The Journal of Computing Sciences in Colleges

Papers of the 41st Annual CCSC Eastern Conference

October 24th-25th, 2025
Arcadia University
Glenside, PA

Abbas Attarwala, Editor
California State University, Chico

George Dimitoglou, Regional Editor
Hood College

Volume 41, Number 3 **October 2025**

The Journal of Computing Sciences in Colleges (ISSN 1937-4771 print, 1937-4763 digital) is published at least six times per year and constitutes the refereed papers of regional conferences sponsored by the Consortium for Computing Sciences in Colleges.

Table of Contents

The Consortium for Computing Sciences in Colleges Board of Directors

CCSC National Partners

The Consortium is very happy to have the following as National Partners. If you have the opportunity please thank them for their support of computing in teaching institutions. As National Partners they are invited to participate in our regional conferences. Visit with their representatives there.

National Partners:
Rephactor
ACM CCECC
ACM2Y
Blossoms
Local Vendors:
evapco
Jones & Bartlett Learning

Welcome to the 2025 CCSC Eastern Conference

Welcome to the Consortium for Computing Sciences in Colleges (CCSC) Eastern's 41st Annual Regional Conference, hosted this year by Arcadia University in Glenside, PA. We are so happy to welcome all of you to North Philly! On behalf of the CCSC Eastern Region and the program committee, we greatly appreciate all the submissions for papers, posters, workshops, panels, and nifty ideas.

We have been working since the beginning of the year to prepare for the conference. We have participants coming from at least 30 different organizations. While the CCSC committee ensured the highest quality of the materials selected for presentations and demonstrations, the Arcadia faculty, staff, and students worked to prepare the locations for the event. We hope you enjoy the activities planned during these two days, and you have a chance to visit the Castle!

It is also a great honor to have Dr. Chen Chaomei from Drexel University as the conference's keynote speaker, inspiring us with the topics of Macroscopic Patterns of Change. Additionally, it is a privilege to have Dr. Tom Way, co-creator of Rephactor, as our banquet speaker with a stunning Magic of Computer Science. We are so pleased to have both of them share their experiences and thoughts on the state-of-the-art ideas and future of computer science education. We are also thankful to our national sponsors, including the ACM Committee for Computing Education in Community Colleges (CCECC) and Blossoms AI.

This year, the conference's steering committee received tremendous interest in paper submission, and the committee members assembled in person in Glenside this past summer to finalize acceptance decisions. We received 22 faculty papers, and 16 were accepted (72%) after double blind peer review. We also have 9 faculty posters. Student submissions are at 4 papers and 30 posters. We also accepted 3 panels, 2 workshops, and 2 Nifty Ideas/Lighting Talks. We are excited about the wide variety of topics covered at the conference.

Lastly, as co-chairs of this year's conference, we are very grateful and honored to work alongside the committee and Arcadia team to ensure this prestigious regional forum for computing sciences is a productive experience. We look forward to meeting you at Arcadia and greatly appreciate your participation and attendance.

Vitaly Ford and Yanxia Jia
Conference Co-Chairs and Hosts

2025 CCSC Eastern Conference Committee

Co-Chairs

Vitaly Ford	Arcadia University
Yanxia Jia	Arcadia University

Papers

George Dimitoglou, Regional Editor	Hood College
Nooh A Bany Muhammad	Frostburg State University

Posters

Karen Anewalt	University of Mary Washington
Peter Hartnett	Frostburg State University
Ian Finlayson	University of Mary Washington

Panels, Workshops, and Tutorials/Nifty Ideas

Zhengping (Jay) Luo	Rider University

Programming Contest

Steven Kennedy	Frostburg State University
David Hovemeyer	Johns Hopkins University
TJ Highley	La Salle University
Jorge Silveyra	Lafayette College

Capture the Flag (CTF)

Zhijiang Chen	Frostburg State University
Mian Qian	Frostburg State University

ConfTool Registration and Submission System

Pranshu Gupta	DeSales University

Regional Board Representative

Michael Flinn	Frostburg State University

Regional Treasurer

Nathan Green	Marymount University

Web Site

John Wright	Juniata College

2025 CCSC Eastern Steering Committee

Elizabeth Adams	James Madison University
Steven Andrianoff	St. Bonaventure University
Karen Anewalt	University of Mary Washington
George Benjamin	Muhlenberg College
Vincent Cicirello	Stockton University
Susan Conrad	Marymount University
George Dimitoglou	Hood College
Michael Flinn, Regional Representative	Frostburg State University
Sister Jane Fritz	St. Joseph's College
Nathan Green, Interim Regional Treasurer (2019–2021)	Marymount University
Pranshu Gupta	DeSales University
David Hovemeyer	Johns Hopkins University
John Meinke	University of Maryland University College
Pipop Nuangpookka	Bay Atlantic University
Karen Paullet	Robert Morris University
Jennifer Polack-Wahl	University of Mary Washington
Donna Schaeffer	Marymount University
John Wright	Juniata College

The Magic of Computer Science[*]

Banquet

Tom Way, Ph.D., Rephactor and Villanova University

Speaker's Bio

Tom Way is the co-creator of Rephactor, an interactive online textbook for computer science. He is a soon-to-be Emeritus Professor at Villanova University, where he has taught for over two decades. A professional magician who once paid his way through college performing magic mostly for children's birthday parties, Tom now combines his expertise in computing and performance to reveal the magic in computer science itself.

[*]Copyright is held by the author/owner.

Macroscopic Patterns of Change Abstract: Disruptive technologies[*]

Keynote

Dr. Chaomei Chen, Drexel University

Disruptive technologies—most visibly the current storm of AI—can forge and force profound and swift changes. The value of once unique expertise may diminish. The reliability of once trustworthy sources may become questionable. In such an environment, adaptation is essential for regaining a competitive edge. In this keynote, I will trace recurring macroscopic patterns of change across art, science, and technology. These patterns reveal high-order insights. With such meta-knowledge, we can better anticipate change, strengthen our strategic position, and sustain lifelong learning—while enriching the pursuit of our aspirations.

Speaker's Bio

Dr. Chaomei Chen is a Professor of Information Science in the College of Computing and Informatics at Drexel University in the USA. He is the Editor-in-Chief of Information Visualization and the Field Chief Editor of Frontiers in Research Metrics and Analytics. His research and scholarly expertise is in

[*]Copyright is held by the author/owner.

the visual analytic reasoning and assessment of critical information concerning the structure and dynamics of complex adaptive systems. He authored a series of books on various related topics, including Representing Scientific Knowledge: The Role of Uncertainty (Springer 2017), The Fitness of Information: Quantitative Assessments of Critical Information (Wiley, 2014), Turning Points: The Nature of Creativity (Springer, 2011), Information Visualization: Beyond the Horizon (Springer 2004, 2006), Mapping Scientific Frontiers: The Quest for Knowledge Visualization (Springer 2003, 2013). He is the creator of the widely used visual analytics software CiteSpace for visualizing and analyzing structural and temporal patterns in scientific literature. His research has been funded and sponsored by the National Science Foundation (NSF), Elsevier, IMS Health, Lockheed Martin, and Pfizer. His earlier research was funded by the European Commission, the Engineering and Physical Sciences Research Council (UK), and the Library and Information Commission (UK). He received a B.Sc. in Mathematics (Nankai University, China), an M.Sc. in Computation (University of Oxford, UK) and a Ph.D. in Computer Science (University of Liverpool, UK).

Approaches to Experiential Learning Leading to Better Job Placement[*]

Panel Discussion

Diane Murphy, Alex Mbaziira
Marymount University
Arlington, Virginia 22207
{diane.murphy, mbaziir}@marymount.edu

Abstract

The classic student internship is often hard to find, most notably in summer 2025 with the impact of cuts in government and industry offerings. Coupled with this is the ongoing reluctance of organizations to hire entry-level talent that cannot show evidence of their skills through direct work experience. As skill-based hiring continues to become the norm for technology job placement, universities and colleges need to become more engaged in simulating the work-based learning approach before students graduate.

A panel of faculty and students will discuss the array of work-based learning options available at their institution and the impact on the students' skills development, including both technical skills and professionalism, and their job placement.

Some of the techniques discussed will be: embedded labs; the competition ecosystem; bootcamps for additional skills development not included in the current programs; developing and operating a cyberclinic; participating in research initiatives; and learning through teaching, mentoring, and other forms as part of community outreach.

