NBGuru: Generating Explorable Data Science Flowcharts to Facilitate Asynchronous Communication in Interdisciplinary Data Science Teams

Panayu Keelawat
University of California San Diego
pkeelawa@ucsd.edu

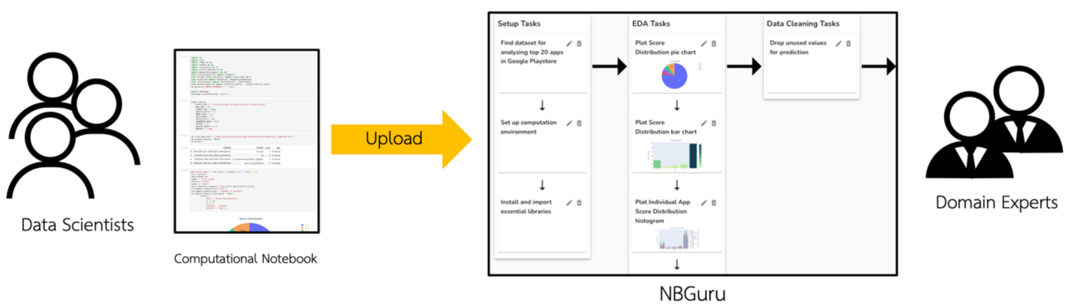

Figure 1: NBGuru extracts information from computational notebooks, and displays it in a multi-stage flowchart format. Domain experts, who work closely with data scientists, can grasp more structured information from NBGuru when they synchronize offline.

ABSTRACT

Data scientists typically work with domain experts in a Data Science (DS) project, resulting in knowledge gaps between roles. Communication holds an immense and difficult workload due to the complicated content, limited meeting time, vast audience backgrounds, etc. Thus, it is almost impossible to build a common ground within the team. Taking a step back, flowcharts and program descriptions have shown to help programmers learn algorithms. However, drawing a flowchart or writing a description takes time and effort. The novel AI-powered search engines can generate elaborate grounded responses with citations. It is then possible to generate flowcharts with text descriptions from code. Therefore, we studied 92 DS flowcharts and 173 code descriptions from top-voted Kaggle notebooks. We propose NBGuru, a flowchart-based communication tool. Users can explore computation steps asynchronously with generated texts and citations. Furthermore, we also discuss the possibility of AI in other collaborative roles.

CCS CONCEPTS

• **Human-centered computing**; • **Interaction design**; • **Software and its engineering**; • **Software creation and management**;

KEYWORDS

data science, interdisciplinary, collaboration, asynchronous communication, computational notebooks, flowchart, artificial intelligence, large language model, on-the-job training

ACM Reference Format:

Panayu Keelawat. 2023. NBGuru: Generating Explorable Data Science Flowcharts to Facilitate Asynchronous Communication in Interdisciplinary Data Science Teams. In *Computer Supported Cooperative Work and Social Computing (CSCW '23 Companion), October 14–18, 2023, Minneapolis, MN, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3584931.3607020

CSCW '23 Companion, October 14–18, 2023, Minneapolis, MN, USA

ACM ISBN 979-8-4007-0129-0/23/10.
https://doi.org/10.1145/3584931.3607020

1 INTRODUCTION

Data Science (DS) is a relatively new field that extracts and extrapolates data insights [24, 48]. Based on its data-driven approach, it can solve many high-impact problems in various domains. A data scientist must know about Computer Science (CS), Mathematics, Artificial Intelligence (AI) and Machine Learning (ML) to be able to make any predictions from data [3]. However, understanding the domain problem is as important as knowing how to analyze them, but most data scientists are lacking the knowledge [33]. Therefore, data scientists typically work as a team with domain experts who can pinpoint important problems in the discipline [45].

Due to knowledge gaps in the DS team, data scientists bear a significant burden of communication [44, 45]. Because data scientists analyze and make predictions on data, they are expected to explain how they approach the problem to their coworkers to validate the results or make decisions [43]. In addition to domain experts, the audience sometimes includes business specialists, software engineers, and other relevant roles if they are in a large organization. Hence, it is not unusual for data scientists to provide foundational AI/ML education to the team [26].

Besides the workload, communicating the work to other team members can also pose significant challenges. Advanced ML algorithms are abstract and esoteric [26]. Finding a time to meet as a team can also be difficult due to each person's other priorities [9]. Even if the team can eventually find time to meet, the meeting duration is generally not sufficient to cover all topics, let alone complex but essential subjects. As a result, data science teams are often not on the same page after each meeting. Combining these challenges with the research nature of constant direction pivoting, communicating, and creating a common ground on what the current problem is and how to solve it can be exceeding tough [33].

