators appear in the second snapshot. This is the case if they exited the platform between December 2017 and November 2020. We further discuss the characteristics of creators who exited the platform below.

[16] The distribution of subscribers over creators is heavily skewed; in other words, there are many more creators with few than creators with many subscribers (the median number of subscribers in Kerkhof (2024) is around 1,000). Thus, to ensure that we have a sufficient number of observations in our main analysis, we decided to use a relatively large initial bandwidth regarding the upper subscriber bound.

4.4.2 Empirical Framework

Identification Strategy

We want to estimate the causal effect of losing access to the partner program on subsequent creator activity. Since losing access is determined by creators' watchtime and their subscriber counts at the time of the rule change, we face the challenge of separating the effect from unobserved creator characteristics that may otherwise drive our estimates. That is, those who are better able to produce engaging content will have more subscribers and more watchtime (hence they do not lose access to the program), and they may be less inclined to change their video supply. In other words, successful YouTubers who maintain access to the program are unlikely to provide a valid counterfactual to those who are demonetized after the rule change. As a result, a naive comparison of the evolution of content provision before and after the rule change between YouTubers who maintain access to the program to those who lose it may yield biased results.

We tackle this challenge by implementing a sharp regression discontinuity design (RDD). The clearly defined subscriber and watchtime thresholds provide us with a quasi-experimental setting that allows us to estimate the *average treatment effect at the cutoffs*: Creators just above and just below the cutoffs are likely to be similar in terms of quality, success, and other (unobserved) characteristics. Moreover, they are unable to precisely manipulate their watchtime and subscriber counts, whereby it is as good as random whether they (just) maintain access to the program or not. Thus, if we focus on creators within a narrow bandwidth around the cutoffs, the content evolution of creators who are just above the cutoff provides a valid counterfactual to the content evolution of creators just below. In other words, our sharp RDD examines comparable creators who differ in terms of being just above or just below the YPP eligibility thresholds, such that we can attribute differences in behavior after the rule change to losing or retaining access to it.

Sharp RD designs are known to provide more credible causal inference than other identification strategies such as difference-in-differences and IV estimation (Lee and Lemieux, 2010). In particular, we not do have to *assume* that the treatment variation is as-good-as-random; rather, the treatment variation is a *consequence* of creators' inability to precisely control their watchtime and subscriber counts (Lee, 2008). Thus, our empirical setting can be analyzed and tested like a randomized experiment, whereby our main estimate corresponds to the causal average treatment effect at the cutoff.

Specific for our setting is that there are two running variables determining treatment status:

subscriber count and watchtime. In our main analysis, we focus on the subscriber threshold for two reasons. First, the definition of watchtime is not always clear to creators, and the metric is not as salient as their subscriber count.[17] As a consequence, we do not expect that being just below the watchtime threshold after the rule change has an equivalently large impact on creators' behavior than being just below the subscriber threshold. Second, in contrast to the subscriber count, we can only approximate watchtime and measure it with noise.[18] Hence, we consider creators just above and just below the subscriber threshold to identify the causal effect of losing access to the YPP and – as proposed by Papay et al. (2010) – include the second running variable, watchtime, as a control. Our key identifying assumption is that the two groups are comparable within a reasonably narrow bandwidth and that subsequent differences in behavior can be attributed to the rule change (Lee and Lemieux, 2010; Imbens and Lemieux, 2008). Specifically, for our main analysis, we select creators who have at least 900, but no more than 1,200 subscribers (we provide robustness checks using both a narrower and wider bandwidth in section 4.5.2). As there are more creators just below the 1,000 subscriber threshold than above, this selection ensures that we analyze comparable creators that are sufficiently close to the threshold, while also maintaining a reasonable sample size that is balanced between treatment and control groups. We can then obtain an unbiased estimate of the effect of losing access by estimating the following equation:

$$Y_i = \alpha + f(Subscribers_i) + \beta \cdot LostAccess_i + \gamma \cdot Watchtime_i + \epsilon_i, \tag{4.1}$$

where Y_i is the outcome of interest, $LostAccess_i$ is a dummy variable indicating whether creator i lost access to the YPP due to an insufficient number of subscribers, and $Subscribers_i$ is our running variable, the subscriber count. $Watchtime_i$ controls for creators' watchtime in the twelve months before the rule change. The coefficient of interest in Equation 4.1 is β, which gives us the local average treatment effect (LATE), that is, the effect of losing access to the YPP, for each regression (Imbens and Angrist, 1994).

[17] See `https://www.qqtube.com/article/youtube-retention-vs-watch-time`, accessed on 13 February 2023).

[18] We would like to emphasize that misclassification of YouTubers on either side of the threshold is impossible. Specifically, YouTubers who do not meet the 1,000 subscriber threshold are definitively demonetized, regardless of their watchtime hours. Thus, misclassification below the threshold is not possible. However, it could be that YouTubers who surpass the 1,000 subscriber threshold fail to meet the 4,000 watchtime hours criterion. This scenario would result in some YouTubers in the control group being incorrectly classified as treated. Consequently, the estimated differences between the treatment and control groups would be smaller than they actually are, leading to conservative estimates. In other words, if misclassification occurs, our estimates would be understated.

The function $f(\cdot)$ captures the underlying relationship between the subscriber count and our outcomes of interest; in particular, we implement a local linear regression approach by letting $f(\cdot)$ be a linear function of subscribers.[19] In addition, we let the slopes of our fitted lines differ on each side of the subscriber threshold by interacting $f(\cdot)$ with $LostAccess_i$ to control for differential trends in $Subscribers_i$. Following Calonico et al. (2020) we use a triangular Kernel function which assigns zero weight to all observations outside of our specified bandwidth, and positive weights to all observations within our bandwidth. The weight is maximized at the threshold and declines symmetrically and linearly going away from the threshold.

Finally, a potential threat to identification is that creators very close to the threshold may manipulate their subscriber count to remain in the YPP, known as bunching. Therefore, we implement a "donut" by excluding creators with a subscriber count between 990 and 1,010 or watchtime between 3,950 and 4,050 hours. In addition, we perform a test for continuity in the subscriber count distribution around the threshold in section 4.4.4.

We use the specification described in Equation 4.1 for the entirety of our analysis, but use different outcomes of interests to investigate different aspects of creator behavior. For each, we perform the analysis both before and after the rule change. Differences in behavior between treated and untreated should only exist after the rule change, but not before. Hence, if the coefficient of interest β is insignificant in the before period, but significant after, we can attribute it to the rule change.[20]

Outcome variables

We consider two main outcome variables: creator activity and the quality of their content. We measure creator activity in terms of their upload frequency, i.e., the average number of monthly video uploads in the six months before and after the rule change. To measure video quality, we leverage information about viewer engagement in terms of "likes" and "dislikes" videos receive. Specifically, we compute the share of likes over all viewer reactions,

[19] We provide robustness checks using quadratic and cubic model fits in section 4.5.2

[20] We are not concerned about potential violations of the Stable Unit Treatment Variance Assumption (SUTVA). YouTubers in the control group were unaffected by the policy change, so we do not anticipate any spillover effects from the treatment group to the control. As for the treatment group, we have two reasons for not being concerned. Firstly, to mitigate any anticipation effects, we designate the month of the announcement (Dec '17) rather than the month of policy implementation (Feb '18) as the treatment date. Secondly, while we cannot definitively confirm whether any YouTubers in the treatment group regained access to the Partnership Program within our timeframe, we do not see this as problematic. Even if they did regain access, they were unquestionably impacted by the policy change, and their response is captured by the average treatment effect at the threshold. Moreover, if these YouTubers resumed their previous behavior – similar to the control group – it would only skew our estimates toward zero, making them overly conservative.

$\frac{\text{Likes}}{\text{Likes+Dislikes}}$, for each video. Based on that, we derive a creator's average quality of content in the six months before and after the rule change.[21]

Finding appropriate measures for video quality is inherently challenging. While there are a few objective measures, such as visual and sound quality, that most viewers can agree upon, many other aspects are subjective and closely tied to the video content itself. Hence, we perceive "likes" and "dislikes" as net measures that reflect all the factors influencing a viewer's evaluation of a video, including its visuals, sound, and careful selection of popular topics and content. Kerkhof (2024) supports this view, demonstrating through an online survey that "likes" and "dislikes" are valid indicators of video quality.

Creator heterogeneity

We study creator heterogeneity along two dimensions: First, we distinguish between mainstream and niche creators. To that end, we adopt the mainstream measure from Kerkhof (2024), which is based on videos' keywords. These are illustrative terms that creators assign to their videos to let YouTube know what the video is about (see Kerkhof (2024) for extensive discussion). For example, a funny cat video might be given the keywords "funny", "cat", and "pet". Specifically, for each month and video category, keywords are ranked by how many video views they attract, that is by their popularity. The upper one percent of keywords in this distribution is then classified as "mainstream". Based on that, we classify videos that exhibit at least one such keyword as "mainstream content". Finally, we compute a creator's proportion of "mainstream content" in the six months before and after the rule change. Since we find that the vast majority of creators exclusively uploads mainstream content, we classify a creator as "niche" if her share of mainstream videos is smaller than one, and as "mainstream" otherwise.

Second, we consider creator experience. To measure this, we use the date of a creator's first video upload and compute the age of her channel in terms of how many months she has been active on the platform until the rule change. As a robustness check, we also measure creator experience in terms of the number of video uploads before the rule change.

4.4.3 Summary Statistics

Table 4.1 shows summary statistics for our main estimation sample of 484 creators who fall within the subscriber count bandwidth that we use (900 to 1,200) and who remained active on the platform after the rule change. The majority (80 %) are below the threshold and lost

[21] Section 4.5.2 shows that our results are robust to using alternative time windows for all three outcome variables.

access to the program. This is the case despite our asynchronous bandwidth selection and shows the "long tail" distribution in subscriber count – as is the case in most media markets, most content creators are relatively unsuccessful (Anderson, 2004). The average creator had 1,026.43 subscribers at the time of the rule change and accumulated about 5,307.68 hours of watchtime in the twelve months before February 2018. Within the subsequent six months, the average creator uploaded an average of 3.248 videos per month, using a monthly average of 41.62 unique keywords, and received an average monthly like share of 91 %. The high share of likes is likely due to rating inflation, which is common on digital platforms (Zervas et al., 2021). Moreover, the average creator uploaded a total of 128.88 videos to her channel before the rule change was implemented. With 60 % the majority of creators in our main sample are classified as mainstream. This is not surprising, as – by definition – this is the most popular and prevalent type of content on the platform. Finally, the average creator has been active for about 35.57 months – that is, for nearly three years – before the rule change.

Crucially, out of the total number of creators that appeared in the first wave of data collection, roughly 46 % had exited the platform by the time of the second wave. Although it is interesting to compare the characteristics of creators who have and have not exited the platform between the two snapshots in 2017 and 2020, we must interpret any differences with care. In particular, it is not clear that these creators exited as a result of the rule change or because they were less successful than the creators who remained active. We show differences between exiting and non-exiting creators in Table D.1. In fact, we observe that creators who exited had even more subscribers and watchtime on average than creators who stayed; similarly, they had on average a higher like share and a smaller proportion lost access to the YPP. However, given that the creators whom we consider are most likely to operate their channel as a hobby in their free time, it is more plausible to assume that many of them simply lost time or interest in participating on the platform. As our main objective is to study creator behavior *after* the rule change, we focus on creators who stayed. In addition, we find differences between the groups in their upload frequency and the use of unique keywords. To account for potential selection into remaining on the platform, we estimate all regression with a Heckman two-stage procedure as a robustness check. In each case, the resulting inverse Mills Ratio is statistically insignificant and our main estimates remain unchanged, indicating that selection issues are no concern.

Table 4.1: Summary statistics

	Mean	Std. Dev.	Min.	Max.	N
Lost Access	0.80	0.40	0	1.00	484
Subscriber count	1026.43	75.48	901	1196.00	484
Watchtime	5307.68	11845.07	3	98600.34	484
Upload frequency	3.248	5.708	0.14	68.29	428
Like share	0.91	0.11	0	1.00	425
Unique keywords	41.62	44.81	1	480.00	428
Lifetime video uploads	128.88	131.69	1	616.00	484
Mainstream	0.60	0.49	0	1.00	484
Age	35.57	17.19	1	107.00	484

Notes: The summary statistics are based on our main sample of creators who did not exit the platform after the rule change.

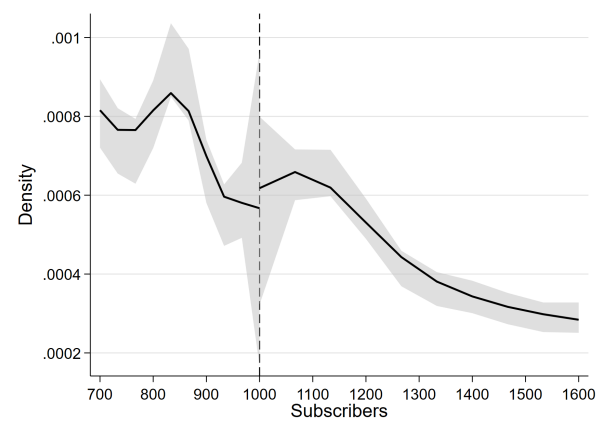
4.4.4 Test for Quasi-Random Assignment

The main identifying assumption of our empirical approach is that losing access to the partnership program is as good as random within the specified subscriber count bandwidth (see e.g. Flammer, 2015). In other words, we assume that creators just above and just below the 1,000 subscribers threshold are similar in all observed and unobserved characteristics except continued access to the YPP after February 2018. We perform two tests for the validity of this assumption.

First, we show that the distribution of subscriber counts is continuous around the threshold. If we would detect a discontinuity at the threshold, this would indicate that assignment to the treatment is in fact not as good as random. For instance, it may be that creators who had been below the threshold before the rule change show efforts to increase their subscriber count to not lose access to the partner program by the time of the rule change. Following Cattaneo et al. (2017), we conduct an automatic manipulation test which does not reject the null of continuity around the threshold ($p = 0.99$; Figure 4.1 visualizes the test). Thus, we find no evidence of a violation of the continuity assumption in our sample.

Second, we check if creators just above and just below the threshold show differences in their behavior even before the rule change. In this case, any observed differences in our main outcomes after the rule change may not be a result of lost access to the YPP, but rather of underlying differences between the two groups of creators. To test this, we estimate Equation 4.1 exclusively based on observations from the six months before the rule change.

Figure 4.1: Test for continuity at the subscriber threshold



Notes: Sample contains creators who did not exit the platform between the first and second waves of data collection.

Table D.2 shows the results. We do not find any statistically significant differences in behavior between creators just above and just below the subscriber threshold before the rule change. Hence, it is plausible to assume that any differences in our outcome variable that we observe afterward can be attributed to losing access to the YPP.

4.5 Results

4.5.1 Main Analysis

Column 1 of Table 4.2 demonstrates that creators who lost access to the partner program uploaded significantly fewer videos to the platform compared to those that did not lose access ($\beta = -2.816$, $p < 0.05$). Specifically, they uploaded 2.816 fewer videos, which – considering the sample mean of 3.248 – is a sizable effect. We show the accompanying RDD plot in Figure D.1a. These provide evidence that the rule change indeed had a deteriorating effect on affected creators' incentives to provide effort in producing content for the platform. Further, it shows that this negative effect, on average, outweighs a potential motivational nudge that arises from creators attempting to regain access to the program.

Next, column 1 of Table 4.3 shows that there is significantly lower content quality among affected creators than those who remained in the partner program ($\beta = -0.048$, $p < 0.1$). Again, this is visualized in Figure D.1b. Specifically, the average like share is 4.8 percentage points lower for the affected creators. While this may not seem much considering the relatively high sample mean of 91 %, we do have to consider that ratings on YouTube are concentrated

Table 4.2: Main Results: Creator activity

	Upload Frequency				
				Experience	
	All	Mainstream	Niche	High	Low
	(1)	(2)	(3)	(4)	(5)
Lost Access	−2.816**	−3.617*	−1.278	−3.82*	−0.841
	(1.387)	(2.098)	(1.427)	(2.038)	(1.559)
Bandwidth	[900,1200]	[900,1200]	[900,1200]	[900,1200]	[900,1200]
Observations	428	253	175	241	187

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime. Sample contains creators who did not exit the platform between the first and second waves of data collection.

in the higher value ranges. In light of this, our estimate suggests a meaningful difference in content quality between treated and untreated creators after the rule change. In addition, this result adds further evidence that the change deteriorated incentives to create content on the platform, which therefore manifests in both subsequent lower activity and quality.

In a next step, we investigate potential sources of heterogeneity in the effect of losing access to the program. First, we investigate differences in the reaction to the rule change between mainstream and niche creators. Results for activity are reported in columns 2 and 3 of Table 4.2, in which we split the sample between the two types. We only find a statistically significant effect for mainstream creators. In addition, the estimated coefficient is considerably larger than for niche creators. These results suggest that taking away extrinsic benefits has a stronger effect on those creators who originally positioned themselves in the most popular segments. In addition, niche creators do not seem to be deterred by losing access to the program. A possible explanation is that their intrinsic motivation had been the primary driver of their activity all along and that the rule change did not affect this.

In columns 2 and 3 of Table 4.3, we again split the sample between mainstream and niche, but this time using average like shares as the dependent variable. We again only find a statistically significant effect for mainstream (column 2, $\beta = -0.061$, $p < 0.05$), but not niche creators (column 3, $\beta = -0.034$, $p > 0.1$). This finding produces interesting insights about underlying motivations: Even after losing access to the partner program, niche creators are unwilling to

Table 4.3: Main Results: Content quality

	Like Share				
				Experience	
	All	Mainstream	Niche	High	Low
	(1)	(2)	(3)	(4)	(5)
Lost Access	−0.048*	−0.061**	−0.034	−0.048	−0.045*
	(0.025)	(0.030)	(0.042)	(0.035)	(0.026)
Bandwidth	[900,1200]	[900,1200]	[900,1200]	[900,1200]	[900,1200]
Observations	425	251	174	240	185

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime. Sample contains creators who did not exit the platform between the first and second waves of data collection.

compromise the quality of their content. A likely explanation is that they are indeed driven by intrinsic motivation and seek to produce content they themselves enjoy, even after the platform withdrew its endorsement.

In a final string of analyses, we are interested in potential heterogeneous effects along the dimension of pre-change experience. Similar to before, we therefore perform a series of sample splits at the median of the experience distribution. We explore heterogeneity in the activity effect in columns 4 and 5 of Table 4.2. We find that only more experienced creators (column 4, $\beta = -3.82$, $p < 0.1$) reduce the frequency at which they upload videos to the platform after the rule change. In contrast, we find no evidence for an effect for less experienced (column 5, $\beta = -0.841$, $p > 0.1$). This suggests that the activity effect is completely driven by more experienced creators, and provides evidence for the relevance of non-pecuniary sources of motivation. We argued that, while financial motivations are unlikely to be a function of prior efforts, more experienced creators had formed a deeper identity-based attachment to the platform. In addition, they had spent a longer time providing effort in the past, which should make a signal of lost valuation and endorsement from the platform all the more impactful.

In columns 4 and 5 of Table 4.3, we only find a significant negative effect on content quality for less experienced creators (column 5, $\beta = -0.045$, $p < 0.1$). However, the size of the estimated coefficient is similar to the estimate for more experienced creators (column 4, $\beta = -0.048$,

$p > 0.1$). While this suggests that both types tended to decrease the quality of their content, we do not find evidence for differences in effect sizes between them.

In all, our results suggest that both effects are completely driven by more experienced creators, who likely formed a stronger identity as "YouTuber" and who had put a lot of effort into producing content before the rule change. Therefore, withdrawn platform support and the potential reputation decline weigh more heavily for them than for less experienced creators.

4.5.2 Additional analyses and robustness

Content diversity

As an additional test, we analyze how the rule change affected creators' content strategy, and in particular its diversity. That is, we study how many different topics, subject areas, or genres they cover on their channel. This is likely shaped by the incentives put into place by the platform owner. Consistent with this notion, some prior research cautions an inefficient duplication of mainstream content when creators receive a share of the ad revenue (Anderson and Gabszewicz, 2006; Wilbur, 2008; Sun and Zhu, 2013), unless the competitive pressure is too great in this segment (Kerkhof, 2024).[22] Similarly, we posit that pursuing a content strategy of greater diversity also reflects an effort to increase the potential audience: covering a greater range of topics bears the potential to match a broader range of consumer tastes. To measure content diversity, we compute a creator's average monthly number of unique keywords in the six months before and after the rule change. A larger number of unique keywords then indicates more diversity in a creator's coverage of different topics, which in turn shows increased experimentation with different types of content, or attempts to appeal to a broader audience that exhibits a wider range of tastes for horizontal content attributes.[23]

Column 1 of Table 4.4 shows that creators who lost access to the partner program exhibit a lower subsequent degree of content diversity compared to those who remained ($\beta = -25.38$, $p < 0.05$). We also show the RDD plot in Figure D.1c. On average, affected creators use 25.38 fewer unique keywords per month to describe their videos. Again, considering the sample mean of 41.62, this implies a sizeable reduction in the range of topics creators cover on their channels. In addition, this indicates that deteriorating incentives reduce creators' attempts to satisfy a broad range of viewer tastes to maximize their audience. Instead, they adjust their

[22] We also investigate creators' subsequent tendency to produce mainstream content as a piece of additional analysis, but do not find an effect.
[23] Similar to our test in Table D.2, we find no statistical differences between treated and untreated YouTubers before the rule change.

strategy towards a more focused approach, perhaps because they now follow more intrinsic sources of motivation. In other words, being "free" of financial or reputational concerns, the rule change enabled them to produce more content they themselves enjoy. Column 7 of Table 6 shows that this change is not driven by lower upload frequency; rather, creators reduce the average number of unique keywords per video.

Columns 2 and 3 of Table 4.4 use again a sample split to investigate differences between mainstream and niche creators' use of unique keywords to describe their content. We find that niche creators exhibit a greater reduction in diversity than mainstream creators after the rule change. The former uses an average of 33.294, and the latter of 19.803 fewer keywords to describe their content after the rule change. At the same time, the coefficient for niche creators is estimated at reduced precision and therefore just statistically insignificant ($p = 0.12$). This is likely the result of the smaller sample size in column 3. While both types reduce the diversity of their content, niche creators do so to a larger extent.

Table 4.4: Content diversity

	Unique Keywords				
				Experience	
	All	Mainstream	Niche	High	Low
	(1)	(2)	(3)	(4)	(5)
Lost Access	−25.380**	−19.803*	−33.294	−34.082**	−8.393
	(10.485)	(10.393)	(21.493)	(15.873)	(12.517)
Bandwidth	[900,1200]	[900,1200]	[900,1200]	[900,1200]	[900,1200]
Observations	428	253	175	241	187

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime. Sample contains creators who did not exit the platform between the first and second waves of data collection.

Together with Section 4.5.1, our findings paint a nuanced pattern in the heterogeneous reactions between mainstream and niche creators, which we believe to be consistent with intrinsic motivations. Compared to mainstream, niche creators are unwilling to compromise the quality of their content. But at the same time, they greatly reduce the scope of topics they cover. Together, this suggests that they rather hone into the areas they are personally excited about, which motivates them to keep quality up. In contrast, we find patterns that are

consistent with deteriorating extrinsic motivations for mainstream creators, who reduce their activity, content quality, and – albeit to a lower extent – content diversity.

Lastly, we study heterogeneity in terms of experience. While more experienced creators show a sizeable and statistically significant reduction in the diversity of their content (column 4, $\beta = -34.082$, $p < 0.05$), we do not find an effect for less experienced creators (column 5, $\beta = -8.393$, $p > 0.1$). Again, this demonstrates the importance of non-pecuniary motivations, which in this case drive more experienced creators to reduce the scope of the topics they cover.

Toxic content

So far, we have analyzed the negative impact of the rule change on content supply motivation. However, it is equally important to evaluate the rule's effectiveness in reducing toxic content from the platform's perspective. Identifying and quantifying toxic content is challenging due to the ambiguity and subjectivity in defining what constitutes toxic content. Additionally, it is impractical for us, as researchers, to screen video content on a large scale.

To provide suggestive evidence of the rule change's impact on reducing toxicity, we conducted a sentiment analysis of video titles. We cleaned all video titles of stopwords and then calculated each title's sentiment score using SentiWS, a dictionary that assigns sentiment scores ranging from -1 (very negative) to +1 (very positive) to more than 30,000 German words.[24] Based on these scores, we computed the average sentiment score for each creator before and after the policy change. Our hypothesis is that more positive wording in titles indicates a lower prevalence of toxic content; for example, a more positive title likely corresponds to less controversial content.

Figure D.2 shows that the average sentiment score of creators' video titles has increased overall, particularly among those who were demonetized. In other words, toxicity decreased. The left panel of Figure D.2 indicates that the average sentiment score before the policy change for those who were later demonetized was around 0.01, both including and excluding creators who left after the policy change. After the policy change, the average sentiment score increased approximately fourfold to about 0.04. For all creators within our bandwidth, taking a more global platform perspective, we also find that the average sentiment score increased by about 50%, from around 0.02 to 0.03.

[24] SentiWS is publicly available. Its most recent version can be downloaded here: `https://wortschatz.uni-leipzig.de/de/download` (Accessed: June 2024).

Robustness Checks

Bandwidth selection We run a series of regressions to test the robustness of our results to alternative modeling and variable choices. First, we use a bandwidth of 900 to 1,200 subscribers in our running variable. There is a trade-off in choosing this range: The wider the bandwidth, the less comparable observations in the treated and untreated groups become. But the narrower, the smaller the sample becomes, which may entail reduced statistical power. Our bandwidth is carefully chosen to hit the sweet spot between including a sufficient number of observations while maintaining their comparability. Here, we test the sensitivity of our results to different ranges in Table D.3. We use a wider range in columns 1, 3, and 5, and a narrower range in columns 2, 4, and 6. For the latter, as expected, standard errors increase due to the smaller sample, yielding statistically insignificant estimates for all outcomes. However, coefficient sizes are very similar to our main specification, which shows that the sample composition does not seem to be affected much. For the former, the coefficient sizes become considerably smaller compared to our main specification, and – with the exception of columns 5 – they become statistically insignificant. This is unsurprising, as the creators we are considering now are less comparable than before, resulting in increased noise in our estimations. Still, they show the same sign as in our main specification. Hence, while we do not believe that these tests are a serious cause for concern about the validity of our modeling choice – especially given our small sample sizes –, we do determine that the precision of our estimates declines with different bandwidth choices.