Professional skills developed include time management, flexibility, elastic thinking, teamwork, and communication.

Audience participation is encouraged.

[*]Copyright is held by the author/owner.

Impact of AI Tools on Faculty Performance at Frostburg State University's Computer Science Department[*]

Panel Discussion

Ying Zheng, Steve Kennedy, Nooh Muhammad,
Yuechen Chen, Zhijiang Chen
Department of Computer Science and Information Technologies
Frostburg State University
Frostburg, MD 21532
{yzheng,sdkennedy,nbany,ychen,zchen}@frostburg.edu

1 Summary

A panel discussion on the impact of AI tools on faculty performance based on a study conducted at a regional state university.

The objective of this study is to investigate the impact of artificial intelligence (AI) tools on faculty performance in the Department of Computer Science at Frostburg State University. The study adopted a mixed-method approach where faculty members were involved in the surveys that were used in data collection about how they use AI and its impact on their teaching. Results from the survey show that there have been significant improvements such as quick marking, student engagement as well as general job satisfaction among others. Furthermore, faculty satisfaction with teaching increased because grading took less time while students were more involved too. However, despite its advantages like those mentioned above; the introduction of this technology faced some challenges including technical hitches during set-up stages which required additional expertise than what most people had at first sight. Recommendations given were based on long-term gains which may only be realized if regular

[*]Copyright is held by the author/owner.

training is provided together with required materials. Having noted down all these advantages brought about by the use of artificial intelligent tools, it can be concluded that AI tools can enhance faculty performance effectiveness on part of teaching although support from the institution is needed for better outcome achievement. These results provide valuable insights for academic institutions looking to adopt AI technologies to enhance educational practices and faculty experiences.

Students Should Learn to Build GUI Applications Without Using IDEs[*]

Tutorial

Penn P. Wu
Cypress College, Cypress, CA 90630
pwu@cypresscollege.edu

Most modern IDEs (Integrated Development Environments), such as Microsoft's Visual Studio (VS), have been adopted by college professors to teach basic programming and the development of basic GUI (graphical user interface) applications. These IDEs support a wide range of platforms such as Windows Forms (WinForms), WPF (Windows Presentation Foundation), WinUI 3, .NET MAUI (Multi-platform App UI), Blazor Hybrid, and so on. While intelligent IDEs provide excellent support for building GUI applications, they abstract away fundamental coding steps required to build the structures and functionalities to construct an interactive GUI application.

For languages like C# and VB.NET, VS provides intelligent visual designer that can generate codes and autocorrect errors. Even without human intervention, VS can seamlessly create a Windows form and its GUI components as well as basic interactive features to make GUI development more automated for beginners. Interestingly, the rich set of drag-and-drop interfaces of IDEs can significantly take away the learning opportunities. The ease of use and automation provided by Visual Studio can shield students from understanding what happens behind the scenes, not to mention compilation and linking. This might hinder their learning of how GUI applications truly work. Students often become highly reliant on IDEs as "click-and-hope" users, not programmers. Students are not only puzzled by the AI-generated codes but also lost on the errors they encounter.

It is generally considered necessary and beneficial for students to learn hand-coding, particularly in a college-level programming course. When GUI application development projects are part of the curriculum, this presentation advocates engaging students in hand-coding GUI applications from scratch

[*]Copyright is held by the author/owner.

without using IDEs, at least to a significant extent for several reasons. First, it helps students obtain a deeper understanding of the framework, the underlying structure, object model, and even-driven nature of Windows forms. Second, students visualize the role of APIs by exploring how “controls” are instantiated, properties are set, and events are wired up in the hand-generated codes. Third, experience how to set properties and styles to manually design the layouts of user interfaces (UIs), implement custom drawing and rendering, and write code to create GUI elements.

Being able to code GUIs manually is a valuable skill in various coding scenarios. Fourth, observe the interoperability with Windows APIs, so students can understand how to access the lower-level Windows APIs, which is necessary knowledge for developing resource-efficient applications. Fifth, students can develop transferable skills. The fundamental concepts of UI development (control instantiation, property manipulation, event handling) learned through hand-coding Windows Forms are transferable to other UI frameworks and programming paradigms. This is a highly demanded skill in the industry.

This presentation will demonstrate how to: (1) design and develop UIs include a Windows form and several interactive controls as its UI components, (2) write Visual C# codes to build the Windows form from scratch by hands without using the Visual Studio IDE, (3) use .Net compliant codes to place UIs components at the designated locations and fix the layout, style, and format of every UIs component, (4) manually compile the code in the Developer Command Prompt which is a console environment, (5) test and execute the self-executable program (.exe) in both console and GUI environments, and (6) compare the file size with a similar program built by Visual Studio IDE. The objective is to explain how hand-coded applications, where developers write the source code directly without relying on visual designers or code generators, offer significant advantages. Once students understand the basics through hand-coding, they can better appreciate and utilize the advanced features of IDEs for increased productivity in more complex projects.

In an era where visual development tools are prevalent, students need an instructor-led discussion in hand-coding of GUI applications, especially in college-level programming courses. This presentation will also showcase how handcoded applications are compact in file size, customizable in every aspect of layout and functionality, minimized in overhead by eliminating unnecessary codes and abstractions added by visual tools, and easier for modification and revision. Students will realize why hand-coding involves deliberate planning and debugging. Audiences will take away the pedagogical approach to prepare instructional materials to guide students through a hands-on coding project. The sample project can inspire instructors to design more examples to lead students to build GUI applications without the immediate assistance of IDE features. The instructional objective is to encourage students to develop hand-coding skills by exposing them to simpler text editors like Microsoft Notepad. Sample lecture notes and a 50-minute lecture video is also furnished upon audiences' request.

Teaching Database Concepts using Data Model/MySQL[*]

Workshop

Syed Ahmed
Computer Science
York Technical College
Rock Hill, SC 29730, USA
sahmed@yorktech.edu

In this workshop Database concepts will be explained using a sample relational database through data modelling using MySQL workbench to create the tables, insert data and run queries. Tables created will implement referential, entity and domain integrity that will also make the database immune to data deletion and update anomalies. A text file will be used to insert data to save time.

Presenter background

I have over thirty-five years of experience of teaching computer science courses that I have taught in Pakistan, Georgia, Nebraska and South Carolina. I have taught Database using Oracle and MySQL to undergraduate students in University of North Georgia for over seventeen years.

Intended audience

Faculty and students who would like to enhance their basic understanding of relational database.

Materials provided

The handout containing detailed step by step instructions for the tutorial will be provided by the presenter.

[*]Copyright is held by the author/owner.

Audio/Visual/Computer requirements

Computer lab with MySQL Workbench version 8.0.30 and projector.

Comparing Human-Written and AI-Generated Code*

Nifty Idea

Carolyn Pe Rosiene[1] and Joel Rosiene[2]
[1] University of Hartford
West Hartford, CT 06117
rosiene@hartford.edu
[2] Eastern Connecticut State University
Willimantic, CT 06226
rosienej@easternct.edu

As AI becomes ubiquitous, expectedly, more students find it tempting to rely on generative-AI to write their code for submission. This assignment tasks the students to compare their own, human-written code (presumably, it really is their own) against AI-generated code that performs the same purpose. The target audience is an upper-level course which covers the fundamental concepts of programming languages and major tools and techniques to implement them including syntax specification; binding and scoping; types and type systems; control structures; data abstraction; procedural abstraction and parameter passing; higher-order functions; and memory management; it also explores key characteristics of major programming paradigms, including their relationship to the imperative programming paradigm.

This assignment is given at the end of the semester when the crucial topics have been covered. Students evaluate both sets of code based on (1) paradigm fidelity, (2) readability and (3) strengths and weaknesses focusing on the differences between both sets of code. This is given in conjunction with a 10-week long project where students investigate a language of their choice, and presents their findings to the class at the end of the semester. This assignment is an addendum to that project to allow them to critically evaluate their code versus AI-generated code.

*Copyright is held by the author/owner.

