From Protocols-to-Prompts (P2P): Mapping the Agentification of Open Networking Education

Venkat Sai Suman Lamba Karanam
Bowling Green State University
Bowling Green, Ohio, USA
vlambak@bgsu.edu

Zahmeeth Sakkaff
West Virginia University Institute of Technology
Beckley, West Virginia, USA
West Virginia University
Morgantown, West Virginia, USA
zahmeeth.sakkaff@mail.wvu.edu

Shubham Sundriyal
Bowling Green State University
Bowling Green, Ohio, USA
ssundr@bgsu.edu

Abstract

AI coding assistants and agentic software workflows since late-2022 have transformed how networking is taught, practiced, and learned. Formal curricula, however, have yet to address this transition. In this paper, we present a large-scale measurement study using 1318 GitHub repositories to gauge how open networking education is evolving in the generative-AI era. The mined 1318 GitHub repositories span the 2009–2026 period and are scored each along three axes, each measuring a pedagogical signal: (1) **Conceptual Depth (CD)** for protocol-level vocabulary density; (2) **Operational Abstraction (OA)** for orchestration and API-layer presence; and (3) **Agent/Tool Dependence (ATD)** for explicit AI framework instrumentation. We compare pre- and post-generative-AI artifacts using a November 2022 cutoff – at the dawn of ChatGPT release.

Using extensive analysis we report the following findings. Within 208 repositories spanning the cutoff, we show a significant decline in mechanism-centric signals ($\Delta_{\text{CD}} = -0.085$, $p < 0.001$) – with a concurrent and consistent rise in orchestration-layer indicators ($\Delta_{\text{OA}} = +0.063$, $p = 0.008$). Interestingly, ATD scores remain uniformly negligible across the entire corpus (mean ≈ 0.002) – a key finding that indicates that agentification is a shift in the tooling substrate rather than of explicit AI instrumentation.

Additionally, our complementary analyses show that Docker adoption increased from 9.6% to 14.3% post-cutoff, GNS3 displaces Mininet as the dominant emulation tool (1.2% $\rightarrow$ 4.3%). From a problem-structure/activity perspective, we report that Parsons-style problems double in prevalence (1.2% $\rightarrow$ 2.5%). Networking and computing testbeds, specifically the NSF PAWR testbeds of FABRIC, COSMOS, POWDER, NRP remain largely absent from open educational repositories in both pre- and post-cutoff epochs. We manually validate our repository-level case studies across five spanning repositories to confirm that these aggregate findings (over 1318 repos) match at the per-repository level. Finally, we present a taxonomy of pedagogical regimes and concrete recommendations for AI-aware networking curricula with the goal of preserving mechanistic understanding while embracing AI contemporary tooling.

Reproducibility Commitment: All code, data, manual validation sample, and artifacts will be released upon publication.

CCS Concepts

• **Social and professional topics** → **Computing / technology policy**; • **Software and its engineering** → **Development frameworks and environments**.

Keywords

networking education, agentification, GitHub mining, repository ecology, pedagogical signals, generative AI, open educational resources, measurement study

ACM Reference Format:
Venkat Sai Suman Lamba Karanam, Zahmeeth Sakkaff, and Shubham Sundriyal. 2026. From Protocols-to-Prompts (P2P): Mapping the Agentification of Open Networking Education. In *Proceedings of Workshop on AI for Networking Education (A4NE '26).* ACM, New York, NY, USA, 10 pages.

1 Introduction

Computer networks pedagogy has long emphasized mechanism-level understanding. For example, packet formats, protocol behavior, routing dynamics, socket programming, congestion control, measurement, and troubleshooting [15] are all taught as integral components towards understanding the field technically. This long-standing pedagogy expected students to build, configure, break, and debug networked systems directly – a hallmark of fields in systems. However, the rise of large language models (LLMs) [6], coding assistants [10], and agentic software workflows [4] has disrupted this teaching model. Learners – both new and experienced – now encounter networking concepts and problems through prompts, AI-generated code, orchestration scripts, and AI-mediated explanations rather than through direct *full manual* construction alone.

In this paper, we ask the following question– *is this pedagogical shift in networking education already observable in the open-source educational ecosystem,* ***before*** *it appears in textbooks or official course catalogs?*

To answer the question, we perform a large-scale study on GitHub [2] repositories that teach networking through labs, tutorials, assignments, notebooks, and reproducible examples, with almost all of them associated with universities or technical institutes adjacent to teaching. Care was taken to chose repositories

A4NE '26, Denver, CO, USA

ACM ISBN 978-1-4503-XXXX-X/2018/06

that sit between formal curricula and everyday practice – all repositories are public, revisable, executable, and used by students and instructors even outside their original institutional context. This corpus allows us to analyze the changes to reveal how networking education is adapting to AI-era tooling in real time. Figure 2 illustrates the timeline that spans the repository collection in this study.

Our main claim is that open networking education (ONE) is undergoing a measurable shift from *mechanism-centric construction* that emphasize conceptual depth toward *orchestration-centric supervision* that relies more on tool-dependent workflows (see Fig. 1). In the former model (mechanism-centric), students focus on implementation, inspection, and debugging networking mechanisms directly. In the latter (orchestration-centric), students focus on developing skills to coordinate diverse set of tools, generated code, APIs, simulators, notebooks, and agents. We must note that this paradigm shift does not necessarily imply a loss of rigor, but rather it relocates where conceptual effort is placed i.e., that relocation has consequences for curriculum design as we empirically support in this study.

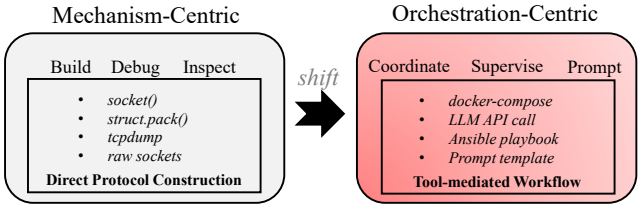

Figure 1: The mechanism-centric to orchestration-centric pedagogical shift. Repositories move from direct protocol construction (socket calls, packet inspection, raw debugging) toward coordinated, tool-mediated workflows (API orchestration, LLM calls, infrastructure-as-code).

The main contributions of this paper are summarized as follows. (1) We mine 1318 GitHub repositories spanning a time period of early-2009 to mid-2026. We compare pre- and post-generative-AI artifacts with a cut-off set at November 2022, when ChatGPT was first introduced broadly. We design three pedagogical axes to support our analysis, namely, **conceptual depth (CD)**, **operational abstraction (OA)**, and **agent/tool dependence (ATD)**. (2) For our empirical analysis, we perform a repository ecology analysis, a pedagogical regime taxonomy, and a set of visualizations that show how protocol reasoning, debugging practice, and AI-mediated workflows are co-evolving as both the networking field and AI-adoption continues to mature. (3) Based on our findings, we provide concrete curriculum design recommendations with the goal of not only preserving mechanistic understanding like the pre-2022 epoch, but also to equip students and instructors with the tooling fluency demanded by the generative-AI era.