A lack of common ground is a serious problem that can lead to skepticism and trustworthiness of other members' output [21, 43]. In fact, this issue is also common in the Software Visualization community. Taking a step back, perhaps the most basic representation of code is a flowchart widely used to describe any program [39]. With its simplicity, it is introduced in almost all CS introductory courses [15]. In Software Engineering (SE), the Kanban board is a popular tool for tracking the work progress of a team [1]. Cards on the board have space to elaborate details of the task. In regards to DS, there are some existing visualization studies [18, 59], but we believe it is also crucial to make the tool less disruptive to their typical work practices; since, with a steep learning curve, the tool may end up not being used [33].

The recent breakthrough in Natural Language Processing (NLP), particularly in Large Language Models (LLMs) [4, 10, 41], exhibits promising AI copilot applications including web search [36], coding [16], and many others. The HCI community has been studying these models as well [6, 59, 62]. Since they can generate texts automatically, these models may reduce the communication workload from data scientists and make communication more effective with automatic software visualization even in situations where synchronous team meetings are not feasible. There have been some works that explore automatic documentation from code [54, 62], but they were not designed based on inter-professional collaboration.

We then propose NBGuru, a novel flowchart-based visualizer that makes asynchronous communication in DS teams more productive. The tool, powered by Bing Chat, visualizes abstract DS work from computational notebooks into a more concrete task representation. It demonstrates DS work as a flowchart, so each team member can understand more about what the code does despite the involved DS materials. This paper presents a comprehensive literature review on the DS communication problem, investigates existing DS flowcharts and code descriptions on Kaggle, and introduces a feasible LLM solution with NBGuru. Our AI serves as a guru who synchronizes the team's project understanding. We further discuss the potential of LLM as a collaborator, trainer, or other roles in CSCW.

2 LITERATURE REVIEW

2.1 Interdisciplinary Collaboration in Data Science Teams

Data scientists are in high demand in nearly all fields, while the supply side remains limited. Most of them come from computational backgrounds, lacking the domain knowledge they are trying to solve [12]. Thus, a data science team usually consists of data scientists and domain experts as core team members, although some teams may include other roles like software engineers or business analysts [42, 45, 61]. Undoubtfully, it is challenging to create a successful data science team with mixed expertise, particularly when the members have low overlapping knowledge backgrounds [26, 52]. Research shows that despite data scientists' efforts to educate their teammates at the beginning of a new project, people still struggle to comprehend the work, and that causes communication frictions [33] and misinterpretation [38, 45, 58]. Apart from communication, knowledge gaps can cause problems like shared information bias [50], task delegation [13], etc. Limited meeting time due to other priorities makes collaboration more difficult [9]. Finally, DS is a research work, which often requires changing topics as part of the process [11], further adding even more complexity to the problem.

Establishing a team's common ground is key to a successful scientific collaboration [23]. A good sign of having common ground is when team members share enough language to discuss within the team [17]. However, creating a mutual understanding is challenged by the aforementioned issues. According to several studies, domain experts find it difficult to trust and verify data scientists' results if the analyses are not comprehensible [33, 56]. A possible solution is to have efficient communication that ensures everyone is on the same page. A study offers evidence that having a person who can translate language from one field to another helps build a common ground in the team [22]. Nevertheless, finding a guru that can translate vocabulary across these highly in-demand skills is nearly impossible. Therefore, this paper explores ways to employ AI web copilot to help DS teams build common ground asynchronously.

2.2 Communicating Software Knowledge

Given the abstract nature of software functionalities, utilization of visualization tools become crucial. Representations like flowcharts have been widely used in CS 101 to show how algorithm works [39]. Kanban boards make code more tangible with cards [1, 49]. Code visualization tools also have been proven to help debug code [7], teach programming [18, 35], etc. Aside from that, code documentation assists software understanding and maintenance as well [29]. Software engineering teams document their software for knowledge sharing. Similarly, students learn programming through algorithm descriptions in academic settings [14].

Regarding DS, the code typically involves complicated ML algorithms and it makes understanding code even more challenging [47]. Computational notebooks are commonly used by data scientists to experiment, document, and visualize their work [20, 27, 55], but making them sharable to others is not trivial [46]. Thus, knowledge communication tools [26], e.g., advanced algorithm visualization [31, 37, 40, 57], labeling facilitator [2], etc. enhance DS collaboration. MIDST [8], a system that is closely relevant to this paper, conveys task delegation and tracking. In the context of multidisciplinary teams, each role prefers a work style that fits well with their professional practices [25]. Since data scientists are familiar with computational notebooks [20, 27, 53], NBGuru is designed to have a shallow learning curve with the flowchart structure generated from notebooks.