Intended Audience: Faculty members working with neurodiverse students
Material Provided: PowerPoint

Higher Education Institution Websites and a Lack of Accessibility[*]

Nifty Idea

Andrea Wentzell
Computer Science Department
Chestnut Hill College
Philadelphia, PA 19118
wentzella@chc.edu

Frequently, over 1.3 billion neurodiverse individuals face poorly developed and designed websites, causing issues with communication due to a lack of accessibility. Many industries fail to implement standards, such as the web content accessibility guidelines (WCAG), effectively on their websites. One such industry that may surprise some is higher education institutions (HEIs). The impact of negative accessibility of HEI websites can cause a lack of interest in these institutions, despite the rate of neurodiverse individuals attending HEIs increasing. This talk addresses where small northeastern catholic institutions stand regarding WCAG 2.0 A/AA standards, error density rates, and types of errors detected through the WAVE WebAIM application and the WebAIM Million listing. Findings show that most small HEIs are failing to properly implement WCAG guidelines and accessibility techniques related to message interactivity.

Intended Audience: Faculty members teaching accessibility and website development/design, students interested in accessibility
Material Provided: PowerPoint
A/V/C Requirements: Computer with access to PowerPoint

[*]Copyright is held by the author/owner.

SimuPilot VR: A LLM-based Black-box Virtual Reality Exploration[*]

Poster Session

Matthew DiGiovanni, Xue Qin
Department of Computing Sciences
Villanova University
Villanova, Pennsylvania, USA
mdigio02, xue.qin@villanova.edu

The Virtual Reality (VR) industry is rapidly expanding, with wide-ranging applications due to its ability to simulate the physical world in a digital environment. A major challenge in VR environments is addressing spatial navigation and user orientation. To address this, we designed a Blackbox solution, Simulated Pilot for Virtual Reality (SimuPilot-VR), an LLM-driven tool designed to assist users in navigating VR environments through screen analysis and task-to-action mapping by prompt engineering. The proposed approach can be directly applied to any VR application, regardless of the hardware devices and platforms used.

In particular, the system captures real-time VR screenshots and then prompts the LLM for a textual description of the identified entities and their spatial relationships with the user. With this report as context, we then design a matching system that collects the user's request as text and matches its intent with one of the pre-defined actions. The action decision is then translated into a sequence of executable controller signals. While performing the sequential actions to complete the user task, the team leverages the LLM to monitor the task-ending conditions and stop sending the controls when the exploration goal is achieved. The SimuPilot-VR was evaluated within two scenarios: one single-target case and one multi-target environment case. Within five tries, the tool successfully reached target objects with an average of 8 and 6 steps for each case, respectively. Only one try in the multi-target scenario failed due to the overshooting of the action (making the item disappear from the view).

[*]Copyright is held by the author/owner.

Ultimately, we discuss future plans, such as context understanding, to address the observed limitations.

Traditional Optimization vs. AI-Driven Power Management*

Poster Session

Blessing Etih-Engo Tewan, Kimberly Grace Allagnon
Adekemi Adepoju
Advisor: Fahmina Nur Salma
Bowie State University
Bowie, Maryland, USA

This study explores how AI-driven power management can optimize mobile app performance while reducing battery consumption compared to traditional optimization methods. Traditional techniques rely on static rules and cannot adapt to user behavior. In contrast, machine learning models analyze usage patterns in real-time, enabling dynamic and personalized energy management. Recent research shows AI methods can extend battery life by up to 40%, improve prediction accuracy, and manage charging more efficiently. By integrating adaptive algorithms and real-time data, AI offers a smarter, more efficient approach to mobile battery optimization on iOS and Android platforms.

*Copyright is held by the author/owner.

Quantum-Safe Cryptography: Addressing Algorithmic Gaps for the Post-Quantum Era[*]

Poster Session

Lashawna Perry and Xannia Simpson
Advisor: Ruth Agada
Bowie State University
Bowie, Maryland, USA

This project explores the development and deployment of post-quantum cryptographic (PQC) algorithms in response to emerging quantum threats. While algorithms like CRYSTALS-Kyber and Dilithium show promise, challenges remain in integrating them into real-world environments. We analyze the roles of NIST, IBM, and Google in advancing PQC, and identify performance gaps, especially in constrained systems. With Qiskit now installed on our lab computer, we plan to take a major step into quantum error correction, a critical factor delaying practical quantum computing, and assess its implications for secure cryptographic design.

[*]Copyright is held by the author/owner.

Fast Fashion and Computer Algorithms[*]

Poster Session

Gracemercy Gichaga, Fahmina Nur Salma
Bowie State University
Bowie, Maryland, USA

This study analyzes how algorithms influence fast fashion trends, and the resulting toll on the environment through waste and growing landfills. The studies reviewed look at the various stages of production, manufacturing, marketing and consumption, that contribute to waste at each stage. Studies recommend the use of AI and machine learning in fashion marketing to allow (1) automation, (2) improved personalization, and (3) enhanced sustainability practices within the industry, to build an efficient and environmentally conscious fashion market.

[*]Copyright is held by the author/owner.

Local Government Supply Chain Cybersecurity: Addressing the Implementation Gap in Resource-Limited Municipalities[*]

Poster Session

Sage Despeignes, Titorian Huggins, Devharsh Trivedi
Department of Computer Science
Bowie State University
Bowie, Maryland, USA
{despeigness0426, hugginst0604, dtrivedi}@students.bowiestate.edu

Local governments are under constant threat of cyberattacks, a risk that has grown as they rely more on third-party vendors and cloud services [5]. However, many small and mid-sized municipalities (10,000–100,000 population) do not have enough resources or staff to put strong cybersecurity practices in place. A recent nationwide survey found that 30% of local governments have no formal cybersecurity programs, and 80% have fewer than five dedicated IT security staff [4].

Even when cybersecurity policies exist, there is often a gap between what is written and what is done. One study found that while many U.S. cities have formal cybersecurity policies, they struggle to apply them in day-to-day operations. This highlights the need for better data management, external audits, and staff training [1]. The same gap shows up in procurement and vendor oversight: most small municipalities do not carefully evaluate the cybersecurity risks that come with working with outside vendors [2].

An analysis of 38 local government cybersecurity policies showed that none addressed all categories of the NIST Cybersecurity Framework, and most did not include third-party risk management [3]. Frameworks like NIST CSF and StateRAMP provide helpful guidelines for managing vendor risk [5], but many small municipalities find them hard to use without additional help.

This ongoing undergraduate research focuses on Maryland municipalities to explore how cybersecurity is handled in procurement policies. We are reviewing

[*]Copyright is held by the author/owner.

local government procurement documents and vendor contracts to see how well they match up with NIST CSF, StateRAMP, and other guidance, and where the gaps are.

Early findings show that statewide efforts can help. For example, Maryland allows local governments to use state contracts to get cybersecurity services at better prices [6]. This research offers practical suggestions like adding clear language in RFPs, setting basic cybersecurity requirements for vendors, and sharing cybersecurity services. These small changes could improve security in local government systems without adding major costs.

References

[1] W. Hatcher, W. L. Meares, and J. Heslen, "The cybersecurity of municipalities in the United States: an exploratory survey of policies and practices," Journal of Cyber Policy, vol. 5, no. 2, pp. 302–325, 2020.

[2] S. T. Hossain, T. Yigitcanlar, K. Nguyen, and Y. Xu, "Local Government Cybersecurity Landscape: A Systematic Review and Conceptual Framework," Applied Sciences, vol. 14, no. 13, p. 5501, 2024.

[3] S. T. Hossain, T. Yigitcanlar, K. Nguyen, and Y. Xu, "Understanding Local Government Cybersecurity Policy: A Concept Map and Framework," Information, vol. 15, no. 6, p. 342, 2024.

[4] Center for Internet Security, "2023 Nationwide Cybersecurity Review: Summary Report," 2023. [Online]. Available: `https://www.cisecurity.org/insights/white-papers/nationwide-cybersecurity-review-2023-summary-report`

[5] S. Descant, "StateRAMP to Offer State, Local Government Secure Vendor Pool," Government Technology, Mar. 2021. [Online]. Available: `https://www.govtech.com/security/stateramp-to-offer-state-local-government-secure-vendor-pool.html`

[6] J. S. Berkowicz, "Uplifting State and Local Procurement into the Modern World: How the Model Procurement Code's Next Revision Could Adopt the New York Department of Finance's Cybersecurity Principles for Its Own Standards," ABA Public Contract Law Section Journal, Dec. 2024. [Online]. Available: `https://www.americanbar.org/groups/public_contract_law/resources/journal/2024-fall/uplifting-state-local-procurement-revivision-new-york-standards/`

Rising Cybercrime Complaints and Youth Vulnerability: A 2021–2024 Trend Analysis*

Poster Session

Haley Reyes, Hoda El-Sayed
Bowie State University
Bowie, Maryland, USA

As cyber threats evolve in sophistication and scope, youth are increasingly at risk. Despite this, reported phishing and spoofing complaints to the FBI's Internet Crime Complaint Center (IC3) fell from over 323,000 in 2021 to 29,836 in 2024 [1][2]. This poster explores the contradiction between the declining number of reported complaints and the rising exposure of young users to cyber threats. Drawing on IC3 annual reports and supplemental research from Internet Matters [3], the UK National Cyber Strategy [5], CISA [4], and FBI cyber awareness efforts [6], the analysis highlights how shifts in attack vectors, underreporting, and gaps in digital literacy contribute to growing youth vulnerability. The findings suggest that the true scale of youth-targeted cybercrime may be masked by outdated reporting categories and a lack of cybersecurity education. This work underscores the need for improved cybercrime reporting structures and early digital literacy initiatives to address the growing risks facing young internet users.