Rest of this paper is organized as follows. Section 2 presents the dataset and methodology. Section 3 presents the scoring framework. Section 4 presents the results. Section 5 presents the discussion and implications. Section 6 concludes the paper.

2 Dataset and Methodology

In this section, we present our methodology for the dataset collection, enrichment, temporal labeling, and file taxonomy used in this study.

2.1 Repository Collection

We collected open networking education repositories from GitHub using its Search API [11, 13]. To this end, we applied 14 keyword queries as follows. *computer networks lab tutorial*, *networking programming assignment*, *socket programming tutorial*, *BGP OSPF lab assignment*, *TCP IP networking course*, *network simulation lab*, *SDN openflow tutorial*, *network security lab assignment*, *mininet tutorial assignment*, *wireshark lab networking*, *computer networking course material*, *network protocols implementation*, *data communications lab*, and *network emulation tutorial*. Additionally, 8 topic-based queries were issues, namely, `topic:computer-networks`, `topic:networking`, `topic:network-programming`, `topic:socket-programming`, `topic:network-security`, `topic:software-defined-networking`, `topic:computer-networking`, and `topic:network-simulation`.

To filter out noise from the collected repositories, we applied the following inclusion criteria: (i) at least two stars, (ii) minimum repository size of 5 KB, (iii) non-fork, non-archived status, and (iv) presence of at least one networking-relevant term in the repository name, description, or topic tags, with explicit exclusion of repositories whose metadata indicated machine learning, computer vision, or unrelated domains. After deduplication, this yielded 1323 repositories. During preprocessing, we excluded five repositories due to corrupt cache files and / or re-fetch failures, thus resulting in a final corpus of 1318 repositories.

2.2 Data Enrichment

For each repository, we collected: (i) full commit history (up to 300 commits), including commit timestamps, messages, and modified file paths; (ii) the complete file tree with file-type labels; (iii) sampled file content, i.e., up to five files per category (Python, notebook, shell, Markdown, configuration) with the first 3,000 characters per file; (iv) the repository README; and (v) declared dependencies from `requirements.txt`, `package.json`, and related manifests. All data was collected via the GitHub REST API in June 2026 and stored as per-repository JSON artifacts.

2.3 Temporal Labeling

We use November 30, 2022, i.e., the public release date of ChatGPT, as the generative-AI inflection point. A repository is labeled *pre-cutoff* if its creation date precedes this threshold and *post-cutoff* otherwise. Repositories with commits on both sides of the threshold are labeled *spanning* and form a longitudinal cohort for within-repository analysis. This yields 925 pre-cutoff, 398 post-cutoff, and 208 spanning repositories.

2.4 File Taxonomy

We categorize repository files into eight types: Python scripts, Jupyter notebooks [14], shell scripts, Markdown/reStructuredText documentation, configuration files (YAML, TOML, JSON, INI), Dockerfiles and Compose files, C/C++ source files, and web files (HTML,

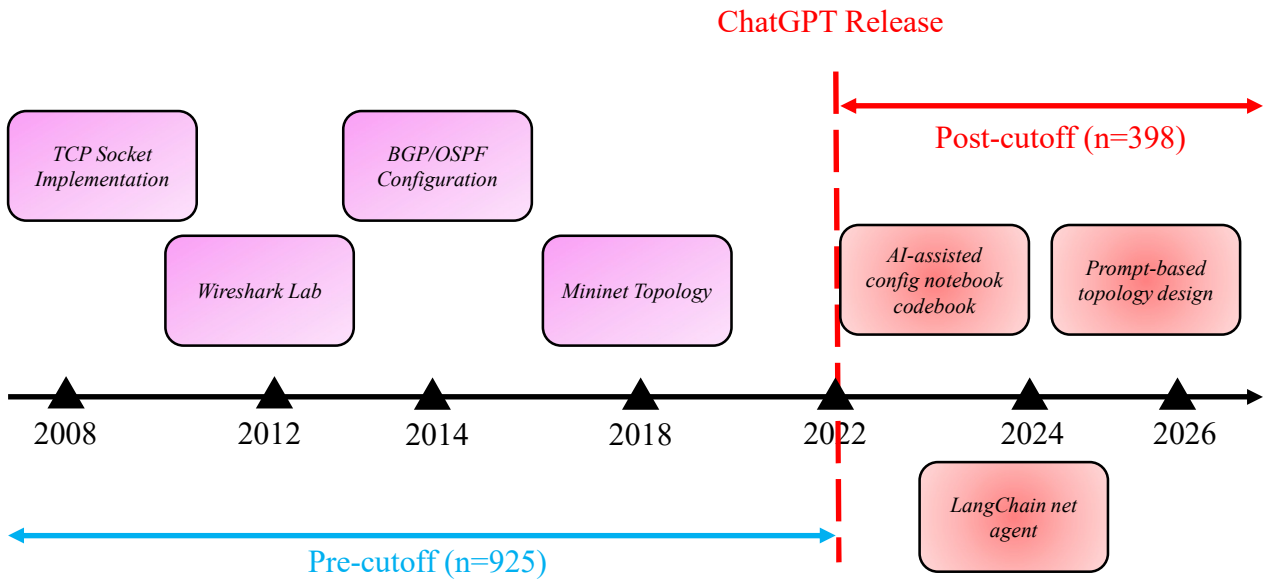

Figure 2: Repository ecosystem timeline (2009–2026). The November 2022 ChatGPT release demarcates pre-cutoff ($n = 925$) and post-cutoff ($n = 398$) cohorts. Representative pre-cutoff artifact types include direct protocol construction labs (TCP socket implementation, Wireshark packet inspection, BGP/OSPF configuration, Mininet topology emulation). Post-cutoff artifacts reflect a shift toward AI-mediated workflows (LangChain network agents, AI-assisted configuration notebooks, prompt-based topology design).

JS, CSS). We designate Python, C/C++, and shell files as *mechanism-oriented* and notebooks, configuration, and Docker files as *abstraction-oriented*, reflecting their association with direct construction versus orchestration-layer workflows, respectively.

2.5 Ethical and Reproducibility Considerations

All repositories analyzed are publicly accessible under open licenses. No personally identifiable information beyond public GitHub usernames was collected. All code, scoring scripts, raw data, and figures will be released as a reproducible artifact upon publication.

3 Scoring Framework

In this section, we present the three-axis scoring framework used to evaluate each repository. We score each repository using keyword-based signal extraction over a text corpus assembled from the repository README and sampled file snippets. All scores are normalized to $[0, 1]$.