In Tables D.4 to D.6, we examine the sensitivity of our results to the choice of bandwidth even more closely. Specifically, for each outcome of interest, we incrementally increase both the upper and lower bandwidths while keeping the other constant. Table D.4 demonstrates that our main results for creator activity, measured by upload frequency, remain consistent in magnitude. With a step-wise increase in the upper bandwidth, which enlarges the control group, all estimates remain highly statistically significant at the 5%- or 1%-level. When increasing the size of the treatment group by incrementally lowering the lower bandwidth, our estimates become slightly larger, but the standard errors also increase. Despite this, all estimates, except for those in column 5, remain statistically significant at the 10%-level.

In Table D.5 and Table 4.4, we perform similar analyses for content quality, measured by like share, and content diversity, measured by the number of unique keywords per month. Consistent with our main results, the estimates remain similar in magnitude and are mostly statistically significant at the 5% or 1% level. Therefore, despite an increase in standard errors in some cases due to alternative bandwidth selections, we conclude that our main results are

robust to different bandwidth selections overall.

Placebo regressions To further support the validity of our empirical strategy, we conducted two placebo regressions. In the first placebo regression, we consider YouTubers within a bandwidth of 600 to 850 subscribers – i.e., only demonetized YouTubers – and use the 700-subscriber threshold as fake treatment. YouTubers below the fake threshold are classified as "treated", and YouTubers above the threshold as "not treated". Then, we re-run our main analyses on this alternative sample of YouTubers. If our main empirical strategy is valid and does not just pick up differences between YouTubers with more or less subscribers, the treatment indicator from the placebo regression should be small and not statistically significant. Table D.7 shows that this is indeed the case for all outcome variables.

Analogously, the treatment indicator should be small and statistically insignificant when we consider YouTubers within a bandwidth of 1,150 to 1,600 subscribers – i.e., only unaffected YouTubers – and use the 1,300-subscriber threshold as fake treatment. Table D.8 confirms that this is the case for all outcome variables.

Higher order polynomials Second, we use local linear regressions in our main analysis, following recommendations from the literature (Gelman and Imbens, 2019). Still, we test the robustness of our results to using higher order polynomials, which we report in Table D.9. Specifically, we fit $f(Subscribers_i)$ in Equation 4.1 with quadratic terms in columns 1, 3, and 5, and we use cubic terms in columns 2, 4, and 6. Across the board, standard errors are increased, rendering the results statistically insignificant. At the same time, however, coefficient sizes tend to be larger than in our main specification, and they continue showing the same sign. Therefore, we consider results qualitatively robust, which is also supported visually by an RDD plot of the quadratic model fit, which we show in Figure D.3 (the cubic model fit is also consistent). In addition, the reduced precision of the estimates can likely be attributed to the increased statistical power required to estimate more complex regression models with reduced sample sizes (see e.g. Gelman, 2018). In all, we are therefore not concerned about the validity of our choice of model fit, but we do determine that the small sample size imposes certain limitations in terms of statistical power and the precision of our estimates.

Alternative time frames Third, our outcomes of interest are calculated based on creator behavior in the six months after the rule change in our main specification. Here, we test the

robustness to using alternative time frames of three and twelve months. Results are reported in Table D.10. They are largely consistent with our main specification. For creator activity (columns 1 and 2) and content diversity (columns 5 and 6), effect sizes are slightly larger when using a shorter window, and slightly smaller when using a longer window. This may suggest that reactions in these dimensions manifest in the relatively short run after the rule change. However, the pattern is reversed for content quality. Here, the coefficient is slightly larger than in our main specification when using a larger time window, and smaller and statistically insignificant when using a shorter one. This may suggest that adjustments to quality only unfold over time. Together, however, these patterns do not cause concerns about our choices in measuring our outcomes.

Alternative experience measure Moreover, we measure creator experience in terms of their time spent on the platform prior to the rule change. Here, we use an alternative measure in their number of videos they had uploaded at that time. The results are reported in Table D.11. They are qualitatively consistent with our main specification. Still, two differences are of note: First, the difference between more and less experienced creators becomes more pronounced in terms of how their content quality changes (models 3 and 4). Second, in contrast to our main specification, we do find a negative effect on content diversity for less experienced creators here. However, this effect is still considerably smaller than for the more experienced. In all, our main results are therefore robust to this alternative measure.

Alternative watchtime calculation Finally, as discussed in Section 4.4.1, a direct measure of channel watchtime is not publicly available. Consequently, we adopt a common approach in the literature by estimating watchtime as the product of video views and 50% of the video's duration. To ensure the robustness of our findings against different watchtime estimations, we also calculated the watchtime variable using 30%, 40%, 60%, and 70% of video duration. These alternative metrics were used as control variables in our analysis. As demonstrated in Table D.12 to Table D.14, our results remain consistent regardless of the watchtime definition employed.

4.6 Discussion and Conclusion

We study how an increase in the eligibility criteria to the YouTube partner program affected the supply of videos from those creators who lost access to it as a result. Such partner programs

are a useful governance tool for platforms: They create incentives to supply high-quality, while also letting them exert some control over creators' activities to prevent "bad faith" and low-quality content. In our setting, following widespread criticism and advertiser boycotts, YouTube significantly increased the criteria to be eligible for its partner program, which made it harder for new creators to participate. In addition, it removed all former participants who did not meet the new requirements. We empirically analyze their reaction to this rule change and specifically investigate how this had an effect on their activity and the quality of the content they create on the platform.

We use a regression discontinuity design to estimate the causal effect of losing access to the program on subsequent creator behavior. The new criteria provide us with a clear threshold in their subscriber count at the time of the rule change. Hence, we compare the supply of videos between those who were just below that threshold to those just above. In our empirical analysis of German creators, we find that those who lost access to the partner program significantly reduced the frequency at which they uploaded new videos to the platform. In addition, the quality and their content decreased. Together, these effects speak to a deterioration of motivation and effort due to the rule change. We also explore effect heterogeneity and find that mainstream creators showed a stronger negative reaction than niche creators. We attribute this to the relative importance of extrinsic and intrinsic motivations: Because niche creators enjoy producing content they themselves like, they are unwilling to compromise its amount and quality. This is further evidenced by our additional analysis of content diversity, which shows that they also focus on a more narrow range of topics after the change. In contrast, because mainstream creators are likely to be relatively more driven by financial and reputational concerns (given their ex-ante positioning in popular segments), the deterioration of these extrinsic motivators causes their adverse reaction to the rule change. And finally, we find that only more, and not less experienced creators show a reaction to losing access to the partner program. Because financial considerations are unlikely to be a function of experience, we attribute this to identity- and attachment-related motivations. More experienced creators had spent more effort on the platform in the past, and they had the opportunity to form a connection to the platform. Hence, they are more likely to identify as "YouTubers". Then, losing the partner status and receiving a signal that the content is not valued by the platform explains the negative reaction only by more, but not less experienced creators.

Contributions Our findings contain contributions to two streams of literature. First, we provide novel insights about changes to a platform's governance strategy, and shifts towards more control in particular. Prior studies show that exerting control is necessary to secure a set of high-quality and innovative complements (Boudreau, 2012; Parker and Van Alstyne, 2018) or to avoid uncontrolled creativity (Geva et al., 2019) and threats to a platform's integrity (Eaton et al., 2015). We add to this discussion by highlighting partnership programs as one governance tool platforms can use to exert more control over complementors' activities, without shutting them out of the platform completely. In particular, we show how an increase in control (by making access to the program more restrictive) can have detrimental effects on the motivation of creators to subsequently exert effort in supporting the platform. Further, recent studies note that it is hard to predict complementor reactions to rule changes more generally (e.g. Jhaver et al., 2018; Koo and Eesley, 2020), due to heterogeneous needs and characteristics (Boudreau and Hagiu, 2009). Not only is this confirmed by our results, but we also clarify the role of complementors' experience and positioning in the creator space drive heterogeneous effects. Together, our contribution to this stream is therefore twofold: We provide insights about how platform partnership programs affect multifaceted content supply motivations, which are important determinants of the effectiveness of such governance attempts. Further, while previous studies have considered ecosystem-level compositional implications of restricting access to the platform, we provide evidence about complementor-level effects of *changing* levels of platform control. Hence, we add novel insights to the discussion around unintended consequences of governance attempts (Tiwana, 2015b; Gawer and Henderson, 2007). Finally, we provide evidence on how complementor-level heterogeneity in their content strategy and experience determines their reaction to governance attempts, which drives effective management of platform ecosystems.

Second, we contribute to the literature on ad-based platform business models that rely on user-generated content for value creation. Prior studies have shown how sharing part of the revenues with creators affects their supply of content (Sun and Zhu, 2013; Tang et al., 2012). We contribute to this discussion in two ways. While prior studies have focused on the aggregate effects of financial incentives, we discuss and provide evidence for the role of creator heterogeneity in determining the effectiveness of revenue-sharing schemes. In doing so, we shed light on how complex financial and non-pecuniary (such as status- or identity-based) sources of motivation drive the effectiveness of these schemes. In addition, we study the effects of losing, rather than gaining, access to these sources of motivation. In particular, under unclear and diverse sources of motivations, this is an important distinction. For example, while financial incentives may be regained after they have been once lost, it is not clear that creators

are able to recover status benefits in a similar way. Hence, the heterogeneous reactions we uncover imply that the financial loss is not a sufficient explanation.

Limitations and future research Finally, our study contains several limitations and point towards future research opportunities. First, we study a small subset of creators in a large platform ecosystem. To identify causal effects of losing access to the YouTube partner program we focus on those who fall within a narrow range of subscriber counts. In addition, these creators are relatively small and unlikely to (individually) contribute a lot of value to the platform. Therefore, they may not be representative of the entire ecosystem. For instance, larger, highly successful creators may enjoy a stronger bargaining position vis-à-vis the platform, eliciting a different reaction to a rule change similar to the one we study. Therefore, future research could look at heterogeneity along the dimension of creator size or success. Second, we are unable to clearly separate financial and non-pecuniary incentives in our study. Instead, we approximate this distinction by investigating heterogeneous reactions between different creator types. Hence, future research could make this distinction explicitly, perhaps in the form of an experiment. Lastly, lessons learned from studying YouTube may not perfectly translate to other empirical contexts. For one, the ecosystem offers both professional and user-generated content. Hence, reactions to increased control may differ in other platform settings, such as app stores, video game consoles, or e-commerce. Moreover, YouTube enjoys a near-monopolistic market position, which may allow it to implement more restrictive rules without many repercussions. Creator reactions could therefore be even stronger in the case of platforms that face rivals, which could provide attractive outside options for them. Finally, the creator economy offers alternative ways to monetize content, most prominently via crowdfunding platforms such as Patreon. This may further limit the effectiveness of financial incentives as a governance tool. Hence, future research should investigate changes to platform control mechanisms in more diverse settings.

Appendices

A Supplementary Materials to Chapter 1

A.1 Tables

Table A.1: Summary statistics

Variable	var	mean	median	min.	max.
Activity					
Commits per user	691,964,175.97	2,298	1,059	25	3,767,493.00
Commit per user per Snapshot	16,341,488.33	230	56	1	1,299,828.00
Technology per user per snapshot	1.23	3	3	1	5.00
Technology per user	1.27	2	2	1	5.00
Programming language per user per snapshot	4.19	7	7	1	18.00
Programming language per user	7.58	3	3	1	17.00
Projects					
Users per project	27.62	2	1	1	2,381.00
Commits per project per snapshot	1.23	19	3	1	1,298,112.00
Stars per project	1,301,008.47	85	0	0	259,118.00
Forks per project	72,701.27	11	0	0	145,997.00
Project age [years]	6.10	5	5	0	13.36
Own project	0.24	0	0	0	1.00
Business share	0.14	1	1	0	1.00
Weekend share	0.09	0	0	0	1.00
Out of hour share	0.11	0	0	0	1.00
Local share	0.05	1	1	0	1.00
Clusters					
Technology per city	0.88	5	5	1	5.00
Technology per city per snapshot	1.84	4	4	1	5.00
Programming language per city	22.05	14	17	1	18.00
Programming language per city per snapshot	27.15	10	11	1	18.00

Table A.2: Top 10 clusters by technology

City	cluster size
Technology 1	
San Jose-San Francisco-Oakland, CA	0.10638
New York-Newark-Bridgeport, NY-NJ-CT-PA	0.08784
Seattle-Tacoma-Olympia, WA	0.05366
Los Angeles-Long Beach-Riverside, CA	0.04403
Indianapolis-Anderson-Columbus, IN	0.04387
Toronto	0.03732
Washington-Baltimore-Northern Virginia, DC-MD-VA-WV	0.03484
Boston-Worcester-Manchester, MA-NH	0.03233
Chicago-Naperville-Michigan City, IL-IN-WI	0.03200
Dallas-Fort Worth, TX	0.02390
Technology 2	
San Jose-San Francisco-Oakland, CA	0.13441
New York-Newark-Bridgeport, NY-NJ-CT-PA	0.09031
Seattle-Tacoma-Olympia, WA	0.05627
Boston-Worcester-Manchester, MA-NH	0.04299
Los Angeles-Long Beach-Riverside, CA	0.04095
Washington-Baltimore-Northern Virginia, DC-MD-VA-WV	0.04039
Toronto	0.03375
Indianapolis-Anderson-Columbus, IN	0.03250
Chicago-Naperville-Michigan City, IL-IN-WI	0.03073
Denver-Aurora-Boulder, CO	0.02385
Technology 3	
San Jose-San Francisco-Oakland, CA	0.14154
New York-Newark-Bridgeport, NY-NJ-CT-PA	0.11862
Seattle-Tacoma-Olympia, WA	0.04749
Chicago-Naperville-Michigan City, IL-IN-WI	0.04181
Los Angeles-Long Beach-Riverside, CA	0.04000
Washington-Baltimore-Northern Virginia, DC-MD-VA-WV	0.03808
Boston-Worcester-Manchester, MA-NH	0.03748
Denver-Aurora-Boulder, CO	0.03744
Toronto	0.03028
Indianapolis-Anderson-Columbus, IN	0.02649
Technology 4	
San Jose-San Francisco-Oakland, CA	0.13405
New York-Newark-Bridgeport, NY-NJ-CT-PA	0.08113
Indianapolis-Anderson-Columbus, IN	0.05250
Seattle-Tacoma-Olympia, WA	0.05092
Los Angeles-Long Beach-Riverside, CA	0.04059
Toronto	0.03623
Washington-Baltimore-Northern Virginia, DC-MD-VA-WV	0.03382
Boston-Worcester-Manchester, MA-NH	0.03259
Chicago-Naperville-Michigan City, IL-IN-WI	0.03208
Dallas-Fort Worth, TX	0.02942
Technology 5	
San Jose-San Francisco-Oakland, CA	0.13050
New York-Newark-Bridgeport, NY-NJ-CT-PA	0.06623
Seattle-Tacoma-Olympia, WA	0.05999
Los Angeles-Long Beach-Riverside, CA	0.04048
Indianapolis-Anderson-Columbus, IN	0.03647
Boston-Worcester-Manchester, MA-NH	0.03572
Washington-Baltimore-Northern Virginia, DC-MD-VA-WV	0.03342
Toronto	0.02954
Dallas-Fort Worth, TX	0.02884
Dayton-Springfield-Greenville, OH	0.02735

Table A.3: User sample

Dep. var.: Commits [log]	(1)	(2)	(3)	(4)	(5)
Min. time intervals with consecutive activity:	1	2	3	4	5
Cluster size [log]	0.1989*	0.1986*	0.1894*	0.1911*	0.2104**
	(0.1082)	(0.1076)	(0.1029)	(0.1018)	(0.1005)
Users	243,443	229,140	124,808	81,459	55,458
Observations	6,363,687	6,320,343	5,295,433	4,636,989	4,053,452
Adj. R^2	0.173	0.180	0.227	0.249	0.261
Dep. var.: Commits [log]	(6)	(7)	(8)	(9)	(10)
Min. time intervals with consecutive activity:	6	7	8	9	10
Cluster size [log]	0.2209**	0.2175**	0.2432**	0.2524**	0.2775**
	(0.1027)	(0.1043)	(0.1081)	(0.1127)	(0.1255)
Users	45,007	38,157	31,669	27,011	21,116
Adjusted R^2	0.268	0.273	0.278	0.284	0.292
Observations	3,722,638	3,470,489	3,197,671	2,961,254	2,527,496

Notes: Robust standard errors clustered at the city × technology level in parenthesis. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table A.4: Cluster size (number of users)

Dep. var.: Commits [log]	(1)	(2)	(3)	(4)	(5)	(6)
Cluster size [absolute, log]	-0.2367***	0.1070	0.0929	0.1966**	0.1935**	0.2777**
	(0.0607)	(0.0785)	(0.0744)	(0.0949)	(0.0962)	(0.1253)
Fixed-effects						
User	Yes	Yes	Yes	Yes	Yes	Yes
Project	Yes	Yes	Yes	Yes	Yes	Yes
Technology	Yes	Yes	Yes	Yes	Yes	Yes
Language	Yes	Yes	Yes	Yes	Yes	Yes
City	Yes	Yes	Yes	Yes	Yes	Yes
Time	Yes	Yes	Yes	Yes	Yes	Yes
Technology × time		Yes	Yes	Yes	Yes	Yes
Language × time			Yes	Yes	Yes	Yes
City × technology				Yes	Yes	Yes
City × language					Yes	Yes
City × time						Yes
Users	21,116	21,116	21,116	21,116	21,116	21,116
Adjusted R^2	0.288	0.289	0.290	0.291	0.291	0.292
Observations	2,527,496	2,527,496	2,527,496	2,527,496	2,527,496	2,527,496

Notes: Language refers to programming language. Robust standard errors clustered at the city × technology level in parenthesis. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table A.5: Robustness (excluding largest projects)

Dep. var.: Commits [log]	(1)	(2)	(3)	(4)	(5)	(6)
Cluster size [log]	0.1006	0.0927	0.0787	0.1625*	0.1586*	0.2773**
	(0.1002)	(0.0674)	(0.0636)	(0.0941)	(0.0954)	(0.1168)
Fixed effects						
User	Yes	Yes	Yes	Yes	Yes	Yes
Project	Yes	Yes	Yes	Yes	Yes	Yes
Technology	Yes	Yes	Yes	Yes	Yes	Yes
Language	Yes	Yes	Yes	Yes	Yes	Yes
City	Yes	Yes	Yes	Yes	Yes	Yes
Time	Yes	Yes	Yes	Yes	Yes	Yes
Technology × time		Yes	Yes	Yes	Yes	Yes
Language × time			Yes	Yes	Yes	Yes
City × technology				Yes	Yes	Yes
City × language					Yes	Yes
City × time						Yes
Users	20,905	20,905	20,905	20,905	20,905	20,905
Adjusted R^2	0.256	0.258	0.260	0.260	0.260	0.261
Observations	2,382,259	2,382,259	2,382,259	2,382,259	2,382,259	2,382,259

Notes: Language refers to programming language. Robust standard errors clustered at the city × technology level in parenthesis. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table A.6: Robustness (excluding most active users)

Dep. var.: Commits [log]	(1)	(2)	(3)	(4)	(5)	(6)
Cluster size [log]	0.0841***	0.0797**	0.0672**	0.1353**	0.1341**	0.1724**
	(0.0321)	(0.0325)	(0.0321)	(0.0595)	(0.0596)	(0.0798)
Fixed effects						
User	Yes	Yes	Yes	Yes	Yes	Yes
Project	Yes	Yes	Yes	Yes	Yes	Yes
Technology	Yes	Yes	Yes	Yes	Yes	Yes
Language	Yes	Yes	Yes	Yes	Yes	Yes
City	Yes	Yes	Yes	Yes	Yes	Yes
Time	Yes	Yes	Yes	Yes	Yes	Yes
Technology × time		Yes	Yes	Yes	Yes	Yes
Language × time			Yes	Yes	Yes	Yes
City × technology				Yes	Yes	Yes
City × language					Yes	Yes
City × time						Yes
Users	20,905	20,905	20,905	20,905	20,905	20,905
Adjusted R^2	0.283	0.284	0.285	0.285	0.285	0.286
Observations	2,277,873	2,277,873	2,277,873	2,277,873	2,277,873	2,277,873

Notes: The 1% most active users (476 users) are excluded. Language refers to programming language. Robust standard errors clustered at the city × technology level in parenthesis. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$.
Sources: GHTorrent, own calculations.

Table A.7: Robustness (excluding largest clusters)

Dep. var.: Commits [log]	(1)	(2)	(3)	(4)	(5)	(6)
Cluster size [log]	0.1239	0.1133	0.0978	0.2006**	0.1973**	0.3003**
	(0.1143)	(0.0824)	(0.0776)	(0.0960)	(0.0975)	(0.1305)
Fixed effects						
User	Yes	Yes	Yes	Yes	Yes	Yes
Project	Yes	Yes	Yes	Yes	Yes	Yes
Technology	Yes	Yes	Yes	Yes	Yes	Yes
Language	Yes	Yes	Yes	Yes	Yes	Yes
City	Yes	Yes	Yes	Yes	Yes	Yes
Time	Yes	Yes	Yes	Yes	Yes	Yes
Technology × time		Yes	Yes	Yes	Yes	Yes
Language × time			Yes	Yes	Yes	Yes
City × technology				Yes	Yes	Yes
City × language					Yes	Yes
City × time						Yes
Users	20,640	20,640	20,640	20,640	20,640	20,640
Adjusted R^2	0.289	0.291	0.293	0.293	0.293	0.294
Observations	2,451,163	2,451,163	2,451,163	2,451,163	2,451,163	2,451,163

Notes: The 5% largest cities (10 cities) are excluded. Language refers to programming language. Robust standard errors clustered at the city × technology level in parenthesis. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table A.8: Quality (forks)

Dep. var.: Commits [log]	(1)	(2)	(3)	(4)	(5)	(6)
Cluster size [log]	0.1438	0.1354	0.1234	0.2779***	0.2742***	0.3789***
	(0.1030)	(0.0851)	(0.0815)	(0.0881)	(0.0891)	(0.1448)
Fixed effects						
User	Yes	Yes	Yes	Yes	Yes	Yes
Project	Yes	Yes	Yes	Yes	Yes	Yes
Technology	Yes	Yes	Yes	Yes	Yes	Yes
Language	Yes	Yes	Yes	Yes	Yes	Yes
City	Yes	Yes	Yes	Yes	Yes	Yes
Time	Yes	Yes	Yes	Yes	Yes	Yes
Technology × time		Yes	Yes	Yes	Yes	Yes
Language × time			Yes	Yes	Yes	Yes
City × technology				Yes	Yes	Yes
City × language					Yes	Yes
City × time						Yes
Users	7,135	7,135	7,135	7,135	7,135	7,135
Adjusted R^2	0.405	0.405	0.406	0.408	0.409	0.411
Observations	427,991	427,991	427,991	427,991	427,991	427,991
$\Delta(\beta_{\text{top10}} - \beta_{\text{all}})$	0.0294	0.0284	0.0305	0.0813	0.0807	0.1012
$\Delta(\beta_{\text{top10}} - \beta_{\text{all}})/\beta_{\text{all}}$	0.2045	0.2097	0.2472	0.2926	0.2943	0.2671

Notes: Regressions based on the top decile of projects by forks. These are 7,135 projects with at least four forks.β_{top10} denotes the estimated coefficient on cluster size. β_{all} refers to the estimated coefficient of cluster size from the corresponding specification in Table 1.1. Robust standard errors clustered at the city × technology level in parenthesis. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

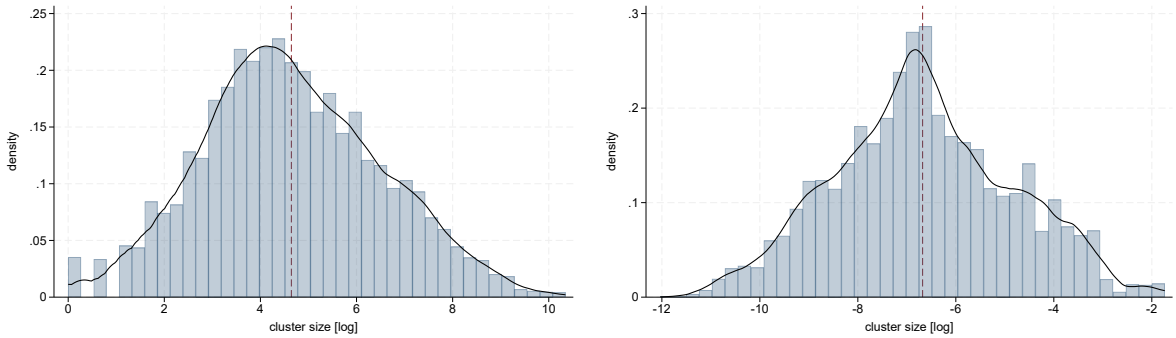
Table A.9: Dynamic estimates

Dep. var.: Commits [log]	(1)	(2)	(3)	(4)	(5)	(6)
$\beta(t = -1)$	0.0001	-0.0006	-0.0004	-0.0001	0.0011	0.0015
	(0.0101)	(0.0089)	(0.0087)	(0.0089)	(0.0091)	(0.0094)
$\beta(t = 0)$	0.1204	0.1120	0.0973	0.1301	0.1302	0.2676**
	(0.1097)	(0.0793)	(0.0753)	(0.1232)	(0.1239)	(0.1267)
$\beta(t = 1)$	-0.0021	-0.0023	-0.0029	-0.0027	-0.0023	-0.0016
	(0.0109)	(0.0111)	(0.0112)	(0.0111)	(0.0112)	(0.0110)
Fixed effects						
User	Yes	Yes	Yes	Yes	Yes	Yes
Project	Yes	Yes	Yes	Yes	Yes	Yes
Technology	Yes	Yes	Yes	Yes	Yes	Yes
Language	Yes	Yes	Yes	Yes	Yes	Yes
City	Yes	Yes	Yes	Yes	Yes	Yes
Time	Yes	Yes	Yes	Yes	Yes	Yes
Technology × time		Yes	Yes	Yes	Yes	Yes
Language × time			Yes	Yes	Yes	Yes
City × technology				Yes	Yes	Yes
City × language					Yes	Yes
City × time						Yes
Users	21,116	21,116	21,116	21,116	21,116	21,116
Adjusted R^2	0.331	0.332	0.333	0.334	0.334	0.335
Observations	1,532,335	1,532,335	1,532,335	1,532,335	1,532,335	1,532,335