2.3 Recent Advances in Natural Language Processing

NLP has recently had a revolutionary breakthrough in LLMs [4, 10, 51]. They have become generic problem solvers for almost any language-related problems, including programming languages [16, 30]. Notably, ChatGPT astounded the world with its remarkable capabilities [41]. Microsoft was one of the first companies that released LLM-integrated products; Bing Chat was included [36].

The HCI community has been studying NLP technologies regarding their impact on humans and designs. A study shows that generative AI could document code in computational notebooks, making DS work more maintainable [34, 54]. Furthermore, it is also possible to generate presentation slides [62], code explanations and demonstrations [19], or even prompts themselves [32]. Lee et al. scrutinized collaborative document authoring with GPT-3 and collected datasets from the writing sessions [28]. Works such as Crosscast [60] and CrossData [6] have explored the intents behind audio podcasts and data descriptions to infer relationships and introduce novel interactions with natural language. We explored Bing Chat abilities and used them to generate code descriptions with web references targeting non-technical readers in interdisciplinary DS teams.

3 FLOWCHARTS AND CODE DESCRIPTIONS IN DATA SCIENCE

What do DS flowcharts look like? How do people describe code in DS? These questions are critical for designing our system. Complementary to the literature review, we studied flowcharts and code descriptions from existing online resources to inform our design. We collected 102 flowcharts from Google and Bing image searches in March 2023. To enhance sampling variation, our search started from broad queries like "*data science flowchart*" and gradually narrowed down to more specific keywords like "*data science feature engineering tasks flowchart*". We filtered out unrelated or duplicate images, and obtained 92 unique data science flowcharts. For code descriptions, we collected 173 descriptions on April 4, 2023, from top-voted data analysis notebooks on Kaggle, a data science community. In our study, we define a code description of a code cell as any artifact(s) produced by the notebook author that helps readers understand the code block better. We conducted open coding on the collected flowcharts and descriptions, and iterated on categorization until saturation. Studying the corpus revealed several outstanding characteristics of flowcharts and code descriptions in DS.

3.1 Flowcharts

Most of the samples describe DS work in high-level abstraction, and they share a lot of common processes. For instance, Exploratory Data Analysis (EDA), data cleaning, classification, etc. High-level flowcharts tend to be simplified versions of the programming flowchart standard [5]. On the other hand, low-level flowcharts are likely to include other symbols like decision, input/output, etc. Additionally, processes are often grouped together to demonstrate higher-level concepts. Flowchart orientations involve horizontal, vertical, cyclic, and mixed. Symbol texts are mainly compact phrases/keywords rather than elaborate sentences.

3.2 Code Descriptions

Code descriptions come in multiple forms, i.e., markdown cells, comments, console outputs, and plots. In most cases, authors use mixed mediums to communicate their thoughts. For low-level ideas, commenting in the code cells and printing standard outputs are prevalently seen. High-level ideas are typically conveyed through markdown cells. The key to successful code documentation boils down to *Why? How?* and *Outcome*. The notebook authors may have different writing styles, but if they can clearly convey *Why the computation is necessary?*, *How to perform the operation?*, and *What is the outcome of that computation?*, then they will have a high chance of getting accolades from their colleagues.

4 NBGURU

NBGuru is a system designed for technical asynchronous communications in DS with flowcharts that are auto generated from notebooks. Based on the literature review and findings from the corpus, NBGuru extracts information from code, and uses them to create flowcharts with elaborate text descriptions.

4.1 User Interface

NBGuru has two pages, homepage and flowchart page. The homepage shows all uploaded notebooks. Here, data scientists can upload a new notebook, and the AI pipeline will be automatically triggered to generate a new flowchart and code descriptions. Data scientists have the ability to edit the content before or after publishing the diagram. The results are stored in Firebase. When the user clicks any of the notebooks in the list, they will be redirected to the flowchart page. The UI design is shown in Figure 2. The execution processes are grouped into eight common stages derived from the corpus, i.e., *Setup*, *EDA*, *Data Cleaning*, *Feature Engineering*, *Model*, *Evaluation*, *Discussion*, and *Miscellaneous Tasks*. The grouping will provide a higher level of abstraction to team members who are not familiar with the area of DS. We design NBGuru to have a description pop-up page for each execution process. Each page will have 1) text elaboration of the associated process with references 2) corresponding code from the notebook. Users can independently learn more about the process by reading the text, checking out citations, or glancing at the code. Users can also search code with descriptions.