3.1 Axis 1: Conceptual Depth (CD)

CD measures the density of mechanism-level vocabulary in a repository. We define two keyword sets: a *positive* set of protocol-level terms (e.g., `three-way handshake`, `congestion window`, `struct.pack`, `tcpdump`, `routing table`, `TTL`) and a *negative* set of abstraction dismissal phrases (e.g., *handled automatically*, *abstracted away*, *black box*). The raw score is the hit ratio:

$$\mathrm{CD}_{\mathrm{raw}} = \frac{|\text{pos hits}| - |\text{neg hits}|}{|\text{pos hits}| + |\text{neg hits}|} \tag{1}$$

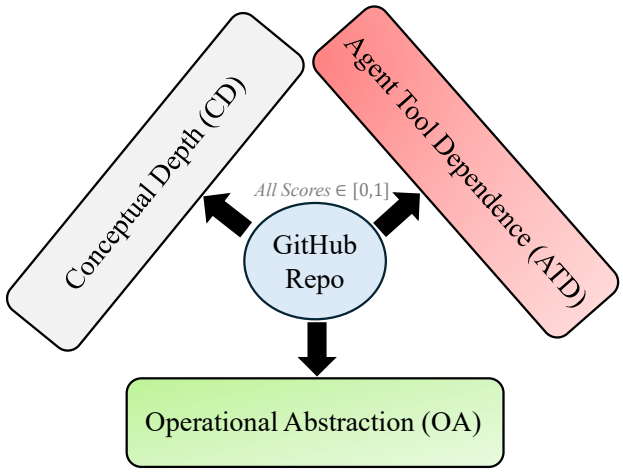

Figure 3: Three-axis scoring framework. Each repository is evaluated on Conceptual Depth (CD), measuring protocol-level vocabulary density; Operational Abstraction (OA), measuring orchestration and API-layer presence; and Agent/Tool Dependence (ATD), measuring explicit AI framework instrumentation. All scores are normalized to $[0, 1]$.

which is normalized to $[0, 1]$. A high CD score indicates a repository dense with direct protocol manipulation, implementation vocabulary, and debugging practice.

To illustrate an example, consider a repository containing the phrase "congestion windows are abstracted away by the library." in its README. In our CD methodology, this phrase contributes one hit to both positive and negative sets via (`congestion window`) and (`abstracted away`) respectively. This results in CD of zero to CD_{raw} for that repository. On the other hand, a repository that discusses congestion windows without any form of dismissal such as the phrase "we implement the congestion window update rule from RFC 5681" in its README contributes positively to CD and hence receives a higher CD_{raw}.

3.2 Axis 2: Operational Abstraction (OA)

OA measures the degree to which a repository operates at the orchestration or API layer rather than at the raw socket or system-call layer. The positive signal set includes cloud and infrastructure terms (e.g., `terraform`, `docker-compose`, `kubectl`, REST API references, SDK mentions), while the negative set captures raw-layer indicators (e.g., `AF_INET`, `SOCK_RAW`, `setsockopt`, `iptables`, `tcpdump`). OA additionally incorporates a structural signal: the fraction of abstraction-oriented files (notebooks, configuration files, Dockerfiles) minus the fraction of mechanism-oriented files (Python scripts, C/C++ sources, shell scripts), blended as:

$$\mathrm{OA} = 0.6 \cdot \mathrm{OA}_{\mathrm{content}} + 0.4 \cdot \mathrm{OA}_{\mathrm{structure}} \tag{2}$$

3.3 Axis 3: Agent/Tool Dependence (ATD)

ATD measures the presence of AI agent and LLM tooling in a repository. The signal sources are: (i) keyword density over AI-related terms in the text corpus (e.g., `langchain`, `openai`, `autogen`, `prompt template`, `function_calling`); (ii) whether any sampled Python file imports a known AI framework; and (iii) whether any declared dependency resolves to an AI library. The three components contribute 0.4 + 0.4 + 0.2, respectively, capped at 1.0.

3.4 Temporal Decomposition

For spanning repositories, we derive per-epoch axis scores from commit-message text and modified file paths rather than from HEAD content, since HEAD reflects only the current state of the repository. Commit messages and file paths are partitioned at the cutoff date, and axis scores are computed separately on the pre- and post-cutoff partitions. The delta $\Delta = \mathrm{score}_{\mathrm{post}} - \mathrm{score}_{\mathrm{pre}}$ is used as the primary longitudinal signal.

3.5 Pedagogical Regime Classification

We classify each repository into one of four regimes based on its CD and OA scores relative to a threshold of 0.5: *mechanism-centric* (CD ≥ 0.5, OA < 0.5), *orchestration-centric* (CD < 0.5, OA ≥ 0.5), *hybrid* (both ≥ 0.5), and *shallow* (both < 0.5). The quadrant boundary of 0.5 is the midpoint of the normalized score range and is not tuned to the data.

3.6 Limitations

We note that keyword-based scoring used in this study is a coarse instrument. It captures surface vocabulary but cannot distinguish between a repository that *teaches* packet headers and one that merely *references* them. Sampled file content (first 3,000 characters, up to five files per category) may miss signals in larger or differently structured repositories. Temporal decomposition via commit messages is a best-effort proxy, as commit text is sparse and often uninformative. These limitations motivate the visualization-driven framing of this paper: our scores are pedagogical signals, not ground-truth labels, and the figures expose distributional patterns rather than rank individual repositories.

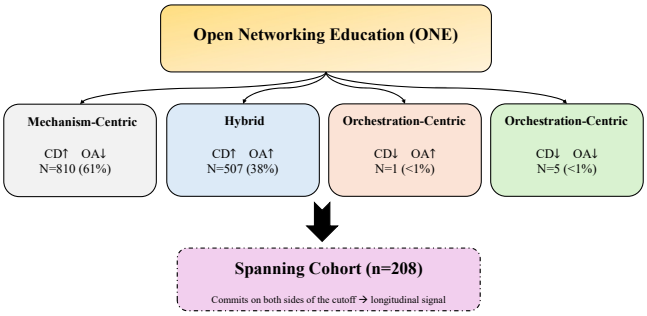

Figure 4: Pedagogical regime taxonomy derived from CD×OA quadrant classification. The corpus is dominated by mechanism-centric (61%) and hybrid (38%) repositories. The 208-repository spanning cohort provides the primary longitudinal signal.

4 Results

In this section, we present the results of the study across the following dimensions: dataset characterization, axis score comparison, spanning repository analysis, repository ecology, agent/tool dependence, technical concept prevalence, package ecosystem shift, problem structure, testbed reliance, manual validation, and repository-level case studies.