References

[1] Internet Crime Complaint Center (IC3), 2021 Internet Crime Report, 2021. [Online]. Available: https://www.ic3.gov/AnnualReport/Reports/2021_IC3Report.pdf

[2] Internet Crime Complaint Center (IC3), 2024 Internet Crime Report, 2024. [Online]. Available: https://www.ic3.gov/AnnualReport/Reports/2024_IC3Report.pdf

*Copyright is held by the author/owner.

[3] A. Katz and A. El Asam, Vulnerable Children in a Digital World, Internet Matters and Youthworks, 2019. [Online]. Available: `https://www.childnet.com/wp-content/uploads/2020/02/Internet-Matters-Report-Vulnerable-Children-in-a-Digital-World.pdf`

[4] Cybersecurity and Infrastructure Security Agency (CISA), "4 Things You Can Do to Keep Yourself Cyber Safe," Fact Sheet, U.S. Department of Homeland Security, Feb. 2023. [Online]. Available: `https://www.cisa.gov/sites/default/files/2023-02/cisa_fact_sheet_4_things_cyber_english_508.pdf`

[5] UK Government, "National Cyber Strategy 2022," Official Publication, HM Government, London, UK, Dec. 2022. [Online]. Available: `https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/1053023/national-cyber-strategy-amend.pdf`

[6] Federal Bureau of Investigation, "FBI Highlights Online Safety Tips During Cybersecurity Awareness Month," FBI Norfolk Field Office, Oct. 5, 2023. [Online]. Available: `https://www.fbi.gov/contact-us/field-offices/norfolk/news/fbi-highlights-online-safety-tips-during-cybersecurity-awareness-month`

Strategies Used by Attackers to Plan and Execute Phishing Emails Targeting Financial Services[*]

Poster Session

Cheryl-Devon Twyman, Nia Allen
Octavia Brewster and Esther Sobo
Bowie State University
Bowie, Maryland, USA
{twymanc1115, allenn1006, brewstero0719, soboe0222}@students.bowiestate.edu

Phishing is a deceptive cyberattack method where attackers impersonate legitimate entities to steal sensitive user information. This research explores phishing's impact on the financial sector, a frequent target due to its valuable data and high transaction volume. We analyzed phishing techniques, common indicators, and prevention strategies used by financial institutions. The findings highlight the need for continuous employee training, multi-factor authentication, and real-time threat monitoring to reduce phishing risks.

[*]Copyright is held by the author/owner.

Exploring the Application of Feedforward Neural Networks in Solving Basic Linear and Quadratic Algebraic Equations[*]

Poster Session

Oluwatoyin Kode, Ayomide Aisida, and Oshoriameh Torhira Aminu
Bowie State University
Bowie, Maryland, USA
okode@bowiestate.edu, {aisidaa0727, aminuo0504}@students.bowiestate.edu

This study investigates the use of Feedforward Neural Networks (FNNs) to solve basic algebraic equations, specifically linear and quadratic forms. While artificial intelligence is often applied to pattern recognition and statistical inference tasks, its application to symbolic mathematical reasoning remains an emerging field. This research explores whether neural networks can effectively learn to solve algebraic equations by identifying relationships between equation structures and their solutions.

A Feedforward Neural Network is a type of artificial neural network where connections between nodes do not form cycles. It processes inputs through layers of interconnected neurons in one direction, from input to output, making it suitable for supervised learning tasks like function approximation. The project explores whether a basic FNN can be trained to accurately solve algebraic equations by learning input-output mappings from a labeled dataset of algebraic expressions and their correct solutions sourced from Kaggle. This enables the model to predict the solution to new, unseen equations. The model is trained on various equation forms and evaluated using performance metrics such as Mean Squared Error (MSE) and solution accuracy. Preliminary findings suggest that the FNN can generalize across multiple equation types and solve expressions even without closed-form analytical solutions.

A key advantage of this approach is the FNN's ability to generate accurate solutions with minimal computational overhead once trained. This opens up exciting possibilities for real-time applications, such as intelligent tutoring

[*]Copyright is held by the author/owner.

systems, potential applications in educational IT systems, and symbolic computation tools for K-12 education. Compared to traditional rule-based solvers, FNNs offer flexibility, scalability, and adaptability to equation structures beyond preset rules. Overall, this work contributes to the integration of machine learning in symbolic mathematics and lays the foundation for more complex neural architectures in future research. This study investigates the use of Feedforward Neural Networks (FNNs) to solve basic algebraic equations, specifically linear and quadratic forms. While artificial intelligence is often applied to pattern recognition and statistical inference tasks, its application to symbolic mathematical reasoning remains an emerging field. This research explores whether neural networks can effectively learn to solve algebraic equations by identifying relationships between equation structures and their solutions.

A Feedforward Neural Network is a type of artificial neural network where connections between nodes do not form cycles. It processes inputs through layers of interconnected neurons in one direction, from input to output, making it suitable for supervised learning tasks like function approximation. The project explores whether a basic FNN can be trained to accurately solve algebraic equations by learning input-output mappings from a labeled dataset of algebraic expressions and their correct solutions sourced from Kaggle. This enables the model to predict the solution to new, unseen equations. The model is trained on various equation forms and evaluated using performance metrics such as Mean Squared Error (MSE) and solution accuracy. Preliminary findings suggest that the FNN can generalize across multiple equation types and solve expressions even without closed-form analytical solutions.

A key advantage of this approach is the FNN's ability to generate accurate solutions with minimal computational overhead once trained. This opens up exciting possibilities for real-time applications, such as intelligent tutoring systems, potential applications in educational IT systems, and symbolic computation tools for K-12 education. Compared to traditional rule-based solvers, FNNs offer flexibility, scalability, and adaptability to equation structures beyond preset rules. Overall, this work contributes to the integration of machine learning in symbolic mathematics and lays the foundation for more complex neural architectures in future research.

Redesigning Cybersecurity Curriculum in Technical Colleges: A Modular Approach to Workforce Readiness*

Poster Session

Hameed Alzahrani
Marymount University
Arlington, Virgnia, USA

As cybersecurity threats increase and job roles diversify, technical colleges face growing pressure to adapt their educational programs to align with workforce needs. Preliminary analysis from a system-wide curriculum review reveals structural challenges, including duplicated foundational content, limited specialization, and underutilization of experiential tools. These issues can hinder student engagement and reduce alignment with industry-defined competencies.

This poster proposes a modular curriculum redesign model that addresses these challenges through flexible course pathways tailored to varied student entry points. The proposed model integrates stackable content blocks that align with key knowledge areas, promotes elective-based specialization, and embeds soft skills and ethical reasoning across technical modules. It also recommends the standardized use of cloud-based cyber ranges and virtual labs to ensure consistent hands-on learning.

The modular approach aims to make the curriculum more adaptive, allowing institutions to accommodate students from diverse technical backgrounds while maintaining rigor and relevance. It also facilitates better faculty collaboration, reduces redundancy, and aligns more clearly with national and international cybersecurity workforce frameworks.

This poster invites feedback from educators, curriculum designers, and workforce partners to refine this model. The goal is to support technical colleges in delivering more agile, role-aligned, and industry-ready cybersecurity education programs.

*Copyright is held by the author/owner.