4.2 System Design

When we planned for development, the major constraint was the newly released Bing Chat. There was no official Bing Chat API that we could use. The workaround for the prototype was to manually

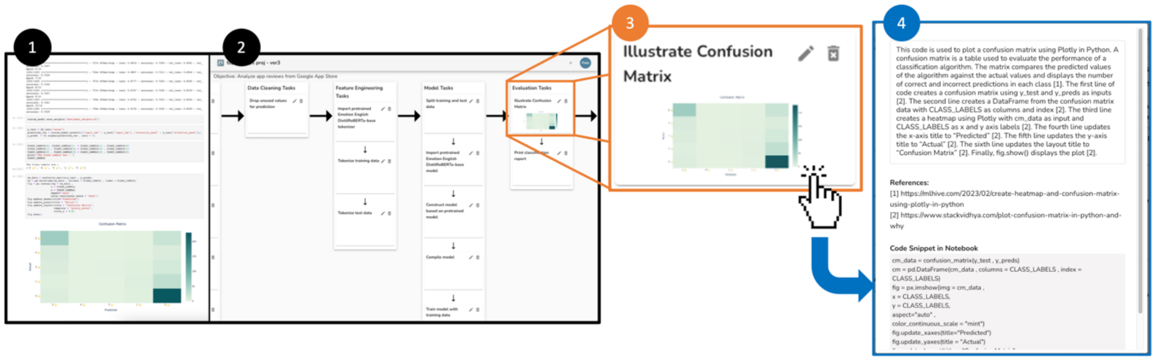

Figure 2: Users can view NBGuru with or without collapsible original notebook panel (1). In the main flowchart panel (2), users can see all of the extracted processes with arrows indicating the execution sequence of the program. Processes are grouped into predefined stages derived from our findings. When users click a process (3), an elaborate process description (4) will be popped up.

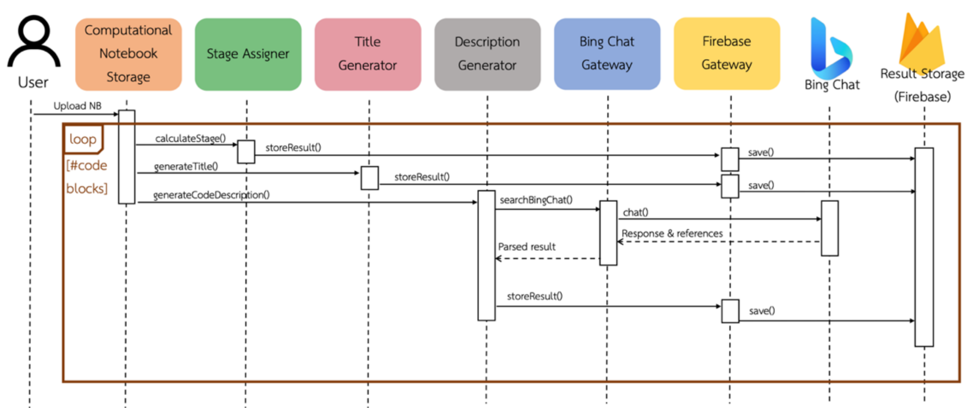

Figure 3: Sequence diagram of flowchart generation when uploading a new computational notebook to NBGuru.

start a Bing Chat session on Microsoft Edge and then copy-paste the cookies to our Bing Chat Gateway before starting the NBGuru server. We developed the entire NBGuru stack with ReactJS. There are three main flows that NBGuru supports: upload a notebook, edit content, and view the flowchart. For the first flow, as shown in Figure 3, Description Generator relies on Bing Chat responses. Bing Chat Gateway decorates the prompt with instructions such as "*Explain this code with the most details and web references:*" before emitting the request. The gateway also cuts out unrelated parts of the chat responses, e.g., "*Sure! Here's an explanation of the code:*", "*I hope this helps!*", etc. We developed our own keyword-matching engines for Stage Assigner and Title Generator based on predefined mappings. We did not use LLM for Title Generator because LLM mostly gives answers that are too long for titles, even if we indicated in the prompt that we needed short responses. Results from Stage Assigner, Title Generator and Description Generator are stored in Firebase. For the second flow, the user will be redirected to the edit page right after all of the results are stored in Firebase. They can overwrite AI-generated content with their own words, or rearrange execution order before publishing the content. They can edit the flowchart after publishing as well by clicking Edit in the flowchart view. When the user views the flowchart, NBGuru renders the flowchart based on the data on Firebase. The original notebooks are exported to HTML and are rendered as iframes when the user expands the side panel. The resulting process descriptions illustrate Why? How? and Outcome from Bing responses and code snippets.

5 LIMITATIONS AND FUTURE WORK

The main limitation is that the prototype has not yet been validated. Also, prompt engineering is not thoroughly explored. Despite that, we have proven that it is possible to compose flowcharts and code documentation with web references using web LLM copilot. As NBGuru is a medium-fidelity prototype, it is theoretically feasible to engineer a higher-fidelity prototype to obtain idea validation. NBGuru highlights the viability of LLM-enhanced web search to advance CSCW in the context of asynchronous collaboration in teams with low-overlapping skills. LLM acts like a guru who bridges people from different backgrounds together, which helps build common ground within the team. It is possible to explore more of how to create a team common ground with LLM. For instance, studying how LLM can help teammates share their work with less friction, e.g., cleaning up messy notebooks, rewriting hacky code, etc., would be interesting. We think that studying ways to use LLM as an on-the-job training instructor will be a fascinating direction as well. From a collaboration standpoint, the exploration of methods to enhance bidirectional communication between data scientists and domain experts remains unaddressed. For example, streamlining high-level requirements to code is an area that can be examined.