4.1 Dataset Characterization

The corpus comprises 1318 GitHub repositories spanning 2009–2026. Using November 2022 as the generative-AI cutoff, we obtain 925 pre-cutoff and 398 post-cutoff repositories. A subset of 208 repositories span the cutoff, i.e., they have commits both before and after November 2022, forming a natural longitudinal cohort for within-repository analysis. Post-cutoff repositories are newer and less mature on average (median stars: 15 vs. 38 for pre-cutoff), reflecting the recency of the post-2022 cohort rather than lower relevance.

4.2 Axis Scores: Cross-Sectional Comparison

Table 1 reports mean axis scores by epoch. CD is significantly higher in post-cutoff repositories ($\mu_{\mathrm{pre}} = 0.824$, $\mu_{\mathrm{post}} = 0.869$, $p = 0.0015$). This is initially counterintuitive, as one would expect newer repositories to show lower mechanism-level engagement. We attribute this to a *selection effect*: post-2022 repositories that explicitly teach networking are disproportionately those that foreground protocol-level content, perhaps as a deliberate contrast to

Table 1: Mean axis scores by epoch (pre/post Nov. 2022 cutoff). Significance via two-sample t-test; $^{**}p < 0.01$.

Axis	Pre	Post	p-value
Conceptual Depth (CD)	0.824	0.869	0.0015^{**}
Operational Abstraction (OA)	0.371	0.395	0.061
Agent/Tool Dependence (ATD)	0.001	0.002	0.274

Table 2: Mean post−pre axis deltas for the 208 spanning repositories. Significance via one-sample t-test against zero; $^{**}p < 0.01$, $^{***}p < 0.001$.

Axis	Δ (post − pre)	p-value
Conceptual Depth (CD)	−0.085	$< 0.001^{***}$
Operational Abstraction (OA)	+0.063	0.008^{**}
Agent/Tool Dependence (ATD)	−0.004	0.451

AI-mediated tooling available elsewhere. OA shows a directional increase ($\mu_{\text{pre}} = 0.371$, $\mu_{\text{post}} = 0.395$) that does not reach significance cross-sectionally ($p = 0.061$).

4.3 Spanning Repository Analysis

The 208 spanning repositories provide the most direct evidence of within-ecosystem change, as they control for repository selection bias. Table 2 reports mean post-minus-pre deltas on commit-message-derived axis scores. CD decreases significantly ($\Delta = -0.085$, $p < 0.001$), while OA increases significantly ($\Delta = +0.063$, $p = 0.008$). These deltas confirm the paper's central thesis: within the same repositories, the balance of pedagogical effort shifted away from mechanism-level construction and toward orchestration-level coordination after the generative-AI inflection point.

4.4 Repository Ecology: Structural Signals

Structural features of the repositories reveal complementary signals of the abstraction shift. Docker adoption increased from 9.6% of pre-cutoff repositories to 14.3% post-cutoff, reflecting a shift toward containerized, reproducible networking environments [12] rather than bare-metal or host-based labs [16]. Notebook presence is low but rising (2.7% pre vs. 3.5% post), consistent with the broader movement toward interactive, executable pedagogy. Fig. 4 shows the pedagogical regime distribution: mechanism-centric repositories decrease from 62.6% to 58.0% post-cutoff, while hybrid repositories increase from 37.0% to 41.5%. Orchestration-centric repositories remain rare (< 1%), indicating the shift is an *overlay* of orchestration-layer concerns atop existing protocol-focused material rather than a wholesale replacement.

4.5 Agent and Tool Dependence: A Null Result with Implications

ATD scores are negligible across the corpus (mean ≈ 0.001–0.002), with only 15 repositories showing any detectable AI-related signal and only 2 containing explicit AI framework imports (e.g., `openai` [7], `langchain`). This null result is itself informative. The pedagogical shift we observe, i.e., rising OA and declining CD, is *infrastructural* rather than *declarative*: repositories are adopting Docker, APIs, and orchestration patterns without yet explicitly embedding AI agents or LLM calls as first-class pedagogical components. Agentification of networking education is currently a shift in *tooling substrate* rather than *explicit AI instrumentation*.

4.6 Technical Concept Prevalence Over Time

Fig. 5 shows how the prevalence of mechanism-layer and orchestration or AI-layer terms evolves across repository creation years. Mechanism-layer terms (TCP, UDP, socket calls, packet-level tooling) remain consistently prevalent throughout the corpus. Their dominance plateaus after 2022, while orchestration and AI-layer terms rise sharply. Docker shows the strongest sustained growth, consistent with the broader adoption of containerized networking environments. Prompt- and agent-related terms, absent before 2022, emerge and accelerate post-cutoff, reflecting the generative-AI inflection point directly.

4.7 Open-Source Package Ecosystem Shift

Fig. 6 shows the shift in declared package dependencies and imports across the pre/post-cutoff cohorts. Data/ML packages rise from 10.2% to 14.3% post-cutoff, indicating that networking education is increasingly incorporating data-driven analysis and measurement workflows. Networking-specific packages (Scapy, pyshark, netmiko) decline from 16.9% to 5.8%, consistent with reduced emphasis on raw packet manipulation. AI/LLM packages remain rare (< 1%) but emerge exclusively in post-cutoff repositories, reinforcing the ATD null result.

4.8 Problem Structure: Parsons-Style vs. Traditional

Fig. 7 examines whether the *structure* of programming problems is shifting alongside the tooling substrate. We detect three signal types: explicit Parsons problems [9, 19] (code-ordering tasks), scaffolded problems (partial implementations with `#TODO`, `#YOUR CODE HERE`, or fill-in markers), and traditional open-ended problems (e.g., "implement from scratch", "build your own"). Parsons-style problems double post-cutoff (1.2% → 2.5%), consistent with AI-assisted generation of structured scaffolded tasks. Traditional problems remain stable (17.2% → 17.6%), indicating open-ended construction has not been displaced. The time-series panel shows scaffolded and traditional problems converging post-2022, i.e., a structural narrowing of the gap between guided and open-ended pedagogy.