Notes: Robust standard errors clustered at the city × technology level in parenthesis. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

A.2 Figures

Figure A.1: Technology cluster size distribution

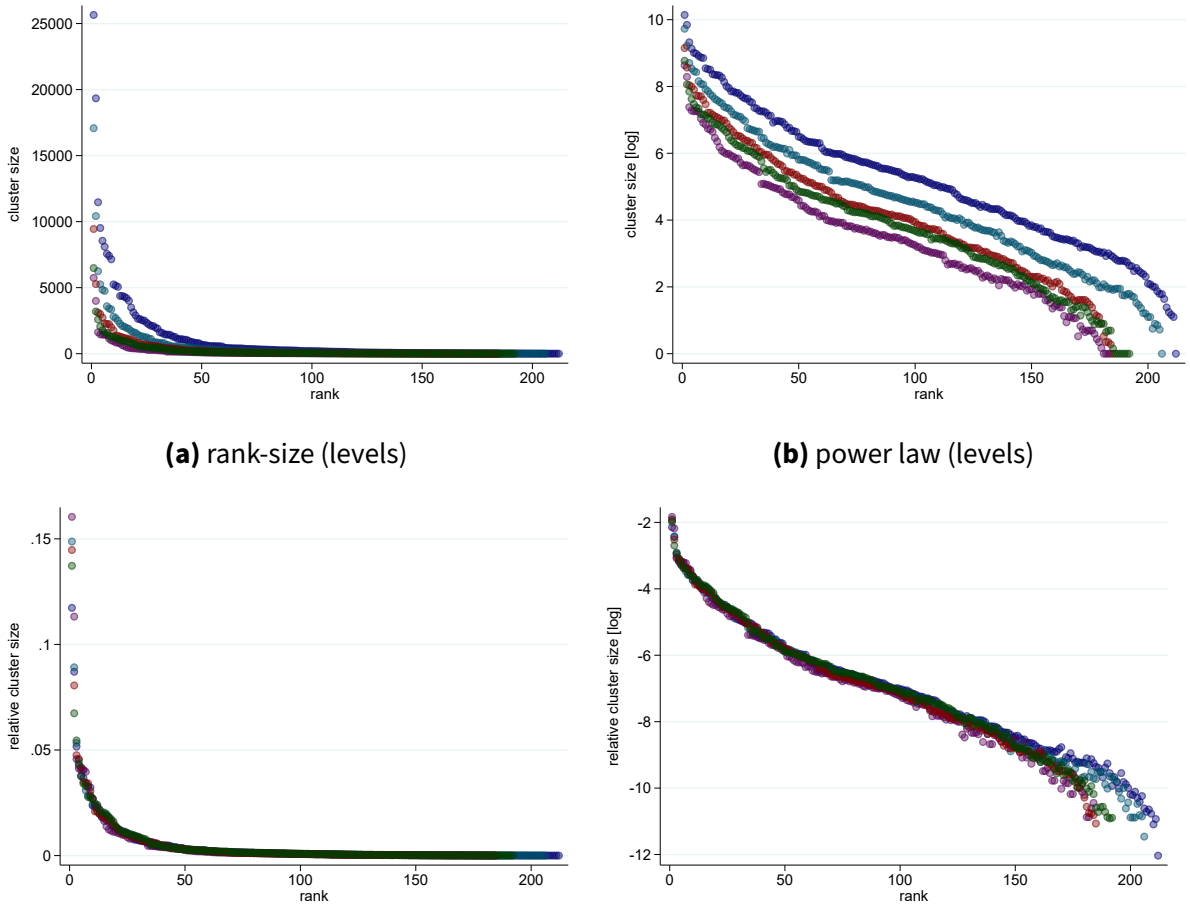


(a) number of users in cluster

(b) cluster size

Sources: GHTorrent, own calculations.

Figure A.2: Agglomeration by technology



(a) rank-size (levels)

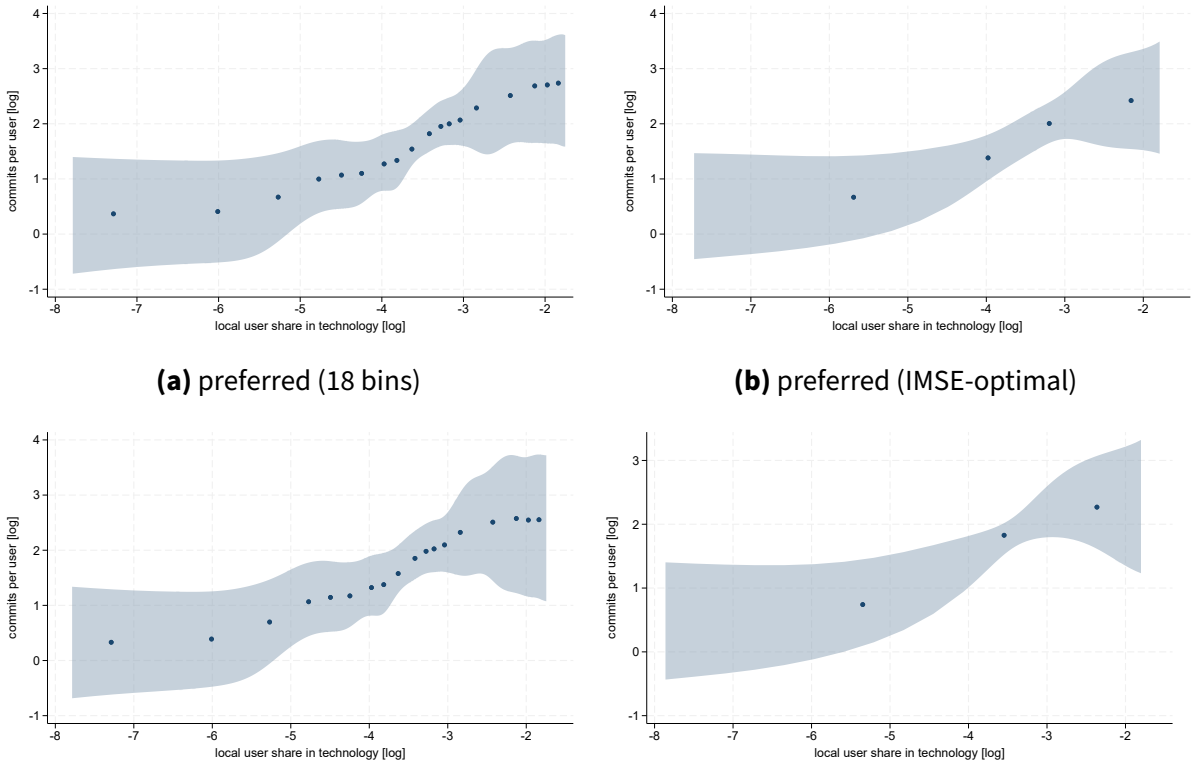
(b) power law (levels)

(c) rank-size (share)

(d) power law (share)

Sources: GHTorrent, own calculations.

Figure A.3: Binscatter specification



(a) preferred (18 bins)

(b) preferred (IMSE-optimal)

(c) extended (18 bins)

(d) extended (IMSE-optimal)

Notes: Graph plots a binscatter representation of the relationship between software engineer productivity and cluster size using `binsreg` (Cattaneo et al., 2021). Our preferred specification includes fixed effects for time, technology, language, project, city, and user as well as for time × city, time × technology, and city × technology. The extended specification additionally features time × language and language × city fixed effects. Shaded areas represent 90% confidence intervals. *Sources:* GHTorrent, own calculations.

B Supplementary Materials to Chapter 2

B.1 Tables

Table B.1: Sample selection

Median	All users	Movers	Δ
Activity			
Commits	6.00	170.00	164.00
commits single projects	*2.00*	*73.00*	*71.00*
commits team projects	*1.00*	*65.00*	*64.00*
Experience	34.00	39.00	5.00
Collaboration			
Projects	2.00	15.00	13.00
single projects	*2.00*	*9.00*	*7.00*
team projects	*2.00*	*5.00*	*3.00*
Quality			
Followers	0.00	5.00	5.00
Stars	0.00	1.30	1.30
stars single projects	*0.00*	*0.10*	*0.10*
Forks	0.00	0.76	0.76
forks single projects	*0.00*	*0.00*	*0.00*

Notes: Experience is measured as tenure on the platform in months since the first commit at the move date. Column Δ reports the absolute difference in median between movers in our sample and all users in the ten *GHTorrent* snapshots we utilize (N = 28,802,543). Column %Δ sets this difference in relation to other movers' median. *Sources:* GHTorrent, own calculations.

Table B.2: Affiliation and job transitions

Affiliation	all movers	job movers	other movers	Δ
Largest 100 firms	28.9 %	28.9 %	27.2 %	+1.7 p.p.
Big tech	*7.2 %*	*7.3 %*	*4.9 %*	*+2.4 p.p.*
Academic	8.9 %	9.0 %	6.3 %	+2.7 p.p.
Other	55.1 %	54.8 %	61.6 %	-6.8 p.p.
Job transitions	anytime	origin	destination	Δ
Largest 100 firms	28.9 %	20.3 %	26.8 %	+6.5 p.p.
Big tech	*7.2 %*	*2.0 %*	*7.1 %*	*+5.1 p.p.*
Academic	8.9 %	9.1 %	7.2 %	-2.0 p.p.
Other	55.1 %	68.6 %	58.9 %	-9.6 p.p.

Notes: Table reports affiliations and job transitions by organization type in shares of the respective sample. Column Δ reports the percentage point difference between job and other movers. *Sources:* GHTorrent, own calculations.

Table B.3: Top origin and destination cities

Origin	**Users**	**Share**	**Destination**	**Users**	**Share**
New York, USA	650	2.84 %	San Francisco, USA	1,307	5.71 %
San Francisco, USA	618	2.70 %	New York, USA	936	4.09 %
London, UK	421	1.84 %	London, UK	763	3.33 %
Bangalore, India	325	1.42 %	Seattle, USA	708	3.09 %
Chicago, USA	311	1.36 %	Bangalore, India	559	2.44 %
Boston, USA	305	1.33 %	Los Angeles, USA	379	1.66 %
Los Angeles, USA	305	1.33 %	Austin, USA	345	1.51 %
Moscow, Russia	305	1.33 %	Toronto, Canada	331	1.45 %
Seattle, USA	273	1.19 %	Chicago, USA	318	1.39 %
Paris, France	247	1.08 %	Boston, USA	315	1.38 %
Cumulative share		15.09 %	Cumulative share		26.05 %

Notes: Table reports the ten largest origin and destination cities in terms of the number of users in our sample. *Sources:* GHTorrent, own calculations.

Table B.4: Domestic moves

Country	Users	Share	
		all	domestic
United States	10,348	45.20 %	63.49 %
India	1,219	5.32 %	7.48 %
United Kingdom	638	2.79 %	3.91 %
Canada	620	2.71 %	3.80 %
China	522	2.28 %	3.20 %
France	436	1.90 %	2.68 %
Germany	417	1.82 %	2.56 %
Russia	375	1.64 %	2.30 %
Poland	195	0.85 %	1.20 %
Australia	194	0.85 %	1.19 %
		65.36 %	91.81 %

Notes: Table reports the ten largest countries in terms of the number of domestic movers in our sample. Shares reported in the third and fourth columns refer to all and to domestic movers, respectively. *Sources:* GHTorrent, own calculations.

Table B.5: Top origin and destination countries

International movers					
Origin	*Users*	*Share*	*Destination*	*Users*	*Share*
United States	1,831	0.28	United States	2,011	0.30
India	817	0.12	United Kingdom	774	0.12
United Kingdom	491	0.07	Canada	506	0.08
Russia	386	0.06	Germany	319	0.05
Canada	384	0.06	Russia	306	0.05
France	267	0.04	Netherlands	290	0.04
Australia	186	0.03	Australia	240	0.04
Italy	165	0.03	Poland	228	0.03
Brazil	163	0.02	France	182	0.03
Germany	151	0.02	Brazil	169	0.03
Inter-continental movers					
Origin	*Users*	*Share*	*Destination*	*Users*	*Share*
United States	1,453	0.34	United States	1,583	0.37
India	793	0.18	United Kingdom	428	0.10
United Kingdom	284	0.07	Russia	287	0.07
Russia	203	0.05	Canada	275	0.06
Australia	180	0.04	Australia	229	0.05
France	144	0.03	Germany	177	0.04
China	130	0.03	Poland	159	0.04
Canada	105	0.02	France	116	0.03
Italy	72	0.02	Netherlands	111	0.03
Poland	72	0.02	Italy	96	0.02

Notes: Table reports the ten largest origin and destination countries in terms of the number of international and inter-continental movers in our sample. *Sources:* GHTorrent, own calculations.

Table B.6: Top origin and destination affiliations

Origin	Share	Destination	Share
Student	0.92 %	Microsoft	2.08 %
Microsoft	0.72 %	Google	2.00 %
University of Washington	0.62 %	Amazon	1.37 %
Freelancer	0.51 %	Facebook	1.00 %
IBM	0.41 %	Red Hat	0.64 %
New York University	0.41 %	Shopify	0.44 %
University of California	0.41 %	IBM	0.37 %
University of Florida	0.41 %	Stanford University	0.31 %
University of Oxford	0.41 %	LinkedIn	0.28 %
Amazon	0.31 %	Apple	0.26 %
	5.13 %		8.75 %

Notes: Table reports the ten most frequently stated affiliations as a percentage of all users with non-empty affiliation information. *Sources:* GHTorrent, own calculations.

Table B.7: Classification of programming languages

Classification	programming language	share	
		lang.	class.
App development	Ruby	5.68 %	
	Go	4.06 %	
	Swift	1.09 %	
	Objective-C	0.65 %	11.48 %
Data engineering	Python	13.03 %	
	R	1.22 %	
	Jupyter Notebook	1.18 %	
	Scala	0.89 %	16.32 %
Low-level programming	C++	5.37 %	
	C	3.33 %	
	C#	2.30 %	
	Rust	1.40 %	
	Assembly	0.08 %	12.48 %
Program routine	Shell	3.16 %	
	PowerShell	0.22 %	3.38 %
Web development	JavaScript	20.91 %	
	HTML	6.65 %	
	Java	6.19 %	
	PHP	4.36 %	
	CSS	4.28 %	
	TypeScript	3.21 %	42.39 %
Other			10.74 %

Notes: The 27 most-used programming languages in terms of commits in the *GHTorrent* are classified, 21 of which are represented in our sample. Classified programming languages account for 89.26% of commits in our sample. *Sources:* GHTorrent, own calculations.

Table B.8: Top-paying programming languages

Classification	programming language	share		median pay	
		lang.	class. cumul.	lang.	class. avg.
Top 30 top-paying languages	Zig	0.009 %		$103,611	
	Erlang	0.145 %		$99,492	
	F#	0.091 %		$99,311	
	Ruby	5.749 %		$98,522	
	Clojure	0.399 %		$96,381	
	Elixir	0.383 %		$96,381	
	Scala	0.894 %		$96,381	
	Perl	0.491 %		$94,540	
	Go	4.087 %		$92,760	
	OCaml	0.365 %		$91,026	
	Objective-C	0.646 %		$90,000	
	Rust	1.365 %		$87,012	
	Swift	1.041 %		$86,897	
	Groovy	0.202 %		$86,271	
	Shell	3.347 %		$85,672	
	Haskell	0.771 %		$85,672	
	Apex	0.015 %		$81,552	
	PowerShell	0.23 %		$81,311	
	SAS	0.002 %		$81,000	
	Lua	0.312 %		$80,690	
	Nim	0.016 %		$80,000	
	Raku	0.001 %		$79,448	
	Python	12.933 %		$78,331	
	Kotlin	0.438 %		$78,207	
	APL	0 %		$77,500	
	Crystal	0.041 %		$77,104	
	TypeScript	3.074 %		$77,104	
	Assembly	0.078 %		$77,010	
	Fortran	0.132 %		$76,104	
	Cobol	0.001 %		$76,000	
	C#	2.314 %	39.572 %	$74,963	$86,008
Other top-paying languages	C++	5.516 %		$74,963	
	Julia	0.416 %		$74,963	
	R	1.217 %		$74,963	
	SQL	0.12 %		$74,963	
	C	3.438 %		$74,351	
	JavaScript	20.381 %		$74,034	
	Solidity	0.007 %		$72,701	
	Ada	0.013 %		$72,656	
	HTML	6.653 %		$71,500	
	CSS	4.264 %		$70,148	
	Prolog	0.018 %		$70,000	
	Delphi	0 %		$69,608	
	GDScript	0.021 %		$69,608	
	VBA	0.002 %		$65,698	
	Visual Basic	0.096 %		$65,000	
	Matlab	0.215 %		$61,735	
	PHP	4.375 %		$58,899	
	Dart	0.221 %	46.973 %	$55,862	$69,536
Not listed			13.455 %		

Notes: Table reports programming languages on the *StackOverflow* list of top-paying technologies. We further distinguish between the top 30 and other listed programming languages. Classified programming languages account for 86.54% of commits in our sample. *Sources:* GHTorrent, StackOverflow, own calculations.

Table B.9: N-grams by project category

Cluster	keywords	% projects
Code	adventofcode; algorithm; algorithms; android; api; app; application; apps; c; class; framework; functions; game; hacktoberfest; ios; javascript; library; module; nodejs; plugin; python; react; server; software; template; testing; tictactoe; tool; ui	7.06
Website	blog; personal; personalwebsite; portfolio; resume; site; website	2.11
File	collection; docs; document; documentation; dotfiles; file; files; githubslideshow; presentation; presentations; scripts	1.17
Education	course; coursera; example; examples; exercise; exercises; freecodecamp; helloworld; homework; learning; nowgithubstarter; programmingassignment; repdata; peerassessment; test	0.85
Data	data; database	0.48
Other		13.06

Notes: Table reports keywords assigned to project type clusters. Projects are assigned to the cluster with most associated uni- or bigrams. We remove stop words and use word stems in our bag-of-words. Keywords search is conducted in project descriptions; 24.73% of projects feature non-empty project descriptions. *Sources:* GHTorrent, own calculations.

Table B.10: Model specification

Model class:	OLS				LPM	NB	PPML
	(1)	(2)	(3)	(4)	(5)	(6)	(7)
Dependent variable:	log	ihs	ihs	ihs	dummy	count	count
Sample:	full	full	geo	change	full	full	full
Job mover × job search	0.1326***	0.1646***	0.1654***	0.1384**	0.0711***	0.4983***	0.1358***
	(0.0119)	(0.0141)	(0.0142)	(0.0548)	(0.0039)	(0.0280)	(0.0521)
Job mover × post move	-0.0851***	-0.1036***	-0.1021***	-0.2804***	-0.0307***	-0.2690***	-0.1707**
	(0.0159)	(0.0190)	(0.0190)	(0.0849)	(0.0056)	(0.0453)	(0.0670)
User FE	×	×	×	×	×	×	×
Month FE	×	×	×	×	×	×	×
Experience FE	×	×	×	×	×	×	×
Adjusted R^2	0.35803	0.35945	0.35958	0.43305	0.34000		
Observations	1,946,413	1,946,413	1,941,317	76,797	1,946,413	1,630,215	1,630,215
# User FE	22,896	22,896	22,838	885	22,896	22,896	22,896

Notes: Results from estimation of Equation 2.2 for different model classes, outcome transformations, and sample definitions. Experience is measured as months since the first commit at move month. Robust standard errors are clustered at the user level. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table B.11: Project ownership and initial forks

IHS(single commits)	project owner		
	(1) own	(2) non-own	(3) no initial forks
Job mover × job search	0.1310***	0.0428***	0.1440***
	(0.0138)	(0.0080)	(0.0149)
Job mover × post move	-0.1157***	0.0024	-0.1088***
	(0.0182)	(0.0097)	(0.0194)
User FE	×	×	×
Month FE	×	×	×
Experience FE	×	×	×
Adjusted R^2	0.33534	0.32483	0.32440
Observations	1,946,413	1,946,413	1,946,413
Users	22,896	22,896	22,896

Notes: Results from estimation of Equation 2.2 by repository ownership and without initial fork projects. Experience is measured as months since the first commit at move month. Robust standard errors are clustered at the user level. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table B.12: Heterogeneity by project types (keywords)

IHS(single commits)	(1) education	(2) data	(3) website	(4) code	(5) files	(6) other
Job mover × job search	0.0030	0.0000	0.0135***	0.0307***	0.0154***	0.1097***
	(0.0024)	(0.0027)	(0.0043)	(0.0069)	(0.0037)	(0.0123)
Job mover × post move	-0.0091***	-0.0089***	-0.0049	-0.0335***	-0.0044	-0.0641***
	(0.0035)	(0.0031)	(0.0056)	(0.0090)	(0.0045)	(0.0163)
User FE	×	×	×	×	×	×
Month FE	×	×	×	×	×	×
Experience FE	×	×	×	×	×	×
Adjusted R^2	0.09276	0.10952	0.15628	0.16827	0.21257	0.31379
Observations	1,946,413	1,946,413	1,946,413	1,946,413	1,946,413	1,946,413
Users	22,896	22,896	22,896	22,896	22,896	22,896

Notes: Results from estimation of Equation 2.2 for different project types, according to keyword-based method. Experience is measured as months since the first commit at move month. Robust standard errors are clustered at the user level. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table B.13: Event study coefficients

IHS(single commits)	(1)	(2)	(3)
Job mover × event_time = -21	0.0126	0.0025	0.0017
	(0.0206)	(0.0217)	(0.0215)
Job mover × event_time = -20	0.0178	0.0283	0.0326
	(0.0208)	(0.0217)	(0.0215)
Job mover × event_time = -19	-0.0397**	-0.0016	0.0042
	(0.0196)	(0.0208)	(0.0207)
Job mover × event_time = -18	-0.0555***	-0.0084	-0.0066
	(0.0193)	(0.0204)	(0.0203)
Job mover × event_time = -17	-0.0328*	-0.0167	-0.0145
	(0.0168)	(0.0178)	(0.0176)
Job mover × event_time = -15	0.1771***	0.0574***	0.0557***
	(0.0188)	(0.0197)	(0.0196)
Job mover × event_time = -14	0.5110***	0.1608***	0.1596***
	(0.0239)	(0.0251)	(0.0252)
Job mover × event_time = -13	0.5415***	0.1787***	0.1807***
	(0.0243)	(0.0251)	(0.0251)
Job mover × event_time = -12	0.6329***	0.2443***	0.2455***
	(0.0277)	(0.0282)	(0.0282)
Job mover × event_time = -11	0.5882***	0.1942***	0.1996***
	(0.0271)	(0.0276)	(0.0278)
Job mover × event_time = -10	0.5708***	0.1640***	0.1675***
	(0.0268)	(0.0272)	(0.0273)
Job mover × event_time = -9	0.4677***	0.1141***	0.1221***
	(0.0264)	(0.0270)	(0.0269)
Job mover × event_time = -8	0.4538***	0.1290***	0.1377***
	(0.0273)	(0.0278)	(0.0277)
Job mover × event_time = -7	0.4278***	0.1339***	0.1475***
	(0.0273)	(0.0278)	(0.0278)
Job mover × event_time = -6	0.4627***	0.1440***	0.1630***
	(0.0287)	(0.0293)	(0.0295)
Job mover × event_time = -5	0.4658***	0.1158***	0.1318***
	(0.0278)	(0.0284)	(0.0285)
Job mover × event_time = -4	0.3806***	0.0759***	0.0967***
	(0.0274)	(0.0276)	(0.0278)
Job mover × event_time = -3	0.3846***	0.0388	0.0654**
	(0.0265)	(0.0272)	(0.0272)
Job mover × event_time = -2	0.3617***	0.0416	0.0690**
	(0.0264)	(0.0271)	(0.0273)
Job mover × event_time = -1	0.4193***	0.0331	0.0738***
	(0.0275)	(0.0283)	(0.0285)
Job mover × event_time = 0	-0.0184	-0.1128***	-0.0799***
	(0.0225)	(0.0237)	(0.0242)
Job mover × event_time = 1	0.1672***	-0.2069***	-0.0380
	(0.0357)	(0.0363)	(0.0360)
Job mover × event_time = 2	0.1323***	-0.2101***	-0.0355
	(0.0391)	(0.0397)	(0.0394)
Job mover × event_time = 3	-0.0117	-0.3078***	-0.1291***
	(0.0379)	(0.0383)	(0.0380)
Job mover × event_time = 4	-0.0196	-0.2641***	-0.0780**
	(0.0338)	(0.0342)	(0.0340)
Job mover × event_time = 5	-0.0234	-0.2527***	-0.0621*
	(0.0364)	(0.0371)	(0.0367)
Job mover × event_time = 6	0.0134	-0.2151***	-0.0197
	(0.0386)	(0.0386)	(0.0381)
Job mover × event_time = 7	-0.3461***	-0.2582***	-0.0785***
	(0.0311)	(0.0309)	(0.0303)
Job mover × event_time = 8	-0.3202***	-0.2582***	-0.0671**
	(0.0302)	(0.0298)	(0.0295)
Job mover × event_time = 9	-0.2907***	-0.2614***	-0.0634**
	(0.0320)	(0.0316)	(0.0313)
Job mover × event_time = 10	-0.3573***	-0.2762***	-0.0725**
	(0.0312)	(0.0310)	(0.0307)
User FE	×	×	×
Month FE		×	×
Experience FE			×
Adjusted R^2	0.28992	0.30870	0.35963
Observations	1,946,413	1,946,413	1,946,413
Users	22,896	22,896	22,896

Notes: Estimates for $t_j \times \text{JobChanger}_i$ based on Equation 2.1 with user, experience and calendar month fixed effects. The outcome is IHS-transformed commits to single-authored projects. The reference month is $t = -16$. Bars show 95% confidence intervals. Standard errors are clustered at the user level. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table B.14: Job search period

Job search period:	(1) [−15, −9]	(2) [−15, −6]	(3) [−15, −3]	(4) [−15, 0]
Job mover × job search	0.1947***	0.1836***	0.1768***	0.1646***
	(0.0154)	(0.0144)	(0.0141)	(0.0141)
Job mover × uncertain	0.1423***	0.1308***	0.0925***	
	(0.0161)	(0.0178)	(0.0203)	
Job mover × post move	-0.1099***	-0.1099***	-0.1100***	-0.1036***
	(0.0184)	(0.0184)	(0.0184)	(0.0190)
User FE	×	×	×	×
Month FE	×	×	×	×
Experience FE	×	×	×	×
Adjusted R^2	0.35946	0.35946	0.35946	0.35945
Observations	1,946,413	1,946,413	1,946,413	1,946,413
Users	22,896	22,896	22,896	22,896
Relation to baseline	+3.01 p.p.	+1.90 p.p.	+1.22 p.p.	baseline
	+18.3 %	+11.5 %	+7.4 %	baseline

Notes: Results from estimation of Equation 2.2 for different definitions of the job serach period. Experience is measured as months since the first commit at move month. Robust standard errors are clustered at the user level. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table B.15: International movers

IHS(single commits)	international		inter-continental	
	(1) yes	(2) no	(3) yes	(4) no
Job mover × job search	0.2027***	0.1474***	0.2335***	0.1483***
	(0.0263)	(0.0167)	(0.0336)	(0.0155)
Job mover × post move	-0.0812**	-0.1124***	-0.1057**	-0.1031***
	(0.0342)	(0.0228)	(0.0435)	(0.0211)
User FE	×	×	×	×
Month FE	×	×	×	×
Experience FE	×	×	×	×
Adjusted R^2	0.36811	0.35640	0.36273	0.35907
Observations	562,982	1,383,431	366,271	1,580,142
Users	6,598	16,298	4,305	18,591

Notes: Results from estimation of Equation 2.2 with IHS-transformed number of commits to (non-)international and (non-)inter-continental single-authored projects. Upward income group moves are defined as moves from developing to developed countries. Upward moves in GDP per capita are based on current 2021 PPP USD. Experience is measured as months since the first commit at move month. Robust standard errors are clustered at the user level. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$.
Sources: GHTorrent, own calculations.