Enhancing RSA Cryptosystem Performance: A Comparative Study of Fermat's Little Theorem and Chinese Remainder Theorem Versus Standard Modular Arithmetic*

Poster Session

Roxan Rockefeller
Bowie State University
Bowie, Maryland, USA

My research focuses on improving the performance of the RSA cryptosystem by comparing mathematical techniques that can replace or optimize the standard modular arithmetic operations RSA normally relies on. RSA is one of the most widely used public-key encryption methods, and its security depends on the difficulty of factoring large prime numbers. However, while RSA is secure, it is not always efficient. The heavy cost of modular exponentiation makes encryption and especially decryption slower, which is a critical limitation in applications where speed is essential.

The main goal of this thesis is to investigate whether methods such as Fermat's Little Theorem (FLT) and the Chinese Remainder Theorem (CRT) can make RSA more computationally efficient compared to traditional modular arithmetic. FLT can simplify exponentiation by reducing large exponents into smaller, equivalent forms, while CRT allows computations to be divided into smaller modular operations and then recombined for the final result. Both approaches offer the potential to reduce execution time without altering RSA's core security. To evaluate this, RSA will be implemented using three approaches: standard modular arithmetic, optimization with FLT, and optimization with CRT. These implementations will be analyzed based on execution time, efficiency in key generation, and scalability with larger key sizes. The comparisons aim to highlight the strengths and trade-offs of each method, offering insight into where each approach is most effective.

*Copyright is held by the author/owner.

Ultimately, this thesis seeks to demonstrate how mathematical optimizations can enhance the real-world usability of RSA. If FLT or CRT consistently provide performance improvements, the results could support more efficient RSA implementations in environments where both speed and security are vital, such as secure communications, digital signatures, and resource-constrained embedded systems.

Edge-Based Object Detection for Visual Assistance Using Raspberry Pi and YOLO[*]

Poster Session

Tarron Montgomery, Malachi Gray, Brandon Wiggins,
Staphord Bengesi and Md Kumruzzaman Sarker
Bowie State University
Bowie, Maryland, USA

Visual impairments create substantial barriers to safe and independent navigation, impacting the daily lives of millions. To address this issue, we propose an edge-based assistive framework that leverages real-time object detection to enhance situational awareness for the visually impaired. This system is centered on a Raspberry Pi, which acts as an edge computing device to process visual data locally, eliminating the need for constant cloud connectivity and reducing latency.

The framework includes an external USB camera mounted on wearable gear, such as glasses or a hat, which continuously captures the user's surroundings. The video feed is processed using the Ultralytics YOLO (You Only Look Once) object detection model hosted on a Flask server running directly on the Raspberry Pi. Detected objects are converted into descriptive audio cues using text-to-speech (TTS) technology, and the processed information is transmitted to the user's smartphone via a custom mobile app called SightAssist. This setup ensures users receive real-time, context-aware feedback to navigate their environment more safely and confidently.

The system emphasizes affordability, portability, and scalability, making it an accessible solution for broader deployment. Prototype testing demonstrates the reliability and responsiveness of the proposed framework in real-world scenarios, validating its effectiveness in assisting visually impaired users. This work highlights the potential of combining edge computing, computer vision, and mobile technologies to develop practical assistive systems aimed at improving autonomy and quality of life.

[*]Copyright is held by the author/owner.

Developing Immersive VR for Anti-Cyberbullying Training*

Poster Session

Sumedha Gajanan Pol and Edward Heimbach
Pennsylvania State University
Department of Information Science and Technology
Center Valley, Pennsylvania, USA
{eah6105, smp6989}@psu.edu

This project focuses on configuring Meta Quest 3 virtual reality hardware and Unity software to develop immersive training experiences as part of an U.S. Bureau of Justice Administration-funded initiative. The primary goal of this project is to support school personnel in effectively recognizing, managing, and responding to bullying and cyberbullying scenarios. Under the guidance of our faculty supervisor, immersive environments are designed to simulate real-life situations, providing educators, counselors, and students with hands-on experimental training that encourages empathy, preparedness, and intervention skills. The technical setup involves integrating Meta Quest 3 with Unity to create fully interactive VR scenarios. This includes application development, configuring settings, managing project assets, optimizing performance, and seamless deployment to headsets. All processes ranging from initial hardware-software interfacing to content distribution are carefully documented to support reproducibility and future scaling of the project. Collaboration with faculty and other project stakeholders ensures that the content is not only technically sound, but also relevant and aligned with project goals. This interdisciplinary project bridges technology and social impact using immersive media to tackle complex social issues in educational environments. The initiative represents a step in leveraging VR for impactful training that allows students and staff to create safer and more supportive educational environments. The poster will describe the development processes, implementation plans, and goals of the project. Screenshots of the applications will also be included.

*Copyright is held by the author/owner.

Simulating and Mitigating Quantum Noise in the Variational Quantum Eigensolver Algorithm[*]

Poster Session

Peter Annis and Abe Kassem
University of Mary Washington
Fredericksburg, Virginia, USA
pannis, akassem2@mail.umw.edu

This research investigates the impact of quantum noise on the Variational Quantum Eigensolver (VQE) algorithm and evaluates multiple error mitigation strategies to improve computational accuracy on NISQ devices. Using IBMs Qiskit quantum computing framework, we implement and analyze VQE simulations for small molecular systems, with simple molecules (e.g., lithium hydride (LiH)) as our primary test cases. Our study employs a multi-faceted approach including noise simulation and characterization, error mitigation techniques, and a comparative analysis. We conduct comprehensive benchmarking across multiple metrics including energy estimation accuracy and computational overhead and attempt to leverage modern approaches from the relevant literature where possible.

Expected outcomes include a quantitative assessment of error mitigation effectiveness across different noise regimes, open-source Jupyter notebooks implementing all techniques for community use and extension, and insights into the interplay between variational optimization and error mitigation in hybrid quantum-classical algorithms. This work contributes to the growing body of knowledge on making quantum algorithms practical for near-term devices, directly supporting the quantum computing community's goal of achieving quantum advantage in scientifically relevant problems despite hardware limitations.

[*]Copyright is held by the author/owner.

Impact of Demographics on COVID-19 Outcomes in New Jersey[*]

Poster Session

Ummu Yuzugulluer, Ching-yu Huang
Kean University
Union, New Jersey, USA

This project examines COVID-19 outcomes in New Jersey using a three-year dataset from March 2020 to March 2023.

We combined the county_data table, which includes daily case and death counts, with the uscities_2024 dataset that provides demographic information such as population, education, income, and health insurance coverage. The combined dataset was cleaned and processed in MariaDB and DBeaver and visualized using Power BI. We transformed the cumulative case and death counts into daily values, aggregated them monthly, and calculated ratios using county populations. These steps allowed us to capture the major COVID waves and compare patterns across counties.

We looked at demographics that might relate to COVID outcomes. Counties with higher health uninsured rates had higher death ratios, with correlation around 0.4. Income and education looked weaker at first. When we split the data into two periods, before vaccines (Jan 2020-Apr 2021) and after vaccines (May 2021-Sep 2022), the differences became clearer. In the first period, education showed almost no relation to death ratios($r=0.04$). In the second period, counties with higher college education levels had lower death ratios($r=-0.23$). This may be connected to vaccine use.

These results give a simple picture of how COVID spread across New Jersey counties and how demographics played a role. The work shows that using database tools and basic visualizations can help track the waves of the pandemic and compare areas over time. It also shows that linking health data with demographics can explain some of the differences between counties and give useful ideas for public health planning.

[*]Copyright is held by the author/owner.

Analyzing Prompt Effectiveness in Generating Feedback for Programming Errors*

Poster Session

Quan Nguyen[1], Chrisma Ndlovu[2]
[1]VNU-HCM High School for the Gifted
Ho Chi Minh City, Vietnam
[2]West Chester University
West Chester, Pennsylvania, USA
student230431@ptnk.edu.vn, cn1030173@wcupa.edu

This project investigates various prompting structures and their corresponding feedback. We aim to understand which prompting structures have the potential to deliver feedback that are most meaningful and actionable for students. A baseline prompt template was developed. A set of twelve different prompts was created and studied. We develop an automated framework with an integrated local LLM to feed various prompts in combination with more than 3,000 error messages. These errors were collected from actual student submissions from a computer system class over four semesters. We observed notable differences in the generated feedback, particularly regarding the level of details and clarity. These results will be integrated into a larger LLM-backed Autograder with Feedback Framework.

*Copyright is held by the author/owner.