4.9 Testbed and Simulator Reliance

Fig. 8 shows how reliance on network testbeds and simulators shifts across the cutoff. GNS3 rises sharply post-cutoff (1.2% → 4.3%, $\Delta = +3.1\%$), displacing Mininet as the dominant emulation tool. Mininet (3.2% → 2.5%) and NS-3 (1.6% → 1.0%) both decline, indicating a shift from programmatic, script-driven emulation toward GUI-mediated topology design, consistent with the broader orchestration-centric trend. NSF PAWR testbeds [1, 5, 20, 21] (FABRIC, COSMOS, POWDER, NRP) remain essentially absent in both

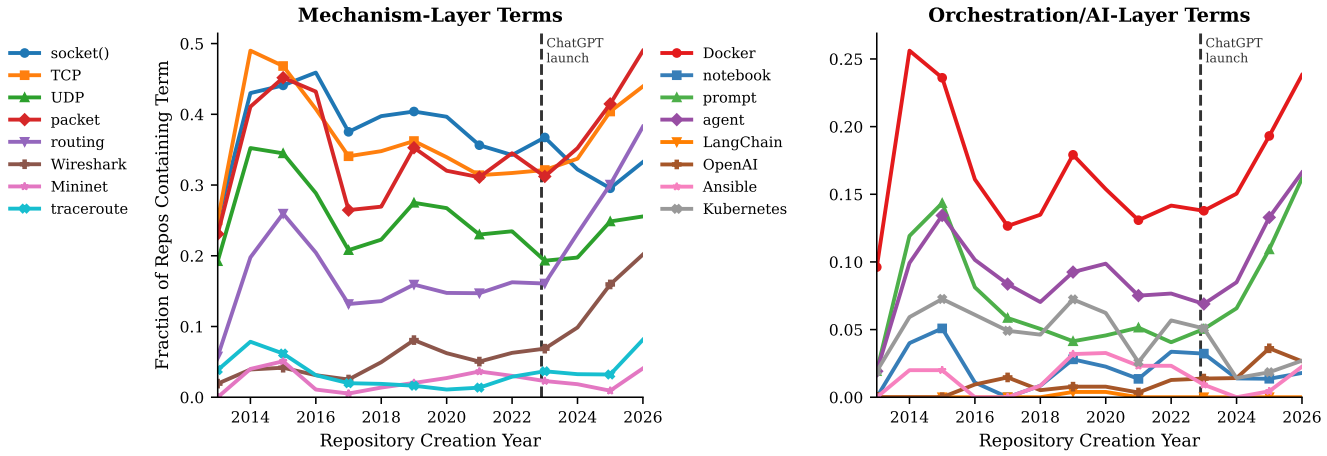

Figure 5: Technical concept prevalence over repository creation year (2013–2026), measured as the fraction of repositories containing each term in their README, description, or commit history. Lines are smoothed with a two-year rolling average. *Left:* Mechanism-layer terms remain prevalent throughout but plateau post-2022. *Right:* Orchestration and AI-layer terms rise sharply after the November 2022 inflection point (dashed line), with LangChain and prompt-related terms showing the steepest post-cutoff growth.

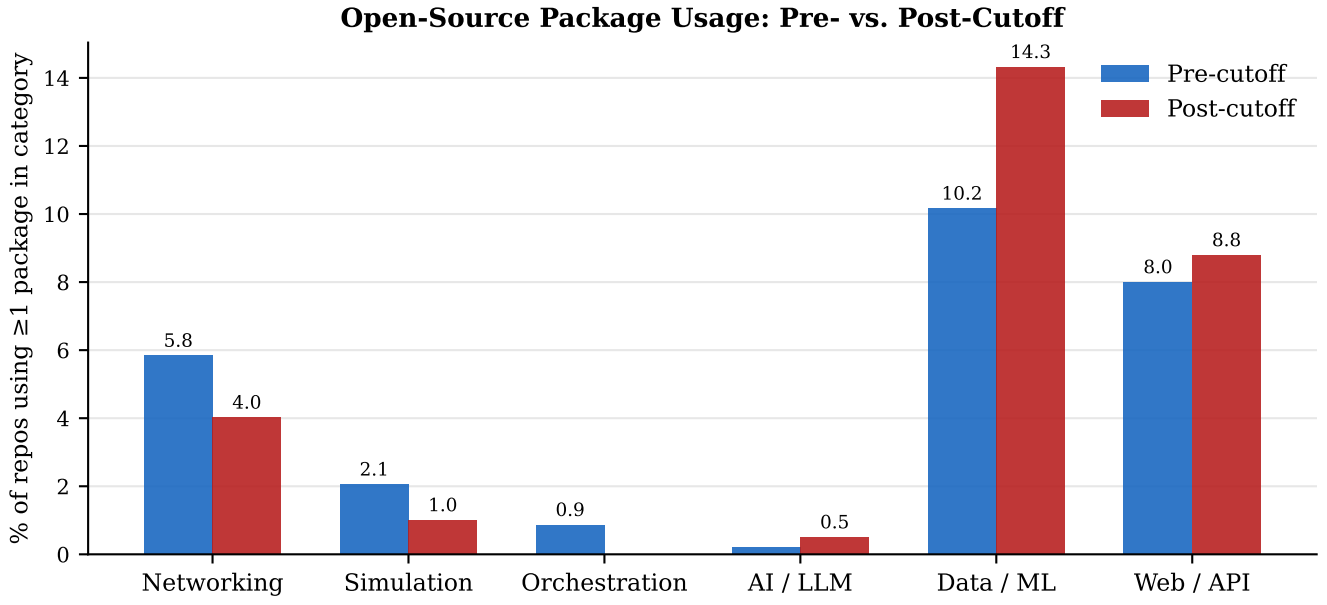

Figure 6: Open-source package category usage in pre- and post-cutoff repositories, measured as the percentage of repositories declaring or importing at least one package from each category. Networking and Simulation usage declines post-cutoff (16.9% → 5.8% and 2.1% → 1.0%, respectively), while Data/ML rises sharply (10.2% → 14.3%) and AI/LLM packages emerge from near-zero (0.3% → 0.5%).

epochs, indicating that large-scale national research infrastructure has not yet diffused into open networking pedagogy.

4.10 Manual Validation of Scoring

We validated our keyword-based axis scores presented in this study by randomly sampling 75 repositories from the larger corpus of 1318 repositories and manually labeling each along the CD and OA axes. One human annotator was assigned to each of the 25 repositories