Table B.16: Upward movers

IHS(single commits)	GDP p. c.		income class	
	(1) other	(2) up	(3) other	(4) up
Job mover × job search	0.1622***	0.1821***	0.1610***	0.2381***
	(0.0149)	(0.0437)	(0.0146)	(0.0512)
Job mover × post move	-0.1034***	-0.1038*	-0.1038***	-0.0949
	(0.0199)	(0.0627)	(0.0195)	(0.0755)
User FE	×	×	×	×
Month FE	×	×	×	×
Experience FE	×	×	×	×
Adjusted R^2	0.36073	0.34025	0.35980	0.33293
Observations	1,776,167	170,246	1,854,956	91,457
Users	20,829	2,067	21,763	1,133

Notes: Results from estimation of Equation 2.2 with IHS-transformed number of commits to (non-)upward single-authored projects in terms of GDP p.c. and income class, respectively. Upward income group moves are defined as moves from developing to developed countries. Upward moves in GDP per capita are based on current 2021 PPP USD. Experience is measured as months since the first commit at move month. Robust standard errors are clustered at the user level. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

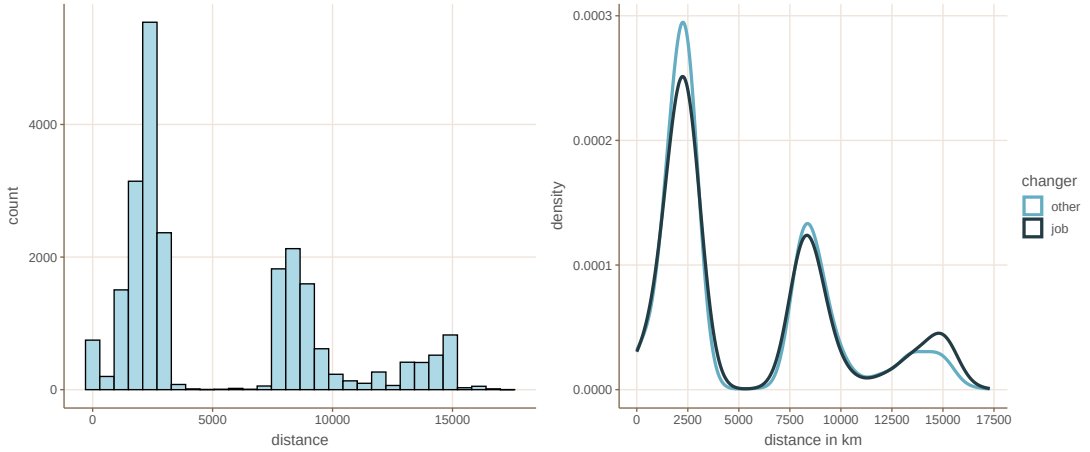
Table B.17: Affiliation

IHS(single commits)	destination						origin			
	median		big tech		academia		median		adademia	
	(1) below	(2) above	(3) no	(4) yes	(5) no	(6) yes	(7) below	(8) above	(9) no	(10) yes
Job mover × job search	0.1770***	0.0068	0.1740***	0.1556***	0.1535***	0.3158***	0.1636***	0.2339**	0.1565***	0.1511***
	(0.0145)	(0.0526)	(0.0212)	(0.0176)	(0.0146)	(0.0459)	(0.0142)	(0.1052)	(0.0519)	(0.0472)
Job mover × post move	-0.0955***	-0.1556**	-0.1001***	-0.0895***	-0.1199***	0.1547**	-0.1038***	-0.0610	-0.1755**	-0.1593**
	(0.0195)	(0.0668)	(0.0296)	(0.0232)	(0.0194)	(0.0717)	(0.0191)	(0.1320)	(0.0758)	(0.0721)
User FE	×	×	×	×	×	×	×	×	×	×
Month FE	×	×	×	×	×	×	×	×	×	×
Experience FE	×	×	×	×	×	×	×	×	×	×
Adjusted R^2	0.35927	0.36002	0.36084	0.35832	0.35823	0.36154	0.35933	0.35989	0.35999	0.36103
Observations	1,900,195	1,369,596	1,553,857	1,715,934	1,900,917	1,368,874	1,935,568	1,334,223	1,361,217	1,368,330
Users	22,387	16,194	18,374	20,207	22,378	16,203	22,767	15,814	16,130	16,212

Notes: Results from estimation of Equation 2.2 with IHS-transformed number of commits to single-authored projects. Median split refers to median size of affiliation in terms of users in the full *GHTorrent* sample. Big tech refers to Google, Amazon, Meta, Apple and Microsoft. Academia refers to students and university affiliations. Destination (origin) refers to users' affiliation before (after) the affiliation change. Experience is measured as months since the first commit at move month. Robust standard errors are clustered at the user level. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

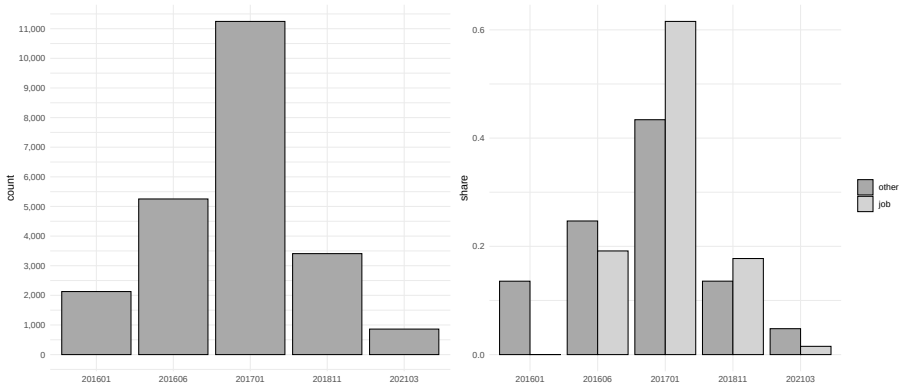
B.2 Figures

Figure B.1: Distribution of move distances



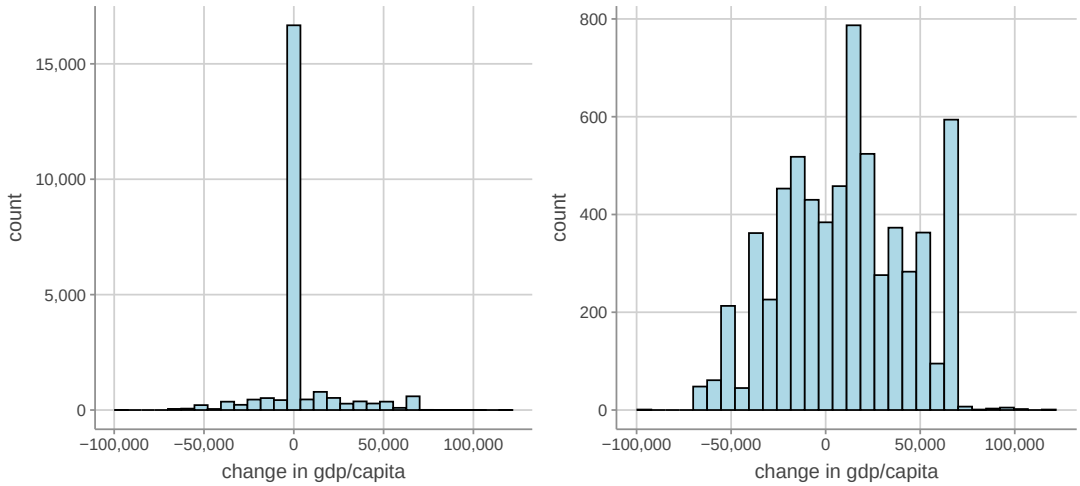
Notes: Histogram on the left shows the distribution of move distances. Estimates on the right show kernel densities for job movers and other movers. *Sources:* GHTorrent, own calculations.

Figure B.2: Distribution of moves across time



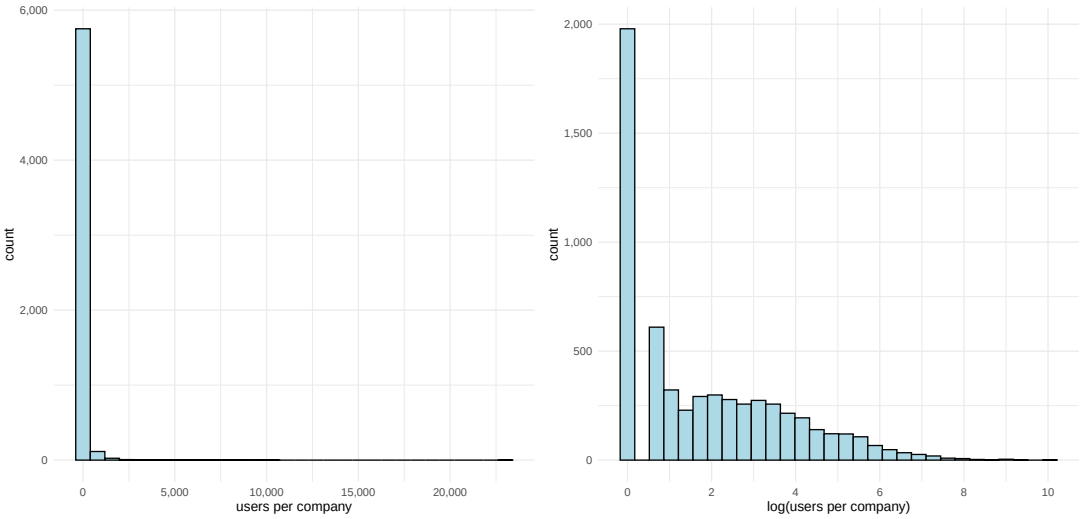
Notes: Histogram on the left shows the distribution of moves across data snapshots. Shares on the right depict the distribution of moves across data snapshots for job movers (dark gray) and other movers (light gray). *Sources:* GHTorrent, own calculations.

Figure B.3: Distribution of income changes



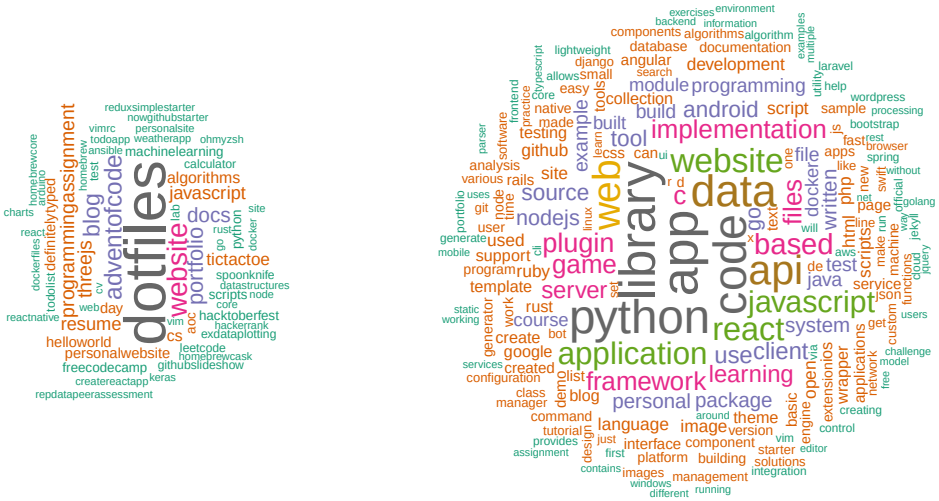
Notes: Histograms depict the distribution of national per capita GDP changes of movers in the full sample (left) and the international sample (right). GDP is measured in current 2021 PPP USD. *Sources:* GHTorrent, World Development Indicators, own calculations.

Figure B.4: Distribution of affiliation size



Notes: Histograms depict the distribution of affiliations with respect to the number of affiliated users in the full *GHTorrent* sample as counts (left) and after logarithmic transformation (right). Note that string-based merging of affiliations is likely imperfect, especially for small firms, which leads to a downward bias of firm size. *Sources:* GHTorrent, own calculations.

Figure B.5: Frequent words in project names and descriptions



Notes: Word clouds show frequently occurring words in single projects of movers. Word size and color represent word frequency in project titles (left) and descriptions (right). Frequency limits are set at 50 (titles) and 100 (descriptions). We remove English stop words, numbers, punctuation, URLs, white space, and the words *project*, *repository/repo*, *simple*, and *using*. *Sources:* GHTorrent, own calculations.

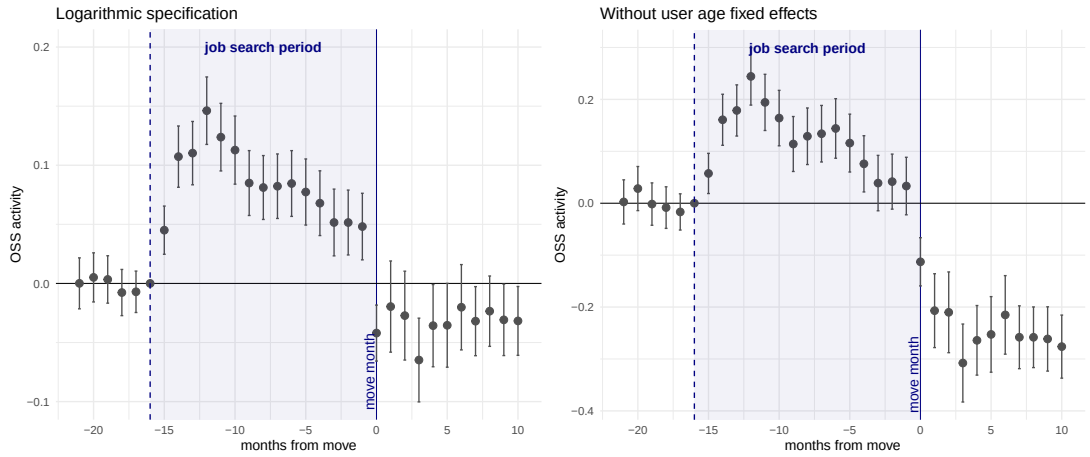
Figure B.6: Heterogeneity by user popularity

Notes: Estimates for $t_j \times \text{JobChanger}_i$ based on Equation 2.1 with user, experience and calendar month fixed effects. The outcome is IHS-transformed commits to single-authored projects in the respective follower quartile (1st quartile: green; 2nd quartile: orange; 3rd quartile: blue and 4th quartile: purple.). The reference month is $t = -16$. Bars show 95% confidence intervals. Standard errors are clustered at the user level. *Sources:* GHTorrent, own calculations.

Figure B.7: Heterogeneity by project age

Notes: Estimates for $t_j \times \text{JobChanger}_i$ based on Equation 2.1 with user, experience and calendar month fixed effects. The outcome is IHS-transformed commits to single-authored new (orange) and old (green) projects. New projects are defined as projects with the date of the first commit in the month under consideration. The reference month is $t = -16$. Bars show 95% confidence intervals. Standard errors are clustered at the user level. *Sources:* GHTorrent, own calculations.

Figure B.8: Event study model robustness



Notes: Estimates for $t_j \times \text{JobChanger}_i$ based on Equation 2.1 with user, experience and calendar month fixed effects. The outcome is logarithmically transformed using $ln(y+1)$ in the left panel and IHS-transformed commits to single-authored projects in the right panel. The reference month is $t = -16$. Bars show 95% confidence intervals. Standard errors are clustered at the user level. *Sources:* GHTorrent, own calculations.

C Supplementary Materials to Chapter 3

C.1 Tables

Table C.1: Summary statistics: users

Median	rockstar	user
Activity		
Commits	4,188	6
Experience	92	34
Followers	4,446	0
Projects		
Total	180	2
one rockstar projects	*160*	*0*
several rockstar projects	*9*	*0*
single projects	*70*	*2*
team projects	*96*	*2*
owned projects	*53*	*1*

Notes: Experience is measured as tenure on the platform in months since the first public commit on *GitHub*. *Sources:* GHTorrent, own calculations.

Table C.2: Project programming languages

Rockstar projects		No rockstar projects	
C	0.26	JavaScript	0.21
Java	0.15	Python	0.10
JavaScript	0.12	Java	0.10
C++	0.08	PHP	0.07
Python	0.04	HTML	0.07

Notes: Table reports the share of projects per programming language for the five most used programming languages in single rockstar project and no rockstar projects, respectively. *Sources:* GHTorrent, own calculations.

Table C.3: Event study coefficients

commits	(1)
event_time = -55	-0.5726***
	(0.0619)
event_time = -5	-0.1093*
	(0.0605)
event_time = -4	-0.1160**
	(0.0561)
event_time = -3	-0.0537
	(0.0508)
event_time = -2	-0.0360
	(0.0384)
event_time = 0	0.2646***
	(0.0459)
event_time = 1	0.0214
	(0.0580)
event_time = 2	-0.0381
	(0.0717)
event_time = 3	-0.1271**
	(0.0638)
event_time = 4	-0.1298*
	(0.0754)
event_time = 5	-0.1973**
	(0.0788)
event_time = 6	-0.2952***
	(0.0691)
event_time = 7	-0.3724***
	(0.0634)
event_time = 8	-0.3501***
	(0.0700)
event_time = 9	-0.3084***
	(0.0784)
event_time = 10	-0.3674***
	(0.0761)
event_time = 11	-0.3644***
	(0.0839)
event_time = 55	-0.6246***
	(0.1140)
Full Set of FE	Yes
Adjusted Pseudo R^2	0.699
Observations	154,067

Notes: Estimates for ι_j based on Equation 3.1 with calendar month, project, project age and programming language x calendar month fixed effects. The reference month is $t = -1$. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table C.4: Event study coefficients (fork activity included)

commits	(1)
event_time = -55	-0.5948***
	(0.0665)
event_time = -5	-0.0822
	(0.0722)
event_time = -4	-0.1145**
	(0.0575)
event_time = -3	-0.0535
	(0.0510)
event_time = -2	-0.0374
	(0.0388)
event_time = 0	0.3703***
	(0.0450)
event_time = 1	0.1704***
	(0.0523)
event_time = 2	0.1286**
	(0.0639)
event_time = 3	0.1044*
	(0.0582)
event_time = 4	0.0962
	(0.0682)
event_time = 5	0.0963
	(0.0741)
event_time = 6	0.0347
	(0.0678)
event_time = 7	-0.0429
	(0.0622)
event_time = 8	-0.0084
	(0.0709)
event_time = 9	0.0492
	(0.0726)
event_time = 10	-0.0289
	(0.0742)
event_time = 11	0.0184
	(0.0899)
event_time = 55	-0.4245***
	(0.1103)
Full Set of FE	Yes
Adjusted Pseudo R^2	0.712
Observations	154,067

Notes: Estimates for t_j based on Equation 3.1. The reference month is $t = -1$. The outcome is the combined number of monthly commits to a project and its forks. For rockstar projects, only forks created in the rockstar contribution month or twelve months post rockstar contribution are considered. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table C.5: Watcher and fork owner

commits	(1) watcher	(2) fork owner
rockstar contribution	0.5893***	0.4568***
	(0.1100)	(0.1115)
post	0.5289***	0.3819***
	(0.1428)	(0.1190)
Month FE	Yes	Yes
Project FE	Yes	Yes
Project Age FE	Yes	Yes
Language FE-Month FE	Yes	Yes
Adjusted Pseudo R^2	0.776	0.807
Observations	154,067	154,067

Notes: Results from PPML estimation of Equation 3.2 with the monthly number of commits to projects by the respective user type. Watcher refers to commits by users that bookmarked, i.e. *watch*, the project prior to committing. Fork owner refers to commits by users that created a project copy prior to committing. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table C.6: Event study coefficients (daily activity)

commits	(1) rockstar followers	(2) followers of rockstar followers
event_time = -6	-0.5141*	0.3501
	(0.3065)	(0.2406)
event_time = -5	-0.7021***	0.2665
	(0.2642)	(0.2521)
event_time = -4	-0.2253	0.3804
	(0.2912)	(0.2449)
event_time = -3	-0.1982	0.1014
	(0.2882)	(0.2847)
event_time = -2	-0.0662	0.2684
	(0.2376)	(0.2604)
event_time = 0	0.9913***	0.6685***
	(0.2333)	(0.2502)
event_time = 1	0.2320	1.168**
	(0.2979)	(0.5811)
event_time = 2	-0.1875	0.6282
	(0.2873)	(0.5162)
event_time = 3	-0.2445	0.0773
	(0.2814)	(0.2887)
event_time = 4	-0.1731	0.5166
	(0.2934)	(0.3632)
event_time = 5	-0.3729	-0.4457
	(0.2969)	(0.3231)
event_time = 6	0.0307	0.2009
	(0.2438)	(0.3360)
event_time = 7	0.0162	-0.1697
	(0.2028)	(0.2976)
event_time = 8	-0.2717	0.2940
	(0.2401)	(0.2651)
event_time = 9	-0.1179	0.0470
	(0.3372)	(0.2763)
event_time = 10	-0.2645	0.1751
	(0.3902)	(0.3031)
event_time = 11	-0.5532**	-0.0160
	(0.2720)	(0.3022)
event_time = 12	0.0911	0.0002
	(0.3088)	(0.2889)
Day FE	Yes	Yes
Project FE	Yes	Yes
Project Age FE	Yes	Yes
Language FE-Day FE	Yes	Yes
Adjusted Pseudo R^2	0.376	0.376
Observations	1,213,381	1,212,008

Notes: Estimates for t_j based on Equation 3.1 with calendar day, project, project age and programming language x calendar day fixed effects. The outcome is the daily number of project commits by rockstar followers or followers of the rockstar followers, which do not follow the rockstar. The reference day is $t = -1$. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table C.7: Rockstar contribution quantity

commits	(1) lines deleted	(2) lines added	(3) lines changed
rockstar contribution	0.7312***	0.9526***	0.7320***
	(0.0821)	(0.0866)	(0.0758)
rockstar contribution × indicator	-0.0551	-0.4267***	-0.0846
	(0.1169)	(0.1210)	(0.1278)
post	-0.0733	0.1548	0.1177
	(0.0890)	(0.1045)	(0.0883)
post × indicator	0.2880**	-0.1277	-0.1410
	(0.1245)	(0.1490)	(0.1547)
Month FE	Yes	Yes	Yes
Project FE	Yes	Yes	Yes
Project Age FE	Yes	Yes	Yes
Language FE-Month FE	Yes	Yes	Yes
Adjusted Pseudo R^2	0.694	0.694	0.694
Observations	154,067	154,067	154,067

Notes: Results from PPML estimation of Equation 3.2 adding an interaction with an indicator variable of interest to study heterogeneous effects. Lines deleted refers to a rockstar contribution deleting code lines. Lines added refers to a rockstar contribution adding code lines. Lines changed refers to a rockstar contribution in total changing above median, i.e. 10, code lines. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table C.8: Rockstar affiliation

commits	(1) freelance	(2) academia	(3) bigtech
rockstar contribution	0.7090***	0.7326***	0.7493***
	(0.0707)	(0.0733)	(0.0753)
rockstar contribution × indicator	0.4220	-0.7225**	-0.2109
	(0.5493)	(0.3315)	(0.1932)
post	0.0732	0.1197	0.1030
	(0.0710)	(0.0737)	(0.0845)
post × indicator	-0.2361	-1.203***	-0.1499
	(0.3400)	(0.4102)	(0.2558)
Month FE	Yes	Yes	Yes
Project FE	Yes	Yes	Yes
Project Age FE	Yes	Yes	Yes
Language FE-Month FE	Yes	Yes	Yes
Adjusted Pseudo R^2	0.694	0.696	0.694
Observations	154,067	154,067	154,067

Notes: Results from PPML estimation of Equation 3.2 adding an interaction with an indicator variable of interest to study heterogeneous effects. Freelance refers to stating *freelance* as the affiliation. Academia refers to university affiliations. Specifically, users stating *university, college, institute, universiteit, universidad, or universitat* in their affiliation are assigned to academia. Big tech refers to Google, Amazon, Meta, Apple and Microsoft. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table C.9: Number of contributors

users	(1) new	(2) total
rockstar contribution	1.018***	0.8776***
	(0.0539)	(0.0566)
post	-0.3412***	-0.0809
	(0.0550)	(0.0568)
Month FE	Yes	Yes
Project FE	Yes	Yes
Project Age FE	Yes	Yes
Language FE-Month FE	Yes	Yes
Adjusted Pseudo R^2	0.355	0.431
Observations	154,067	154,067

Notes: Results from PPML estimation of Equation 3.2 with the number of monthly users as the dependent variable. New users refer to the monthly number of users contributing the first time to a project. Total refers to the total monthly number of contributors to a project. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table C.10: Heterogeneity by project age

	project age quarter			
commits	(1) first	(2) second	(3) third	(4) fourth
rockstar contribution	-0.1112	0.8469***	0.5199***	0.9221***
	(0.1943)	(0.1434)	(0.0850)	(0.1125)
post	-0.5024**	0.3966**	0.1067	0.2875***
	(0.2181)	(0.1657)	(0.0887)	(0.1092)
Month FE	Yes	Yes	Yes	Yes
Project FE	Yes	Yes	Yes	Yes
Project Age FE	Yes	Yes	Yes	Yes
Language FE-Month FE	Yes	Yes	Yes	Yes
Clusters	2,561	2,822	3,025	3,429
Adjusted Pseudo R^2	0.651	0.707	0.684	0.668
Observations	50,737	68,486	75,739	93,721

Notes: Results from PPML estimation of Equation 3.2 with monthly number of commits to projects in the respective project age quarter at the month of rockstar contribution as the dependent variable. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table C.11: Heterogeneity by project age (fork activity included)

	project age quarter			
commits	(1) first	(2) second	(3) third	(4) fourth
rockstar contribution	0.1961	1.028***	0.6027***	0.9383***
	(0.2136)	(0.1458)	(0.0897)	(0.1258)
post	-0.0001	0.7041***	0.3434***	0.4166***
	(0.2532)	(0.1649)	(0.0842)	(0.1154)
Month FE	Yes	Yes	Yes	Yes
Project FE	Yes	Yes	Yes	Yes
Project Age FE	Yes	Yes	Yes	Yes
Language FE-Month FE	Yes	Yes	Yes	Yes
Adjusted Pseudo R^2	0.667	0.724	0.693	0.678
Observations	52,890	70,639	77,892	95,874

Notes: Results from PPML estimation of Equation 3.2 with the combined number of monthly commits to projects and its forks in the respective project age quarter at the month of rockstar contribution as the dependent variable. For rockstar projects, only forks created in the rockstar contribution month or twelve months post rockstar contribution are considered. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$.
Sources: GHTorrent, own calculations.

