Effort Opacity *

Sahiba Chopra

Haas School of Business, UC Berkeley

Abstract

Generative AI can raise individual productivity, but organizational value also depends on the downstream evaluation and integration of what workers produce. This paper argues that AI may improve production while reducing the efficiency of that downstream allocation. I develop a theory of *effort opacity*: a condition in which the effort behind a contribution becomes difficult to observe. In collaborative settings, evaluators rely not only on the contribution itself but also on visible traces of effort when deciding whether work is worth approving, developing, or building upon. When these traces are less visible, the artifact becomes less informative about the investment that produced it, diminishing the signal that collaborators need when deciding how much of their own effort to reciprocate. This can lead to potentially valuable contributions being deprioritized or underdeveloped even when their observable quality is high. Drawing on signaling and social exchange theories, I argue that effort opacity arises when technologies, production systems, or organizational arrangements decouple output quality from the visibility of the effort that produced it—generating a misallocation of evaluative attention that is invisible in individual productivity data. While effort opacity can arise from many sources, generative AI provides a particularly salient contemporary instance. The theory, however, applies whenever changes in the production process make effort less legible to those who must evaluate or build on others' work. I examine this argument across three studies of software development. First, an observational analysis of 13,614 pull requests across 166 GitHub repositories reveals that contributions predicted to be AI-generated are more likely to be rejected, receive less substantive feedback, and remain open longer—patterns that persist among pull requests passing identical automated quality checks. Second, a pre-registered detection experiment with 126 software developers shows that the behavioral premise holds: developers distinguish AI-generated from human-written code above chance, and code

*I am grateful to Mathijs de Vaan and Abhishek Nagaraj for their guidance and support throughout this project. I also thank Don Moore and Solène Delecourt for valuable input on the experimental design. Large language models were used for copy-editing assistance. The code detection experiment was pre-registered on OSF (osf.io/76nrt). I gratefully acknowledge financial support from the Institute for Business Innovation at the UC Berkeley Haas School of Business. All errors are my own.

perceived as AI-generated receives lower effort ratings. Third, a within-subjects pilot experiment holds code quality constant while varying AI disclosure and visible effort cues, finding that AI disclosure reduces perceived contributor competence and human contribution without changing perceived code quality, while making effort visible attenuates the penalty. Together, these findings suggest that productivity gains from AI may not translate smoothly into organizational value when the evaluation systems that allocate attention and developmental investment can no longer rely on effort as an informational input.

1 Introduction

Collaborative work depends on evaluation. Memos must be reviewed, code must be approved, analyses must be vetted, and recommendations must be incorporated into ongoing work. In each case, an evaluator examines a submitted artifact and makes a judgment—not only about the work itself, but about the person behind it: whether the contributor is careful, invested, and likely to be responsive in subsequent interaction. These inferences matter because evaluation is costly. Reviewers must decide how to allocate scarce attention and where developmental effort is most likely to pay off (Blau, 1964; Gouldner, 1960; Spence, 1973).

How do evaluators make these inferences? Largely through effort. Observable traces of investment—the depth of reasoning, the care in documentation, the evidence of iterative refinement—have historically served as credible signals of a contributor's engagement and commitment. Effort works as a signal precisely because it is costly: it would be irrational to expend significant time and cognitive resources on a submission unless the contributor genuinely valued the collaboration (Spence, 1973). In this way, effort functions as the primary information infrastructure of collaborative exchange—the mechanism through which collaborators make each other's intentions legible.

This paper develops a theory of what happens when that infrastructure breaks. I define *effort opacity* as a condition in which the observable features of a work product decouple from the human effort that produced them, rendering effort inference unreliable. When a technology makes it possible to produce high-quality output with substantially less effort, the historical coupling between output characteristics and underlying investment weakens. Evaluators can no longer distinguish work that reflects deep engagement from work that merely appears adequate, and the information system that sustains reciprocity, competence assessment, and relational trust degrades.

Effort opacity is not new. Every major technology that decoupled output from effort triggered a recognizable pattern: the printing press made scholarly reproduction trivial, disrupting attribution norms that had relied on the visible labor of hand-copying; factory production dissolved the craft guild's reliance on individual effort legibility, prompting institutional responses from piece-rate pay to labor unions; computer-aided design made architectural drafting effortless, leaving senior partners unable to evaluate junior associates through their drawings. In each of these cases, the underlying theoretical mechanisms—costly signaling (Spence, 1973) and reciprocal exchange (Blau, 1964)—were disrupted when technology weakened the link between effort and observable output. In each case, evaluators initially penalized or resisted the technology, and institutions eventually developed new mechanisms to restore some form of effort legibility. The recurrence of this pattern is itself evidence that effort legibility is not a sentimental preference but a structural necessity: societies keep rebuilding it because collaboration cannot sustain itself without some mechanism for making intentions credible.

Generative artificial intelligence represents the most powerful and scalable instance of this disruption. Unlike earlier productivity tools that automated formatting or transmission while leaving substantive reasoning visible, AI can assist with the creation of knowledge-work artifacts themselves

(Brynjolfsson et al., 2025; Dell'Acqua et al., 2023; Noy & Zhang, 2023; Peng et al., 2023; Demirer et al., 2026; Daniotti et al., 2026). A polished pull request, a well-structured memo, a thorough analysis—each could reflect hours of careful human work or minutes of AI prompting. The artifact looks the same either way. This means AI does not merely reduce the time required to produce work; it reduces the visibility of the human contribution embedded in the work, interfering with a signal that many evaluative systems have always relied on.

I test the theory in the AI context, where effort opacity is most acute, using two primary sources of evidence. First, I analyze 13,614 pull requests from 4,907 contributors across 166 Python repositories on GitHub. Using a fine-tuned ModernBERT classifier to identify contributions with strong AI signatures, I find that AI-predicted contributions face 31 percentage points higher rejection rates, receive less substantive feedback, and remain open longer—patterns that strengthen in larger repositories where reviewer attention is more constrained. The classifier identifies contributions on the basis of stylistic signatures—precisely the kind of perceivable features that the theory predicts evaluators would also respond to. A pre-registered detection experiment with 126 software developers provides direct evidence for this bridge: developers distinguish AI-generated from human-written code above chance, and perceived low effort is a primary cue driving that judgment. The detection experiment thus validates that the stylistic features captured by the classifier correspond to features that human evaluators can perceive and react to. Second, a within-subject pilot review experiment holds the underlying code constant while varying AI disclosure and visible process evidence: AI disclosure reduces perceived contributor competence and human contribution without changing perceived code quality, while visible effort cues increase willingness to prioritize a contributor for future review over and above quality and competence ratings.

Together, these findings suggest that technology adoption can reshape how submitted work is interpreted—not only by changing what is produced, but by changing what evaluators can infer from it. For management and strategy research, this identifies a margin that productivity-centered accounts tend to overlook. Even when a tool raises individual output, organizational value still depends on the complementary actors whose approval, attention, and developmental investment are required for that output to matter (Teece, 1986). The question is not only how much a technology can help workers produce, but whether the evaluation systems organizations rely on can function when artifacts are no longer a dependable window into human engagement.

The paper proceeds as follows. I develop the theory of effort opacity, grounding it in social exchange theory and signaling theory, and derive predictions about evaluator behavior. I then test these predictions in the AI context, beginning with the GitHub field setting—which includes a pre-registered detection experiment validating the assumption that developers can perceive AI involvement—followed by a pilot review experiment that isolates the effort opacity mechanism under controlled conditions. A formal model specifying both the contributor's and evaluator's problems is provided in Appendix A.

2 Theory

2.1 The Baseline: How Collaboration Works When Effort Is Visible

Consider a collaborative setting in which a contributor produces a work artifact and a reviewer decides what to do with it—accept it, request revisions, or reject it. The contributor chooses how much effort to invest. Effort is costly, but it produces two things: it raises the quality of the output, and it leaves visible traces in the artifact—documentation, reasoning, iteration marks, thoroughness. In the baseline, these two things are *correlated*: higher effort tends to produce both higher quality and more visible process traces. But the coupling is not perfect. Even before AI, quality and visible effort could diverge. A capable contributor might submit functional code that solves the problem but omit documentation, skip edge-case reasoning, or provide a terse pull request description—producing adequate quality with low visible engagement. Conversely, a contributor working at the boundary of their competence might invest heavily in a problem and produce a result that is competent but unremarkable, accompanied by detailed process traces that reveal genuine struggle. The baseline is therefore better described as one in which quality and effort are *positively correlated and jointly observable*, not one in which they are identical.

The reviewer observes the artifact and makes two inferences from it. The first is a quality judgment: is this work good enough to accept? The second is an effort judgment: is this contributor worth investing in? The second inference matters because the reviewer's time is scarce (March & Simon, 1958; Boudreau & Jeppesen, 2015). If the reviewer requests revisions, that costs the reviewer effort—they must write detailed feedback, explain what needs to change, develop the contributor. Reviewers only want to make that investment if they believe the contributor is genuinely engaged and will follow through (Blau, 1964; Gouldner, 1960).

The reviewer's decision is therefore not simply a quality threshold. It is a joint assessment: quality tells the reviewer whether the work *could* be valuable; effort tells the reviewer whether investing in this contributor *will* be valuable. Because quality and effort can diverge even in the baseline, reviewers attend to both dimensions. High quality paired with high visible effort leads to acceptance or investment in revisions. High quality paired with low visible effort—the capable but disengaged contributor—produces skepticism and deprioritization. The system works because, in the baseline, reviewers can usually read effort off the artifact with reasonable accuracy, and that information governs how they allocate their own scarce attention (Spence, 1973; Gouldner, 1960; Kollock, 1999).

This informational role of effort is well grounded in several theoretical traditions. In social exchange theory, visible investment sustains the norms of reciprocity on which collaborative relationships depend; evaluators calibrate their own engagement to the investment they perceive from contributors (Blau, 1964; Gouldner, 1960). In signaling theory, costly effort functions as a credible signal of unobservable attributes—competence, commitment, type—in settings where these cannot be directly verified (Spence, 1973). Research on joint action further suggests that computing a part-

ner's effort is not merely a heuristic but a cognitive prerequisite for efficient coordination (Török et al., 2021). And in equity theory, monitoring input ratios across collaborators requires that inputs be at least partially observable (Adams, 1965). Across these frameworks, the common thread is that effort serves as an informational input to evaluation, not merely an incidental correlate of output quality.

Three features of artifact-based collaborative settings make this effort signal especially consequential. First, *evaluation is costly*: reviewers cannot invest equally in every submission, so they rely on available cues to triage attention. Second, *artifacts have historically carried effort information*: observable features of work products provided a noisy but useful basis for inferring contributor engagement. Third, *alternative signals are often limited*: in many collaborative systems, evaluators lack rich prior knowledge of a contributor's capability or motivation, giving the artifact itself unusual inferential weight (Tsay et al., 2014; de Vaan et al., 2015).