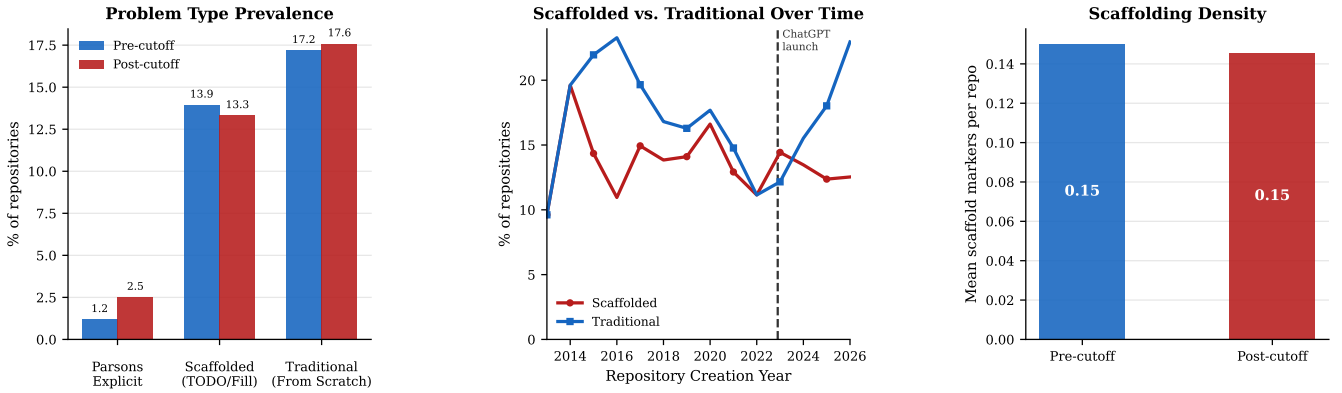

Figure 7: Problem structure prevalence across pre- and post-cutoff repositories. *Left:* Parsons-style explicit problems double post-cutoff while traditional problems remain stable. *Center:* Scaffolded and traditional prevalence over time, with convergence post-2022. *Right:* Mean scaffold marker density is unchanged, indicating broader adoption rather than deeper scaffolding.

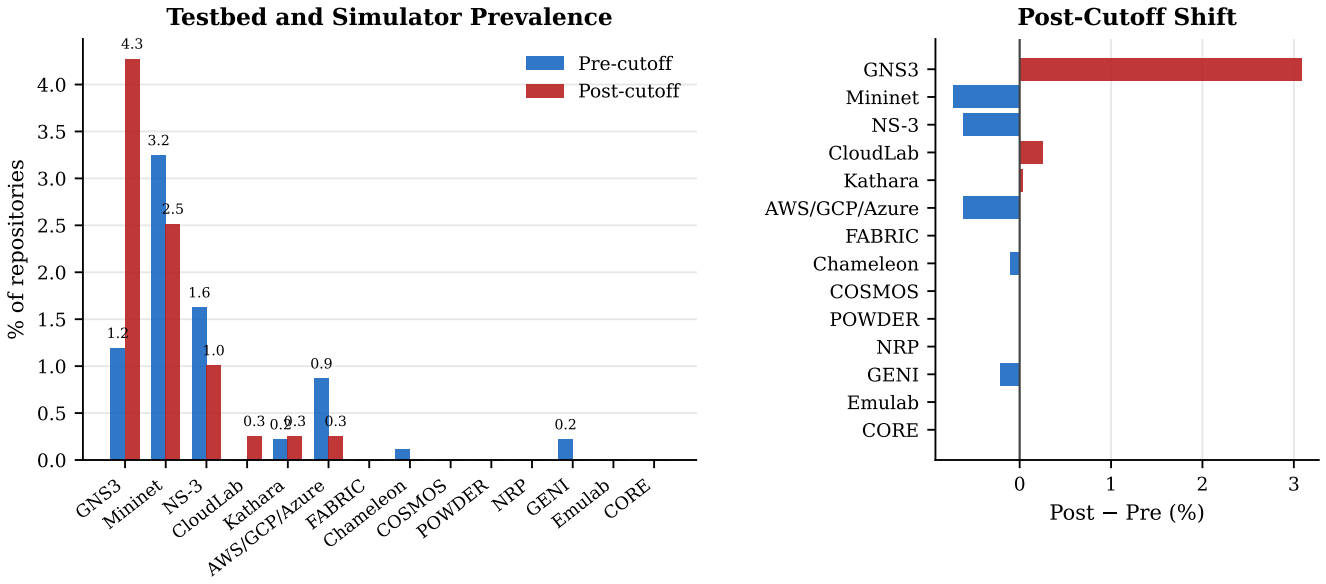

Figure 8: Testbed and simulator prevalence across pre- and post-cutoff repositories. *Left:* GNS3 rises sharply post-cutoff while Mininet and NS-3 decline. *Right:* Post-minus-pre deltas show GNS3 as the dominant gainer, with NSF PAWR testbeds remaining near-zero in both epochs.

– to visit the associated GitHub URL, review the README and representative source code files, and assign binary labels: CD= 1 if the repository teaches mechanism-level networking, CD= 0 otherwise; OA= 1 if the repository uses orchestration or API-layer tooling, OA= 0 otherwise.

We then compared these human annotated labels against the binarized labels that were assigned by our automated scripts. With a threshold set to = 0.5, for CD, we found the accuracy to be 92% and Cohen's k = 0.807. For OA, we found the accuracy to be 92% and Cohen's k = 0.802. We found that when errors do occur, for CD they were mostly false negatives, while they were false positives for OA. Our manual validation indicates overall agreement between our keyword-based data analysis pipeline and the human judgment.

4.11 Repository-Level Case Studies

Fig. 10 presents five spanning repositories that exhibit the strongest pre/post-2022 pedagogical shift. `PROGNOSISTool/main`, a protocol model-learning toolkit, shows the sharpest transition: OA surges post-2022 (Δ_{OA} = +0.844) while CD declines (Δ_{CD} = −0.172), reflecting adoption of higher-level inference APIs over raw protocol construction. `pojntfx/weron`, a WebRTC overlay network, shows a sharp OA spike immediately after November 2022 (Δ_{OA} = +0.523).

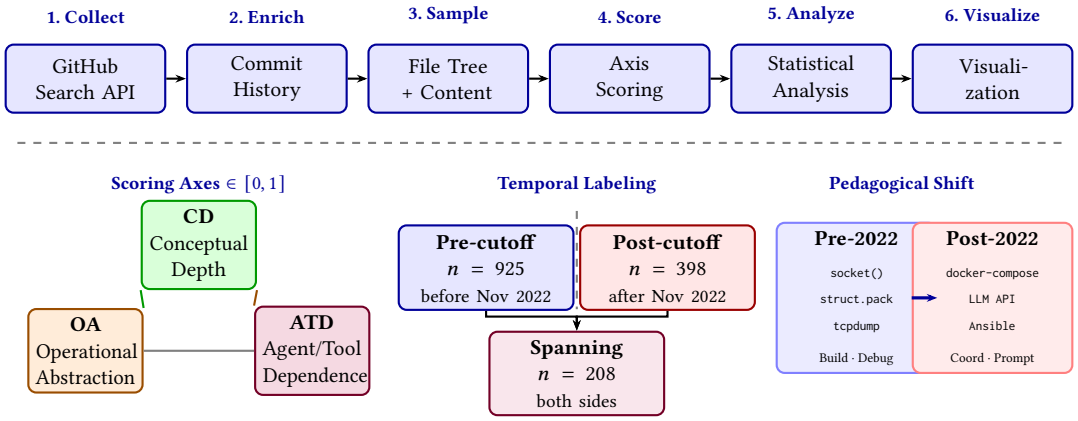

Figure 9: Methodology for manual scoring validation. *Top row:* The six-stage data pipeline from GitHub collection to visualization. *Bottom left:* The three-axis scoring framework (CD, OA, ATD), each normalized to $[0, 1]$. *Bottom center:* Temporal labeling using the November 2022 ChatGPT release as the generative-AI inflection point, yielding pre-cutoff, post-cutoff, and spanning cohorts. *Bottom right:* The pedagogical shift from mechanism-centric (socket-level construction) to orchestration-centric (API/LLM-mediated) repository content.