6 CONCLUSION

Data scientists hold an immense workload of communicating their work to others in the team to build common ground. Besides, diverse audience backgrounds, esoteric ML content, and limited meeting time are the main contributors to making communication even more arduous. Representing code as flowcharts or text descriptions would be helpful, yet they require non-trivial effort to create. The recent LLM innovation shows promising generation capabilities, so we studied DS flowcharts and code descriptions. Then, we propose NBGuru as a visualization tool that fits with DS teams' asynchronous, iterative, and interdisciplinary nature. Materializing abstract contents with trained NLP models makes DS more comprehensible to other team members and reduces data scientists' communication workload. Code snippets are wrapped with meaningful text explanations; thus, non-technical audiences will not be inundated with cryptic information unless they purposely view the task to deep dive into the issue themselves. NBGuru demonstrates the potential of using LLM to build common ground. The work paves the way for future research in investigating LLM's potential roles in CSCW.

ACKNOWLEDGMENTS

I would like to thank Andrew Head, Sam Lau, and Philip Guo for their valuable feedback. I also appreciate Thipok Thammakulangkul's inputs on the topic as a data scientist.

REFERENCES

[1] Muhammad Ovais Ahmad, Jouni Markkula, and Markku Oivo. 2013. Kanban in software development: A systematic literature review. In *Proceedings of the 2013 39th Euromicro Conference on Software Engineering and Advanced Applications*, IEEE 9-16, DOI: 10.1109/SEAA.2013.28.

[2] Zahra Ashktorab, Michael Desmond, Josh Andres, Michael Muller, Narendra Nath Joshi, Michelle Brachman, Aabhas Sharma, Kristina Brimijoin, Qian Pan, Christine T. Wolf, Evelyn Duesterwald, Casey Dugan, Werner Geyer, and Darrell Reimer. 2021. AI-Assisted Human Labeling: Batching for Efficiency without Overreliance. *Proc. ACM Hum.-Comput. Interact.* 5, CSCW1, Article 89 (April 2021), 27 pages. https://doi.org/10.1145/3449163.

[3] Sumon Biswas, Mohammad Wardat, and Hridesh Rajan. 2022. The art and practice of data science pipelines. In *Proceedings of the 44th International Conference on Software Engineering* (2022). DOI: https://doi.org/10.1145/3510003.3510057.

[4] T. Brown *et al.* 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33, 1877-1901.

[5] Ned Chapin. 1970. Flowcharting with the ANSI standard: A tutorial. *ACM Computing Surveys (CSUR)* 2, 2, 119-146.

[6] Zhutian Chen and Haijun Xia. 2022. CrossData: Leveraging Text-Data Connections for Authoring Data Documents. In *Proceedings of CHI Conference on Human Factors in Computing Systems*, New Orleans, LA, USA, 2022. DOI: https://doi.org/10.1145/3491102.3517485.

[7] Benjamin Cosman, Madeline Endres, Georgios Sakkas, Leon Medvinsky, Yao-Yuan Yang, Ranjit Jhala, Kamalika Chaudhuri, and Westley Weimer. 2020. PABLO: Helping Novices Debug Python Code Through Data-Driven Fault Localization. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20)*. ACM, New York, NY, USA, 1047–1053. https://doi.org/10.1145/3328778.3366860.

[8] Kevin Crowston, Jeffery Saltz, Niraj Sitaula, and Yatish Hegde. 2021. Evaluating MIDST, A System to Support Stigmergic Team Coordination. *Proc. ACM Hum.-Comput. Interact.* 5, CSCW1, Article 36 (April 2021), 24 pages. https://doi.org/10.1145/3449110.

[9] Yuri Demchenko, Luca Comminiello, and Gianluca Reali. 2019. Designing Customisable Data Science Curriculum Using Ontology for Data Science Competences and Body of Knowledge. In *Proceedings of the 2019 International Conference on Big Data and Education (ICBDE'19)*. ACM, New York, NY, USA, 124–128. https://doi.org/10.1145/3322134.3322143.

[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv: 1810.04805*, 2018.

[11] Vasant Dhar. 2013. Data science and prediction. *Commun. ACM* 56, 12 (December 2013), 64–73. https://doi.org/10.1145/2500499.