Table C.12: Issues

issues	(1) new	(2) updated	(3) closed
rockstar contribution	0.7380	0.3814***	-0.2295***
	(0.5896)	(0.0641)	(0.0497)
post	1.345***	0.0207	0.0920
	(0.4819)	(0.0824)	(0.1115)
Month FE	Yes	Yes	Yes
Project FE	Yes	Yes	Yes
Project Age FE	Yes	Yes	Yes
Language FE-Month FE	Yes	Yes	Yes
Adjusted Pseudo R^2	0.928	0.880	0.970
Observations	154,082	154,082	154,082

Notes: Results from PPML estimation of Equation 3.2 with the monthly number of issues of the respective type as the dependent variable. New refers to newly opened issues, updated to changed issues, and closed to finished issues. Language refers to programming language. Robust standard errors in parentheses are reported. *Sources:* GHTorrent, own calculations. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$.

Table C.13: Mechanism

	log(project duration) (1)	log(non-zero months) (2)	log(total commits) (3)
rockstar contribution	1.072***	1.580***	0.4961***
	(0.0198)	(0.0382)	(0.0531)
Project Age FE	Yes	Yes	Yes
Language FE	Yes	Yes	Yes
Projects	4,394	4,394	4,394
Adjusted R^2	0.679	0.555	0.106
Observations	4,394	4,394	4,394
Avg. dependent variable	34.06	629.7	582.6

Notes: Results from estimating an OLS model on project level outcomes of interest. Project duration is the total number of months between a project's first and last commit. Non-zero months are the total number of months with non-zero commits to a project. Total commits is the total number of monthly commits to a project. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table C.14: Rockstar definition

	rockstar definition		
commits	(1) upper 25%	(2) upper 10%	(3) upper 5%
rockstar contribution	0.5949***	0.7094***	0.7424***
	(0.1576)	(0.0706)	(0.0703)
post	-0.2045	0.0731	-0.0229
	(0.1789)	(0.0710)	(0.0927)
Month FE	Yes	Yes	Yes
Project FE	Yes	Yes	Yes
Project Age FE	Yes	Yes	Yes
Language FE-Month FE	Yes	Yes	Yes
Projects	2,760	4,394	3,772
Adjusted Pseudo R^2	0.641	0.694	0.677
Observations	58,085	154,067	126,578

Notes: Results from PPML estimation of Equation 3.2 using alternative rockstar definitions. Upper 25%, upper 10%, and upper 5% refer to rockstars that are in the upper 25%, upper 10%, and upper 5% of the user-follower distribution given a user has at least one follower. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

Table C.15: ATT: placebo

commits	(1)	(2)	(3)
placebo contribution	0.0770	0.0770	-0.0142
	(0.1404)	(0.1405)	(0.1782)
placebo post	-0.0574	-0.0574	-0.0293
	(0.1167)	(0.1168)	(0.1317)
Day FE	Yes	Yes	Yes
Project FE	Yes	Yes	Yes
Project Age FE		Yes	Yes
Language FE-Day FE			Yes
Projects	3,942	3,942	3,942
Adjusted Pseudo R^2	0.467	0.467	0.523
Observations	833,309	833,309	833,309

Notes: Results from PPML estimation of Equation 3.2 with the number of daily project commits as the dependant variable. Placebo treatment is a randomly drawn day between project start and 60 days before the rockstar contribution. Placebo contribution is an indicator equal to one if calendar day is the placebo treatment day. Placebo post is an indicator equal to one for all calendar days larger than placebo treatment day. Sample includes the project activity up until 30 days before rockstar contribution. Language refers to programming language. Robust standard errors in parentheses are reported. * $p > 0.1$, ** $p > 0.05$, and *** $p > 0.01$. *Sources:* GHTorrent, own calculations.

Table C.16: Comparison group

commits	(1) no matching	(2) only treated
rockstar contribution	0.7534***	0.6675***
	(0.0587)	(0.0677)
post	-0.3199***	-0.0763
	(0.1060)	(0.0738
Month FE	Yes	Yes
Project FE	Yes	Yes
Project Age FE	Yes	Yes
Language FE-Month FE	Yes	Yes
Projects	978,022	1,913
Adjusted Pseudo R^2	0.433	0.722
Observations	93,544,108	108,517

Notes: Results from PPML estimation of Equation 3.2. No matching refers to comparing all single rockstar contribution projects to all no rockstar projects, i.e. no matching was done. Only treated refers to limiting the sample to the single rockstar projects, i.e. comparing earlier vs. later treated. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

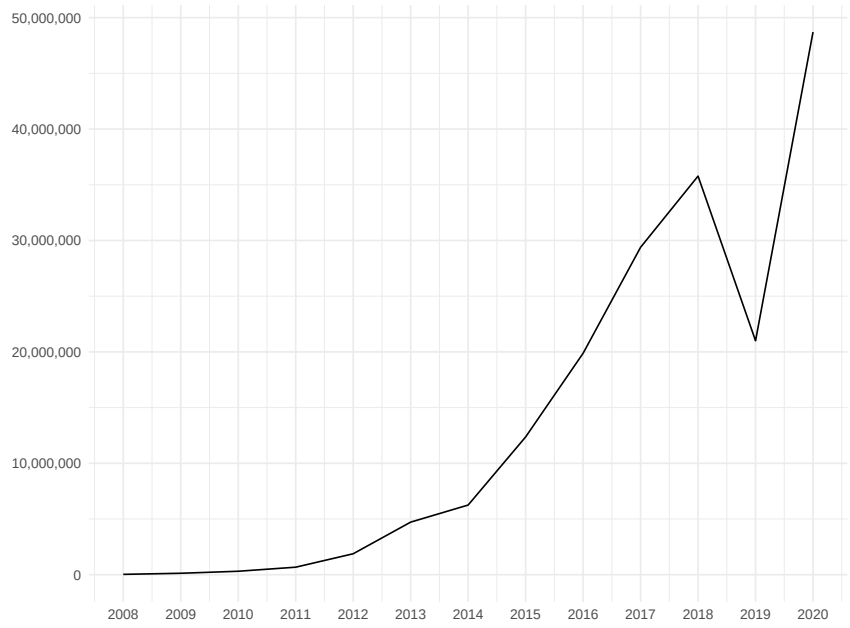
Table C.17: Single vs. several contributions

commits	(1)	(2)	(3)
single × rock. contribution	0.5639***	0.5639***	0.5821***
	(0.0708)	(0.0708)	(0.0759)
single × post	-0.0881	-0.0881	-0.1125
	(0.1269)	(0.1269)	(0.1388)
Month FE	Yes	Yes	Yes
Project FE	Yes	Yes	Yes
Project Age FE		Yes	Yes
Language FE-Month FE			Yes
Projects	2,546	2,546	2,546
Adjusted Pseudo R^2	0.767	0.767	0.797
Observations	146,884	146,884	146,884
Avg. Commits	25.78	25.78	25.78

Notes: Results from PPML estimation of Equation 3.2 comparing single rockstar contribution projects to several rockstar contribution projects. Language refers to programming language. Robust standard errors in parentheses are reported. *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.1$. *Sources:* GHTorrent, own calculations.

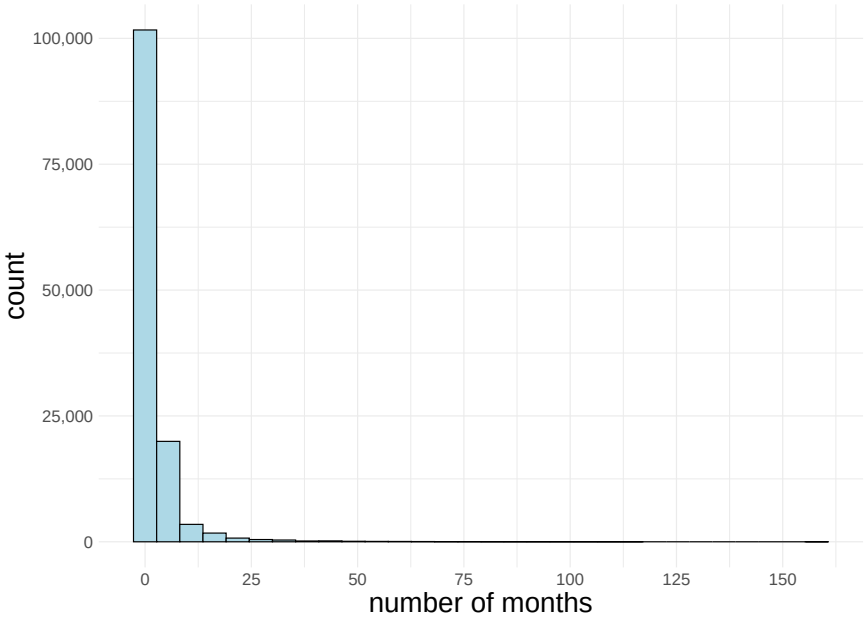
C.2 Figures

Figure C.1: New *GitHub* projects created per year



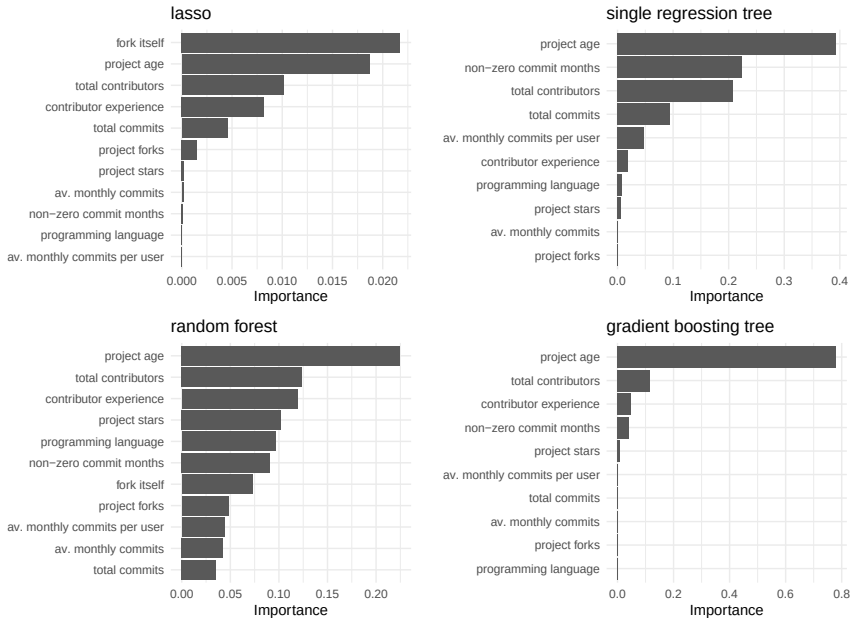
Notes: Figures displays the yearly number of new *GitHub* projects created. *Sources:* GHTorrent, own calculations.

Figure C.2: Distribution of rockstar contributions



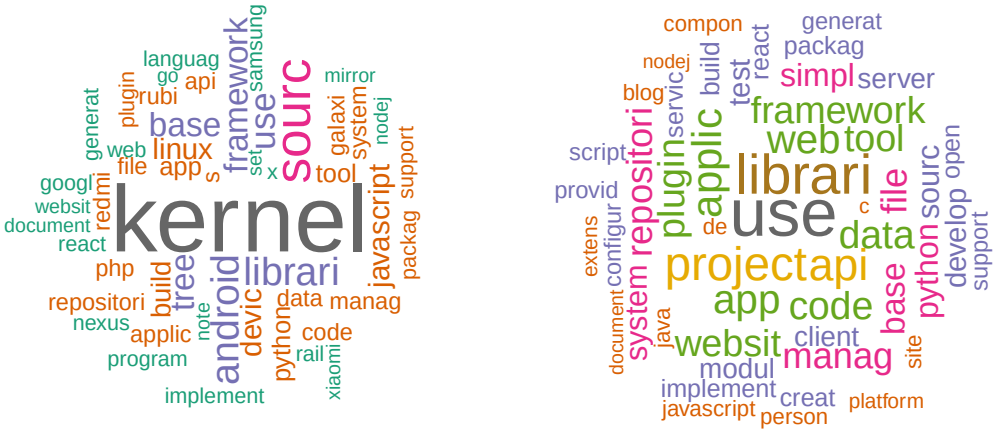
Notes: Histogram shows the distribution of the number of months with a rockstar contribution per project. *Sources:* GHTorrent, own calculations.

Figure C.3: Variable importance analysis



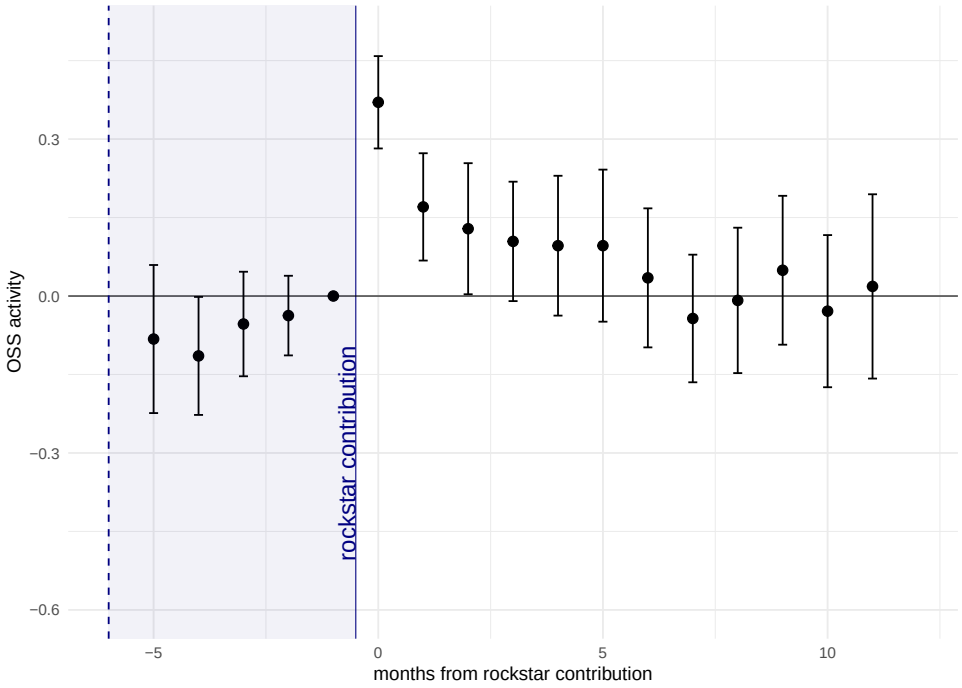
Notes: The figure displays the variable importance predictions for projects receiving a single rockstar contribution after running a LASSO, single regression tree, random forest or gradient boosting tree. Contributor experience is measured as the median user experience in a project. For single rockstar projects, I take the variable values in the calendar month before the rockstar contribution occurs. *Sources:* GHTorrent, own calculations.

Figure C.4: Frequent words in project descriptions



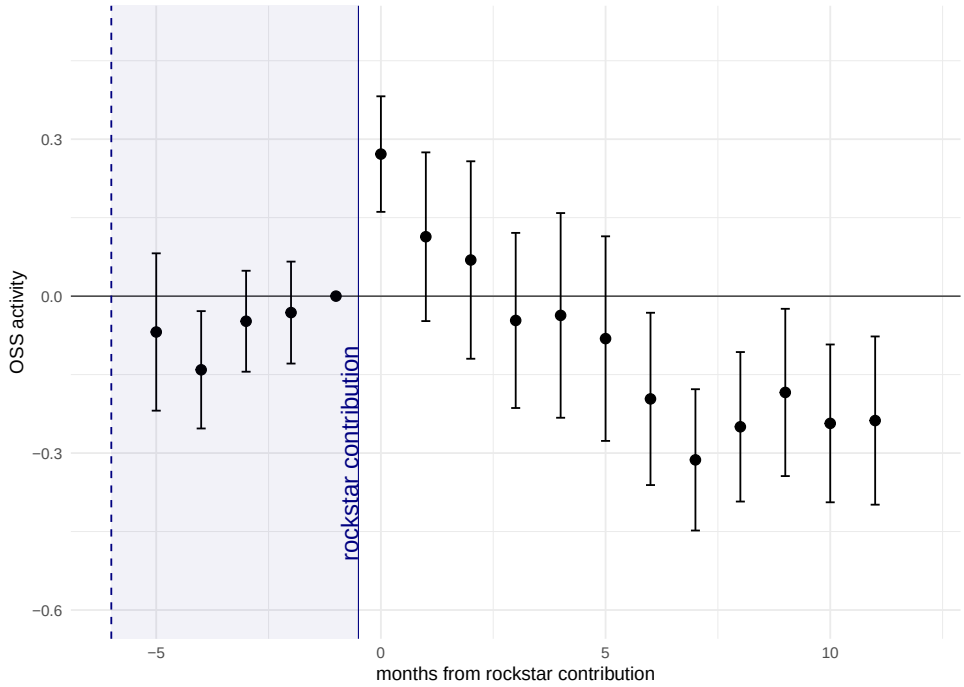
Notes: Word clouds show frequently occurring words in project descriptions. Word size and color represent word frequency in single rockstar (left) and no rockstar (right) projects. The maximum number of displayed words is set to 25. I remove stop words, white space, and use word stems before creating unigrams. 81.38% of single rockstar projects and 79.63% of no rockstar projects contain non-empty project descriptions. *Sources:* GHTorrent, own calculations.

Figure C.5: Event study estimates (fork activity included)



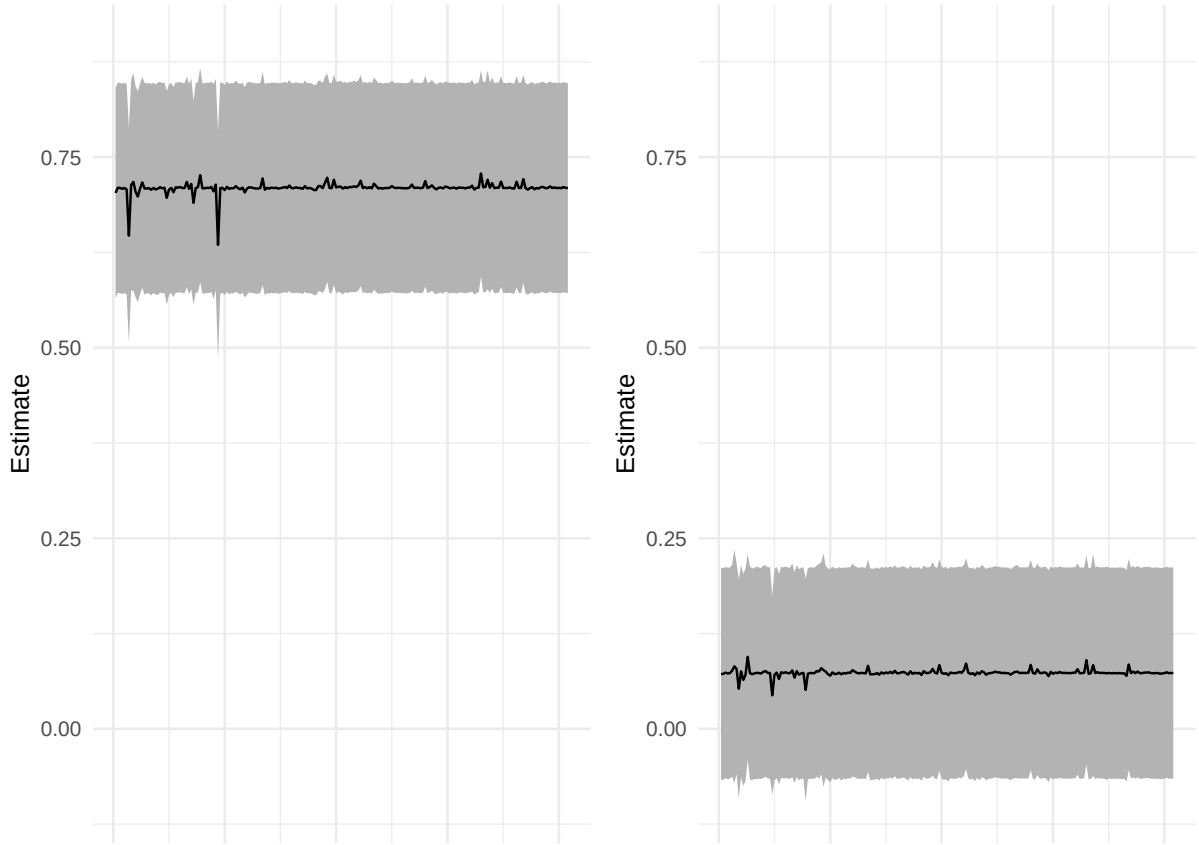
Notes: Estimates for t_j based on Equation 3.1 with calendar month, project, project age and programming language x calendar month fixed effects. The outcome is the combined number of monthly commits to a project and its forks. For rockstar projects, only forks created in the rockstar contribution month or twelve months post rockstar contribution are considered. The reference month is $t = -1$. Bars show 95% confidence intervals.
Sources: GHTorrent, own calculations.

Figure C.6: Event study estimates (balanced sample)



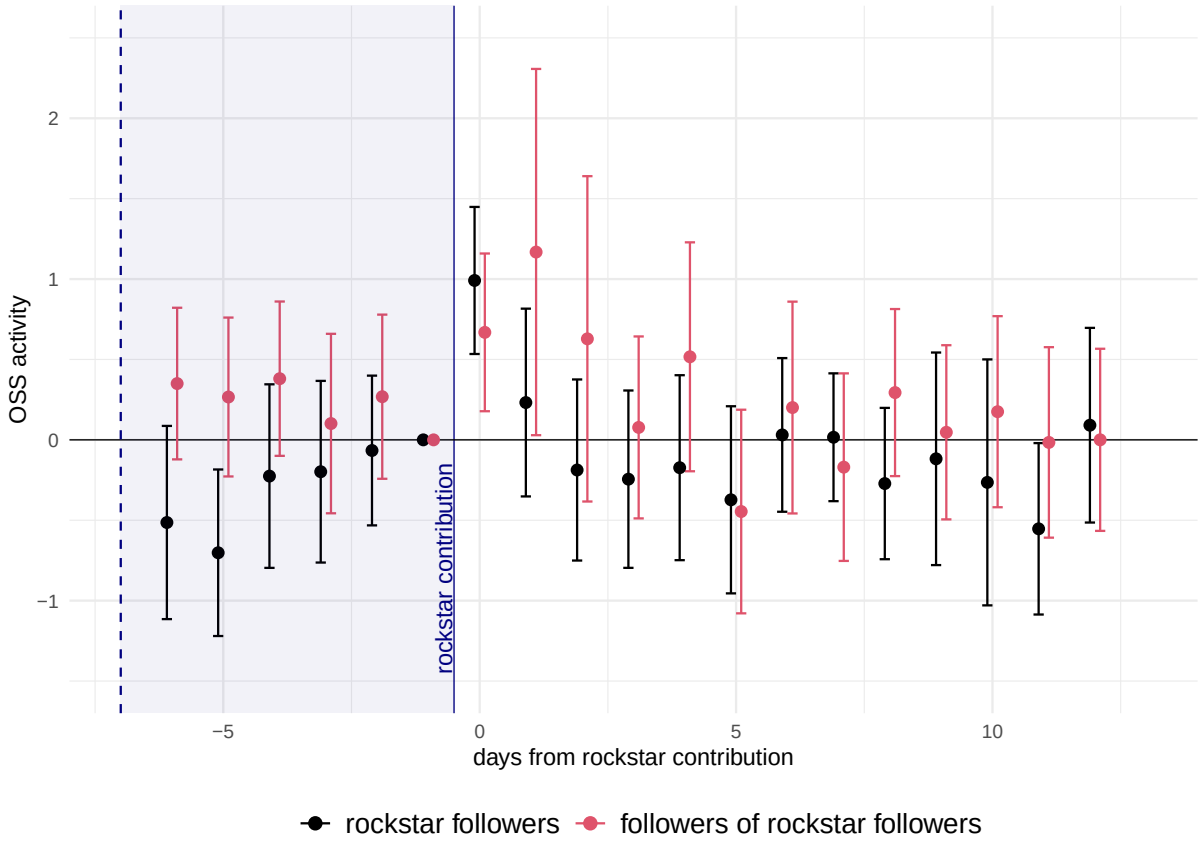
Notes: Estimates for t_j based on Equation 3.1 with calendar month, project, project age and programming language x calendar month fixed effects. The sample is balanced over the period of analysis. The reference month is $t = -1$. Bars show 95% confidence intervals.
Sources: GHTorrent, own calculations.