`Yortw/RSSDP`, a long-running .NET SSDP implementation, shows a sustained CD decline ($\Delta_{CD} = -0.329$) as protocol-level commit activity gives way to configuration and dependency management. `CiscoDevNet/virl2-client`, a client library for Cisco's network simulation platform, transitions toward orchestration-layer patterns post-2022 ($\Delta_{OA} = +0.462$). `adulau/DomainClassifier` shows a modest but consistent shift ($\Delta_{OA} = +0.309$) as the project incorporates higher-level classification APIs. Across all five repositories, CD either declines or holds flat while OA rises, confirming the aggregate spanning-cohort result at the per-repository level.

5 Discussion and Implications

Before we present our discussion, we distinguish between data-driven findings and any interpretive claims used in this study. The results presented in Sec. 4 are purely empirical. Axis score deltas, package prevalence shifts, testbed adoption rates, and problem structure changes are directly measured from the collected corpus. However, the discussion that follows in this section draws on these empirical measurements in tandem with interpretation aimed at connecting infrastructure-level signals (e.g., Docker adoption, GNS3 rise) to pedagogical causes (e.g., AI-era tooling preferences). We flag any interpretive claims explicitly when they are made and thus encourage readers to treat them as hypotheses that warrant future study rather than conclusions established purely by the data. We aim to unify these implications with our pedagogical observations and give recommendations.

5.1 The Pedagogical Shift is Real but Infrastructural

All results in this study consistently support that open networking education is changing. However, this change is at the substrate-level rather than surface-level. In other words, we found that teaching is being done not (yet) *with* AI, but rather *through* infrastructures that AI has made mainstream or readily-accessible due to automation [8]. For example, we found that docker, orchestration tooling, and data-driven workflows are being increasingly adopted, while raw socket programming, which used to be highly prevalent, and simulation-specific packages are declining as teaching tools. Our ATD null result shown in Sec. 4.5 is not a failure of our analysis, but rather the null result is the finding itself. This result supports our claim that the agentification of open networking education is currently a shift in what tools are *assumed* rather than what tools are *taught*.

5.2 Mechanism-Level Understanding is Not Disappearing

Our finding that traditional open-ended problems remaining stable through the pre- and post-cutoff (17.2% → 17.6%) tells us that mechanism-layer terms still hold importance in teaching. This is additionally supported by higher CD scores and CD scores in post-cutoff repositories. We claim that this is because the educators who create public networking repositories post-2022 increasingly place emphasis on items like protocol-level construction. This is *perhaps* a deliberate reaction to the pervasive availability of AI-generated explanations to the students. This likely indicates that open repository ecosystem may be self-correcting to account for AI making surface-level networking content cheap to produce. This is supported by our finding that repositories that accumulate more stars and forks are the ones that whose depth and rigor AI fails to replicate solely.

5.3 Parsons Problems as an AI-Era Pedagogical Signal

Although the doubling of Parsons-style problem prevalence post-cutoff (1.2% → 2.5%) appears numerically small, we note that this is directionally important. This is because Parsons problems are highly suited in curriculum generating aided by AI. They are easy to produce from existing implementations and also reduce cognitive load for novices [9] for both instructors and students.

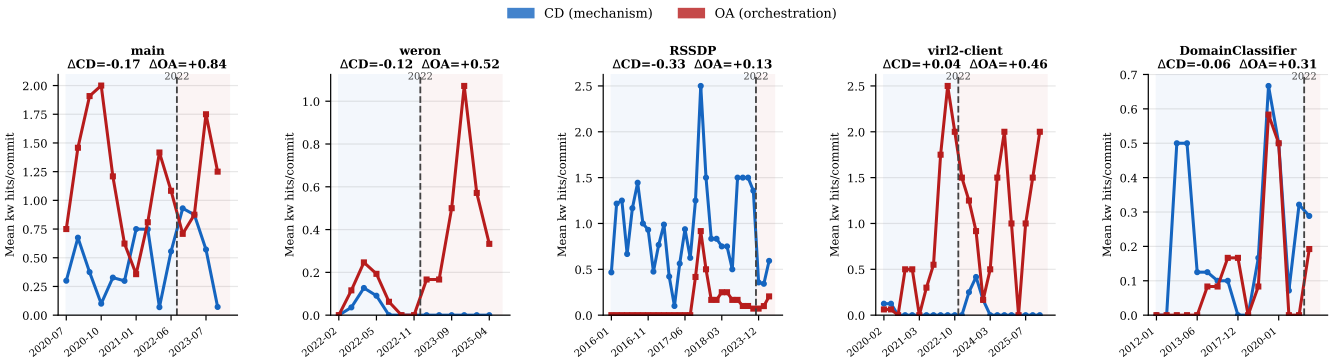

Figure 10: Commit-level CD and OA trends for five spanning repositories. Each panel shows monthly mean keyword hits per commit for mechanism-layer (CD, blue) and orchestration-layer (OA, red) signals. The dashed vertical line marks November 2022. Shaded regions indicate pre- (blue) and post-cutoff (red) periods.

Additionally, they support automated grading. This rise in Parson-style problems likely reflects instructors adopting LLMs to scaffold the existing code [17] into guided exercises for students, rather than asking them to implement from scratch manually. Cognitively, this shift lowers the load and moves the activity-style from *synthesis to sequencing*, as outlined by the Bloom's taxonomy [3], where higher-order synthesis focuses on "creating" and lower-order sequencing focuses on "understanding or applying".

5.4 Implications for Curriculum Design

The results suggest three directions for AI-aware networking curricula.

Preserve mechanistic anchors. Because mechanism-level vocabulary and construction tasks are the primary manners in which systems concepts are best learned, we urge that networking curricula resist the temptation to fully delegate implementation to AI tools. Networking as a field lies on direct mechanistic interaction with packet headers, socket calls, and protocol state machines etc. Hence, networking pedagogy must emphasize direct engagement with packet headers, socket calls, and protocol state machines etc. because these will remain as the irreplaceable foundation of networking, no matter the advancement in AI and the intuition gained from them will remain irreplaceable [15].