2.2 Effort Opacity: What Changes When Effort Becomes Hard to See

I define *effort opacity* as the degree to which the effort underlying a contribution is unobservable to those who evaluate it. To understand what opacity disrupts, it is useful to clarify the relationship between quality, effort, and visible traces in the baseline. Quality and effort are correlated but distinct. In the pre-AI baseline, effort tends to produce higher quality, so quality provides a coarse signal of engagement. But quality alone is a noisy proxy for effort—a contributor may produce adequate work with minimal investment, or invest heavily in a difficult problem and produce a result that is competent but unremarkable. What makes effort legible in practice is not quality per se but the *visible traces* that accompany effortful production: documentation of reasoning, evidence of iteration, engagement with edge cases, diagnostic specificity. These traces carry information about the contributor's investment that quality alone does not. A polished artifact with no visible process trace and a polished artifact accompanied by detailed reasoning may be of equivalent quality, but they communicate different things about the person behind them.

Effort opacity arises when the artifact still reveals quality but the visible traces that distinguish high-effort from low-effort production become unreliable. The reviewer can assess whether the work looks good, but the process signals that previously helped the reviewer infer engagement are degraded or absent.

This changes the reviewer's behavior in two ways. First, the reviewer becomes more selective about acceptance. Because the reviewer's evaluation weighs both quality and inferred effort, opacity degrades the effort component of the assessment. The reviewer can no longer confidently distinguish a high-effort contributor from a low-effort one, so the reviewer may raise the bar—requiring higher observable quality before accepting, because quality is the only reliable signal left. Marginal contributions that would have been accepted under transparency may be rejected under opacity.

Second, the reviewer pulls back on developmental investment. The logic of requesting revisions and providing substantive feedback depends on the reviewer believing the contributor will follow

through. That belief is grounded in perceived effort (Blau, 1964; Gouldner, 1960). When effort is opaque, the reviewer's expected return from developmental investment drops—they cannot tell if they are investing in someone who will reciprocate. So the reviewer shifts from development (costly, high-return-if-reciprocated) toward simple gatekeeping (cheap, low-risk). The result is less substantive feedback, more binary accept-or-reject decisions (Kollock, 1999).

Opacity also changes the contributor's incentives. If effort is no longer visible, the returns to effort fall. The contributor still needs quality above the threshold to be accepted, but the additional return from effort—the developmental help from the reviewer, the relational goodwill—is diminished. The contributor may rationally reduce effort and rely more on whatever produces quality cheaply. This further degrades the system: less effort invested means less visible effort, which means more opacity and less trust.

It is worth distinguishing effort opacity from several adjacent mechanisms that could produce superficially similar patterns. *AI stigma* or source aversion would predict that any known AI involvement lowers evaluation, regardless of the artifact's characteristics or the contributor's visible engagement. *Authorship discounting* would predict that evaluators reduce credit when they believe a contributor is not the sole author, again regardless of process visibility. *Competence inference* would predict that AI disclosure signals lower contributor skill. Effort opacity is distinct because it operates through the *informational* channel: the artifact becomes a noisier signal of the contributor's investment, and the evaluator's response is a rational adjustment to degraded information rather than a blanket penalty for AI involvement or a judgment about the contributor's underlying ability. This distinction generates a specific empirical prediction that the alternatives do not: if effort opacity is the operative mechanism, then providing supplementary effort signals should attenuate the penalty even when AI involvement is disclosed. I test this prediction in the pilot experiment.

The concept is not specific to any single technology. The same mechanism has recurred across centuries of technological change, and institutions have responded in two broadly distinct ways. Some responses attempt to *restore* effort legibility through alternative signals: the printing press severed the link between a text and the visible labor of hand-copying, prompting authors to restore effort signals through elaborate prefaces and personal seals; in law, electronic research tools made citation depth trivial, forcing firms to shift evaluation toward oral questioning and direct interaction, where effort remained observable; computer-aided design allowed mediocre draftspersons to produce clean output, and architecture firms initially resisted CAD before eventually shifting evaluation upstream to conceptual design and client interaction, where effort was still legible. Other responses *bypass* effort-based evaluation entirely by tying compensation directly to output: the factory system dissolved craft guilds' reliance on individual effort legibility, and piece-rate pay emerged as a response that abandoned effort observation in favor of pure output measurement—an institutional adaptation to opacity rather than a restoration of legibility. Labor unions, by contrast, represented a collective response to the loss of individual evaluative standing. The distinction matters for the present argument: the theory predicts that under effort opacity, evaluators may either seek alternative effort signals or retreat to output-based gatekeeping. Both responses appear in the

historical record.

2.3 How AI Intensifies Effort Opacity

Generative AI enters this framework by breaking the relationship between effort and quality. Before AI, quality was primarily a function of effort: if a contributor wanted high-quality output, they had to work hard. Effort and quality were coupled, which is why observing quality was informative about effort.

AI introduces a second input to the production process. A contributor can now achieve high quality with low effort by relying on AI assistance. The cost of producing quality falls. This is the productivity gain that the existing literature documents (Brynjolfsson et al., 2025; Dell'Acqua et al., 2023; Noy & Zhang, 2023; Peng et al., 2023). But the consequence the paper identifies is that when AI capability is high, observing high quality no longer tells the reviewer whether effort was high. A polished pull request could reflect ten hours of careful work or ten minutes of prompting. The artifact looks the same either way. AI does not just introduce opacity as an abstract possibility—it mechanically decouples the two things that were previously bundled (quality and effort), making quality uninformative about effort.

The nature of this signal degradation deserves emphasis. In Section 2.2, I described effort opacity as noisier inference—the reviewer faces greater uncertainty about effort given the artifact. AI intensifies this in a specific direction: it raises the floor of quality achievable at any effort level, so that low-effort contributions now *look* high-effort because AI polishes them. The signal is not merely noisy but systematically compressed from below. The reviewer cannot use low quality as a reliable signal of low effort anymore, because AI eliminates the low-quality outcomes that previously carried information. The result is that the reviewer faces both greater uncertainty and a prior that shifts downward: if the reviewer suspects AI involvement, the posterior estimate of effort drops even when the artifact appears polished, because polish is no longer diagnostic.

The consequence is exactly what the baseline opacity dynamics predict, but intensified: reviewers become more selective, pull back on developmental feedback, and the collaborative system becomes less efficient at identifying and developing valuable contributors—even as the average quality of submitted artifacts may actually improve. The theory does not claim that AI-generated work is necessarily lower quality. The claim is that AI changes what reviewers can *infer* from work products, even when the products themselves are of comparable quality. A formal model specifying both the contributor's and reviewer's objective functions is provided in Appendix A.

For effort opacity to produce evaluative consequences in the AI context, reviewers must be able to perceive—at least imperfectly—when a submission may have been AI-assisted. If AI-assisted work were completely indistinguishable from unaided work, there would be no basis for differential treatment. This assumption is plausible given evidence that AI-generated outputs often exhibit recognizable stylistic regularities (Bukhari et al., 2023), and I test it directly in Section 4.7. Additionally, observational evidence faces a selection concern: contributors may turn to AI for

tasks they understand less well or are less committed to. I address this empirically but note that the field design is correlational.

2.4 Predictions

The argument yields two central predictions about reviewer behavior under effort opacity, which I test in the AI context where the disruption is most acute.

When effort becomes opaque, reviewers face greater uncertainty about whether a submission merits their investment. Because reviewers in collaborative settings weigh not only the artifact's apparent quality but also the contributor's inferred engagement (Blau, 1964; Gouldner, 1960), a weakening of the effort signal raises the effective threshold for acceptance. Reviewers who cannot distinguish high-engagement from low-engagement contributions may become more selective as a rational response to a noisier signal environment (March & Simon, 1958; Dietvorst et al., 2015).

Hypothesis 1 (Heightened selectivity): AI-predicted contributions are associated with higher rejection rates than non-AI contributions, controlling for observable contribution characteristics.

Effort opacity should affect not only whether contributions are accepted but also how reviewers engage with them. In the framework of social exchange, reviewers calibrate their own investment—the depth and developmental quality of their feedback—to the investment they perceive from contributors (Gouldner, 1960; Kollock, 1999). When effort is opaque, the expected return to developmental investment is uncertain, because the reviewer cannot reliably identify contributors who will benefit from and reciprocate their attention. One response is to redirect evaluative effort from collaborative development toward lower-cost gatekeeping: rather than investing in feedback that improves the contribution, reviewers shift toward directive responses that place the burden of demonstrating engagement on the contributor. This prediction is distinct from total disengagement; reviewers may not reduce total engagement but instead convert developmental investment into screening.

Hypothesis 2 (Feedback reallocation): AI-predicted contributions are associated with less developmental feedback and a shift toward directive evaluative responses, as reviewers reallocate effort from collaborative improvement to verification under uncertainty.

3 Evidence from Open-Source Software Review

3.1 Empirical Setting

Open-source software (OSS) provides a useful setting for this argument because it is a transparent version of a broader organizational process. A contributor submits a work product, and another actor decides whether to approve it, improve it, or reject it. GitHub's pull request (PR) system makes both sides of that exchange observable. Contributors submit code changes for review by project maintainers, who act as both quality gatekeepers and contributor mentors (Tsay et al., 2014;

Dabbish et al., 2012). In doing so, they solve a repeated screening and investment problem: which contributions should be integrated, and which contributors are worth further developmental effort (Boudreau & Jeppesen, 2015; Lerner & Tirole, 2002).

OSS is also substantively important. It underpins much of the modern software economy (Nagle, 2019; Nagaraj, 2021; Gortmaker, 2024), and generative AI diffused into software development earlier than into many other forms of knowledge work. At the same time, the logic I study is not specific to software. Code review is analytically useful precisely because it resembles many other organizational evaluation processes in which managers or peers must decide how much attention to devote to a submitted artifact.

The collaborative norms of open source make effort especially salient (von Krogh et al., 2003; O'Mahony & Ferraro, 2007). Semi-structured interviews with open-source maintainers (n=5) reinforced this point. Concerns centered less on AI per se than on whether contributors appeared genuinely invested. As one maintainer explained, "I don't care if someone is using AI to generate code, as long as they demonstrate effort." Another put the issue in explicitly evaluative terms: "We want to help people become better developers, but I need to make sure that the effort I put into teaching isn't going to waste—we want to help the person, not help the AI."

3.2 Data

I classify pull requests as AI-generated using a fine-tuned ModernBERT model. Human training examples are drawn from GitHub commits prior to June 2021 (before AI coding assistants); AI examples are generated via a "describe-and-generate" pipeline repeated across more than ten language models to prevent overfitting to any single tool's style. On held-out data, the classifier achieves a 1.2% false positive rate on human code and 74–100% accuracy across AI-generated samples. PRs classified as AI-generated prior to July 2021 are recoded as non-AI. The sample is restricted to pull requests containing at least one complete function with a minimum of 10 lines of added code.

Figure 1 plots the proportion of AI-predicted pull requests over time. The share rises sharply after ChatGPT's release in late 2022, reaching approximately 30% by 2025—tracking the known timeline of AI coding tool adoption.

The final sample comprises 13,614 pull requests from 4,907 contributors across 166 Python repositories (2019–2025), restricted to repositories created prior to 2021. I focus on Python because early code-generation models were trained predominantly on it (Chen et al., 2021), and restricting to a single language eliminates confounds from heterogeneous LLM performance. These repositories are dependencies for over 1 million PyPI packages, forming critical ecosystem infrastructure. Table 1 presents summary statistics.