Figure C.7: Leave-one-out estimates



Notes: Figures plots the estimates of rockstar contribution (left) and post (right) of a leave-one-out exercise, based on Equation 3.2 with calendar month, project, project age, and programming language x calendar month fixed effects. For the regressions, successively one rockstar was omitted. The figure plots a total of 204 estimates. The lines are the point estimates and the shading indicates 95% confidence intervals. *Sources:* GHTorrent, own calculations.

Figure C.8: Event study estimates (daily activity)

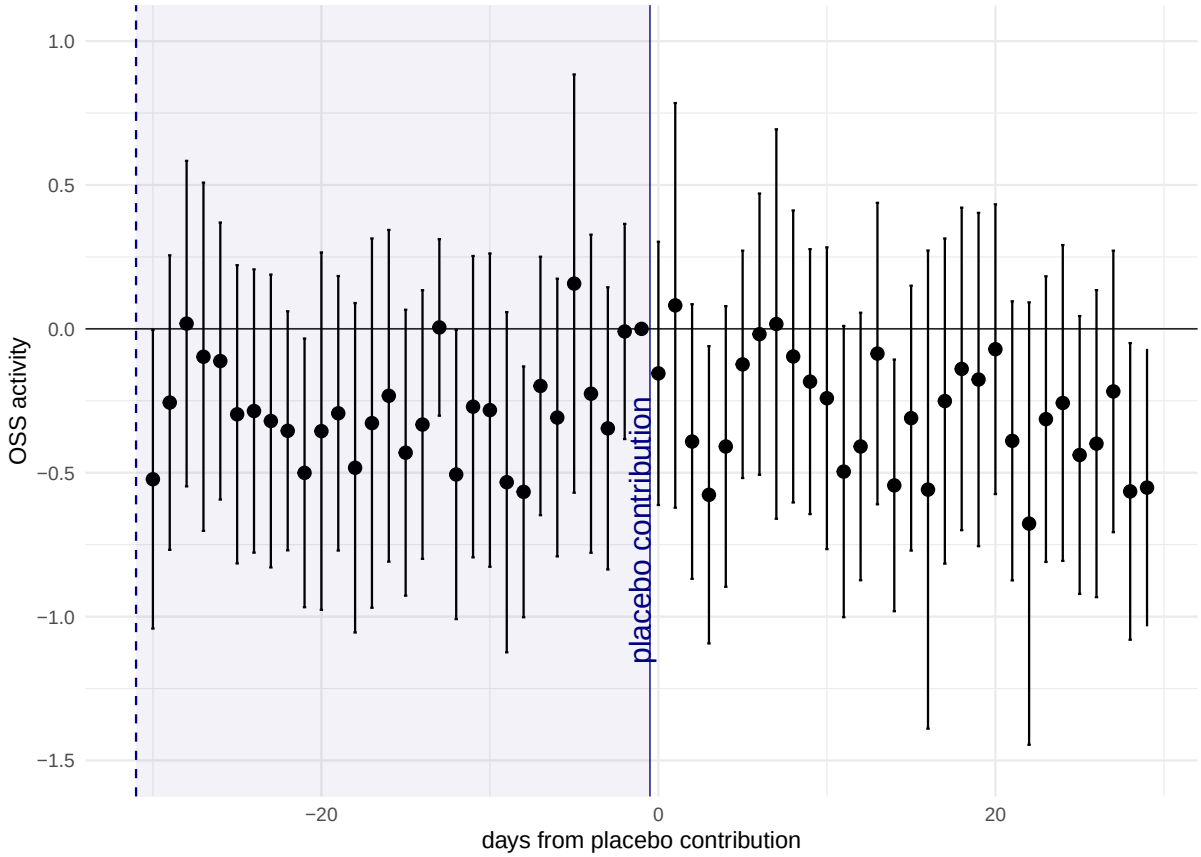


Notes: Estimates for t_j based on Equation 3.1 with calendar day, project, project age and programming language x calendar day fixed effects. The outcome is the daily number of commits to a project by rockstar followers (black), and followers of rockstar followers, which do not follow the rockstar (red). The reference day is $t = -1$. Bars show 95% confidence intervals. *Sources:* GHTorrent, own calculations.

Figure C.9: Frequent words in rockstar commit messages

Notes: Word cloud shows frequently occurring words in rockstar commit messages to single rockstar projects. Word size and color represent word frequency. Frequency limit is set at 25. I remove stop words, white space, and the word *signedoffbi*, and use word stems before creating unigrams. *Sources:* GHTorrent, own calculations.

Figure C.10: Placebo event study estimates



Notes: Estimates for t_j based on Equation 3.1 with calendar day, project, project age and programming language x calendar day fixed effects. The outcome is the daily number of project commits. Placebo treatment is a randomly drawn day between project start and 60 days before the rockstar contribution. Placebo contribution is an indicator equal to one if calendar day is the placebo treatment day. Placebo post is an indicator equal to one for all calendar days larger than placebo treatment day. Sample includes the project activity up until 30 days before rockstar contribution. * $p > 0.1$, ** $p > 0.05$, and *** $p > 0.01$. *Sources:* GHTorrent, own calculations.

D Supplementary Materials to Chapter 4

D.1 Tables

Table D.1: Difference in means between Exit and Non-Exit

	Non-Exit	Exit	Diff.	
Subscriber count	1026.82	1097.15	-70.33	***
	(77.33)	(83.17)	(118.06)	
Watchtime	5308.02	6444.02	-1136.00	
	(11786.76)	(20154.47)	(23859.67)	
Upload frequency	0.76	0.6	0.16	***
	(0.05)	(0.04)	(0.06)	
Like share	0.91	0.92	-0.01	
	(0.11)	(0.09)	(0.14)	
Unique keywords	33.63	29.21	4.42	***
	(35.33)	(23.97)	(45.17)	
Mainstream	0.59	0.53	0.06	*
	(0.49)	(0.50)	(0.73)	
Experience	35.59	35.09	0.50	
	(17.16)	(22.15)	(28.92)	

Notes: *** p<0.01, ** p<0.05, * p<0.1. Standard deviations in parentheses. The table reports the results of two-tailed t-tests on the difference in means between creators who appeared in the second wave of data collection (Non-Exit) and those that did not (Exit).

Table D.2: Manipulation test

	Upload Frequency (1)	Like Share (2)
Lost Access	−0.173	−0.035
	(0.209)	(0.024)
Bandwidth	[900,1200]	[900,1200]
Observations	428	425

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models use local linear regressions and control for watchtime. The sample consists of creators who did not exit the platform before the first and second waves of data collection. The outcome variables are calculated based on activity in the six months before the rule change.

Table D.3: Robustness: Bandwidth

	Upload Frequency		Like Share		Unique Keywords	
	(1) wide	(2) narrow	(3) wide	(4) narrow	(5) wide	(6) narrow
Lost Access	−1.207	−3.079	−0.010	−0.039	−11.503*	−30.236
	(0.809)	(2.154)	(0.016)	(0.045)	(6.136)	(18.554)
Bandwidth	[800,1400]	[950,1100]	[800,1400]	[950,1100]	[800,1400]	[950,1100]
Observations	827	263	819	261	827	263

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime. Sample contains creators who did not exit the platform between the first and second waves of data collection. The table reports results using different bandwidths of the running variable (subscriber count).

Table D.4: Robustness: Bandwidth Upload Frequency

	Upload Frequency			
	(1)	(2)	(3)	(4)
Lost Access	-1.983**	-1.875**	-1.969***	-2.004***
	(0.889)	(0.768)	(0.708)	(0.602)
Bandwidth	[900,1400]	[900,1600]	[900,1800]	[900,3000]
Observations	685	897	1083	1666

	Upload Frequency			
	(1)	(2)	(3)	(4)
Lost Access	-2.042	-2.386*	-2.479*	-2.525*
	(1.333)	(1.325)	(1.317)	(1.313)
Bandwidth	[800,1200]	[700,1200]	[600,1200]	[500,1200]
Observations	570	812	1100	1422

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime. Sample contains creators who did not exit the platform between the first and second waves of data collection. The table reports results using different bandwidths of the running variable (subscriber count).

Table D.5: Robustness: Bandwidth Like Share

	Like Share			
	(1)	(2)	(3)	(4)
Lost Access	-0.020	-0.017	-0.015	-0.020
	(0.023)	(0.023)	(0.022)	(0.022)
Bandwidth	[900,1400]	[900,1600]	[900,1800]	[900,3000]
Observations	678	888	1074	1649

	Like Share			
	(1)	(2)	(3)	(4)
Lost Access	-0.038**	-0.035**	-0.034**	-0.033**
	(0.019)	(0.017)	(0.016)	(0.015)
Bandwidth	[800,1200]	[700,1200]	[600,1200]	[500,1200]
Observations	566	807	1092	1408

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime. Sample contains creators who did not exit the platform between the first and second waves of data collection. The table reports results using different bandwidths of the running variable (subscriber count).

Table D.6: Robustness: Bandwidth Keywords

	Unique Keywords			
	(1)	(2)	(3)	(4)
Lost Access	-16.421**	-17.223**	-16.639***	-15.945***
	(7.451)	(6.845)	(6.161)	(6.042)
Bandwidth	[900,1400]	[900,1600]	[900,2500]	[900,3000]
Observations	685	897	1485	1666

	Unique Keywords			
	(1)	(2)	(3)	(4)
Lost Access	-20.515**	-20.755**	-22.705**	-22.581**
	(9.585)	(9.368)	(9.279)	(9.219)
Bandwidth	[800,1200]	[700,1200]	[600,1200]	[500,1200]
Observations	570	812	1100	1422

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime. Sample contains creators who did not exit the platform between the first and second waves of data collection. The table reports results using different bandwidths of the running variable (subscriber count).

Table D.7: Placebo test: 700 Subscriber threshold

	(1) Upload Frequency	(2) Like Share	(3) Unique Keywords
Lost Access	-0.244	0.011	-6.730
	(0.572)	(0.015)	(7.537)
Bandwidth	[600,850]	[600,850]	[600,850]
Observations	597	592	597

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime.

Table D.8: Placebo test: 1,300 Subscriber threshold

	(1) Upload Frequency	(2) Like Share	(3) Unique Keywords
Lost Access	2.302	-0.009	11.688
	(1.413)	(0.019)	(9.143)
Bandwidth	[1150,1600]	[1150,1600]	[1150,1600]
Observations	495	489	495

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime.

Table D.9: Robustness: Higher order polynomials

	Upload Frequency		Like Share		Unique Keywords	
	(1) quadratic	(2) cubic	(3) quadratic	(4) cubic	(5) quadratic	(6) cubic
Lost Access	−3.649	−2.1	−0.048	−0.021	−32.774	−38.769
	(2.393)	(3.595)	(0.047)	(0.091)	(20.703)	(35.788)
Bandwidth	[900,1200]	[900,1200]	[900,1200]	[900,1200]	[900,1200]	[900,1200]
Polynomials	2	3	2	3	2	3
Observations	428	428	425	425	428	428

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime. Sample contains creators who did not exit the platform between the first and second waves of data collection. The table reports results using different higher order polynomials to fit $f(Subscribers_i)$ in equation 4.1.

Table D.10: Robustness: Different time windows

	Upload Frequency		Like Share		Unique Keywords	
	(1)	(2)	(3)	(4)	(5)	(6)
	3 months	12 months	3 months	12 months	3 months	12 months
Lost Access	−4.281**	−2.182**	−0.036	−0.051*	−30.583**	−23.706**
	(2.111)	(0.981)	(0.025)	(0.028)	(12.462)	(9.438)
Bandwidth	[900,1200]	[900,1200]	[900,1200]	[900,1200]	[900,1200]	[900,1200]
Observations	359	465	357	463	359	465

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime. Sample contains creators who did not exit the platform between the first and second waves of data collection. The table reports results using measures calculated based on creator activity within three and twelve months after the rule change.

Table D.11: Robustness: Alternative Experience Measure

	Upload Frequency		Like Share		Unique Keywords	
	(1)	(2)	(3)	(4)	(5)	(6)
	More	Less	More	Less	More	Less
Lost Access	−4.897**	−0.325	−0.018	−0.075**	−32.967*	−15.466**
	(2.44)	(0.369)	(0.030)	(0.038)	(18.993)	(7.542)
Bandwidth	[900,1200]	[900,1200]	[900,1200]	[900,1200]	[900,1200]	[900,1200]
Observations	222	206	222	203	222	206

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime. Sample contains creators who did not exit the platform between the first and second waves of data collection. The table reports results using the number of videos uploaded before the rule change as an experience measure.

Table D.12: Robustness: Watchtime Upload Frequency

	Upload Frequency				
	(1)	(2)	(3)	(4)	(5)
	50%	30%	40%	60%	70%
Lost Access	-2.816**	-2.816**	-2.816**	-2.816**	-2.816**
	(1.387)	(1.387)	(1.387)	(1.387)	(1.387)
Bandwidth	[900,1200]	[900,1200]	[900,1200]	[900,1200]	[900,1200]
Observations	428	428	428	428	428

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime.

Table D.13: Robustness: Watchtime Like Share

	Like Share				
	(1)	(2)	(3)	(4)	(5)
	50%	30%	40%	60%	70%
Lost Access	-0.048*	-0.048*	-0.048*	-0.048*	-0.048*
	(0.025)	(0.025)	(0.025)	(0.025)	(0.025)
Bandwidth	[900,1200]	[900,1200]	[900,1200]	[900,1200]	[900,1200]
Observations	425	425	425	425	425

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime.

Table D.14: Robustness: Watchtime Unique Keywords

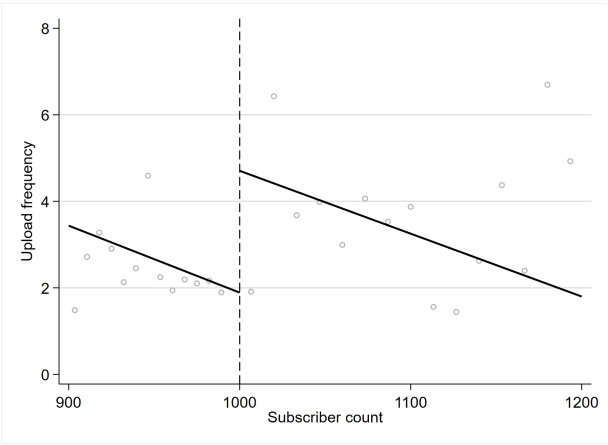
	Unique Keywords				
	(1)	(2)	(3)	(4)	(5)
	50%	30%	40%	60%	70%
Lost Access	-26.666**	-25.380**	-25.380**	-25.380**	-25.380**
	(10.636)	(10.485)	(10.485)	(10.485)	(10.485)
Bandwidth	[900,1200]	[900,1200]	[900,1200]	[900,1200]	[900,1200]
Observations	428	428	428	428	428

Notes: *** p<0.01, ** p<0.05, * p<0.1. Robust standard errors in parentheses are reported. All models control for watchtime.

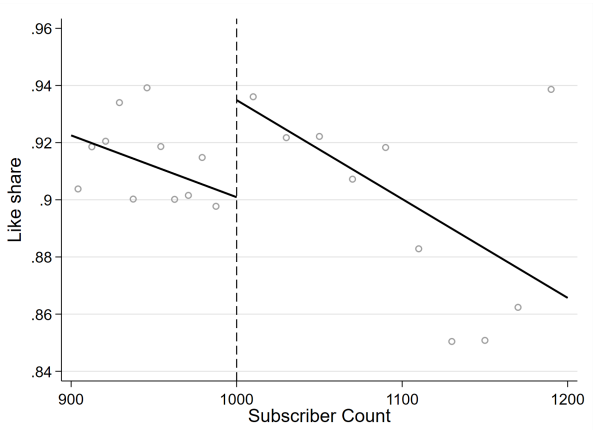
D.2 Figures

Figure D.1: RDD plots: Main results and content diversity

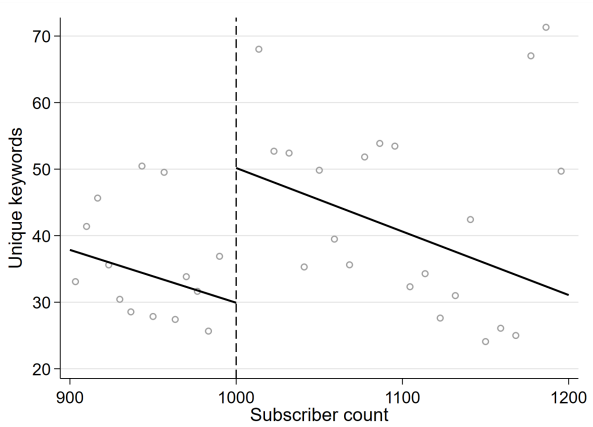
(a) Upload frequency



(b) Like share

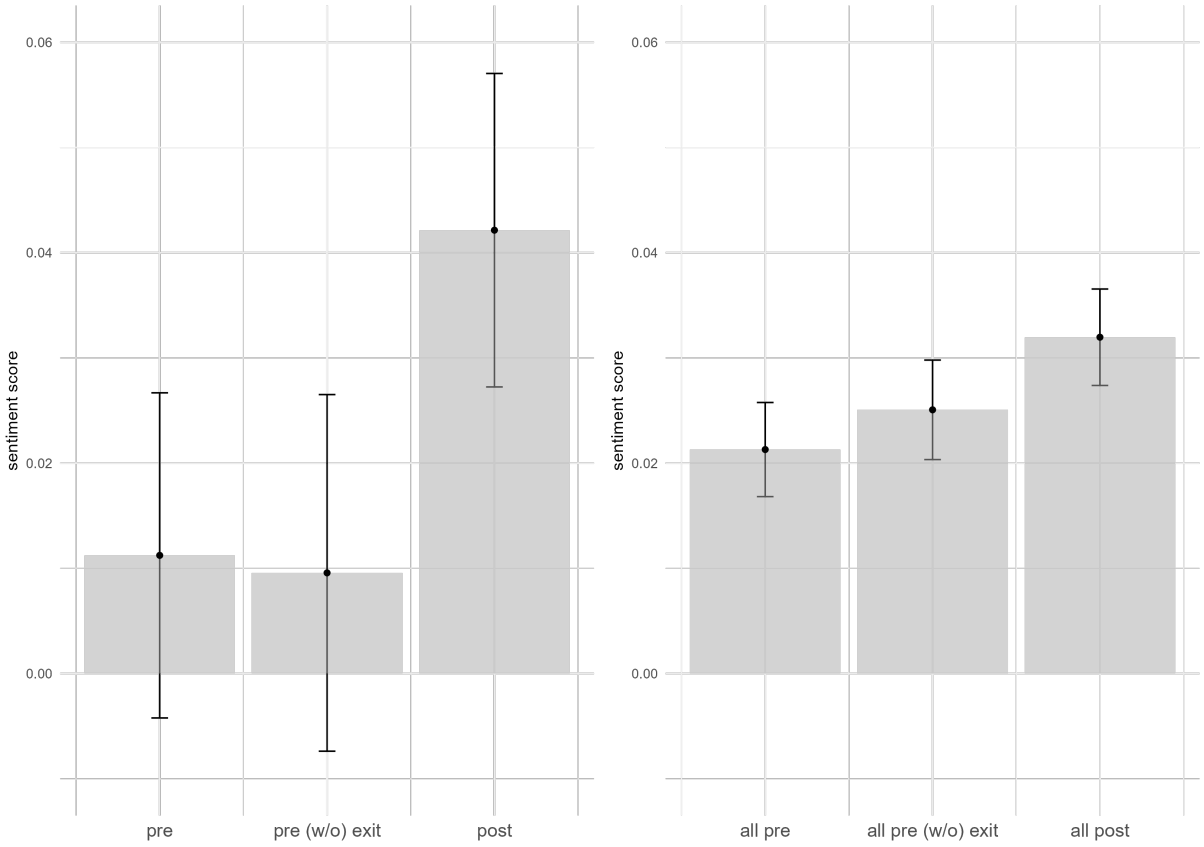


(c) Unique keywords



Notes: RDD plots using a linear model fit. Sample contains creators who did not exit the platform between the first and second waves of data collection.

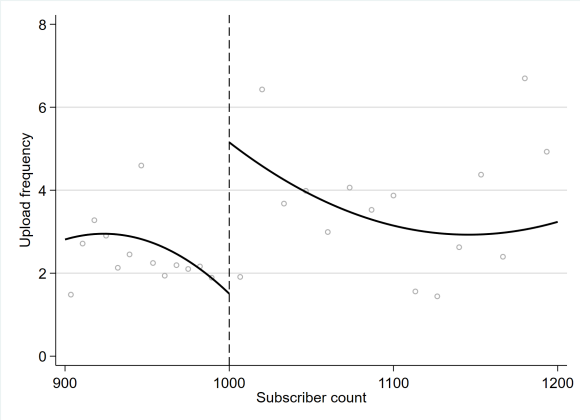
Figure D.2: Average sentiment scores of video titles



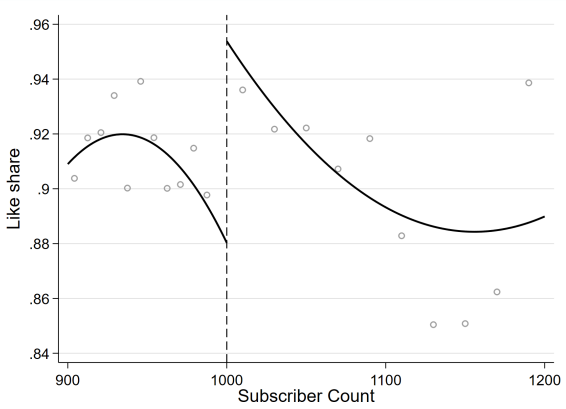
Notes: Average sentiment scores of video titles, before and after the policy change, of YouTubers within our main subscriber bandwidth. The left panel focuses on the average sentiment score for all YouTubers who were eventually demonetized. The right panel shows the average sentiment score for all YouTubers within our bandwidth.

Figure D.3: RDD plots: Robustness – Quadratic model fit

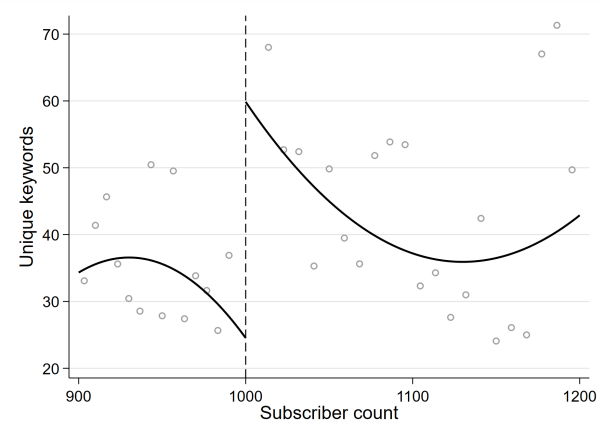
(a) Upload frequency



(b) Like share



(c) Unique keywords



Notes: RDD plots using a quadratic model fit. Sample contains creators who did not exit the platform between the first and second waves of data collection.

Bibliography

Abou El-Komboz, L. and Goldbeck, M. (2024). Career concerns as public good: The role of signaling for open source software development. *ifo Working Paper 405*.

Abreu, M., Faggian, A., and McCann, P. (2015). Migration and inter-industry mobility of uk graduates. *Journal of Economic Geography*, 15(2):353–385.

Acemoglu, D., Makhdoumi, A., Malekian, A., and Ozdaglar, A. (2022). Learning from reviews: The selection effect and the speed of learning. *Econometrica*, 90(6):2857–2899.

Acikalin, U. U., Caskurlu, T., Hoberg, G., and Phillips, G. M. (2022). Intellectual property protection lost and competition: An examination using machine learning. *NBER Working Paper 30671*.

Acs, Z. J., Anselin, L., and Varga, A. (2002). Patents and innovation counts as measures of regional production of new knowledge. *Research Policy*, 31(7):1069–1085.

Adrian, F., Miriam, F., and Michael, P. S. (2017). *The Human Face of Global Mobility: A Research Agenda*. Routledge.

Agrawal, A., Lacetera, N., and Lyons, E. (2016). Does standardized information in online markets disproportionately benefit job applicants from less developed countries? *Journal of International Economics*, 103:1–12.

Agrawal, B. A. and Goldfarb, A. (2008). Restructuring research: Communication costs and the democratization of university innovation. *American Economic Review*, 98(4):1578–1590.

Aguiar, L., Reimers, I., and Waldfogel, J. (2024). Platforms and the transformation of the content industries. *Journal of Economics & Management Strategy*, 33(2):317–326.

Aguiar, L. and Waldfogel, J. (2018). Quality Predictability and the Welfare Benefits from New Products: Evidence from the Digitization of Recorded Music. *Journal of Political Economy*, 126(2):492–524.