Embrace infrastructural fluency. While the rise of Docker, orchestration tooling, and data-driven workflows has co-evolved with AI adoption in pedagogy, we claim that this rise reflects genuine industry alignment [18]. As such, networking curricula should treat containerization, API-mediated configuration, and measurement pipelines as first-class skills that go hand-in-hand with raw socket programming rather than as supplementary or optional teaching components.

Be deliberate about scaffolding. We emphasize that not all scaffolding is equal. Parsons problems and fill-in-the-blank exercises are legitimate pedagogical tools [19] and are not inherently inferior to open-ended problems [19]. However, their increasing adoption raises a practical concern for instructors– while it is easy to generate and scale creating scaffolded exercises, it is also easy to confuse volume for pedagogical rigor. There is a need to make a clear distinction between two types of scaffolding. (1) Scaffolding efforts that progressively build toward independent construction. For example, these are exercises where the scaffold is eventually removed. This type of scaffolding develops the mechanistic intuition. (2) Scaffolding that replaces or permanently removes the need for construction. For example, these are exercises where students never write code from scratch, and hence miss on developing critical nuances in implementation (e.g., in protocol design). This scaffolding type does not develop student's mechanistic understanding and hence may not be ideal.

6 Conclusion

In this paper, we presented a large-scale measurement study of how open networking education is evolving in the generative-AI era. To this end, we mined a corpus of 1318 GitHub repositories spanning 2009–2026 – with a cutoff period set at Nov. 2022 to separate pre- and post-epochs to account for broad AI adoption. We introduced and scored the repositories along three axes, i.e., **Conceptual Depth (CD)**, **Operational Abstraction (OA)**, and **Agent/Tool Dependence (ATD)**, and analyzed 208 spanning repositories longitudinally. We show that the pedagogical shift due to AI is outwardly measurable, and more importantly, infrastructural. Our within-repository commit analysis showed that mechanism-centric signals are declining ($\Delta_{CD} = -0.085$, $p < 0.001$) and orchestration-layer signals are rising ($\Delta_{OA} = +0.063$, $p = 0.008$) after the Nov. 2022 inflection point. Other important findings include – Docker adoption, Data/ML package usage, GNS3 prevalence, and Parsons-style problem prevalence all rising since the start of the post-cutoff, while explicit AI framework instrumentation surprisingly remains negligible. In the future, we will extend our analysis to non-English repositories, validate the axis scores against human annotation. Finally, we aim to track whether the infrastructural shift eventually surfaces as explicit AI instrumentation as the networking field as a whole continues to mature.

References

[1] Ilya Baldin, Anita Nikolich, James Griffioen, Indermohan Inder S. Monga, Kuang-Ching Wang, Tom Lehman, and Paul Ruth. 2019. FABRIC: A National-Scale Programmable Experimental Network Infrastructure. *IEEE Internet Computing* 23, 6 (2019), 38–47. doi:10.1109/MIC.2019.2958545

[2] John D Blischak, Emily R Davenport, and Greg Wilson. 2016. A quick introduction to version control with Git and GitHub. *PLoS computational biology* 12, 1 (2016), e1004668.

[3] Benjamin S Bloom et al. 1956. *Taxonomy of Educational Objectives: The Classification of Educational Goals. Handbook 1: Cognitive Domain.* Longman New York, New York.

[4] Rishi Bommasani et al. 2021. On the Opportunities and Risks of Foundation Models. *CoRR* abs/2108.07258 (2021). arXiv:2108.07258 https://arxiv.org/abs/2108.07258

[5] Joe Breen et al. 2020. POWDER: Platform for Open Wireless Data-driven Experimental Research. In *Proceedings of the 14th International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization* (London, United Kingdom) *(WiNTECH '20)*. Association for Computing Machinery, New York, NY, USA, 17–24. doi:10.1145/3411276.3412204

[6] Tom Brown et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems* 33 (2020).

[7] Mark Chen et al. 2021. Evaluating large language models trained on code. In *arXiv preprint arXiv:2107.03374.*

[8] Arghavan Moradi Dakhel et al. 2023. GitHub Copilot AI pair programmer: Asset or liability? *Journal of Systems and Software* 203.

[9] Paul Denny, Andrew Luxton-Reilly, and Beth Simon. 2008. Evaluating a new exam question: Parsons problems. In *Proceedings of the Fourth international Workshop on Computing Education Research.* 113–124.

[10] GitHub. 2023. GitHub Copilot: AI pair programmer. https://github.com/features/copilot.

[11] Georgios Gousios, Bogdan Vasilescu, Alexander Serebrenik, and Andy Zaidman. 2014. Lean GHTorrent: GitHub data on demand. In *Proceedings of the 11th Working Conference on Mining Software Repositories.*

[12] Nikhil Handigol et al. 2012. Reproducible network experiments using container-based emulation. In *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies.*

[13] Eirini Kalliamvakou et al. 2014. The promises and perils of mining GitHub. In *Proceedings of the 11th Working Conference on Mining Software Repositories.*

[14] Thomas Kluyver et al. 2016. Jupyter development team. *Jupyter notebooks-a publishing format for reproducible computational workflows. In Loizides F, Scmidt B (eds.), Positioning and Power in Academic Publishing: Players, Agents and Agendas* (2016), 87–90.

[15] James F Kurose and Keith W Ross. 2016. *Computer Networking: A Top-Down Approach* (7 ed.). Pearson.

[16] Bob Lantz, Brandon Heller, and Nick McKeown. 2010. A network in a laptop: Rapid prototyping for software-defined networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks.*

[17] Juho Leinonen et al. 2023. Comparing Code Explanations Created by Students and Large Language Models. In *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1* (Turku, Finland) *(ITiCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 124–130. doi:10.1145/3587102.3588785

[18] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. 2008. OpenFlow: enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.* 38, 2 (March 2008), 69–74. doi:10.1145/1355734.1355746

[19] Dale Parsons and Patricia Haden. 2006. Parson's programming puzzles: a fun and effective learning tool to introduce software engineering. *Conferences in Research and Practice in Information Technology* 52 (2006).

[20] Dipankar Raychaudhuri et al. 2020. Challenge: COSMOS: A city-scale programmable testbed for experimentation with advanced wireless. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking* (London, United Kingdom) *(MobiCom '20)*. Association for Computing Machinery, New York, NY, USA, Article 14, 13 pages. doi:10.1145/3372224.3380891

[21] Derek Weitzel et al. 2025. The National Research Platform: Stretched, Multi-Tenant, Scientific Kubernetes Cluster. In *Practice and Experience in Advanced Research Computing 2025: The Power of Collaboration (PEARC '25)*. Association for Computing Machinery, New York, NY, USA, Article 69, 5 pages. doi:10.1145/3708035.3736060