[12] Usama Fayyad and Hamit Hamutcu. 2021. How Can We Train Data Scientists When We Can't Agree on Who They Are? *Harvard Data Science Review* 3, 1 (2021).

[13] Heather Ford. 2014. Big Data and Small: Collaborations between ethnographers and data scientists. *Big Data & Society* 1.

[14] Ursula Fuller, Colin G. Johnson, Tuukka Ahoniemi, Diana Cukierman, Isidoro Hernán-Losada, Jana Jackova, Essi Lahtinen, Tracy L. Lewis, Donna McGee Thompson, Charles Riedesel, and Errol Thompson. 2007. Developing a computer science-specific learning taxonomy. *SIGCSE Bull.* 39, 4 (December 2007), 152–170. https://doi.org/10.1145/1345375.1345438.

[15] Daniela Giordano and Francesco Maiorana. 2015. Teaching algorithms: Visual language vs flowchart vs textual language. In *Proceedings of the 2015 IEEE Global Engineering Education Conference (EDUCON)*, IEEE, 499-504, DOI: 10.1109/EDUCON.2015.7096016.

[16] GitHub. 2023. Copilot. Retrieved from https://github.com/features/copilot.

[17] Michael E. Gorman. 2008. Scientific and technological expertise. *Journal of psychology of science and technology* 1.

[18] Philip J. Guo. 2013. Online python tutor: embeddable web-based program visualization for cs education. In *Proceeding of the 44th ACM technical symposium on Computer science education (SIGCSE '13)*. ACM, New York, NY, USA, 579–584. https://doi.org/10.1145/2445196.2445368.

[19] Andrew Head, Codanda Appachu, Marti A. Hearst, and Björn Hartmann. 2015. Tutorons: Generating context-relevant, on-demand explanations and demonstrations of online code. In *Proceedings of the 2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 3-12, DOI: 10.1109/VLHCC.2015.7356972.

[20] Andrew Head, Fred Hohman, Titus Barik, Steven M. Drucker, and Robert DeLine. 2019. Managing Messes in Computational Notebooks. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Paper 270, 1–12. https://doi.org/10.1145/3290605.3300500.

[21] Sungsoo Ray Hong, Jessica Hullman, and Enrico Bertini. 2020. Human Factors in Model Interpretability: Industry Practices, Challenges, and Needs. In *Proc. ACM Hum.-Comput. Interact.* 4, CSCW1, Article 68 (May 2020), 26 pages. https://doi.org/10.1145/3392878.

[22] Youyang Hou and Dakuo Wang. 2017. Hacking with NPOs: Collaborative Analytics and Broker Roles in Civic Data Hackathons. *Proc. ACM Hum.-Comput. Interact.* 1, CSCW, Article 53 (November 2017), 16 pages. https://doi.org/10.1145/3134688.

[23] Xinlan Emily Hu, Rebecca Hinds, Melissa Valentine, and Michael S. Bernstein. 2022. A "Distance Matters" Paradox: Facilitating Intra-Team Collaboration Can Harm Inter-Team Collaboration. *Proc. ACM Hum.-Comput. Interact.* 6, CSCW1, Article 48 (April 2022), 36 pages. https://doi.org/10.1145/3512895.

[24] Michael I. Jordan and Tom M. Mitchell. 2015. Machine learning: Trends, perspectives, and prospects. *Science* 349, 6245 (2015), 255-260. DOI: https://doi.org/10.1126/science.aaa8415.

[25] Ju Yeon Jung, Tom Steinberger, John L. King, and Mark S. Ackerman. 2022. How Domain Experts Work with Data: Situating Data Science in the Practices and Settings of Craftwork. *Proc. ACM Hum.-Comput. Interact.* 6, CSCW1, Article 58 (April 2022), 29 pages. https://doi.org/10.1145/3512905.

[26] Sean Kross and Philip J. Guo. 2019. Practitioners Teaching Data Science in Industry and Academia: Expectations, Workflows, and Challenges. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Paper 263, 1–14. https://doi.org/10.1145/3290605.3300493.

[27] Sam Lau, Ian Drosos, Julia M. Markel, and Philip J. Guo. 2020. The Design Space of Computational Notebooks: An Analysis of 60 Systems in Academia and Industry. In *Proceedings of the 2020 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, 1-11. DOI: 10.1109/VL/HCC50065.2020.9127201.

[28] Mina Lee, Percy Liang, and Qian Yang. 2022. CoAuthor: Designing a Human-AI Collaborative Writing Dataset for Exploring Language Model Capabilities. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. ACM, New York, NY, USA, Article 388, 1–19. https://doi.org/10.1145/3491102.3502030.

[29] Timothy C. Lethbridge, Janice Singer, and Andrew Forward. 2003. How software engineers use documentation: The state of the practice. *IEEE software*, 20, 35-39.