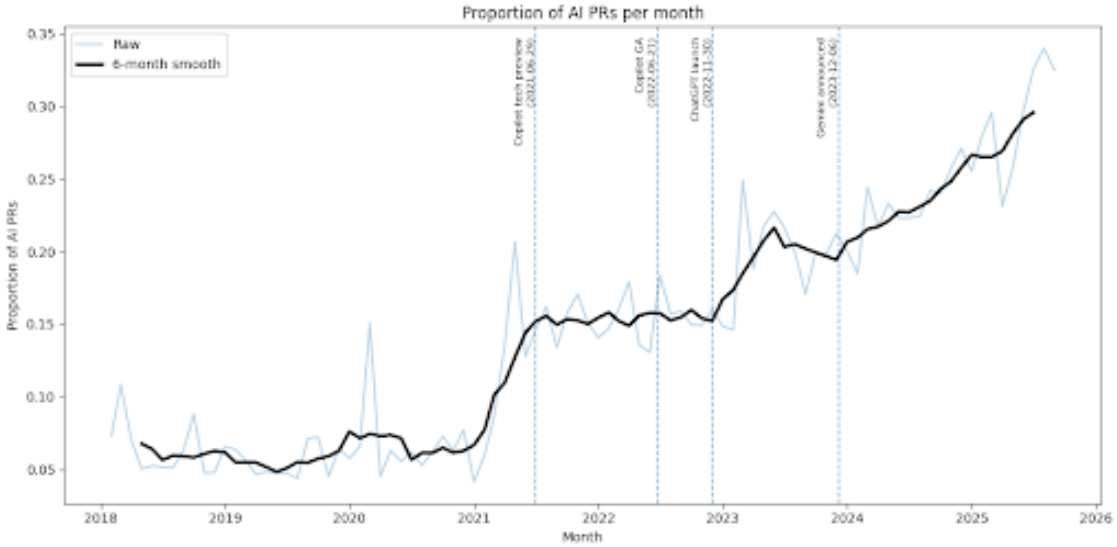


Figure 1: Proportion of AI-Predicted Pull Requests Over Time

Notes: Figure plots the raw classifier output—the monthly proportion of pull requests classified as AI-generated by the ModernBERT model—before the recoding described in the text. The non-zero proportions prior to mid-2021 reflect the classifier's false positive rate on human-written code (approximately 1.2% on held-out pre-2021 data). In the analysis sample, all PRs classified as AI-generated prior to July 2021 are recoded as non-AI; this figure is provided to show the raw trend. The gray line shows raw monthly proportions; the black line shows a 6-month smoothed trend. Vertical dashed lines indicate major AI tool releases: GitHub Copilot (June 2021), ChatGPT (November 2022), GPT-4 (March 2023), and Claude 3.5 Sonnet (June 2024).

Panel A: Repository-Level (N = 166 repos)	Mean	Median	Std. Dev.	Min	Max
Pull requests per repository	82.01	20.00	221.87	5	2,463
Contributors per repository	30.25	11.50	99.44	3	1,237
Avg. rejection rate (repo-level)	0.171	0.137	0.155	0.000	0.813
Avg. AI PR share (repo-level)	0.092	0.057	0.122	0.000	0.720
Panel B: Pull-Request Level (N = 13,614 PRs)	Mean	Median	Std. Dev.	Min	Max
Rejected (binary)	0.214	0.000	0.410	0	1
AI PR (binary)	0.110	0.000	0.313	0	1
Time to close (days)	42.21	5.85	127.04	0.0001	2,424.06
Panel C: Pull-Request Review Activity (N = 13,614 PRs)	Mean	Median	Std. Dev.	Min	Max
General comments	1.123	0.000	2.156	0	58
Inline comments	0.972	1.000	1.392	0	34
Rejection after first comment (binary)	0.057	0.000	0.232	0	1
Constructive feedback (binary)	0.793	1.000	0.405	0	1

Table 1: Summary Statistics for Repository-Level and Pull-Request-Level Variables

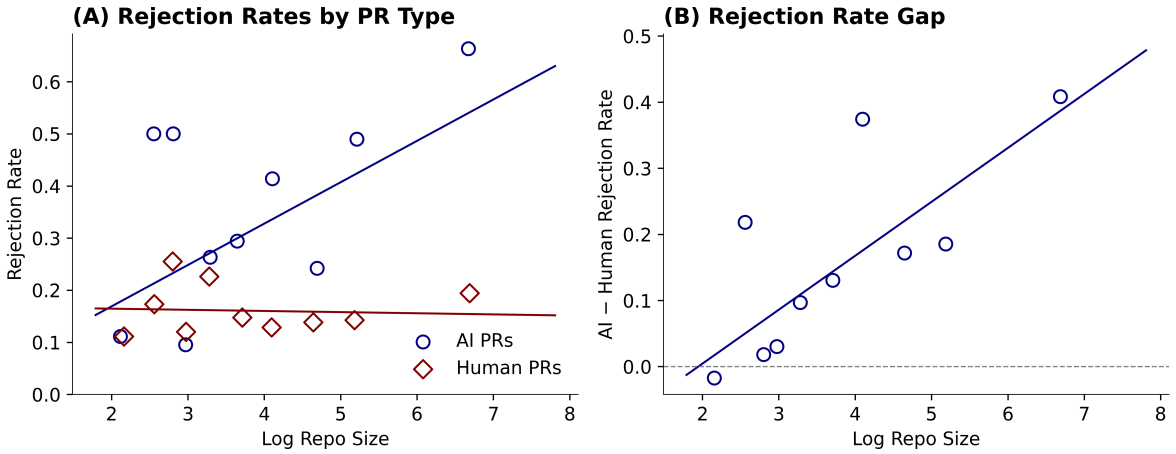


Figure 2: Rejection Rates by AI Status and Repository Size

Notes: Panel A plots binned rejection rates for AI-generated (navy circles) and human-generated (maroon diamonds) pull requests against repository size, with fitted linear trends. Panel B plots the within-repository gap (AI minus human rejection rate). Repository size is measured as the log of total pull requests in the repository. AI-generated contributions face higher rejection rates across all repository sizes, with the gap widening in larger repositories.

Figure 2 shows that the AI rejection penalty widens as repositories grow larger, consistent with effort opacity being amplified where maintainers face greater attention constraints.

3.3 Observable Differences Between AI and Non-AI Contributions

Table 2 presents differences between AI-predicted and non-AI pull requests. AI-predicted contributions add more code (median 74 vs. 51 lines), delete less (median 1 vs. 3 lines), and exhibit lower cyclomatic complexity—a pattern suggesting additive generation rather than integrative refactoring. These observable signatures may inform reviewer intuitions, even without conscious "AI detection." The regression analyses control for these characteristics; the associations between AI prediction and reviewer behavior persist, suggesting evaluators respond to something beyond the technical features of the code itself.

	Non-AI PRs	AI PRs	Difference
Panel A: Submission Characteristics (Median)			
Initial Lines Added	51.00	69.00	18.00**
Initial Lines Deleted	3.00	1.00	−2.00***
Initial Files Changed	2.00	2.00	0.00***
Cyclomatic Complexity (Mean)	2.79	2.66	−0.12***
Cyclomatic Complexity (Max)	13.00	10.00	−3.00***
Maintainability Index	41.65	44.43	2.78
Panel B: Outcomes (Mean)			
Rejection Rate	0.173	0.544	0.371***
Days to Close	40.8	55.0	14.2***
Inline Comments	1.02	0.62	−0.40***
General Comments	1.12	1.14	0.02
Observations	12,115	1,499	

Notes: Panel A presents median values of pull request characteristics at initial submission; medians are reported due to heavy-tailed distributions. Panel B presents mean values of outcome variables. Significance levels from Wilcoxon rank-sum tests (Panel A) and t-tests (Panel B): * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$.

Table 2: Observable Characteristics of AI-Predicted vs. Non-AI Pull Requests

3.4 Estimation Strategy

AI adoption is endogenous, so I present cross-sectional results and attempt to rule out quality-based screening by controlling for code complexity and maintainability metrics. A key assumption is that maintainers can perceive AI involvement; recent work confirms above-chance detection of AI-generated text (Russell et al., 2025), and the classifier identifies contributions with strong observable signatures that human reviewers would also likely notice.

I estimate the relationship between AI-predicted pull requests and evaluator responses using the following specification:

$$Y_{ijt} = \beta \mathrm{AI}_{ijt} + \gamma X_{ijt} + \alpha_j + \delta_t + \epsilon_{ijt} \tag{1}$$

where Y_{ijt} is the outcome for pull request i in repository j submitted in month t. Outcomes include rejection (binary), inline comments (count), and general comments (count). AI_{ijt} is an indicator for whether the pull request is predicted to be AI-generated. X_{ijt} is a vector of controls for code complexity (initial deletions, additions, and changed files) and code quality (mean and maximum cyclomatic complexity, mean maintainability index). α_j and δ_t are repository and month fixed

effects, respectively. Standard errors are clustered at the contributor level to account for within-author correlation.

The coefficient β captures the conditional association between predicted AI status and evaluator behavior, not a causal effect. Repository fixed effects absorb time-invariant project differences; month fixed effects account for temporal trends.

3.4.1 Measuring Comment Constructiveness

I measure comment constructiveness using the VADER sentiment analyzer (Hutto & Gilbert, 2014), classifying a comment as constructive if its negative sentiment score is below 0.20. This threshold was validated through manual inspection of comments near the classification boundary.

4 Results

4.1 Initial Reactions to AI PRs

I begin by examining reviewer behavior at first engagement, before iterative dialogue confounds the signal. By isolating what happens after a maintainer's first comment, I obtain a cleaner measure of reviewer response. All specifications control for submission scope (lines added, deleted, files changed) and code quality at initial submission, measured via cyclomatic complexity (McCabe, 1976) and maintainability index (Oman & Hagemeister, 1992).

Table 3 reports results. Panel A examines rejection after a single comment; Panel B examines constructiveness of the first comment.

	(1)	(2)	(3)
Panel A: Dependent variable = Rejection after first comment			
AI PR	0.0776***	0.0502***	0.0517***
	(0.0120)	(0.0099)	(0.0096)
Repository FE	No	Yes	Yes
Month FE	No	No	Yes
Controls[a]	Yes	Yes	Yes
Observations	13,614	13,614	13,614
R^2	0.011	0.077	0.090
Panel B: Dependent variable = Comment is Constructive			
AI PR	-0.0858***	-0.0516***	-0.0360*
	(0.0194)	(0.0140)	(0.0141)
Repository FE	No	Yes	Yes
Month FE	No	No	Yes
Controls[a]	Yes	Yes	Yes
Observations	13,614	13,614	13,614
R^2	0.004	0.140	0.152

[a] All models control for initial additions, initial deletions, and initial changed files. Robust standard errors clustered by author_login are reported in parentheses.
Significance levels: * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.

Table 3: Cross-Section of AI-PRs on Rejection After First Comment and Constructiveness of First Comment

With both repository and month fixed effects (Column 3), AI-predicted pull requests are 5.17 percentage points more likely to be rejected after one comment ($p < 0.001$)—a 108% increase over the 4.8% baseline. First comments on AI-predicted PRs are also 3.6 percentage points less likely to be constructive ($p < 0.05$). These patterns persist after quality controls, consistent with external evidence that AI-generated code is not systematically lower quality (Phan et al., 2025; Sarkar, 2025).

4.2 The Iterative Review Process

Table 4 extends the analysis to outcomes across the full review process.