Aihounton, G. and Henningsen, A. (2021). Units of measurement and the inverse hyperbolic sine transformation. *The Econometrics Journal*, 24(2):334–351.

Alcácer, J. and Chung, W. (2007). Location strategies and knowledge spillovers. *Management Science*, 53(5):760–776.

Alexander, J. (2018). YouTube's lesser-known creators worry for the future after major monetization changes (update). *Polygon* (January 17). https://www.polygon.com/2018/1/17/16900474/youtube-monetization-small-creators-adsense.

Alexander, J. (2019). The golden age of youtube is over. *The Verge* (April 5). https://www.theverge.com/2019/4/5/18287318/youtube-logan-paul-pewdiepie-demonetization-adpocalypse-premium-influencers-creators.

Amior, M. (2015). Why are higher skilled workers more mobile geographically?: The role of the job surplus. *CEPR Working Paper 1338*.

Amior, M. (2019). Education and geographical mobility: The role of the job surplus. *CEP Working Paper 1616*.

Anderson, A., Huttenlocher, D., Kleinberg, J., and Leskovec, J. (2013). Steering User Behavior with Badges. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 95–106.

Anderson, C. (2004). The Long Tail. *Wired Magazine (October)*, pages 170–177.

Anderson, M. and Magruder, J. (2012). Learning from the crowd: Regression discontinuity estimates of the effects of an online review database. *The Economic Journal*, 122(563):957–989.

Anderson, S. P. and Gabszewicz, J. J. (2006). The media and advertising: a tale of two-sided markets. *Handbook of the Economics of Art and Culture*, 1:567–614.

Andersson, M., Klaesson, J., and Larsson, J. P. (2014). The sources of the urban wage premium by worker skills: Spatial sorting or agglomeration economies? *Papers in Regional Science*, 93(4):727–747.

Andreessen, M. (2011). Why software is eating the world. *Wall Street Journal*, 20(2011):C2.

Anenberg, E., Kuang, C., and Kung, E. (2022). Social learning and local consumption amenities: Evidence from yelp. *The Journal of Industrial Economics*, 70(2):294–322.

Angrist, J. D. (2014). The perils of peer effects. *Labour Economics*, 30:98–108.

Athey, S. and Ellison, G. (2014). Dynamics of open source movements. *Journal of Economics & Management Strategy*, 23(2):294–316.

Atkin, D., Chen, M. K., and Popov, A. (2022). The returns to face-to-face interactions: Knowledge spillovers in silicon valley. *NBER Working Paper 30147*.

Atkinson, R., Muro, M., and Whiton, J. (2019). The case for growth centers. *Brookings Institution: Washington, DC, USA*.

Au, C.-C. and Henderson, J. V. (2006). Are chinese cities too small? *The Review of Economic Studies*, 73(3):549–576.

Audretsch, D. and Feldmann, M. (1996). R&D spillovers and the geography of innovation and production. *American Economic Review*, 86:630–640.

Aum, S. and Shin, Y. (2024). Is software eating the world? *NBER Working Paper 32591*.

Autor, D. H., Dorn, D., and Hanson, G. H. (2013). The china syndrome: Local labor market effects of import competition in the united states. *American Economic Review*, 103(6):2121–2168.

Ayoubi, C., Pezzoni, M., and Visentin, F. (2017). At the origins of learning: Absorbing knowledge flows from within the team. *Journal of Economic Behavior & Organization*, 134:374–387.

Azoulay, P., Graff Zivin, J. S., and Wang, J. (2010). Superstar extinction. *The Quarterly Journal of Economics*, 125(2):549–589.

Badashian, A. S., Esteki, A., Gholipour, A., Hindle, A., and Stroulia, E. (2014). Involvement, contribution and influence in github and stack overflow. In *Proceedings of 24th Annual International Conference on Computer Science and Software Engineering*, pages 19–33.

Bahar, D., Choudhury, P., Kim, D. Y., and Koo, W. W. (2022). Innovation on wings: Nonstop flights and firm innovation in the global context. *Management Science*.

Bailey, M., Cao, R., Kuchler, T., and Stroebel, J. (2018). The economic effects of social networks: Evidence from the housing market. *Journal of Political Economy*, 126(6):2224–2276.

Bailey, M., Johnston, D., Kuchler, T., Stroebel, J., and Wong, A. (2022). Peer effects in product adoption. *American Economic Journal: Applied Economics*, 14(3):488–526.

Bakshy, E., Rosenn, I., Marlow, C., and Adamic, L. (2012). The role of social networks in information diffusion. In *Proceedings of the 21st international conference on World Wide Web*, pages 519–528.

Baldwin, R. (2017). *The Great Convergence: Information Technology and the New Globalization*. Harvard University Press.

Baldwin, R. and Dingel, J. I. (2022). Telemigration and development: On the offshorability of teleworkable jobs. In *Robots and AI*, pages 150–179. Routledge.

Balgova, M. (2020). Leaping into the unknown? the role of job search in migration decisions. mimeo.

Balland, P. A., Jara-Figueroa, C., Petralia, S. G., Steijn, M. P., Rigby, D. L., and Hidalgo, C. A. (2020). Complex economic activities concentrate in large cities. *Nature Human Behaviour*, pages 1–10.

Banerjee, A. V. (1992). A simple model of herd behavior. *The Quarterly Journal of Economics*, 107(3):797–817.

Bean, R. (2024). How moderna is embracing data and ai to transform drug discovery. *Forbes*.

Becker, R. A. and Wilks, A. R. (2018). Package 'maps'. `https://cran.r-project.org/web/packages/maps/maps.pdf`. Accessed: 2021-05-11.

Bellégo, C., Benatia, D., and Pape, L. (2022). Dealing with logs and zeros in regression models. *CREST Working Paper*.

Bellemare, M. F. and Wichman, C. J. (2020). Elasticities and the inverse hyperbolic sine transformation. *Oxford Bulletin of Economics and Statistics*, 82(1):50–61.

Ben-Porath, Y. (1967). The production of human capital and the life cycle of earnings. *Journal of Political Economy*, 75(4):352–365.

Bergstrom, T., Blume, L., and Varian, H. (1986). On the private provision of public goods. *Journal of Public Economics*, 29(1):25–49.

Bettencourt, L. M. A., Lobo, J., Helbing, D., Kühnert, C., and West, G. B. (2007). Growth, innovation, scaling, and the pace of life in cities. *Proceedings of the National Academy of Sciences*, 104(17):7301–7306.

Bhargava, H. K. (2022). The creator economy: Managing ecosystem supply, revenue sharing, and platform design. *Management Science*, 68(7):5233–5251.

Bikard, M. and Marx, M. (2020). Bridging academia and industry: How geographic hubs connect university science and corporate technology. *Management Science*, 66(8):3425–3443.

Bikhchandani, S., Hirshleifer, D., Tamuz, O., and Welch, I. (2024). Information cascades and social learning. *Journal of Economic Literature*, 62(3):1040–1093.

Bikhchandani, S., Hirshleifer, D., and Welch, I. (1992). A theory of fads, fashion, custom, and cultural change as informational cascades. *Journal of Political Economy*, 100(5):992–1026.

Bilodeau, M. and Slivinski, A. (1996). Toilet cleaning and department chairing: Volunteering a public service. *Journal of Public Economics*, 59(2):299–308.

Bissyandé, T. F., Lo, D., Jiang, L., Réveillere, L., Klein, J., and Le Traon, Y. (2013). Got issues? who cares about it? a large scale investigation of issue trackers from github. In *2013 IEEE 24th international symposium on software reliability engineering (ISSRE)*, pages 188–197. IEEE.

Bitzer, J. and Geishecker, I. (2010). Who contributes voluntarily to oss? an investigation among german it employees. *Research Policy*, 39(1):165–172.

Bitzer, J., Schrettl, W., and Schröder, P. J. H. (2007). Intrinsic motivation in open source software development. *Journal of Comparative Economics*, 35(1):160–169.

Bitzer, J. and Schröder, P. J. H. (2005). Bug-fixing and code-writing: The private provision of open source software. *Information Economics and Policy*, 17(3):389–406.

Bitzer, J. and Schröder, P. J. H. (2007). Open source software, competition and innovation. *Industry and Innovation*, 14(5):461–476.

Blau, G. (1994). Testing a two-dimensional measure of job search behavior. *Organizational Behavior and Human Decision Processes*, 59(2):288–312.

Blincoe, K., Sheoran, J., Goggins, S., Petakovic, E., and Damian, D. (2016). Understanding the popular users: Following, affiliation influence and leadership on github. *Information and Software Technology*, 70:30–39.

Bliss, C. and Nalebuff, B. (1984). Dragon-slaying and ballroom dancing: The private supply of a public good. *Journal of Public Economics*, 25(1-2):1–12.

Bloom, N., Schankerman, M., and Van Reenen, J. (2013). Identifying technology spillovers and product market rivalry. *Econometrica*, 81(4):1347–1393.

Boldrin, M. and Levine, D. K. (2013). The case against patents. *Journal of Economic Perspectives*, 27(1):3–22.

Bonina, C., Koskinen, K., Eaton, B., and Gawer, A. (2021). Digital platforms for development: Foundations and research agenda. *Information Systems Journal*, 31(6):869–902.

Borges, H., Hora, A., and Valente, M. T. (2016). Understanding the factors that impact the popularity of github repositories. In *2016 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 334–344. IEEE.

Boudreau, K. J. (2010). Open Platform Strategies and Innovation: Granting Access vs. Devolving Control. *Management Science*, 56(10):1849–1872.

Boudreau, K. J. (2012). Let A Thousand Flowers Bloom? An Early Look at Large Numbers of Software App Developers and Patterns of Innovation. *Organization Science*, 23(5):1409–1427.

Boudreau, K. J. and Hagiu, A. (2009). Platform Rules: Multi-sided Platforms as Regulators. *Platforms, Markets and Innovation*, 1:163–191.

Branch, T. A., Côté, I. M., David, S. R., Drew, J. A., LaRue, M., Márquez, M. C., Parsons, E. C. M., Rabaiotti, D., Shiffman, D., Steen, D. A., et al. (2024). Controlled experiment finds no detectable citation bump from twitter promotion. *Plos one*, 19(3):e0292201.

Bresnahan, T. F., Brynjolfsson, E., and Hitt, L. M. (2002). Information technology, workplace organization, and the demand for skilled labor: Firm-level evidence. *The Quarterly Journal of Economics*, 117(1):339–376.

Brucks, M. S. and Levav, J. (2022). Virtual communication curbs creative idea generation. *Nature*, 605(7908):108–112.

Brynjolfsson, E., Hu, Y., and Smith, M. D. (2003). Consumer surplus in the digital economy: Estimating the value of increased product variety at online booksellers. *Management Science*, 49(11):1580–1596.

Buera, F. J. and Kaboski, J. P. (2012). The rise of the service economy. *American Economic Review*, 102(6):2540–2569.

Burbidge, J. B., Magee, L., and Robb, A. L. (1988). Alternative transformations to handle extreme values of the dependent variable. *Journal of the American Statistical Association*, 83(401):123–127.

Burtch, G., He, Q., Hong, Y., and Lee, D. (2022). How do peer awards motivate creative content? experimental evidence from reddit. *Management Science*, 68(5):3488–3506.

Burtch, G., Hong, Y., Bapna, R., and Griskevicius, V. (2018). Stimulating Online Reviews by Combining Financial Incentives and Social Norms. *Management Science*, 64(5):2065–2082.

Butler, S., Gamalielsson, K., Lundell, B., Brax, C., Sjöberg, J., Mattsson, A., Gustavsson, T., Feist, J., and Lönroth, E. (2019). On company contributions to community open source software projects. *IEEE Transactions on Software Engineering*, 47(7):1381–1401.

Cabral, L. and Li, L. (2015). A Dollar for Your Thoughts: Feedback-Conditional Rebates on eBay. *Management Science*, 61(9):2052–2063.

Cai, H., Chen, Y., and Fang, H. (2009). Observational learning: Evidence from a randomized natural field experiment. *American Economic Review*, 99(3):864–882.

Cairncross, F. (1997). *The Death of Distance: How the Communications Revolution Will Change Our Lives*. Harvard Business School Press.

Calonico, S., Cattaneo, M. D., and Farrell, M. H. (2020). Optimal Bandwidth Choice for Robust Bias-corrected Inference in Regression Discontinuity Designs. *The Econometrics Journal*, 23(2):192–210.

Carlino, G. and Kerr, W. R. (2015). Agglomeration and innovation. *Handbook of Regional and Urban Economics*, 5:349–404.

Carlino, G. A., Carr, J., Hunt, R. M., and Smith, T. E. (2012). The agglomeration of r&d labs. *Federal Reserve Bank of Philadelphia*.

Carlino, G. A., Chatterjee, S., and Hunt, R. M. (2007). Urban density and the rate of invention. *Journal of Urban Economics*, 61(3):389–419.

Casalnuovo, C., Vasilescu, B., Devanbu, P., and Filkov, V. (2015). Developer onboarding in GitHub: the role of prior social links and language experience. In *Proceedings of the 2015 10th joint meeting on foundations of software engineering*, pages 817–828.

Catalini, C. (2018). Microgeography and the direction of inventive activity. *Management Science*, 64(9):4348–4364.

Cattaneo, M. D., Crump, R. K., Farrell, M. H., and Feng, Y. (2021). Binscatter regressions. *Stat. J. vv*, pages 1–49.

Cattaneo, M. D., Crump, R. K., Farrell, M. H., and Feng, Y. (2024). On binscatter. *American Economic Review*, 114(5):1488–1514.

Cattaneo, M. D., Jansson, M., and Ma, X. (2017). Simple local regression distribution estimators. mimeo.

Ceccagnoli, M., Forman, C., Huang, P., and Wu, D. (2012). Cocreation of Value in a Platform Ecosystem! The Case of Enterprise Software. *MIS Quarterly*, pages 263–290.

Chamberlain, A. (2015). Why is hiring taking longer. *Glassdoor Policy Brief*.

Chamberlin, J. (1974). Provision of collective goods as a function of group size. *American Political Science Review*, 68(2):707–716.

Chan, H. F., Önder, A. S., Schweitzer, S., and Torgler, B. (2023). Twitter and citations. *Economics Letters*, 231:111270.

Chattergoon, B. and Kerr, W. R. (2022). Winner takes all? tech clusters, population centers, and the spatial transformation of us invention. *Research Policy*, 51(2):104418.

Chen, J. and Roth, J. (2024). Logs with zeros? some problems and solutions. *The Quarterly Journal of Economics*, 139(2):891–936.

Chevalier, J. and Ellison, G. (1999). Career concerns of mutual fund managers. *The Quarterly Journal of Economics*, 114(2):389–432.

Ciriaci, D. (2014). Does university quality influence the interregional mobility of students and graduates? the case of italy. *Regional Studies*, 48(10):1592–1608.

Claussen, J., Kretschmer, T., and Mayrhofer, P. (2013). The Effects of Rewarding User Engagement: The Case of Facebook Apps. *Information Systems Research*, 24(1):186–200.

Coelho, J., Valente, M. T., Silva, L. L., and Shihab, E. (2018). Identifying unmaintained projects in github. In *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, pages 1–10.

Cohen, J. E. and Lemley, M. A. (2001). Patent scope and innovation in the software industry. *Calif. L. Rev.*, 89:1.

Cohn, J. B., Liu, Z., and Wardlaw, M. I. (2022). Count (and count-like) data in finance. *Journal of Financial Economics*, 146(2):529–551.

Colombo, M. G., Piva, E., and Rossi-Lamastra, C. (2014). Open innovation and within-industry diversification in small and medium enterprises: The case of open source software firms. *Research Policy*, 43(5):891–902.

Combes, P.-P., Duranton, G., Gobillon, L., and Roux, S. (2010). Estimating agglomeration economies with history, geology, and worker effects. In *Agglomeration economics*, pages 15–66. University of Chicago Press.

Combes, P.-P. and Gobillon, L. (2015). The empirics of agglomeration economies. In *Handbook of Regional and Urban Economics*, volume 5, pages 247–348. Elsevier.

Constantinides, P., Henfridsson, O., and Parker, G. G. (2018). Introduction—Platforms and Infrastructures in the Digital Age. *Information Systems Research*, 29(2):381–400.

Conti, A., Gupta, V., Guzman, J., and Roche, M. P. (2023). Incentivizing innovation in open source: Evidence from the github sponsor program. *NBER Working Paper 31668*.

Conti, A., Peukert, C., and Roche, M. P. (2021). Beefing it up for your investor? open sourcing and startup funding: Evidence from github. *Harvard Business School Working Paper 22-001*.

Cornelissen, T., Dustmann, C., and Schönberg, U. (2017). Peer effects in the workplace. *American Economic Review*, 107(2):425–56.

Correia, S., Guimarães, P., and Zylkin, T. (2020). Fast poisson estimation with high-dimensional fixed effects. *The Stata Journal*, 20(1):95–115.

Cusumano, M. A., Gawer, A., and Yoffie, D. B. (2019). *The business of platforms: Strategy in the age of digital competition, innovation, and power*, volume 320. Harper Business New York.

Dabbish, L., Stuart, C., Tsay, J., and Herbsleb, J. (2012). Social coding in github: transparency and collaboration in an open software repository. In *Proceedings of the ACM 2012 conference on computer supported cooperative work*, pages 1277–1286.

De La Roca, J. and Puga, D. (2017). Learning by working in big cities. *The Review of Economic Studies*, 84(1):106–142.

De Reuver, M., Sørensen, C., and Basole, R. C. (2018). The digital platform: a research agenda. *Journal of Information Technology*, 33(2):124–135.

del Rio-Chanona, M., Laurentsyeva, N., and Wachs, J. (2023). Are large language models a threat to digital public goods? evidence from activity on stack overflow. *arXiv preprint arXiv:2307.07367*.

Demary, V. and Rusche, C. (2018). The economics of platforms. *IW-Analysen*, 123.

Deming, D. and Noray, K. (2020a). Earnings dynamics, changing job skills, and STEM careers. *The Quarterly Journal of Economics*, pages 1965–2005.

Deming, D. J. and Noray, K. (2020b). Earnings dynamics, changing job skills, and stem careers. *The Quarterly Journal of Economics*, 135(4):1965–2005.

Dinerstein, M., Einav, L., Levin, J., and Sundaresan, N. (2018). Consumer Price Search and Platform Design in Internet Commerce. *American Economic Review*, 108(7):1820–1859.

Dohmke, T. (2023). 100 million developers and counting. *GitHub Blog*.

Draca, M., Sadun, R., and van Reenen, J. (2007). Ict and productivity: A review of the evidence. *Handbook of Information and Communication Technologies*.

D'Mello, M. and Sahay, S. (2007). 'i am kind of a nomad where i have to go places and places...': Understanding mobility, place and identity in global software work from india. *Information and Organization*, 17(3):162–192.

Eaton, B., Elaluf-Calderwood, S., Sørensen, C., and Yoo, Y. (2015). Distributed Tuning of Boundary Resources: The Case of Apple's iOS Service System. *MIS Quarterly*, 39(1):217–244.

Eisenmann, T., Parker, G., and Van Alstyne, M. W. (2006). Strategies for Two-Sided Markets. *Harvard Business Review*, 84:92–101.

Elazhary, O., Storey, M., Ernst, N., and Zaidman, A. (2019). Do as i do, not as i say: Do contribution guidelines match the github contribution process? In *2019 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 286–290. IEEE.

Ellison, G. and Glaeser, E. L. (1997). Geographic concentration in us manufacturing industries: a dartboard approach. *Journal of Political Economy*, 105(5):889–927.

Fackler, T. and Laurentsyeva, N. (2020). Gravity in online collaborations: Evidence from GitHub. *CESifo Forum*, 21(3).

Falck, O., Heimisch-Roecker, A., and Wiederhold, S. (2021). Returns to ict skills. *Research Policy*, 50(7):104064.

Fershtman, C. and Gandal, N. (2004). The determinants of output per contributor in open source projects: An empirical examination. *CEPR Working Paper 4329*.

Fershtman, C. and Gandal, N. (2011). Direct and Indirect Knowledge Spillovers: The "Social Network" of Open-Source Projects. *The RAND Journal of Economics*, 42(1):70–91.

Finch, T., O'Hanlon, N., and Dudley, S. P. (2017). Tweeting birds: online mentions predict future citations in ornithology. *Royal Society Open Science*, 4(11):171371.

Firaz, A. (2022). How long do software engineers stay at a job? *LinkedIn Blog*.

Flammer, C. (2015). Does corporate social responsibility lead to superior financial performance? a regression discontinuity approach. *Management Science*, 61(11):2549–2568.

Foerderer, J., Lueker, N., and Heinzl, A. (2021). And the winner is…? the desirable and undesirable effects of platform awards. *Information Systems Research*, 32(4):1155–1172.

Forbes (2023). The global 2000. `https://www.forbes.com/lists/global2000/`. Accessed: 2024-06-12.

Forman, C. and Goldfarb, A. (2022). Concentration and agglomeration of it innovation and entrepreneurship: Evidence from patenting. In *The Role of Innovation and Entrepreneurship in Economic Growth*, pages 95–122. University of Chicago Press.

Fuller, J., Langer, C., and Sigelman, M. (2022). Skills-based hiring is on the rise. *Harvard Business Review*, 11.

Gallus, J. (2017). Fostering Public Good Contributions with Symbolic Awards: A Large-Scale Natural Field Experiment at Wikipedia. *Management Science*, 63(12):3999–4015.

Ganguli, I., Lin, J., and Reynolds, N. (2020). The paper trail of knowledge spillovers: Evidence from patent interferences. *American Economic Journal: Applied Economics*, 12(2):278–302.

Gawer, A. and Henderson, R. (2007). Platform Owner Entry and Innovation in Complementary Markets: Evidence from Intel. *Journal of Economics & Management Strategy*, 16(1):1–34.

Gelman, A. (2018). You need 16 times the sample size to estimate an interaction than to estimate a main effect. *Statistical Modeling, Causal Inference, and Social Science Blog* (March 15). `https://statmodeling.stat.columbia.edu/2018/03/15/need-16-times-sample-size-estimate-interaction-estimate-main-effect/`.

Gelman, A. and Imbens, G. (2019). Why high-order polynomials should not be used in regression discontinuity designs. *Journal of Business & Economic Statistics*, 37(3):447–456.

Gentzkow, M., Kelly, B., and Taddy, M. (2019). Text as data. *Journal of Economic Literature*, 57(3):535–574.

Gerosa, M., Wiese, I., Trinkenreich, B., Link, G., Robles, G., Treude, C., Steinmacher, I., and Sarma, A. (2021). The shifting sands of motivation: Revisiting what drives contributors in open source. *IEEE International Conference on Software Engineering (ICSE)*, pages 1046–1058.

Geva, H., Barzilay, O., and Oestreicher-Singer, G. (2019). A Potato Salad with a Lemon Twist: Using a Supply-Side Shock to Study the Impact of Opportunistic Behavior on Crowdfunding Platforms. *MIS Quarterly*, 43(4):1227–1248.

Ghazawneh, A. and Henfridsson, O. (2013). Balancing Platform Control and External Contribution in Third-Party Development: The Boundary Resources Model. *Information Systems Journal*, 23(2):173–192.

Gillespie, T. (2018). The Logan Paul YouTube controversy and what we should expect from internet platforms. *Vox* (January 16). `https://www.vox.com/the-big-idea/2018/1/12/16881046/logan-paul-youtube-controversy-internet-companies`.

GIT (2021). `https://git-scm.com/`. Accessed: 2021-05-31.

GitHub (2021). The 2021 state of the octoverse. *Company Report*.

Glaeser, E. L. (1999). Learning in cities. *Journal of Urban Economics*, 46(2):254–277.

Glaeser, E. L., Sacerdote, B. I., and Scheinkman, J. A. (2003). The social multiplier. *Journal of the European Economic Association*, 1(2-3):345–353.

Goes, P. B., Guo, C., and Lin, M. (2016). Do Incentive Hierarchies Induce User Effort? Evidence from an Online Knowledge Exchange. *Information Systems Research*, 27(3):497–516.

Goldbeck, M. (2023). Bit by bit: Colocation and the death of distance in software developer networks. *CRC Discussion Paper 422*.

Goldfarb, A. and Tucker, C. (2019). Digital economics. *Journal of Economic Literature*, 57(1):3–43.

Gousios, G. (2013). The ghtorent dataset and tool suite. *IEEE Conference on Mining Software Repositories (MSR)*, pages 233–236.

Granovetter, M. S. (1973). The strength of weak ties. *American Journal of Sociology*, 78(6):1360–1380.

Greenwood, M. J. (1973). The geographic mobility of college graduates. *The Journal of Human Resources*, 8(4):506–515.

Greenwood, M. J. (1975). Research on internal migration in the united states: A survey. *Journal of Economic Literature*, pages 397–433.

Grisold, T., Gau, M., and Yoo, Y. (2021). Coding like a rockstar: The role of social influence on action patterns in github. In *ICIS*.

Guzman, J. and Stern, S. (2020). The state of american entrepreneurship: New estimates of the quantity and quality of entrepreneurship for 32 us states, 1988–2014. *American Economic Journal: Economic Policy*, 12(4):212–243.

Haapanen, M. and Tervo, H. (2012). Migration of the highly educated: Evidence from residence spells of university graduates. *Journal of Regional Science*, 52(4):587–605.

Hagiu, A. (2007). Merchant or two-sided platform? *Review of Network Economics*, 6(2).

Hakim Orman, W. (2008). Giving it away for free? the nature of job-market signaling by open-source software developers. *The BE Journal of Economic Analysis & Policy*, 8(1).

Hann, I.-H., Roberts, J. A., and Slaughter, S. (2004). Why developers participate in open source software projects: An empirical investigation. *International Conference on Information Systems (ICIS)*.

Hann, I.-H., Roberts, J. A., and Slaughter, S. A. (2013). All are not equal: An examination of the economic returns to different forms of participation in open source software communities. *Information Systems Research*, 24(3):520–538.

Hars, A. and Ou, S. (2002). Working for free? motivations for participating in open-source projects. *International Journal of Electronic Commerce*, 6(3):25–39.