[30] Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. 2022. A survey of transformers. *AI Open* 3, 111-132. DOI: https://doi.org/10.1016/j.aiopen.2022.10.001.

[31] Dongyu Liu, Weiwei Cui, Kai Jin, Yuxiao Guo, and Huamin Qu. 2018. DeepTracker: Visualizing the Training Process of Convolutional Neural Networks. *ACM Trans. Intell. Syst. Technol.* 10, 1, Article 6 (January 2019), 25 pages. https://doi.org/10.1145/3200489.

[32] Michael Xieyang Liu, Advait Sarkar, Carina Negreanu, Benjamin Zorn, Jack Williams, Neil Toronto, and Andrew D. Gordon. 2023. "What It Wants Me To Say": Bridging the Abstraction Gap Between End-User Programmers and Code-Generating Large Language Models. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 598, 1–31. https://doi.org/10.1145/3544548.3580817.

[33] Yaoli Mao, Dakuo Wang, Michael Muller, Kush R. Varshney, Ioana Baldini, Casey Dugan, and Aleksandra Mojsilović. 2019. How Data ScientistsWork Together With Domain Experts in Scientific Collaborations: To Find The Right Answer Or To Ask The Right Question?. In *Proceedings of the ACM on Human-Computer Interaction* 3, GROUP (2019), 1-23. DOI: https://doi.org/10.1145/3361118.

[34] Andrew M Mcnutt, Chenglong Wang, Robert A Deline, and Steven M. Drucker. 2023. On the Design of AI-powered Code Assistants for Notebooks. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 434, 1–16. https://doi.org/10.1145/3544548.3580940.

[35] Leonel Merino, Mohammad Ghafari, Craig Anslow, and Oscar Nierstrasz. 2018. A systematic literature review of software visualization evaluation. *Journal of Systems and Software* 144, 165-180. DOI: https://doi.org/10.1016/j.jss.2018.06.027.

[36] Microsoft. 2023. Bing Chat. Retrieved from https://www.bing.com/chat.

[37] Yao Ming, Huamin Qu, and Enrico Bertini. 2019. RuleMatrix: Visualizing and Understanding Classifiers with Rules. *IEEE Transactions on Visualization and Computer Graphics* 25, 1, 342-352. DOI: 10.1109/TVCG.2018.2864812.

[38] Regina Moirano, Marisa Analía Sánchez, and Libor Štěpánek. 2020. Creative interdisciplinary collaboration: A systematic literature review. *Thinking Skills and Creativity* 35, 100626.

[39] Ike Nassi and Ben Shneiderman. 1973. Flowchart techniques for structured programming. *SIGPLAN Not.* 8, 8 (August 1973), 12–26. https://doi.org/10.1145/953349.953350.

[40] Harsha Nori, Samuel Jenkins, Paul Koch, and Rich Caruana. 2019. Interpretml: A unified framework for machine learning interpretability. arXiv:1909.09223. Retrieved from https://arxiv.org/abs/1909.09223.

[41] OpenAI. 2023. ChatGPT. Retrieved from https://openai.com/blog/chatgpt.

[42] Soya Park, April Yi Wang, Ban Kawas, Q. Vera Liao, David Piorkowski, and Marina Danilevsky. 2021. Facilitating Knowledge Sharing from Domain Experts to Data Scientists for Building NLP Models. In *26th International Conference on Intelligent User Interfaces (IUI '21)*. ACM, New York, NY, USA, 585–596. https://doi.org/10.1145/3397481.3450637.

[43] Samir Passi and Steven J. Jackson. 2018. Trust in Data Science: Collaboration, Translation, and Accountability in Corporate Data Science Projects. In *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 1-28. DOI: https://doi.org/10.1145/3274405.

[44] D. J. Patil. 2011. *Building data science teams*. O'Reilly Media, Inc.

[45] David Piorkowski, Soya Park, April Yi Wang, Dakuo Wang, Michael Muller, and Felix Portnoy. 2021. How AI Developers Overcome Communication Challenges in a Multidisciplinary Team: A Case Study. In *Proceedings of the ACM on Human-Computer Interaction* 5, CSCW (2021), 1-25. DOI: https://doi.org/10.1145/3449205.

[46] Adam Rule, Aurélien Tabard, and James D. Hollan. 2018. Exploration and Explanation in Computational Notebooks. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. Association for Computing Machinery, New York, NY, USA, Paper 32, 1–12. https://doi.org/10.1145/3173574.3173606.

[47] Mariam Salloum, Daniel Jeske, Wenxiu Ma, Vagelis Papalexakis, Christian Shelton, Vassilis Tsotras, and Shuheng Zhou. 2021. Developing an Interdisciplinary Data Science Program. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education (SIGCSE '21)*. ACM, New York, NY, USA, 509–515. https://doi.org/10.1145/3408877.3432454.