	(1)	(2)	(3)	(4)
Panel A: Dependent variable = Rejected				
AI PR	0.3625***	0.3027***	0.3141***	0.1932***
	(0.0193)	(0.0157)	(0.0154)	(0.0139)
Panel B: Dependent variable = Inline Comments				
AI PR	−0.3950***	−0.2477***	−0.2544***	−0.0766
	(0.0501)	(0.0477)	(0.0497)	(0.0489)
Panel C: Dependent variable = General Comments				
AI PR	0.0199	0.2555*	0.3050**	0.3990***
	(0.0962)	(0.1027)	(0.1063)	(0.1039)
Panel D: Dependent variable = Proportion Constructive				
AI PR	−0.0979***	−0.0596***	−0.0456***	–
	(0.0176)	(0.0130)	(0.0129)	–
Repository Fixed Effects	No	Yes	Yes	Yes
Month Fixed Effects	No	No	Yes	Yes
Maintainer Fixed Effects	No	No	No	Yes
Observations	13,614	13,614	13,614	13,614

Notes: This table reports linear probability model estimates for Panels A and D, and OLS estimates for Panels B and C. AI PR is an indicator for whether the pull request is classified as AI-assisted by the fine-tuned ModernBERT classifier. All specifications include controls for initial additions, initial deletions, initial changed files, and code quality metrics (cyclomatic complexity and maintainability index); coefficients omitted for brevity. Standard errors clustered at the contributor level are reported in parentheses.
* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$.

Table 4: Cross Section of AI-Predicted Pull Requests on Review Outcomes

With repository and month fixed effects (Column 3), AI pull requests are 31.4 percentage points more likely to be rejected ($p < 0.001$), receive 0.25 fewer inline comments (a 26% reduction, $p < 0.001$), and receive 0.31 more general comments (a 28% increase, $p < 0.01$). The simultaneous decrease in high-effort and increase in low-effort feedback reveals the shape of the new equilibrium: reviewers reallocate effort rather than simply disengaging.

4.2.1 Objective Quality Controls: CI/CD Test Outcomes

I collect data on automated test outcomes from CI/CD systems for 70 repositories (635 PRs). AI-predicted PRs do fail automated tests more often than human PRs: 42.7% versus 51.9% pass rate ($p = 0.026$), confirming that some portion of the evaluative penalty reflects genuine quality

differences. Effort opacity does not require that AI-assisted code is equally good—it requires that the penalty *exceeds* what quality differences warrant. The key comparison is therefore between AI and human PRs with identical test outcomes. Among PRs that pass all automated tests, AI-predicted contributions still face a 27.7 percentage point higher rejection probability than human contributions ($p < 0.001$). This residual penalty—applied to code with objectively equivalent test performance—represents evaluative friction beyond quality screening. Conditioning on test outcomes reduces the unconditional penalty only modestly (from 30.1 to 27.7 percentage points), indicating that the bulk of the AI penalty operates independent of objective quality differences captured by automated testing. Moreover, the interaction between AI status and test failure is statistically insignificant ($p = 0.649$), indicating that reviewers apply a blanket AI penalty regardless of whether the code passes or fails tests. This blanket penalty is consistent with effort opacity: reviewers respond to inferred AI involvement per se, not to differential quality revealed by testing.

Table 5 presents additional robustness checks with progressively richer fixed effects on the full 13,614-observation sample. The AI rejection penalty remains large and highly significant (11–16 percentage points, $p < 0.001$) across all specifications, including models with author, repository, and reviewer fixed effects simultaneously.

	Rejected (0/1)				
	(1)	(2)	(3)	(4)	(5)
AI-Generated PR	0.124***	0.111***	0.148***	0.157***	0.137***
	(0.012)	(0.015)	(0.012)	(0.013)	(0.017)
Additional Controls:					
Repo Selectivity	✓	✓	✓	✓	✓
Reviewer Selectivity[a]	✓	✓	✓	✓	✓
First-Time Contributor	✓	✓	✓	✓	✓
Author Experience	✓	✓	✓	✓	✓
Author FE	No	Yes	No	No	Yes
Repository FE	No	No	Yes	No	Yes
Reviewer FE	No	No	No	Yes	Yes
R^2	0.511	0.681	0.515	0.509	0.490
Observations	13,511	10,235	13,511	13,335	8,851

Notes: Linear probability models predicting PR rejection. All models include the controls listed above where not absorbed by fixed effects. Robust standard errors in parentheses. *** p¡0.001, ** p¡0.01, * p¡0.05.

[a] Leave-one-out measure calculated excluding the focal PR.

Table 5: Robustness: AI Rejection Penalty with Controls and Fixed Effects

4.3 Heterogeneity by Repository Size

Table 6 examines whether effort opacity effects vary with repository context. If effort opacity operates through attention allocation, its effects should be amplified where evaluators face greater demands on their time.

	(1) Rejected	(2) Any review comment	(3) General comments	(4) Share constructive
AI PR	0.2314***	-0.2424***	0.0786	-0.0200
	(0.0218)	(0.0617)	(0.1476)	(0.0180)
AI PR × Large Repo	0.1290***	-0.0218	0.3495.	-0.0418.
	(0.0305)	(0.0939)	(0.1905)	(0.0240)
Repository FE	Yes	Yes	Yes	Yes
Month FE	Yes	Yes	Yes	Yes
Observations	13,614	13,614	13,614	13,614
R^2	0.237	0.239	0.115	0.184

Notes: Each column reports coefficients from a separate regression with repository and month fixed effects. Robust standard errors clustered by author_login in parentheses. All models include baseline controls (initial additions, initial deletions, initial changed files, mean and max cyclomatic complexity, and mean maintainability index), omitted for brevity. Large Repo is a time-invariant indicator for repositories in the top decile by total pull requests; its main effect is absorbed by the repository fixed effects and is therefore not reported. The interaction AI PR × Large Repo is identified because AI status varies across pull requests within repositories. Significance levels: * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$; "." indicates $p < 0.10$.

Table 6: Heterogeneous Effects of AI-generated Code by Repository Size

The AI × Large Repo interaction is 0.129 ($p < 0.001$) for rejection, indicating the penalty is 13 percentage points larger in high-activity repositories. The shift toward low-effort general comments is also concentrated in large repositories (interaction = 0.350, $p < 0.1$). These patterns reinforce the theoretical mechanism: effort opacity disrupts reciprocal exchange most where evaluators must triage scarce attention.

4.4 Robustness Checks

One alternative explanation is that AI lowers the cost of initiating a pull request, so contributors submit more speculative contributions and abandon them when faced with reviewer feedback. I construct an indicator for whether a pull request went stale, identified by inline comments flagging the contribution as inactive. After controlling for this measure, the AI coefficient remains 22.4 percentage points ($p < 0.001$), compared to 31.4 percentage points in the main specification. The majority of the effect persists among actively maintained contributions.

Another concern is that AI-predicted contributions disproportionately come from newcomers to a repository. Adding an indicator for first-time contributors leaves the AI coefficients substantively

unchanged: 30.9 percentage points higher rejection, 0.25 fewer inline comments, 0.28 more general comments.

I assess robustness to omitted variable bias using the method developed by Oster (2019). Moving from no controls to the basic code-complexity controls, the coefficient on AI-predicted status is stable (0.363 to 0.362) while R^2 increases modestly (0.069 to 0.071), yielding a δ that far exceeds Oster's threshold of 1. However, this comparison uses a restricted set of covariates. Table 5 shows that richer observable controls—including repo selectivity, reviewer selectivity, author experience, and first-time contributor status—attenuate the AI coefficient more substantially (to 0.124 in Column 1). The relevant Oster comparison is therefore between the uncontrolled and the richest observable specification. Using Column 1 of Table 5 as the controlled model ($\hat{\beta} = 0.124$, $R^2 = 0.511$), the proportional selection ratio remains above 1, but the margin is narrower than the restricted comparison suggests. I therefore interpret the robustness evidence as indicating that unobserved confounders would need to be large relative to the observed controls to fully explain away the penalty, while acknowledging that the observational design cannot rule out selection on unobservables definitively.

Beyond reviewer behavior, effort opacity also appears to generate system-level inefficiencies. AI-predicted contributions take 30.4 additional days to close ($p < 0.001$), with the delay concentrated in large repositories (AI × Large Repo = 18.3 days, $p < 0.05$). Full results are reported in Appendix B.

4.5 Testing the Detection Assumption: A Pre-Registered Experiment

The observational analysis rests on a behavioral premise: evaluators must be able to perceive, at least imperfectly, that some artifacts bear the signature of AI assistance. I test this assumption directly using a pre-registered code detection experiment with 126 software developers recruited through Prolific (630 trial-level observations; pre-registered on OSF, osf.io/76nrt). Each participant evaluated five code snippets, classifying each as human-written or AI-generated and rating how much effort the code appeared to reflect. Trial order, button order, and question order were randomized.

Participants distinguished AI-generated from human-written code above chance: overall accuracy was 64.0%, with signal-detection sensitivity $d' = 0.632$. Performance was asymmetric—developers correctly identified human-written code 72.3% of the time but AI-generated code only 56.6% of the time—indicating a conservative strategy of defaulting toward human authorship. Crucially, perceived effort strongly predicted source attribution: each one-point decrease in perceived effort was associated with a 4.6 percentage point increase in AI classification ($b = -0.046$, $p < .001$). Developers were more likely to label code as AI-generated when the artifact appeared to reflect less human investment.

A suppression analysis further clarifies the mechanism. When considered alone, AI-generated code was rated as *higher* effort than human-written code ($b = +0.482$, $p < .001$), suggesting that AI output is, if anything, more polished in appearance. But when actual and perceived source are

modeled jointly, code *perceived* as AI received lower effort ratings ($b = -0.605$, $p < .001$), while the actual AI coefficient increased ($b = +0.670$, $p < .001$). This suppression pattern indicates that the association between AI and low perceived effort is not driven by surface features of the code—it is conditional on the reviewer having formed a belief that the submission is AI-generated. The evaluative consequence of AI use operates through the inferential process by which reviewers attribute effort, not through observable quality differences. Full experimental details, tables, and figures are reported in Appendix C.

5 Pilot Review Experiment on AI Disclosure and Effort Signals

The observational study shows that AI-predicted pull requests face substantially higher rejection rates, fewer substantive review comments, and longer time-to-close than human-authored submissions. The detection experiment provides evidence that developers can perceive AI involvement above chance and that perceived effort organizes those judgments. Together, these findings are consistent with the effort opacity mechanism. However, the causal chain—that AI code faces evaluative penalties *because* it is perceived to be low effort—cannot be isolated in the observational data. The GitHub evidence is cross-sectional, and AI status, effort visibility, code quality, and contributor history all covary in ways that preclude clean attribution.

The review experiment is designed to close that gap. It holds the underlying code artifact fixed and independently varies the two theoretically active ingredients: whether the contributor discloses AI use, and whether the submission provides visible evidence of human effort. This 2×2 design allows a direct test of the effort opacity mechanism. Specifically, whether AI disclosure generates an evaluative penalty, whether high effort visibility attenuates that penalty, and whether the interaction between the two follows the pattern the theory predicts.