Enhancing the Testing, Training and Learning for Driver License with Deep Learning*

Poster Session

Taruna Suryawanshi and Tracy Nyamnjoh
Advsior: Osman Guzide
Shepherd University
Shepherdstown, West Virginia, USA

This study explored the use of deep learning to train an artificial intelligence model to enhance driver license testing and learning by classifying driving images. The model vehicle was taken from the open-source resource Microsoft AirSim, and was used within Unity's Unreal Engine environment. Our research focused on good and bad driving behavior. The data collected were snapshots categorized as either "on road" or "off road". To create this AI model, we utilize TensorFlow and Keras to normalize our data. We built a convolutional neural network (CNN) using TensorFlow's Sequential API. The model was trained over 20 epochs, with performance being evaluated on accuracy, precision, and recall. The results of this training showed that the model was able to identify the visual features between the "on road" images and the "off road" images. Once the model was trained, it can be used on new directories containing various images in various weather conditions. By applying CNN to driving imagery, our research highlights how machine learning can easily enhance driving safety features by recognizing good and bad driving. While this research focused on visual classification, future research could develop models that can classify additional data – such as speed and steering angle. With more time and advancements, this study could also lead to the research of aircraft training.

*Copyright is held by the author/owner.

Identifying Emotions in Text[*]

Poster Session

Jassiris Nunez
Advisor: Navya Martin Kollapally
Department of Computer Science and Technology
Kean University
Union, New Jersey, USA
nunezjas@kean.edu

With the rapid growth of social media, millions of people express their emotions through short posts, often revealing deep personal feelings. In light of tragedies like school shootings, analyzing these posts could provide early intervention and promote mental health awareness. AI, particularly using BERT transformer models, has shown the ability to detect emotions such as love, sadness, anger, and fear from text [3].

This project builds and tests an AI-based emotion detection system using BERT and real-world social media data. The model examines each word in a sentence to output the emotion conveyed [2]. We implemented Python scripts and drew inspiration from Google's GoEmotions dataset [1]. While the model accurately identifies single emotions, it struggles with mixed emotions or shorter posts, often defaulting to neutral. Longer posts, over 100 words, provided more context and improved accuracy.

We collected over 2,000 posts from Twitter and Reddit related to depression and anxiety, manually filtering to 600 posts per platform ranging from 100–500 words. Posts were annotated with 4–5 fundamental emotions to evaluate AI accuracy against human judgment. We predict that longer posts enhance emotion recognition, expecting model accuracy of 75–80% for selected emotions. Future improvements include fine-tuning for shorter posts, expanding the model as a mental health chatbot, and possibly creating a browser extension to alert authorities to concerning content.

Challenges included API limitations, mixed emotions, and neutral default predictions. Manual data collection allowed targeted selection of posts that

[*]Copyright is held by the author/owner.

tested the model's capabilities.

In summary, this study highlights both the potential and limitations of transformer-based models like BERT for social media emotion detection. Results show the importance of contextual depth in text for accurate emotion recognition and suggest promising applications in digital mental health support. Future work may explore sarcasm detection to further enhance AI's ability to understand nuanced human emotions.

References

[1] Demszky, D., Movshovitz-Attias, D., Ko, J., Cowen, A., Nemade, G., & Ravi, S. (2020). GoEmotions: A Dataset of Fine-Grained Emotions. `https://doi.org/10.18653/v1/2020.acl-main.372`

[2] Koufakou, A., Garciga, J., Paul, A., Morelli, J., & Frank, C. (2022). Automatically Classifying Emotions based on Text: A Comparative Exploration of Different Datasets. `https://doi.org/10.1109/ICTAI56018.2022.00056`

[3] Teng, S., Chai, S., Liu, J., Tateyama, T., Lin, L., & Chen, Y. W. (2024, 6-8 Jan. 2024). Multi-Modal and Multi-Task Depression Detection with Sentiment Assistance. 2024 IEEE International Conference on Consumer Electronics (ICCE)

Leveraging Multi-Agent AI Architectures and Large Language Models for Stock Price Prediction and Automated Trading[*]

Poster Session

Rachael Poffenberger and Anna Hou
Advisor: Weidong Liao
Department of Computer Science, Mathematics and Engineering
Shepherd University
Shepherdstown, West Virgnia, USA

We present a prototype that investigates how multi-agent AI and Large Language Models (LLMs) can support stock prediction and automated trading. Our system coordinates specialized agents:

1. a sentiment agent that uses an LLM to distill market sentiment from news, forums, and social media;
2. a prediction agent that evaluates price-only models (linear and polynomial regression, decision tree, random forest, LSTM, and transformer) on historical data; and
3. a decision agent that fuses signals to propose trades in a paper-trading simulator.

Using public API to fetch available daily price data and a chosen set of machine learning algorithms, we observe that simple, well-regularized linear regression with rolling features outperforms more complex models for 1–5 day forecasts on noisy equities such as AAPL, while polynomial regression and tree-based models are sensitive to large swings, and LSTM/transformer variants tend to overfit on limited data. Incorporating the LLM sentiment score modestly improves prediction consistency during news-heavy periods and reduces obvious false positives.

[*]Copyright is held by the author/owner.

Our results suggest that a modular multi-agent design combining lightweight statistical models with LLM-based sentiment can yield clearer, more reliable signals than complex deep models alone. We also discuss limitations and outline next steps, including richer feature engineering, broader asset coverage, and live execution safeguards. This work presents a practical and extensible framework for practicing AI programming, as well as teaching and learning AI-driven trading concepts.

Can virtual reality help people with mental illness?: Improving mood and relieving stress[*]

Poster Session

Bill Nguyen
Chestnut Hill College
Philadelphia, Pennsylvania, USA
nguyenb@chc.edu

This research study will investigate how virtual reality (VR) can serve as an effective or complementary tool for mental health professionals alongside their traditional methods in improving mental health, particularly those who have experienced anxiety and depression. Due to recent technological advancements in immersive technology, this study will evaluate the impact that virtual reality-supported therapy can have as an effective method compared to traditional methodologies. By using mixed methods, the research will combine a quantitative assessment with qualitative participant feedback to help understand the efficacy, engagement, and overall value of a VR-based intervention. This will incorporate participant surveys before and after the VR session to obtain knowledge or insight into the participants' mood changes and outlook.

[*]Copyright is held by the author/owner.

Evaluating LMS Usability and Accessibility for Students and Instructors*

Poster Session

Kenisha Thapa and Diya Adhikari
Advisor: Vladislav D. Veksler
Computer Science
Caldwell University
Caldwell, New Jersey, USA
{kthapa1, dadhikari, vveksler}@caldwell.edu

This human-centered study of Learning Management System (LMS) effectiveness presents a multimethod approach involving surveys, heuristic evaluation, Universal Design for Learning (UDL), and accessibility audits of widely used LMS applications: Blackboard Learn, Blackboard Ultra, and Canvas. Our goal was to examine effectiveness and accessibility from both student and instructor perspectives, and to provide best practices for LMS design and development.

We collected survey data capturing experiences of students (n = 45) and instructors (n = 50) with Blackboard. Students generally found routine tasks such as submitting assignments (M = 4.67, mean on a 1–5 scale) or posting to discussion boards (M = 4.15) straightforward, though course layouts often caused navigational overload. Instructors reported greater challenges with quiz creation (M = 3.20), gradebook management (M = 3.98), and module organization (M = 3.48).

Heuristic evaluations confirmed these differences, showing that Blackboard supported basic functions but became confusing for complex activities like grading and module organization. Canvas offered a cleaner, more structured interface for students but introduced workflow inefficiencies for instructors. Accessibility audits (WAVE, axe DevTools) revealed contrasting barriers: Blackboard Ultra relied heavily on ARIA (Accessible Rich Internet Applications) attributes instead of semantic HTML and included unlabeled buttons and missing alter-

*Copyright is held by the author/owner.

native text, while Canvas demonstrated stronger labeling and structure but critical issues in grading and quiz features. UDL analysis highlighted gaps in engagement and system feedback across both platforms.

Overall, Blackboard proved functional for repetitive student tasks, while Canvas provided a streamlined student experience but required refinement for instructor workflow and accessibility. This research underscores the importance of integrating usability, accessibility, and inclusivity into LMS development. Future work will expand surveys and interviews and propose a combined evaluation framework that institutions can adopt to assess and improve LMS effectiveness.