[48] Jeff Saltz, Nicholas Hotz, David Wild, and Kyle Stirling. 2018. Exploring project management methodologies used within data science teams. In *Proceedings of the 24th Americas Conference on Information Systems (AMCIS '18)*, 2018, New Orleans, LA.

[49] Jeffrey S. Saltz and Nicholas Hotz. 2020. Identifying the most Common Frameworks Data Science Teams Use to Structure and Coordinate their Projects. In *Proc. 2020 IEEE International Conference on Big Data (Big Data)*, DOI: 10.1109/BigData50022.2020.9377813.

[50] Garold Stasser and William Titus. 1985. Pooling of unshared information in group decision making: Biased information sampling during discussion. *Journal of personality and social psychology* 48, 1467, 1985.

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30.

[52] Theresa Velden. 2013. Explaining field differences in openness and sharing in scientific communities. In *Proceedings of the 2013 conference on Computer supported cooperative work (CSCW '13)*. ACM, New York, NY, USA, 445–458. https://doi.org/10.1145/2441776.2441827.

[53] April Yi Wang, Anant Mittal, Christopher Brooks, and Steve Oney. 2019. How Data Scientists Use Computational Notebooks for Real-Time Collaboration. *Proc. ACM Hum.-Comput. Interact.* 3, CSCW, Article 39 (November 2019), 30 pages. https://doi.org/10.1145/3359141.

[54] April Yi Wang, Dakuo Wang, Jaimie Drozdal, Michael Muller, Soya Park, Justin D. Weisz, Xuye Liu, Lingfei Wu, and Casey Dugan. 2022. Documentation Matters: Human-Centered AI System to Assist Data Science Code Documentation in Computational Notebooks. *ACM Trans. Comput.-Hum. Interact.* 29, 2, Article 17 (April 2022), 33 pages. https://doi.org/10.1145/3489465.

[55] April Yi Wang, Zihan Wu, Christopher Brooks, and Steve Oney. 2020. Callisto: Capturing the "Why" by Connecting Conversations with Computational Narratives. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. ACM, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376740.

[56] Dakuo Wang, Justin D. Weisz, Michael Muller, Parikshit Ram, Werner Geyer, Casey Dugan, Yla Tausczik, Horst Samulowitz, and Alexander Gray. 2019. Human-AI Collaboration in Data Science: Exploring Data Scientists' Perceptions of Automated AI. *Proc. ACM Hum.-Comput. Interact.* 3, CSCW, Article 211 (November 2019), 24 pages. https://doi.org/10.1145/3359313.

[57] James Wexler, Mahima Pushkarna, Tolga Bolukbasi, Martin Wattenberg, Fernanda Viégas, and Jimbo Wilson. 2020. The What-If Tool: Interactive Probing of Machine Learning Models. *IEEE Transactions on Visualization and Computer Graphics* 26, 56-65. DOI: 10.1109/TVCG.2019.2934619.

[58] Elaine Wittenberg-Lyles, Debra Parker Oliver, George Demiris, Paula Baldwin, and Kelly Regehr. 2008 .Communication Dynamics in Hospice Teams: Understanding the Role of the Chaplain in Interdisciplinary Team Collaboration. *Journal of Palliative Medicine* 11, 1330-1335, DOI: 10.1089/jpm.2008.0165.

[59] Aoyu Wu, Yun Wang, Xinhuan Shu, Dominik Moritz, Weiwei Cui, Haidong Zhang, Dongmei Zhang, and Huamin Qu. 2021. AI4VIS: Survey on Artificial Intelligence Approaches for Data Visualization. *IEEE Transactions on Visualization and Computer Graphics*. DOI: 10.1109/TVCG.2021.3099002.

[60] Haijun Xia, Jennifer Jacobs, and Maneesh Agrawala. 2020. Crosscast: Adding Visuals to Audio Travel Podcasts. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST '20)*. Association for Computing Machinery, New York, NY, USA, 735–746. https://doi.org/10.1145/3379337.3415882.

[61] Amy X. Zhang, Michael Muller, and Dakuo Wang. 2020. How do Data Science Workers Collaborate? Roles, Workflows, and Tools. *Proc. ACM Hum.-Comput. Interact.* 4, CSCW1, Article 22 (May 2020), 23 pages. https://doi.org/10.1145/3392826.

[62] Chengbo Zheng, Dakuo Wang, April Yi Wang, and Xiaojuan Ma. 2022. Telling Stories from Computational Notebooks: AI-Assisted Presentation Slides Creation for Presenting Data Science Work. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. ACM, New York, NY, USA, Article 53, 1–20. https://doi.org/10.1145/3491102.3517615.