Haussen, T. and Uebelmesser, S. (2018). Job changes and interregional migration of graduates. *Regional Studies*, 52(10):1346–1359.

Hendricks, K., Weiss, A., and Wilson, C. (1988). The war of attrition in continuous time with complete information. *International Economic Review*, pages 663–680.

Herbst, D. and Mas, A. (2015). Peer effects on worker output in the laboratory generalize to the field. *Science*, 350(6260):545–549.

Hersche, M. and Moor, E. (2020). Identification and estimation of intensive margin effects by difference-in-difference methods. *Journal of Causal Inference*, 8(1):272–285.

Hertel, G., Niedner, S., and Herrmann, S. (2003). Motivation of software developers in open source projects: An internet-based survey of contributors to the linux kernel. *Research Policy*, 32(7):1159–1177.

Hindle, A., German, D. M., and Holt, R. (2008). What do large commits tell us? a taxonomical study of large commits. In *Proceedings of the 2008 international working conference on Mining software repositories*, pages 99–108.

Ho, S. Y. and Rai, A. (2017). Continued voluntary participation intention in firm-participating open source software projects. *Information Systems Research*, 28(3):603–625.

Hoffmann, M., Nagle, F., and Zhou, Y. (2024). The value of open source software. *Harvard Business School Strategy Unit Working Paper 24-038*.

Holmström, B. (1999). Managerial incentive problems: A dynamic perspective. *The Review of Economic Studies*, 66(1):169–182.

Huang, P., Ceccagnoli, M., Forman, C., and Wu, D. (2013). Appropriability mechanisms and the platform partnership decision: Evidence from enterprise software. *Management Science*, 59(1):102–121.

Huang, P., Lyu, G., and Xu, Y. (2022). Quality regulation on two-sided platforms: Exclusion, subsidization, and first-party applications. *Management Science*, 68(6):4415–4434.

Huang, P. and Zhang, Z. (2016). Participation in open knowledge communities and job-hopping. *MIS Quarterly*, 40(3):785–806.

Huang, Y. and Chung, W. (2019). Rockstar effect in distributed project management: A study of github social networks. In *2019 Pre-ICIS SIGDSA Symposium*. Association for Information Systems.

Hukal, P., Henfridsson, O., Shaikh, M., and Parker, G. (2020). Platform Signaling for Generating Platform Content. *MIS Quarterly*, 44(3):1177–1205.

Imbens, G. W. and Angrist, J. D. (1994). Identification and estimation of local average treatment effects. *Econometrica*, pages 467–475.

Imbens, G. W. and Lemieux, T. (2008). Regression Discontinuity Designs: A Guide to Practice. *Journal of Econometrics*, 142(2):615–635.

Jackson, C. K. and Bruegmann, E. (2009). Teaching students and teaching each other: The importance of peer learning for teachers. *American Economic Journal: Applied Economics*, 1(4):85–108.

Jacobides, M. G., Cennamo, C., and Gawer, A. (2018). Towards a Theory of Ecosystems. *Strategic Management Journal*, 39(8):2255–2276.

Jaffe, A. B., Trajtenberg, M., and Henderson, R. (1993). Geographic localization of knowledge spillovers as evidenced by patent citations. *The Quarterly Journal of Economics*, 108.

Jain, S. and Qian, K. (2021). Compensating online content producers: A theoretical analysis. *Management Science*, 67(11):7075–7090.

Jhaver, S., Karpfen, Y., and Antin, J. (2018). Algorithmic Anxiety and Coping Strategies of Airbnb Hosts. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–12.

Johnson, J. P. (2002). Open Source Software: Private Provision of a Public Good. *Journal of Economics & Management Strategy*, 11(4):637–662.

Johnson, K. P. and Kort, J. R. (2004). 2004 redefinition of the bea economic areas. *Survey of Current Business*, 75(2):75–81.

Jones, B. F. (2009). The burden of knowledge and the ”death of the renaissance man”: Is innovation getting harder? *Review of Economic Studies*, 76(1):283–317.

Jones, B. F. (2010). Age and great invention. *The Review of Economics and Statistics*, 92(1):1–14.

Kaltenberg, M., Jaffe, A. B., and Lachman, M. E. (2023). Invention and the life course: Age differences in patenting. *Research Policy*, 52(1):104629.

Kapoor, R. and Agarwal, S. (2017). Sustaining Superior Performance in Business Ecosystems: Evidence from Application Software Developers in the iOS and Android Smartphone Ecosystems. *Organization Science*, 28(3):531–551.

Kerkhof, A. (2024). Advertising and content differentiation: Evidence from youtube. *The Economic Journal*, page ueae043.

Kerr, W. R. and Robert-Nicoud, F. (2020). Tech clusters. *Journal of Economic Perspectives*, 34(3):50–76.

Khern-am-nuai, W., Kannan, K., and Ghasemkhani, H. (2018). Extrinsic Versus Intrinsic Rewards for Contributing Reviews in an Online Platform. *Information Systems Research*, 29(4):871–892.

Khondhu, J., Capiluppi, A., and Stol, K.-J. (2013). Is it all lost? a study of inactive open source projects. In *Open Source Software: Quality Verification: 9th IFIP WG 2.13 International Conference, OSS 2013, Koper-Capodistria, Slovenia, June 25-28, 2013. Proceedings 9*, pages 61–79. Springer.

Kodrzycki, Y. K. (2001). Migration of recent college graduates: Evidence from the national longitudinal survey of youth. *New England Economic Review*, pages 13–34.

Koetsier, J. (2020). YouTube Will Now Show Ads On All Videos Even If Creators Don’t Want Them. *Forbes* (November 18). `https://www.forbes.com/sites/johnkoetsier/2020/11/`

18/youtube-will-now-show-ads-on-all-videos-even-if-creators-dont-want-them/?sh=155ae06e4913.

Kogan, L., Papanikolaou, D., Seru, A., and Stoffman, N. (2017). Technological innovation, resource allocation, and growth. *The Quarterly Journal of Economics*, 132(2):665–712.

Koo, W. W. and Eesley, C. E. (2020). Platform Governance and the Rural–Urban Divide: Sellers' Responses to Design Change. *Strategic Management Journal*.

Korkmaz, G., Calderón, J. B. S., Kramer, B. L., Guci, L., and Robbins, C. A. (2024). From github to gdp: A framework for measuring open source software innovation. *Research Policy*, 53(3):104954.

Krishnamurthy, S. (2006). On the intrinsic and extrinsic motivation of free/libre/open source (floss) developers. *Knowledge, Technology & Policy*, 18(4):17–39.

Krishnamurthy, S., Ou, S., and Tripathi, A. K. (2014). Acceptance of monetary rewards in open source software development. *Research Policy*, 43(4):632–644.

Lakhani, K. R. and von Hippel, E. (2003). How Open Source Software Works: "Free" User-to-User Assistance. *Research Policy*, 32(6):923–943.

Lakhani, K. R. and Wolf, R. G. (2005). Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects. In Feller, J., Brian, F., A., H. S., and Lakhani, K. R., editors, *Perspectives on Free and Open Source Software.*, pages 3–21. MIT Press, Cambridge, MA.

Langenkamp, M. and Yue, D. N. (2022). How open source machine learning software shapes ai. In *Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society*, pages 385–395.

Larch, M., Wanner, J., Yotov, Y. V., and Zylkin, T. (2019). Currency unions and trade: A ppml re-assessment with high-dimensional fixed effects. *Oxford Bulletin of Economics and Statistics*, 81(3):487–510.

Lee, D. S. (2008). Randomized experiments from non-random selection in us house elections. *Journal of Econometrics*, 142(2):675–697.

Lee, D. S. and Lemieux, T. (2010). Regression Discontinuity Designs in Economics. *Journal of Economic Literature*, 48(2):281–355.

Lee, G. K. and Cole, R. E. (2003). From a firm-based to a community-based model of knowledge creation: The case of the linux kernel development. *Organization Science*, 14(6):633–649.

Lee, M. J., Ferwerda, B., Choi, J., Hahn, J., Moon, J. Y., and Kim, J. (2013). Github developers use rockstars to overcome overflow of news. In *CHI'13 Extended Abstracts on Human Factors in Computing Systems*, pages 133–138.

Leppämäki, M. and Mustonen, M. (2009). Skill signalling with product market externality. *The Economic Journal*, 119(539):1130–1142.

Lerner, J. and Seru, A. (2022). The use and misuse of patent data: Issues for finance and beyond. *The Review of Financial Studies*, 35(6):2667–2704.

Lerner, J. and Tirole, J. (2002). Some simple economics of open source. *The Journal of Industrial Economics*, 50(2):197–234.

Lerner, J. and Tirole, J. (2005a). The economics of technology sharing: Open source and beyond. *Journal of Economic Perspectives*, 19(2):99–120.

Lerner, J. and Tirole, J. (2005b). The scope of open source licensing. *Journal of Law, Economics, and Organization*, 21(1):20–56.

Lifshitz-Assaf, H. and Nagle, F. (2021). The digital economy runs on open source. here's how to protect it. *Harvard Business Review*.

Lima, A., Rossi, L., and Musolesi, M. (2014). Coding together at scale: Github as a collaborative social network. *Proceedings of the international AAAI conference on web and social media*, 8(1):295–304.

Lin, Y.-K. and Rai, A. (2024). The scope of software patent protection in the digital age: evidence from alice. *Information Systems Research*, 35(2):657–672.

Locke, E. A. and Latham, G. P. (2002). Building a Practically Useful Theory of Goal Setting and Task Motivation: A 35-year Odyssey. *American Psychologist*, 57(9):705.

Loh, J. and Kretschmer, T. (2023). Online communities on competing platforms: Evidence from game wikis. *Strategic Management Journal*, 44(2):441–476.

Long, J. (2009). Open source software development experiences on the students' resumes: Do they count? *Journal of Information Technology Education: Research*, 8(1):229–242.

Luca, M. (2015). User-Generated Content and Social Media. In *Handbook of Media Economics*, volume 1, pages 563–592. Elsevier.

Luca, M. (2017). Designing online marketplaces: Trust and reputation mechanisms. *Innovation Policy and the Economy*, 17(1):77–93.

Lucas, R. E. (1988). On the mechanics of economic development. *Journal of Monetary Economics*, 22(1):3–42.

Ma, M. and Agarwal, R. (2007). Through a Glass Darkly: Information Technology Design, Identity Verification, and Knowledge Contribution in Online Communities. *Information Systems Research*, 18(1):42–67.

Machin, S., Salvanes, K. G., and Pelkonen, P. (2012). Education and mobility. *Journal of the European Economic Association*, 10(2):417–450.

MacKinnon, J. G. and Magee, L. (1990). Transforming the dependent variable in regression models. *International Economic Review*, pages 315–339.

Maggi, L., Gkatzikis, L., Paschos, G., and Leguay, J. (2018). Adapting caching to audience retention rate. *Computer Communications*, 116:159–171.

Manski, C. F. (1993a). Dynamic choice in social settings: Learning from the experiences of others. *Journal of Econometrics*, 58(1-2):121–136.

Manski, C. F. (1993b). Identification of endogenous social effects: The reflection problem. *The Review of Economic Studies*, 60(3):531–542.

Marlow, J. and Dabbish, L. (2013). Activity traces and signals in software developer recruitment and hiring. *ACM SIGCHI Conference on Computer Supported Cooperative Work (CSCW)*, pages 145–156.

Mas, A. and Moretti, E. (2009). Peers at work. *American Economic Review*, 99(1):112–45.

McDonald, N. and Goggins, S. (2013). Performance and participation in open source software on GitHub. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, pages 139–144. CHI EA '13, Paris, France.

Mens, T., Goeminne, M., Raja, U., and Serebrenik, A. (2014). Survivability of software projects in gnome–a replication study. In *7th International seminar series on advanced techniques & tools for software evolution (SATToSE)*, pages 79–82.

Miklós-Thal, J. and Ullrich, H. (2015). Belief precision and effort incentives in promotion contests. *The Economic Journal*, 125(589):1952–1963.

Miklós-Thal, J. and Ullrich, H. (2016). Career prospects and effort incentives: Evidence from professional soccer. *Management Science*, 62(6):1645–1667.

Mohan, N. and Kyncl, R. (2018). Additional Changes to the YouTube Partner Program (YPP) to Better Protect Creators. *YouTube Official Blog* (January 16). https://blog.youtube/news-and-events/additional-changes-to-youtube-partner.

Moretti, E. (2004). Workers' education, spillovers, and productivity: evidence from plant-level production functions. *American Economic Review*, 94(3):656–690.

Moretti, E. (2011). Social learning and peer effects in consumption: Evidence from movie sales. *The Review of Economic Studies*, 78(1):356–393.

Moretti, E. (2021). The effect of high-tech clusters on the productivity of top inventors. *American Economic Review*, 111(10):3328–75.

Mullahy, J. and Norton, E. C. (2022). Why transform y? a critical assessment of dependent-variable transformations in regression models for skewed and sometimes-zero outcomes. *NBER Working Paper 30735*.

Myatt, D. P. and Wallace, C. (2002). Equilibrium selection and public-good provision: The development of open-source software. *Oxford Review of Economic Policy*, 18(4):446–461.

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

Nagle, F. (2019). Open Source Software and Firm Productivity. *Management Science*, 65(3):1191–1215.

Nagle, F. (2022). Strengthening digital infrastructure: A policy agenda for free and open source software. *Brookings Institution Policy Brief*.

Nagle, F., Wheeler, D. A., Lifshitz-Assaf, H., Ham, H., and Hoffman, J. (2020). Report on the 2020 foss contributor survey. *The Linux Foundation Core Infrastructure Initiative*.

Nicas, J. (2017). Google's YouTube Has Continued Showing Brands' Ads With Racist and Other Objectionable Videos. *The Wall Street Journal* (March 24). https://www.wsj.com/articles/googles-youtube-has-continued-showing-brands-ads-with-racist-and-other-objectionable-videos-1490380551.

Oestreicher-Singer, G. and Sundararajan, A. (2012). Recommendation Networks and the Long Tail of Electronic Commerce. *MIS Quarterly*, 36(1):65–83.

Osterloh, M. and Rota, S. (2007). Open source software development: Just another case of collective invention? *Research Policy*, 36(2):157–171.

Ouimet, P. and Tate, G. (2020). Learning from coworkers: Peer effects on individual investment decisions. *The Journal of Finance*, 75(1):133–172.

O'Neil, M., Muselli, L., Cai, X., and Zacchiroli, S. (2022). Co-producing industrial public goods on github: Selective firm cooperation, volunteer-employee labour and participation inequality. *New Media & Society*.

Palfrey, T. R. and Rosenthal, H. (1984). Participation and the provision of discrete public goods: A strategic analysis. *Journal of Public Economics*, 24(2):171–193.

Pallais, A. (2014). Inefficient hiring in entry-level labor markets. *American Economic Review*, 104(11):3565–3599.

Papay, J. P., Murnane, R. J., and Willett, J. B. (2010). The Consequences of High School Exit Examinations for Low-performing Urban Students: Evidence from Massachusetts. *Educational Evaluation and Policy Analysis*, 32(1):5–23.

Parker, G. and Van Alstyne, M. (2018). Innovation, Openness, and Platform Control. *Management Science*, 64(7):3015–3032.

Patel, R. (2024). Software development statistics: Market trends and insights. *Radix*.

Peterson, K. (2013). The github open source development process. *url: http://kevinp.me/github-process-research/github-processresearch. pdf (visited on 05/11/2017)*.

Piopiunik, M., Schwerdt, G., Simon, L., and Woessmann, L. (2020). Skills, signals, and employability: An experimental investigation. *European Economic Review*, 123:103374.

Popper, B. (2017). YouTube will no longer allow creators to make money until they reach 10,000 views. *The Verge* (April 6). `https://www.theverge.com/2019/4/5/18287318/youtube-logan-paul-pewdiepie-demonetization-adpocalypse-premium-influencers-creators`.

Puranam, P., Alexy, O., and Reitzig, M. (2014). What's 'new' about new forms of organizing? *Academy of Management Review*, 39(2):162–180.

Pursey, J. (2022). 7 Facts About The 1983 Video Game Crash. *GameRant* (April 17). `https://gamerant.com/1983-video-game-crash-facts-details-trivia/`.

Raveendran, M., Puranam, P., and Warglien, M. (2022). Division of labor through self-selection. *Organization Science*, 33(2):810–830.

Ren, Y., Kraut, R., and Kiesler, S. (2007). Applying Common Identity and Bond Theory to Design of Online Communities. *Organization Studies*, 28(3):377–408.

Rietveld, J., Ploog, J. N., and Nieborg, D. B. (2020). Coevolution of Platform Dominance and Governance Strategies: Effects on Complementor Performance Outcomes. *Academy of Management Discoveries*, 6(3):488–513.

Rietveld, J., Schilling, M. A., and Bellavitis, C. (2019). Platform Strategy: Managing Ecosystem Value through Selective Promotion of Complements. *Organization Science*, 30(6):1232–1251.

Rietveld, J., Seamans, R., and Meggiorin, K. (2021). Market orchestrators: The effects of certification on platforms and their complementors. *Strategy Science*, 6(3):244–264.

Roberts, J. A., Hann, I.-H., and Slaughter, S. A. (2006). Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the apache projects. *Management Science*, 52(7):984–999.

Rosenthal, S. S. and Strange, W. C. (2020). How close is close? The spatial reach of agglomeration economies. *Journal of Economic Perspectives*, 34(3):27–49.

Sacerdote, B. (2001). Peer effects with random assignment: Results for dartmouth roommates. *The Quarterly Journal of Economics*, 116(2):681–704.

Sacerdote, B. (2011). Peer effects in education: How might they work, how big are they and how much do we know thus far? In *Handbook of the Economics of Education*, volume 3, pages 249–277. Elsevier.

Sacerdote, B. (2014). Experimental and quasi-experimental analysis of peer effects: two steps forward? *Annu. Rev. Econ.*, 6(1):253–272.

Shah, S. K. (2006). Motivation, Governance, and the Viability of Hybrid Forms in Open Source Software Development. *Management Science*, 52(7):1000–1014.

Simplemaps (2021). United states cities database. `https://simplemaps.com/data/us-cities`. Accessed: 2021-05-10.

Smirnova, I., Reitzig, M., and Alexy, O. (2022). What makes the right oss contributor tick? treatments to motivate high-skilled developers. *Research Policy*, 51(1):104368.

Solimano, A. (2006). *The International Mobility of Talent and Its Impact on Global Development: An Overview*. ECLAC.

Spence, M. (1973). Job market signaling. *The Quarterly Journal of Economics*, pages 355–374.

StackOverflow (2020). Stackoverflow developer survey 2020. https://insights.stackoverflow.com/survey/2020#overview. Accessed: 2024-09-09.

Startlin (2016). History of github. *Infographic*.

Statt, N. (2017). YouTube is facing a full-scale advertising boycott over hate speech. *The Verge* (March 24). https://www.theverge.com/2017/3/24/15053990/google-youtube-advertising-boycott-hate-speech.

Steinwender, C. (2018). Real effects of information frictions: When the states and the kingdom became united. *American Economic Review*, 108(3):657–96.

Stewart, K. J. and Gosain, S. (2006). The impact of ideology on effectiveness in open source software development teams. *MIS Quarterly*, pages 291–314.

Sun, L. and Abraham, S. (2021). Estimating dynamic treatment effects in event studies with heterogeneous treatment effects. *Journal of econometrics*, 225(2):175–199.

Sun, M. and Zhu, F. (2013). Ad revenue and content commercialization: Evidence from blogs. *Management Science*, 59(10):2314–2331.

Sun, Y., Dong, X., and McIntyre, S. (2017). Motivation of User-Generated Content: Social Connectedness Moderates the Effects of Monetary Rewards. *Marketing Science*, 36(3):329–337.

Surakka, S. (2007). What subjects and skills are important for software developers? *Communications of the ACM*, 50(1):73–78.

Sutherland, W. and Jarrahi, M. H. (2018). The sharing economy and digital platforms: A review and research agenda. *International Journal of Information Management*, 43:328–341.

Synopsys (2023). Open source security and risk analysis report 2023. *Report*.

Tae, C. J., Luo, X., and Lin, Z. (2020). Capacity-Constrained Entrepreneurs and Their Product Portfolio Size: The Response to a Platform Design Change on a Chinese Sharing Economy Platform. *Strategic Entrepreneurship Journal*, 14(3):302–328.

Tambe, P., Hitt, L., Rock, D., and Brynjolfsson, E. (2020). Digital capital and superstar firms. *NBER Working Paper 28285*.

Tan, X., Zhou, M., and Fitzgerald, B. (2020). Scaling open source communities: an empirical study of the linux kernel. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, pages 1222–1234.

Tang, Q., Gu, B., and Whinston, A. B. (2012). Content contribution for revenue sharing and reputation in social media: A dynamic structural model. *Journal of Management Information Systems*, 29(2):41–76.

Tiwana, A. (2013). *Platform Ecosystems: Aligning Architecture, Governance, and Strategy*. Morgan Kauffmann.

Tiwana, A. (2015a). Evolutionary Competition in Platform Ecosystems. *Information Systems Research*, 26(2):266–281.

Tiwana, A. (2015b). Platform Desertion by App Developers. *Journal of Management Information Systems*, 32(4):40–77.

Tonia, T., Van Oyen, H., Berger, A., Schindler, C., and Künzli, N. (2020). If i tweet will you cite later? follow-up on the effect of social media exposure on article downloads and citations. *International journal of public health*, 65:1797–1802.

Toubia, O. and Stephen, A. T. (2013). Intrinsic vs. Image-Related Utility in Social Media: Why do People Contribute Content to Twitter? *Marketing Science*, 32(3):368–392.

Tubino, L., Cain, A., Schneider, J.-G., Thiruvady, D., and Fernando, N. (2020). Authentic individual assessment for team-based software engineering projects. *IEEE International Conference on Software Engineering, Education and Training (CSEE&T)*, pages 71–81.

Vasilescu, B., Van Schuylenburg, S., Wulms, J., Serebrenik, A., and van den Brand, M. (2014). Continuous integration in a social-coding world: Empirical evidence from github. In *2014 IEEE international conference on software maintenance and evolution*, pages 401–405. IEEE.

Venhorst, V., van Dijk, J., and van Wissen, L. (2011). An analysis of trends in spatial mobility of dutch graduates. *Spatial Economic Analysis*, 6(1):57–82.

Verspagen, B. and Schoenmakers, W. (2004). The spatial dimension of patenting by multinational firms in europe. *Journal of Economic Geography*, 4(1):23–42.

Vidoni, M. (2022). A systematic process for mining software repositories: Results from a systematic literature review. *Information and Software Technology*, 144:106791.

von Krogh, G., Haefliger, S., Spaeth, S., and Wallin, M. W. (2012). Carrots and rainbows: Motivation and social practice in open source software development. *MIS Quarterly*, pages 649–676.

von Krogh, G., Spaeth, S., and Lakhani, K. R. (2003). Community, joining, and specialization in open source software innovation: A case study. *Research Policy*, 32(7):1217–1241.

von Proff, S., Duschl, M., and Brenner, T. (2017). Motives behind the mobility of university graduates: A study of three german universities. *Review of Regional Research*, 37:39–58.

Wachs, J., Nitecki, M., Schueller, W., and Polleres, A. (2022). The geography of open source software: Evidence from github. *Technological Forecasting and Social Change*, 176:121478.

Wagner, S. and Ruhe, M. (2018). A systematic review of productivity factors in software development. *arXiv Preprint 1801.06475*.

Waldfogel, J. (2017). How digitization has created a golden age of music, movies, books, and television. *Journal of Economic Perspectives*, 31(3):195–214.

Waldinger, F. (2012). Peer effects in science: Evidence from the dismissal of scientists in nazi germany. *The Review of Economic Studies*, 79(2):838–861.

Wang, Y., Wang, L., Hu, H., Jiang, J., Kuang, H., and Tao, X. (2022). The influence of sponsorship on open-source software developers' activities on github. *IEEE Computers, Software, and Applications Conference (COMPSAC)*, pages 924–933.

Wareham, J., Fox, P. B., and C. Giner, J. L. (2014). Technology Ecosystem Governance. *Organization Science*, 25(4):1195–1215.

Webb, M., Short, N., Bloom, N., and Lerner, J. (2018). Some facts of high-tech patenting. *NBER Working Paper 24793*.

Wen, W., Ceccagnoli, M., and Forman, C. (2016). Opening up intellectual property strategy: Implications for open source software entry by start-up firms. *Management Science*, 62(9):2668–2691.

Wilbur, K. C. (2008). A two-sided, empirical model of television advertising and viewing markets. *Marketing Science*, 27(3):356–378.

Wojcicki, S. (2017). Expanding our work against abuse of our platform. *YouTube Official Blog* (December 5). `https://blog.youtube/news-and-events/expanding-our-work-against-abuse-of-our`.

Wright, N. L., Nagle, F., and Greenstein, S. (2023). Open source software and global entrepreneurship. *Research Policy*, 52(9):104846.

Wu, Y. and Zhu, F. (2022). Competition, contracts, and creativity: Evidence from novel writing in a platform market. *Management Science*, 68(12):8613–8634.

Wuchty, S., Jones, B. F., and Uzzi, B. (2007). The increasing dominance of teams in production of knowledge. *Science*, 316(5827):1036–1039.

Xu, L., Nian, T., and Cabral, L. (2020). What makes geeks tick? a study of stack overflow careers. *Management Science*, 66(2):587–604.

Young, H. P. (2009). Innovation diffusion in heterogeneous populations: Contagion, social influence, and social learning. *American Economic Review*, 99(5):1899–1924.

YouTube Official Blog (2012). Being a YouTube Creator Just Got Even More Rewarding. *YouTube Official Blog* (April 12). `https://blog.youtube/news-and-events/being-youtube-creator-just-got-even`.

Zeitlyn, D. (2003). Gift economies in the development of open source software: Anthropological reflections. *Research Policy*, 32(7):1287–1291.

Zervas, G., Proserpio, D., and Byers, J. W. (2021). A first look at online reputation on airbnb, where every stay is above average. *Marketing Letters*, 32(1):1–16.