5.1 Design and Measures

The experiment uses a within-subject 2×2 design crossing AI disclosure (disclosed vs. not disclosed) with visible effort (high vs. low). Twenty-eight software developers were recruited through Prolific; twenty-six remained after minimal quality filters, yielding 104 trial-level observations.

A key design choice is holding the underlying code artifact constant across conditions. In collaborative software development, the code itself provides an ambiguous and incomplete signal of the effort behind it—the final diff could be similar whether a developer spent three hours reasoning through the problem or generated it in thirty seconds. While reviewers do extract some effort cues from code (as the detection experiment confirms), the richer and more diagnostic effort signals are typically expressed in the *process trace* surrounding the contribution: the pull request description, the commit history, and the diagnostic reasoning a contributor makes visible. This is precisely what AI use disrupts. When contributors offload the generative work to a model, the process trace becomes thin or absent, even if the output is indistinguishable from a carefully crafted human

submission. Holding code quality constant isolates this mechanism: any differences in evaluation must be driven by what reviewers infer about the contributor's process, not by differential code quality confounding the comparison. Each participant reviewed four pull requests, each set in a distinct fictional open-source Python repository. Using four separate fictional projects gives each PR a realistic, self-contained context and prevented participants from treating the four contributions as submissions to the same codebase, which could have introduced social comparison dynamics unrelated to the experimental manipulation. The code used in the experiment is AI-generated; prior to the study, 5 experienced software developers evaluated the code and found no meaningful quality differences across the four stimuli, confirming that the diffs were suitable for holding quality constant across conditions. Across the four PRs, each participant encountered each cell of the 2×2 design exactly once—high effort with AI disclosed, high effort without disclosure, low effort with AI disclosed, and low effort without disclosure—with assignment counterbalanced across participants. The underlying code diff is identical across all versions of each pull request; only the submission context varied. Appendix D provides screenshots of two experimental conditions applied to a single representative code artifact, illustrating the high-effort, AI-disclosed condition (Figure 6) and the low-effort, no-disclosure condition (Figure 7).

The primary dependent variable is *future-review priority*: after reviewing all four contributors, participants ranked whom they would prioritize reviewing in the future. This measure captures willingness to allocate scarce evaluative attention—the downstream consequence effort opacity theory predicts. Secondary outcomes include acceptance likelihood, perceived code quality, perceived contributor competence, and perceived human contribution. Manipulation checks confirm that the treatments were effective. High-effort submissions were rated as substantially more detailed than low-effort submissions (5.69 vs. 2.88 on a seven-point scale; b=2.81b = 2.81 b=2.81, $p < .001$ $p < .001$ $p < .001$), and 96% of participants correctly recalled the AI-disclosure cue.

The choice to operationalize AI use through explicit disclosure rather than through covert AI generation is deliberate and serves two purposes. First, the theoretical claim is not that AI use is always detected, but that the *perception* of AI use, however it arises, triggers an inferential penalty by disrupting effort attribution. Disclosure is the cleanest way to manipulate that perception experimentally while holding everything else constant. Covert generation would confound the perception manipulation with actual differences in code characteristics, reintroducing the identification problem the experiment is designed to solve. Second, explicit disclosure represents an increasingly relevant real-world condition. Organizational AI policies, journal submission requirements, and platform-level transparency norms are progressively mandating or incentivizing disclosure (Nature, 2023). Understanding how disclosure affects evaluation is therefore directly policy-relevant, independent of the covert case. To the extent that the results generalize, they suggest that the effort opacity mechanism operates through perceived AI involvement, meaning, the findings apply equally to settings where AI use becomes apparent through stylistic cues, platform flags, or institutional mandates rather than through voluntary self-report.

5.2 Main Results

Two patterns stand out. First, AI disclosure shifts attribution while leaving the artifact itself largely unchanged in evaluators' eyes. Holding the code constant, disclosed AI use reduces perceived human contribution by 1.77 points and perceived contributor competence by 0.90 points on seven-point scales, but it does not significantly reduce perceived code quality or acceptance likelihood. That dissociation is theoretically important: reviewers are not saying that the code is worse; they are crediting the contributor less for the same underlying work.

Second, visible process evidence partially repairs that attribution problem. High-effort contributors who disclosed AI use were prioritized at roughly the same level as high-effort contributors who did not disclose AI use. The sharpest penalty fell on low-effort contributors who disclosed AI use, who were ranked lowest for future review. In other words, disclosure becomes most damaging when the submission provides little independent evidence of human engagement.

Outcome	High Effort, No AI	High Effort, AI Disclosed	Low Effort, No AI	Low Effort, AI Disclosed
Acceptance likelihood (0–100)	77.92	74.92	64.23	65.54
Perceived code quality (1–7)	5.54	5.50	4.73	4.92
Perceived competence (1–7)	5.54	5.00	4.92	3.65
Perceived human contribution (1–7)	5.62	4.23	5.19	3.04
Future-review priority (1 = highest)	2.27	2.12	2.58	3.04

Notes: $N = 26$ per condition cell in the within-subject design. Lower values on future-review priority indicate greater willingness to allocate future evaluative attention.

Table 7: Pilot experiment: cell means by condition

Table 8 presents the primary mixed-effects estimates. Visible effort increases acceptance likelihood, perceived code quality, perceived competence, and perceived human contribution, and it improves future-review priority. AI disclosure, by contrast, strongly depresses competence and human-contribution judgments but does not significantly affect code quality, acceptance, or priority on average. The interaction terms are directionally consistent with the theory but estimated imprecisely, which is unsurprising given the pilot sample size. I therefore treat the interaction evidence as diagnostic rather than confirmatory.

Outcome	Effort	AI Disclosure	Effort × AI	ICC
Acceptance likelihood	11.54**	−0.85	−4.31	.13
	(3.92)	(3.92)	(7.84)	
Perceived code quality	0.69**	0.08	−0.23	.03
	(0.22)	(0.22)	(0.45)	
Perceived competence	0.98***	−0.90***	0.73	.05
	(0.23)	(0.23)	(0.46)	
Perceived human contribution	0.81***	−1.77***	0.77	.06
	(0.25)	(0.25)	(0.49)	
Future-review priority	−0.62**	0.15	−0.62	.00
	(0.21)	(0.21)	(0.42)	

Notes: Linear mixed-effects models with participant random intercepts. Predictors are effect coded. Standard errors in parentheses. Lower values on future-review priority indicate greater priority. $^{***}p < .001$, $^{**}p < .01$, $^{*}p < .05$.

Table 8: Pilot experiment: mixed-effects estimates

5.3 Future-Review Priority and Direct Reciprocity

Because future-review priority is the experiment's primary measure of evaluative attention, I examine it in greater detail. Figure 3 shows the central pattern. The low-effort contributor who disclosed AI use receives the lowest priority, whereas the two high-effort contributors are treated similarly regardless of disclosure. That pattern suggests that process evidence does not remove attributional ambiguity entirely, but it appears sufficient to restore willingness to invest future attention.

Table 9 examines whether visible effort predicts future-review priority only because it changes competence or quality beliefs. In the baseline specification, effort improves priority by 0.62 rank points ($p = .004$). When perceived competence is added, the coefficient shrinks but remains directionally large. When both competence and code quality are included, the effort coefficient remains marginally significant ($b = -0.39$, $p = .084$). That pattern is consistent with a direct reciprocity channel: reviewers allocate future attention not only because the contributor appears more capable, but also because the submission communicates meaningful investment. A cluster-bootstrap mediation analysis yields the same conclusion. Roughly 36% of the effort effect on future-review priority operates through perceived competence, while the remaining 64% appears to be direct.

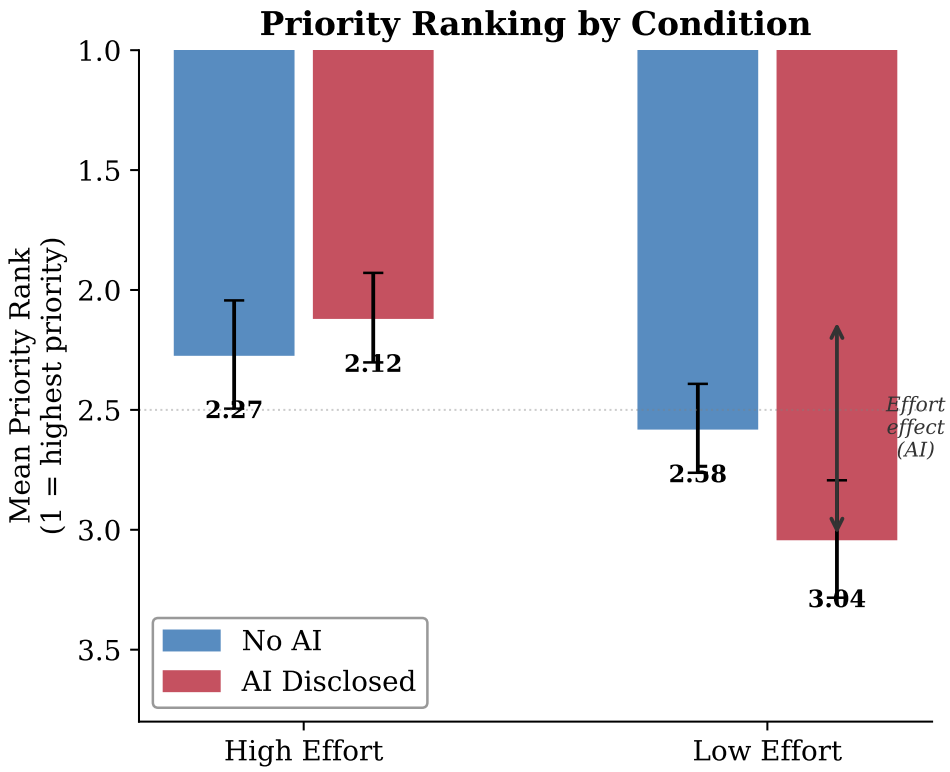


Figure 3: Future-review priority by AI disclosure and visible effort. Lower values indicate greater priority. The sharpest penalty falls on low-effort contributors who disclose AI use, while high-effort contributors are treated similarly whether or not AI is disclosed.

Predictor	Model 1	Model 2	Model 3
Effort	−0.62**	−0.40†	−0.39†
	(0.21)	(0.22)	(0.23)
AI Disclosure	0.15	−0.04	−0.01
	(0.21)	(0.22)	(0.23)
Effort × AI	−0.62	−0.46	−0.50
	(0.42)	(0.42)	(0.43)
Perceived competence		−0.22*	−0.18†
		(0.09)	(0.10)
Perceived code quality			−0.07
			(0.11)

Notes: Linear mixed-effects models with participant random intercepts. Lower values indicate greater future-review priority. Standard errors in parentheses. †$p < .10$, *$p < .05$, **$p < .01$.

Table 9: Pilot experiment: sequential mixed-effects models for future-review priority

5.4 Additional Analyses

The supplementary analyses sharpen the mechanism in three ways. First, the substantive pilot patterns are not driven by a particular code snippet or by review order. Adding stimulus fixed effects and trial-order controls leaves the main coefficients nearly unchanged. The effort effect on future-review priority remains $b = -0.61$ ($p < .01$), and the effects of effort and AI on competence and human contribution remain large and statistically significant.