References

[1] Bradley, V. M. (2021). Learning management system (LMS) use with online instruction. International Journal of Technology in Education, 4(1), 68–92. `https://files.eric.ed.gov/fulltext/EJ1286531.pdf`

[2] S. Ghazal, H. Al-Samarraie and H. Aldowah, "I am Still Learning: Modeling LMS Critical Success Factors for Promoting Students' Experience and Satisfaction in a Blended Learning Environment," in IEEE Access, vol. 6, pp. 77179-77201, 2018, DOI:10.1109/ACCESS.2018.2879677

[3] M. Bjornholt Karapetian, "Universal Design for Learning (UDL) and Learning Management System (LMS) Considerations to Facilitate Online Learning Outcomes for Logistics Workforce Development." Order No. 31332835, California State University, Long Beach, United States – California, 2024.

[4] "10 Usability Heuristics for User Interface Design," Nielsen Norman Group. `https://www.nngroup.com/articles/ten-usability-heuristics/`

Using Static Application Security Testing; Combining Security with Development*

Poster Session

Blake Douglas Hatcher
Advisor: Brian Heinold
Mount St. Mary's University
Emmitsburg, Maryland, USA
b.d.hatcher@email.msmary.edu, heinhold@msmary.edu

Application security is increasingly becoming a point of focus for developers who have security sensitive services. According to Verified Market Research, the global DevSecOps market is projected to grow from USD 9.72 billion in 2024 to USD 22.72 billion by 2032, showing a compound annual growth rate of over 13% [1]. Within this sphere one of the most popular tools is Static Application Security Testing (SAST) which is predicted to grow from 0.64 billion dollars in 2024 to 1.24 billion dollars by 2034 [2]. More than 65% of enterprises have already adopted SAST within their DevSecOps pipelines, and cloud-based deployments account for 58% of current usage, underscoring a clear industry trend toward scalable, automated solutions. This poster outlines the integration of SAST within the DevSecOps lifecycle and highlights its benefits, challenges, and practical applications. Over the summer, I worked with the CECOM Software Assurance Lab where Black Duck was leveraged for projects. While specifics are classified, this poster demonstrates how SonarQube, an open-source SAST platform, can provide similar value in academic settings. By embedding SonarQube scans into the development pipeline, developers receive immediate feedback on insecure coding practices, injection flaws, and hardcoded secrets before software reaches production.

The benefits of this integration are significant. Organizations with mature DevSecOps practices resolve vulnerabilities 11.5 times faster than less advanced peers [3], while also reducing remediation costs by addressing flaws early in the lifecycle. In addition, continuous scanning ensures that compliance requirements are consistently met, reducing the burden on security teams. Despite

*Copyright is held by the author/owner.

challenges such as false positives and the need for developer training, the trajectory of adoption is clear: DevSecOps, powered by tools like SonarQube, is rapidly becoming a standard in building resilient, trustworthy software. This poster demonstrates how integrating SAST tools into development not only strengthens security but also accelerates delivery and reinforces confidence in modern IT infrastructure.

References

[1] Verified Market Research. (2024, May). DevSecOps Market Size and Forecast. `https://www.verifiedmarketresearch.com/product/devsecops-market`

[2] Global Growth Insights. (2024, June). Static Application Security Testing (SAST) Software Market Report 2024–2034. `https://www.globalgrowthinsights.com/market-reports/static-application-security-testing-sast-software-market-103342`

[3] Veritis. (2024, July). 14 Statistics That Shed Light Upon DevSecOps Opportunities and Challenges. `https://www.veritis.com/blog/14-statistics-that-shed-light-upon-devsecops-opportunities-and-challenges`

Detecting Plagiarism through Visualization of Coding Activity Logs*

Poster Session

Keepa Maharjan[1] and Nievanik Thapa Shrestha[2]
Advisor: Vladislav D. Veksler[1]
[1]Caldwell University
Caldwell, New Jersey, USA
[2] University of Texas Arlington
Arlington, Texas, USA
{kmaharjan, vveksler}@caldwell.edu, nxt3969@mavs.uta.edu

The advancement of AI-assisted coding tools has made plagiarism detection in programming assignments increasingly challenging. Traditional plagiarism detection approaches, which typically analyze only the final submitted code (e.g., similarity metrics and classifiers), often fail to capture behavioral cues embedded in edit logs. This project investigates visualization-based techniques to expose these cues and assist plagiarism detection.

Using Python assignment logs collected via the LectureAssign VSCode extension, five types of visualizations were generated: (i) edit ratio plots, (ii) size versus cursor position charts, (iii) burst and pause visualizations, (iv) timeline heatmaps of editing activity, and (v) smoothed cumulative character count series. Playback within VSCode confirmed that the observed patterns corresponded to authentic coding behavior, with smoothing particularly enhancing the interpretability of dense cumulative character count data.

Effectiveness was assessed through a survey of computer science students, who ranked visualizations based on their usefulness for detecting plagiarism. The size versus cursor position chart proved to be the most informative, as cursor movement consistently indicated copied work. Timeline heatmaps and smoothed cumulative character counts were also considered valuable, while other approaches offered limited additional insight.

*Copyright is held by the author/owner.

These findings suggest that cursor-based interactions constitute key signals for plagiarism detection. Future work will refine smoothing methods, extend them to additional visualization types, and develop automated anomaly detection tools capable of seamless integration into grading workflows, providing scalable and transparent support for academic integrity.

RAG-Based Privacy Policy Analysis for Mental Health Apps[*]

Poster Session

Muhan Ding, Ziyu Kang, Erchen Qu and Jie Xu
Advisor: Yanxia Jia
Arcadia University
Glenside, Pennsylvania, USA
{mding_02, zkang@arcadia.edu, equ, jxu_02, jiay}@arcadia.edu

Privacy policies play a vital role in explaining how mobile apps collect, use, store, and share user data. For mental health apps, safeguarding privacy is especially critical due to the sensitivity of the information involved. However, these policies are often lengthy and complex, making manual analysis both time-consuming and error-prone for researchers and practitioners. Large Language Models (LLMs) offer a promising avenue for automating policy analysis, yet their reliability is limited by hallucination.

We developed a Retrieval-Augmented Generation (RAG)-based Question Answering (QA) system, implemented with Haystack, to analyze privacy policies of mental health apps. The system extracts answers to privacy-related questions from the privacy policy documents, for example, whether the app collects user data, what kinds of data are collected, whether user data is shared with third parties, and which third parties receives it. By integrating advanced RAG techniques and evaluation methods with Haystack, our approach reduces hallucination risks while improving the accuracy and trustworthiness of automated policy analysis.

[*]Copyright is held by the author/owner.

Tranquilify: Selective Sound Suppression with Embedded Hardware[*]

Poster Session

Katherine Connelly
Advisor: Jeffrey Bush
Moravian University
Bethlehem, Pennsylvania, USA
`{connellyk, bushj}`@moravian.edu

This project aims to develop wearable technology specifically for assisting individuals suffering from misophonia, a condition in which certain sounds trigger intensely negative emotional, physiological, and behavioral responses. There is limited research and minimal knowledge on this matter, and the prevalence of triggers like chewing, coughing, and sniffling presents continuous challenges for those affected. To address the current gap in effective coping strategies, the intention is to create headphones that block all external sounds and then reproduce filtered, trigger-free audio to the wearer

To suppress specific sounds, an embedded system will be used within the headphones. Audio is collected with two electret microphones spaced for binaural processing, filtered using machine learning on the embedded system, and played back with standard earbuds. This method enables the headphones to capture audio from their surroundings, eliminate trigger sounds, and provide real-time playback to the user.

The objective is to design hardware suitable for running the algorithm with minimal latency (50 milliseconds or less), which is crucial for the user's ability to engage normally with their environment. Additionally, emphasis will be placed on gathering data to utilize in training the machine learning algorithm, with a focus on versatility to make the headphones suitable for a range of different sound triggers.

[*]Copyright is held by the author/owner.

Machine Learning Approach to Real-time Selective Sound Suppression[*]

Poster Session

Yousuf Kanan
Advisor: Jeffrey Bush
Moravian University
Bethlehem, Pennsylvania, USA
{kanany, bushj}@moravian.edu

This project aims to design and implement a real-time machine learning model that filters out specific sound triggers associated with misophonia, such as coughs and throat clearing, while allowing all other environmental sounds, such as conversations, to pass through. Traditional noise-canceling systems block entire frequency ranges or crude patterns, but using a machine learning model trained to recognize and suppress only the sounds identified as triggers. The software component of this project consists of separating sounds from the environment into multiple sources using DUET, a sound separation algorithm [1]. Then, running each individual sound through a machine learning model to determine whether or not the sound is a trigger. The non-trigger noises will then be reconstructed, allowing them to be played back to the user.