Second, the conditional effects line up closely with the theory. Table 11 shows that visible effort matters much more for future-review priority when AI is disclosed ($b = -0.92$, $p = .002$) than when it is not ($b = -0.31$, $p = .289$). Likewise, the competence penalty from AI disclosure is modest in the high-effort condition ($b = -0.54$, $p = .058$) but more than doubles in the low-effort condition ($b = -1.27$, $p < .001$). The human-contribution penalty from AI disclosure is also significantly larger when visible effort is weak (−2.15 vs. −1.39). Across all four code stimuli, the low-effort AI-disclosed cell produces the lowest perceived human contribution and, in most cases, the lowest future-review priority.

Third, the pilot clarifies the distinction between attribution and evaluation. Figure 4 shows that AI disclosure sharply lowers perceived competence and human contribution while leaving code quality essentially unchanged. This is the core signature of effort opacity in the experiment: evaluators interpret the person behind the artifact differently even when the artifact itself is held constant.

Predictor	Priority	Acceptance	Competence	Human contribution
Effort	-0.61^{**}	11.62^{**}	0.99^{***}	0.81^{**}
	(0.21)	(3.95)	(0.24)	(0.25)
AI Disclosure	0.21	−0.59	-0.93^{***}	-1.78^{***}
	(0.21)	(3.96)	(0.24)	(0.25)
Effort × AI	−0.58	−2.35	0.78	0.72
	(0.43)	(7.93)	(0.47)	(0.50)
Trial order	−0.02	−1.67	−0.02	−0.06
	(0.10)	(1.83)	(0.11)	(0.12)

Notes: All models include participant random intercepts as well as stimulus fixed effects (not shown). None of the stimulus indicators or the trial-order coefficient is statistically significant at conventional levels. Standard errors are in parentheses. $^{***}p < .001$, $^{**}p < .01$.

Table 10: Pilot experiment: robustness to stimulus and review-order controls

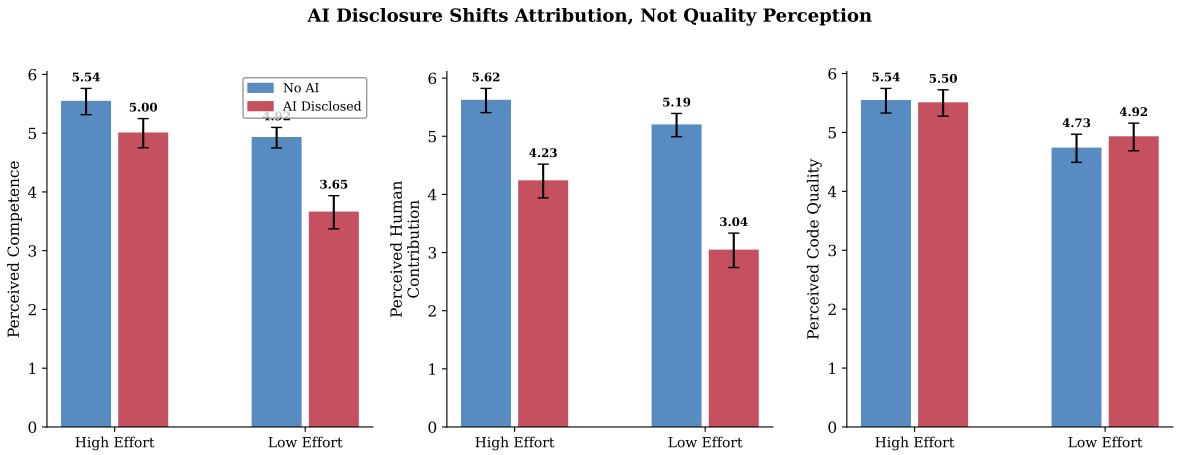


Figure 4: AI disclosure reduces perceived contributor competence and human contribution while leaving the underlying code unchanged. The figure visualizes the attributional shift at the heart of effort opacity.

Outcome	Effect of Effort		Effect of AI Disclosure	
	No AI	AI Disclosed	High Effort	Low Effort
Future-review priority	−0.31	−0.92**	−0.15	0.46
	(0.29)	(0.29)	(0.26)	(0.30)
Perceived competence	0.62*	1.35***	−0.54†	−1.27***
	(0.28)	(0.28)	(0.28)	(0.28)
Perceived human contribution	0.42	1.19**	−1.39***	−2.15***
	(0.27)	(0.39)	(0.31)	(0.29)
Acceptance likelihood	13.69**	9.39†	−3.00	1.31
	(4.25)	(5.31)	(4.07)	(6.37)

Notes: Entries are simple slopes from exploratory mixed-effects models. Lower values on future-review priority indicate greater priority. The strongest effort effect on future-review priority appears when AI is disclosed, and the largest AI penalties for competence and human contribution appear when visible effort is low. $^{***}p < .001$, $^{**}p < .01$, $^{*}p < .05$, $^{\dagger}p < .10$.

Table 11: Pilot experiment: conditional effects for key outcomes

Taken together, the experimental evidence sharpens the interpretation of the field patterns. The detection experiment shows that developers can form AI-suspect judgments from the artifact and that perceived effort helps organize those judgments. The pilot review experiment then shows that once AI is made salient, reviewers credit contributors less for the same underlying work unless the submission supplies strong, human-readable evidence of process and engagement. Importantly, that pilot pattern is not driven by a particular code snippet or by review order, and the future-review results suggest that effort matters partly through a direct reciprocity channel rather than only through

competence or quality beliefs. The field evidence therefore reads less like a pure quality story and more like a broader problem of attribution, evaluative attention, and organizational absorption.

6 Discussion and Conclusion

This paper develops a theory of *effort opacity*—the condition in which the effort behind a contribution becomes difficult to observe, disrupting the information that collaborative systems depend on—and tests it in the context of generative AI. The central implication is that AI may improve individual productivity while generating a misallocation of evaluative attention: contributions that would merit developmental investment become harder to identify when effort signals degrade, and the collaborative system becomes less efficient at screening, developing, and integrating valuable work.

Across the field analysis and pilot review experiment, the evidence is consistent with this argument. In the field setting, contributions with strong AI signatures are associated with more rejection, less developmental feedback, and slower integration—and a pre-registered detection experiment shows that developers can perceive AI involvement above chance, with perceived effort as a primary organizing cue (Appendix C). In the pilot review experiment, AI disclosure lowers perceived contributor competence and human contribution even when the underlying code is held constant, while visible process evidence increases future-review priority. The combined pattern is consistent with the interpretation that AI affects evaluator inferences about the contributor, not only evaluations of the artifact, though the observational design does not permit definitive causal attribution.

The paper makes three contributions to management and strategy research. First, it offers a general theory of effort opacity that applies to any technology that decouples output from effort, not only to AI. By grounding the argument in social exchange theory and signaling theory, and by showing that the same dynamic has recurred from the printing press to factory production to computer-aided design, the paper positions effort opacity as a recurring structural problem in collaborative work that AI merely intensifies and scales.

Second, the paper contributes to research on governance and evaluation in collaborative systems. The results are more consistent with a reallocation of evaluator effort than with simple disengagement. In the field data, detailed developmental feedback becomes less common while lower-effort gatekeeping responses become more common. In the pilot, future-review priority responds to visible process evidence even after accounting for competence and code-quality beliefs. This suggests that evaluators shift from development to gatekeeping under opacity, and that providing visible evidence of human engagement offers a design principle for restoring productive evaluation.

Third, the paper contributes a conceptual lens for thinking about the organizational consequences of technology adoption more broadly. From a strategy perspective, the value of any productivity-enhancing technology depends not only on what it enables the focal producer to do, but also on

how it affects complementary actors whose approval, coordination, and attention are required for value creation (Teece, 1986). Effort opacity identifies one reason why individual productivity gains may fail to translate into organizational value.

Several features of the research design warrant discussion. The field analysis is observational: ground-truth AI use is unobservable, so I rely on a classifier to identify contributions that exhibit strong stylistic signatures of AI generation. This measurement strategy fits the theory—what matters for evaluator response is whether a contribution *appears* AI-assisted, not whether AI was definitively used—but it means I cannot rule out that contributors who use AI differ from non-users in unobserved ways that also affect review outcomes. In particular, contributors may turn to AI for tasks they understand less well, for speculative contributions they are less committed to, or for work that is architecturally misaligned—any of which could generate higher rejection rates for reasons unrelated to evaluator inference about effort. I address this with controls for newcomer status, PR abandonment, and observable contribution characteristics, and the Oster (2019) sensitivity analysis suggests that implausibly large unobserved confounding would be required to explain away the main patterns. The CI/CD exercise provides a useful benchmark, but passing automated tests does not fully absorb architectural-fit issues or integration difficulty. I therefore interpret the observational findings as documenting a robust association between AI-predicted status and evaluator behavior, rather than as definitive evidence of reviewer inference. A natural alternative explanation is that evaluators are simply screening on quality. I address this by controlling for cyclomatic complexity and maintainability index, and the AI coefficient remains large; external evidence further weakens a pure quality account, as Phan et al. (2025) find AI-generated code scores higher on maintainability and Sarkar (2025) find experienced developers accept AI code at higher rates.

A related question concerns the conceptual bridge between the three empirical components. The field study relies on a classifier trained to distinguish pre-2021 human code from synthetic AI code; the detection experiment measures how human developers perceive code snippets; and the pilot experiment manipulates explicit AI disclosure alongside process traces. The classifier captures perceivable stylistic signatures of AI generation—precisely the kind of features that the theory predicts evaluators would respond to—and the detection experiment provides direct evidence that human developers do in fact respond to these features: they detect AI involvement above chance and use perceived effort as an organizing cue. The detection experiment therefore serves as the empirical bridge between the classifier-based field measure and the theory's focus on evaluator perception. That said, the current design does not demonstrate that the specific pull requests flagged by the classifier in the GitHub sample were perceived as AI-suspect by their actual reviewers. Readers should interpret the three studies as converging evidence for the same theoretical construct, with the detection experiment providing the strongest available link between classifier-detected style and human perception.

The operationalization of feedback reallocation (Hypothesis 2) also invites further discussion. The distinction between inline and general comments, combined with the VADER sentiment threshold, provides a useful first approximation of the shift from developmental to gatekeeping

engagement, but these measures do not cleanly separate mentoring from screening. A substantive alternative is that maintainers leave fewer inline comments on architecturally misaligned contributions simply because granular edits are inefficient, rather than because they are withholding developmental investment. Future work incorporating analysis of actual comment content—mentoring specificity, verification demands, or expressions of abandonment—would more directly substantiate the proposed mechanism.

The pilot review experiment is intentionally small and diagnostic, designed to isolate the attributional mechanism under tightly controlled conditions and to inform a larger follow-on study. I place more weight on the direction, consistency, and theoretical specificity of the pilot patterns than on any single coefficient, and the interaction evidence should be interpreted as suggestive rather than conclusive. The pilot results—where AI disclosure lowers perceived competence and human contribution while code quality remains unchanged—are also compatible with source stigma or diluted authorship attribution, rather than solely a reciprocity-based effort inference. The high-visible-effort condition introduces detailed process documentation that may make the submission objectively easier to review, meaning the effort manipulation covaries with reviewability. Disentangling these channels is an important task for the larger follow-on experiment.

I study open-source software, where reciprocity norms and mentorship orientations make effort signals especially salient; the magnitudes may therefore represent an upper bound relative to settings with weaker relational expectations. More broadly, the theory predicts that effort opacity should be most consequential in settings where evaluation is artifact-based, prior ties are weak, and alternative signals of contributor engagement are limited. In settings with strong formal roles, repeated interaction, or rich non-artifact signals (e.g., direct observation of work process, long-standing team relationships), the mechanism may be muted because evaluators have other bases for inferring effort. Establishing these boundary conditions empirically—identifying when effort opacity translates from software review to other collaborative domains, and when formal structures or relational history substitute for artifact-based effort signals—is an important direction for future research. For the present, I view OSS not as a narrow special case but as a strategically revealing setting in which a broader organizational problem becomes visible.

6.1 Conclusion

The productivity gains from generative AI are real and well-documented. This paper argues that they are incomplete as an account of organizational value, because AI may simultaneously degrade the information on which downstream evaluation depends. The result is a potential misallocation of evaluative attention: contributions that would merit developmental investment become harder to identify, and the collaborative system becomes less efficient at screening, developing, and integrating valuable work—even as the average quality of submitted artifacts may improve.

Effort opacity is the mechanism. Whenever a technology decouples the quality of an artifact from the effort behind it, it changes not only what workers produce but what evaluators can learn from

what workers produce. This paper develops that argument and tests it in the context of generative AI, where the disruption is most powerful. The results suggest that this inferential disruption has measurable consequences: for whose work gets approved, for who receives developmental feedback, and for how evaluators allocate the scarce attention that determines whose contributions get incorporated over time. The problem is general, not AI-specific: the same dynamic has recurred across centuries of technological change, and societies have always rebuilt effort legibility—through authorship norms, piece-rate pay, oral examinations, and other institutional innovations. Whether organizations will adapt evaluation practices to restore those signals in the AI era, or whether effort opacity becomes a persistent feature of augmented work, remains an open question. What the evidence makes clear is that the organizational returns to technology adoption will depend not only on the capabilities of the tools, but on the robustness of the human systems through which technology-assisted work is ultimately judged.

7 References

References

Adams, J. S. (1965). Inequity in social exchange. In L. Berkowitz (Ed.), *Advances in Experimental Social Psychology* (Vol. 2, pp. 267–299). Academic Press.

Blau, P. M. (1964). *Exchange and Power in Social Life*. Wiley.

Boudreau, K. J., & Jeppesen, L. B. (2015). Unpaid crowd complementors: The platform network effect mirage. *Strategic Management Journal*, 36(12), 1761–1777.

Brynjolfsson, E., Rock, D., & Syverson, C. (2018). Artificial intelligence and the modern productivity paradox: A clash of expectations and statistics. *NBER Working Paper No. 24001*.

Brynjolfsson, E., Li, D., & Raymond, L. (2025). Generative AI at work. *The Quarterly Journal of Economics, 140(2), 889-942.*.

Bukhari, S., Tan, B., & De Carli, L. (2023). Distinguishing AI- and human-generated code: A case study. *Proceedings of the 2023 Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses (SCORED '23)*, 23–32.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. d. O., Kaplan, J., ... & Zaremba, W. (2021). Evaluating large language models trained on code. arXiv preprint arXiv:2107.03374.

Dabbish, L., Stuart, C., Tsay, J., & 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*, 1277–1286.

Daniotti, S., Wachs, J., Feng, X., & Neffke, F. M. H. (2026). Who is using AI to code? Global diffusion and impact of generative AI. *Science*, 391(6787), 831–835.

Dell'Acqua, F., McFowland, E., Mollick, E. R., Lifshitz-Assaf, H., Kellogg, K., Rajendran, S., Krayer, L., Candelon, F., & Lakhani, K. R. (2023). Navigating the jagged technological frontier: Field experimental evidence of the effects of AI on knowledge worker productivity and quality. *Harvard Business School Working Paper No. 24-013*.

Demirer, M., Cui, Z., Jaffe, S., Musolff, L., Peng, S., & Salz, T. (2026). Production and adoption of generative AI: Evidence from 1.2 million developers. *Management Science* (forthcoming).

de Vaan, M., Stark, D., & Vedres, B. (2015). Game changer: The topology of creativity. *American Journal of Sociology*, 120(4), 1144–1194.

Dietvorst, B. J., Simmons, J. P., & Massey, C. (2015). Algorithm aversion: People erroneously avoid algorithms after seeing them err. *Journal of Experimental Psychology: General*, 144(1), 114–126.

Eltabakh, T. M., Soudi, N. N., & Shawky, D. (2024, December). Quality of AI-generated vs. human-generated code. In *2024 34th International Conference on Computer Theory and Applications (ICCTA)* (pp. 200–205). IEEE.

GitHub. (2023). Octoverse: The state of open source and rise of AI in 2023. *GitHub Blog*.

Gortmaker, J. (2024). Open source software policy in industry equilibrium. *Working Paper*.

Gorwa, R., & Veale, M. (2024). Moderating model marketplaces: Platform governance puzzles for AI intermediaries. *Law, Innovation and Technology*, 1–51.

Gouldner, A. W. (1960). The norm of reciprocity: A preliminary statement. *American Sociological Review*, 25(2), 161–178.

Hutto, C., & Gilbert, E. (2014). VADER: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the International AAAI Conference on Web and Social Media*, 8(1), 216–225.

Kollock, P. (1999). The economies of online cooperation: Gifts and public goods in cyberspace. In M. A. Smith & P. Kollock (Eds.), *Communities in Cyberspace* (pp. 220–239). Routledge.

Lakhani, K. R., & Wolf, R. G. (2005). Why hackers do what they do: Understanding motivation and effort in free/open source software projects. In J. Feller, B. Fitzgerald, S. Hissam, & K. R. Lakhani (Eds.), *Perspectives on Free and Open Source Software* (pp. 3–22). MIT Press.

Lerner, J., & Tirole, J. (2002). Some simple economics of open source. *The Journal of Industrial Economics*, 50(2), 197–234.

Lin, N. (2001). *Social Capital: A Theory of Social Structure and Action*. Cambridge University Press.

March, J. G., & Simon, H. A. (1958). *Organizations*. Wiley.

McCabe, T. J. (1976). A complexity measure. *IEEE Transactions on Software Engineering*, SE-2(4), 308–320.

McElheran, K., Li, J. F., Brynjolfsson, E., Kroff, Z., Dinlersoz, E., Foster, L., & Zolas, N. (2024). AI adoption in America: Who, what, and where. *Journal of Economics & Management Strategy*, 33(2), 375–415.

Nature. (2023). Tools such as ChatGPT threaten transparent science; here are our ground rules for their use. *Nature*, 613(7945), 612.

Nagaraj, A. (2021). Information seeding and knowledge production in online communities: Evidence from OpenStreetMap. *Management Science*, 67(8), 4908–4934.

Nagle, F. (2019). Open source software and firm productivity. *Management Science*, 65(3), 1191–1215.

Noy, S., & Zhang, W. (2023). Experimental evidence on the productivity effects of generative artificial intelligence. *Science*, 381(6654), 187–192.

O'Mahony, S., & Ferraro, F. (2007). The emergence of governance in an open source community. *Academy of Management Journal*, 50(5), 1079–1106.

Oman, P., & Hagemeister, J. (1992). Metrics for assessing a software system's maintainability. *Proceedings of the IEEE Conference on Software Maintenance*, 337–344.

Oster, E. (2019). Unobservable selection and coefficient stability: Theory and evidence. *Journal of Business & Economic Statistics*, 37(2), 187–204.

Padhye, R., Mani, S., & Sinha, V. S. (2014). A study of external community contribution to open-source projects on GitHub. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, 332–335.

Peng, S., Kalliamvakou, E., Cihon, P., & Demirer, M. (2023). The impact of AI on developer productivity: Evidence from GitHub Copilot. *Working Paper*.

Phan, T., et al. (2025). AI-generated code quality: A comparative analysis of human and machine-written software. *Working Paper*.

Piskorski, M. J., & Toffel, M. W. (2013). A social strategy: How we profit from social media. *Harvard Business Review Press*.

Russell, R., et al. (2025). People who frequently use ChatGPT for writing tasks are accurate and robust detectors of AI-generated text. *Working Paper*.

Sarkar, S. K. (2025). AI Agents, Productivity, and Higher-Order Thinking: Early Evidence From Software Development. Available at SSRN 5713646.

Spence, M. (1973). Job market signaling. *The Quarterly Journal of Economics*, 87(3), 355–374.

Teece, D. J. (1986). Profiting from technological innovation: Implications for integration, collaboration, licensing and public policy. *Research Policy*, 15(6), 285–305.

Török, G., Pomiechowska, B., Csibra, G., & Sebök, M. (2021). Rationality in joint action: Maximizing coefficiency in coordination. *Psychological Science*, 32(10), 1624–1636.

Tsay, J., Dabbish, L., & Herbsleb, J. (2014). Influence of social and technical factors for evaluating contribution in GitHub. In *Proceedings of the 36th International Conference on Software Engineering*, 356–366.

von Krogh, G., Spaeth, S., & Lakhani, K. R. (2003). Community, joining, and specialization in open source software innovation: A case study. *Research Policy*, 32(7), 1217–1241.

Zhang, X., Yu, Y., Gousios, G., & Rastogi, A. (2022). Pull request decisions explained: An empirical overview. *IEEE Transactions on Software Engineering*, 49(2), 849–871.

A Formal Model of Effort Opacity

This appendix provides a simple model of effort opacity that formalizes the verbal argument in Section 2. Consider a setting with two players: a contributor C who produces a work artifact, and a reviewer E who decides whether to accept it and how much developmental feedback to invest.

The contributor chooses effort $e \in [0, 1]$ and has access to a production technology with capability $\tau \in [0, 1]$. Effort and technology jointly determine quality:

$$q = q(e, \tau), \qquad \frac{\partial q}{\partial e} > 0, \quad \frac{\partial q}{\partial \tau} > 0 \tag{2}$$

The artifact also generates observable cues $\mathbf{s} = (s_1, \ldots, s_k)$—process traces such as documentation, reasoning, and iteration evidence, distinct from output quality q—from which the reviewer infers effort:

$$\mathbf{s} = g(e, \tau) + \boldsymbol{\eta} \tag{3}$$

where $\boldsymbol{\eta}$ is noise and g satisfies $\partial g/\partial e > 0$, $\partial g/\partial \tau > 0$, and $\partial^2 g/\partial e\, \partial \tau < 0$: the marginal contribution of effort to observable cues decreases as technology capability increases.

The contributor selects effort to maximize expected payoff:

$$\max_{e} \; P(\text{accept} \mid e, \tau) \cdot B - c(e, \tau) \tag{4}$$

where B is the value of acceptance and $c(e, \tau)$ is the cost of effort, with $\partial c/\partial e > 0$ and $\partial c/\partial \tau \leq 0$.

The reviewer forms beliefs about both quality and effort from the observable cues. Let $\hat{q} = \hat{q}(\mathbf{s})$ denote the reviewer's estimate of output quality and $\hat{e} = f(\mathbf{s})$ the reviewer's estimate of contributor effort, both inferred from $\mathbf{s}$. The reviewer then makes two decisions. First, an acceptance decision:

$$V = \alpha\, \hat{q} + (1 - \alpha)\, \hat{e} \geq \bar{V} \tag{5}$$

where $\alpha < 1$ captures that reviewers weigh the contributor's apparent investment, not only the artifact's perceived adequacy. Note that $\hat{q}$ is the reviewer's inference about quality from the noisy cues, not true quality $q(e, \tau)$, which is unobserved. Second, a feedback investment decision:

$$\max_{r} \; \hat{e} \cdot \theta(r) - c_E(r) \tag{6}$$

where $\theta(r)$ is the improvement from feedback and the return is scaled by $\hat{e}$ because development is more valuable when the contributor is believed to be genuinely engaged. The first-order condition yields $r^* = r(\hat{e})$ with $r' > 0$.

Effort opacity is defined as:

$$\Omega(\tau) \equiv 1 - \text{Corr}(\hat{e},\, e \mid \tau) \tag{7}$$

As τ increases, the cross-partial compresses the mapping from effort to cues, Ω rises, and the reviewer's inference becomes uninformative—even as output quality may remain high.

Proposition 1 (Evaluative Penalty Under Opacity)**.** *Fix perceived quality $\hat{q}$. As τ increases, $\Omega(\tau)$ rises. Technology-assisted contributions receive lower evaluations V than contributions with equivalent perceived quality that are not technology-assisted, because $\hat{e}$ is discounted. Rejection rates increase.*

Proposition 2 (Feedback Reallocation)**.** *Under high effort opacity, r^* declines. Reviewers shift from developmental engagement toward gatekeeping, because they cannot reliably identify contributions whose contributors would reciprocate their investment.*

The model also shows that the penalty can be attenuated. If the contributor provides supplementary signals $\mathbf{s}^+$ that are informative about e independently of the technology-mediated cues, the reviewer's inference improves: $\hat{e} = f(\mathbf{s}, \mathbf{s}^+)$, and $\Omega(\tau \mid \mathbf{s}^+) < \Omega(\tau)$.

B Work Efficiency

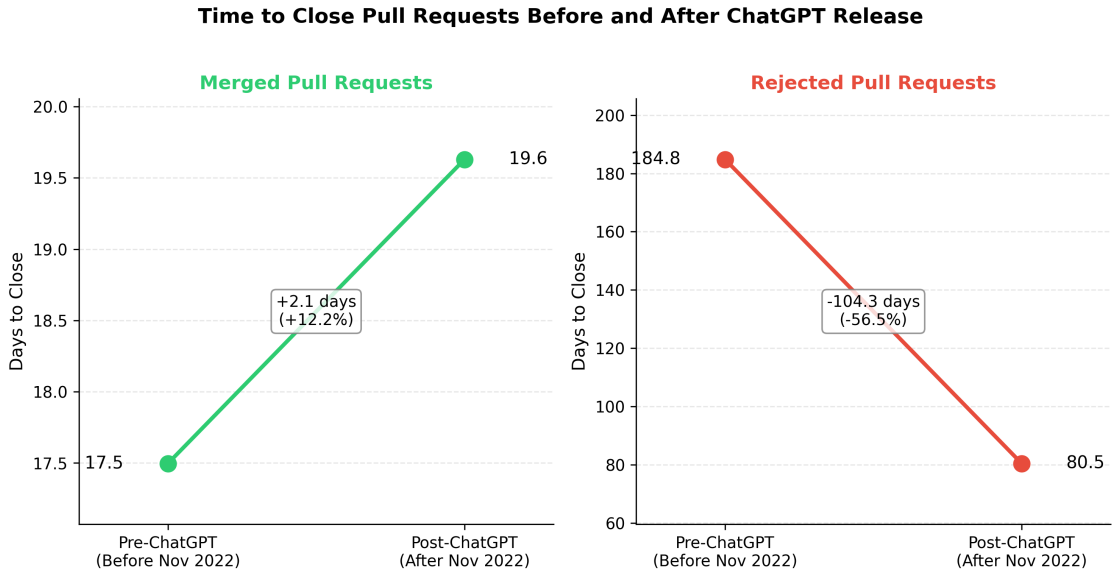


Figure 5: Average Number of Days Taken to Close a PR Before and After ChatGPT

	Main Effect Models			Interaction Models		
	(1)	(2)	(3)	(4)	(5)	(6)
AI PR	14.46***	21.45***	30.35***	1.881	10.07	19.59**
	(3.666)	(5.180)	(5.071)	(5.401)	(6.115)	(6.810)
Large Repo	–	–	–	-17.01***	–	–
				(2.369)		
AI PR × Large Repo	–	–	–	23.18**	19.56**	18.32*
				(7.362)	(7.278)	(7.248)
Repo FE	No	Yes	Yes	No	Yes	Yes
Month FE	No	No	Yes	No	No	Yes
Observations	13,614	13,614	13,614	13,614	13,614	13,614
R^2	0.0016	0.0848	0.0992	0.0055	0.0854	0.2412

Notes: All models include controls for initial deletions, initial additions, initial changed files, and code quality metrics (coefficients omitted for brevity). Standard errors clustered at the repository level in FE models. Large Repo is a time-invariant indicator absorbed by repository fixed effects in Columns (5)–(6); its main effect is reported only in Column (4), which omits repository fixed effects. The interaction AI PR × Large Repo is identified because AI status varies within repositories. Significance levels: *** p<0.001, ** p<0.01, * p<0.05, . p<0.1.

Table 12: Cross-section of AI-predicted PRs and Repo Size on Time to Close PRs

C Code Detection Experiment: Full Results

This appendix reports the full design, results, and analyses from the pre-registered code detection experiment summarized in Section 4.7 of the main text. Full details are available at osf.io/76nrt.

D Experimental Stimuli

Figures 6 and 7 show two experimental conditions applied to representative code artifacts, as displayed to participants. Figure 6 shows the high-effort, AI-disclosed condition: the pull request description includes diagnostic reasoning, iterative problem-solving, and codebase-specific references, and an "AI-assisted" badge appears in the PR header. Figure 7 shows the low-effort, no-disclosure condition: the description is sparse and generic, and no AI disclosure cue appears. Note that each condition is applied to a different fictional repository; the two figures illustrate different codebases.

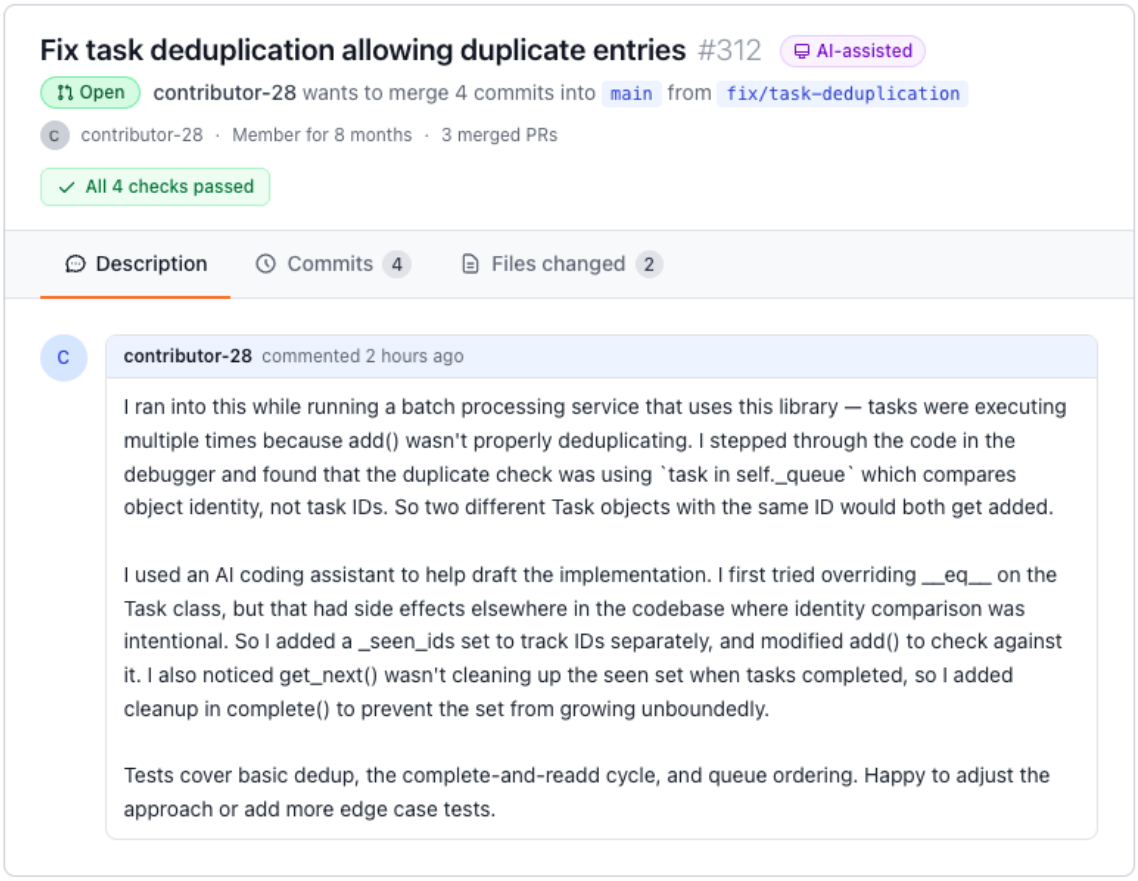


Figure 6: High-effort, AI-disclosed condition (Stimulus A: task deduplication). The PR description displays diagnostic reasoning, iteration, and codebase-specific references. An "AI-assisted" badge appears in the pull request header.

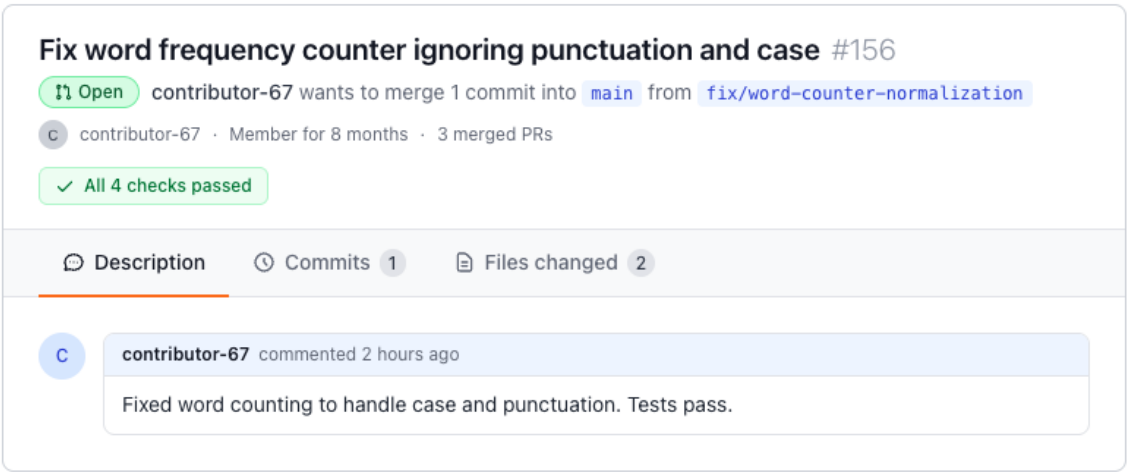


Figure 7: Low-effort, no-disclosure condition (Stimulus B: word frequency counter). The PR description is sparse and generic, and no AI disclosure cue appears.