A major challenge in this project lies in hardware limitations; for the brain to remain unaffected by lag, system latency must stay less than 50 ms due to synchronization between audio and visual stimuli [2]. This constraint requires balancing computational power with a compact form factor. This project is seeking to create assistive technology that improves the quality of life for those with misophonia and other auditory processing disorders, while advancing the field of embedded machine learning.

References

[1] Rickard, S. (2007). The DUET blind source separation algorithm. In Blind

[*]Copyright is held by the author/owner.

speech separation (pp. 217-241). Dordrecht: Springer Netherlands.

[2] Veluri, B., Itani, M., Chan, J., Yoshioka, T., & Gollakota, S. (2023, October). Semantic hearing: Programming acoustic scenes with binaural hearables. In Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (pp. 1-15)

Monitoring Online Misinformation Using LLM-Based Models*

Poster Session

Sarah Ainani
Advisor: Navya Martin Kollapally
Department of Computer Science and Technology
Kean University
Union, New Jersey, USA

The proliferation of misinformation on digital platforms has become a significant concern for individuals seeking reliable information, particularly in critical situations. Messages, broadcasts, weather reports, and government announcements could be manipulated, influencing the public's perception. Social media networks, with their evolving content reaching a broad user base, have become common ground for the circulation of misleading narratives and ideas. Regardless of intent, its consequences have proven severe, with disruption of emergency responses and amplifying fear and division within society. Crisis events such as wildfires highlight the importance of addressing misinformation. During natural disasters, accurate and precise information is key to effective public safety measures. False evacuation notices, false location, bold claims with no proven evidence can move civilians into hazardous areas and delay compliance with instructions given. Vosoughi, Roy, and Aral [1] demonstrated that false news spreads rapidly on twitter and shows a broader reporting compared to factual reporting, which is 70% more likely to be retweeted. This further complicates the role of emergency services, senators, leaders, and public figures.

Traditional misinformation detection models that use keyword-based filters or logic classifiers, these models offer standard effectiveness when applied to the Twitter content. These models tend to classify sarcastic or partial truths as not high in confidence, meaning not likely to be factual. In contrast, transformer-based Large Language Models such as RoBERTa offer context to their reasoning, which allows for interpretation and tests the model on its knowledge. This

*Copyright is held by the author/owner.

study evaluates whether such LLM models can detect fatal events such as the California wildfires and examines the likelihood of misinformation from tweets.

This study highlights that a content aware LLM model on an annotated wildfire Twitter dataset will significantly outperform traditional classifier models in detecting misinformation. The study further shows that misinformation in critical events shows a linguistic pattern that the model picks up, such as conspiracy theories or exaggerated claims, which is captured by the transformer-based classifier.

The foundation of this study was the gathering of tweets from the California wildfire timeline. Tweets were collected between January and April 2025, covering political commentary, emergency news reports, and live updates. A total of 156 tweets were chosen reflecting the general landscape of the twitter website. Attention was given to balance the dataset by organizing them by categories, opinion, news, and personal bias, to keep the data represented and identified, rather than leaning on factual samples. For example, tweets including verified updates like a Sky News report on wildfire containment numbers. It was predicted that transformer based LLM models would achieve a higher accuracy race compared to baseline classifiers. Results confirmed this hypothesis, and the logistic regression classifier achieved a 68% accuracy on tweets with truth but contain personal opinions, while the rule-based model scored a 58%. The RoBERta base misinformation bot did deliver the best results as predicted, giving a 85% confidence rate, with news article linked tweets were screened as true, while heavily personal statements were caught at a higher rate, where the other models remained neutral. Qualitative analysis shows that the strengths of large language models are in handling complex emotions versus factual statements. For example, one tweet from a user claims, "This is how quickly the fire ran completely out of control. California Governor Gavin Newsom and Mayor Karen Bass failed to ensure the state was prepared; even worse, they cut funding." This received a 0.19 confidence, making the likelihood of this being false, and the twitter user stated conspiracy theories or stating broad claims.

These results confirm the predicted outcome they also support the findings of Thorne [2], who emphasized that verification and context are essential for misinformation detection. Similarly, Hussain [3], who discussed that the transformer-based models outperform traditional models.

The broader societal relevance of this research, however, relies on its application to public media forums during crisis events. In wildfire scenarios such as the California Wildfires in early 2025, misinformation can cause residents to panic or underestimate the risk. False claims about where the fire was started or how many casualties have been reported create a rift in communication. This also puts emergency responders at risk due to negligence or lack of reliable information, creating a need to assist more people. A component of the misinformation bot highlights a common critic of artificial intelligence models and their black and white classifications. By providing justifications, clarifications, and context, this proves essential for public agents and social media platforms to further the technology for detecting misinformation.

Beyond accuracy, the significance is in the ability of these models to determine human attitudes and provide an explanation of the output. Future work would include multilingual datasets and multimodal misinformation features for videos. By highlighting the misinformation detection during extreme crises, this study shows the potential of the LLM approach to give out clear and correct information.

References

[1] Vosoughi, S., Roy, D. & Aral, S. (2018). The Spread of True and False News Online. Science, 359(6380), 1146–1151. `https://doi.org/10.1126/science.aap9559`

[2] Thorne, J., Vlachos, A., Christodoulopoulos, C., & Mittal, A. (2018, June 1). FEVER: a Large-scale Dataset for Fact Extraction and VERification. ACLWeb; Association for Computational Linguistics. `https://doi.org/10.18653/v1/N18-1074`

[3] Hussain, F. G., Wasim, M., Hameed, S., Hameed, S., Asim, M. N., & Dengel, A. (2025, January). (PDF) fake news detection landscape: Datasets, data modalities, AI approaches, their challenges, and future perspectives. Fake News Detection Landscape: Datasets, Data Modalities, AI Approaches, Their Challenges, and Future Perspectives.

Emotion Detection Via Text*

Poster Session

Ricardo Urbaez
Advisor: Navya Martin Kollapally
Department of Computer Science and Technology
Kean University
Union, New Jersey, USA

The growth of user-generated text on digital platforms has produced an immense amount of emotionally rich content. Understanding these emotions is important for mental health research, human–computer interaction, conversational systems, and digital well-being. Yet, emotion detection remains one of the most difficult challenges in natural language processing because online language is shaped by slang, sarcasm, mixed tones, and cultural nuance. This project asks whether large language models (LLMs) can reliably capture how people actually express emotions in everyday online communication.

To explore this question, we developed a reproducible pipeline in Python consisting of four main tasks: collecting Reddit data via the official API, preprocessing text to remove inconsistencies, applying LLMs for classification, and mapping outputs to structured emotion labels. We used the GoEmotions dataset, which defines twenty-seven fine-grained emotions, as the foundation for classification and evaluation. From more than two thousand Reddit posts, we created a benchmark of six hundred posts from mental health communities such as r/depression and r/SuicideWatch. Each post was labeled with up to five emotions by two annotators, with disagreements resolved through discussion to ensure reliability.

Preliminary findings show that models captured frequent emotions such as admiration, approval, and gratitude, reflecting the supportive tone of these communities. However, rare emotions like pride and grief were harder to detect. The models performed well on short, direct statements but struggled with sarcasm, idiomatic expressions, and posts containing mixed emotional signals. These limitations underscore the challenges of applying LLMs in real-world

*Copyright is held by the author/owner.

contexts. The next phase involves formal evaluation using precision, recall, F1 scores, and confusion matrices. We will also compare models such as Qwen, Claude, and Gemini using the same benchmark. This work contributes a transparent framework for testing and highlights both the potential and limitations of LLMs in emotion detection.

Phishing in Academia[*]

Poster Session

Isha Salma Renner
Advisor: Hoda El-Sayed
Department of Computer Science
Bowie State University
Bowie, Maryland, USA
renneri0901@students.bowiestate.edu, helsayed@bowiestate.edu

This research investigates user vulnerability to phishing attacks in academic environments and explores AI-powered strategies for prevention. It synthesizes findings from behavioral and psychological studies to understand why students and faculty remain susceptible to phishing attempts, even with existing training and technical defenses. Special attention is given to the role of generative AI, which enables adversaries to craft realistic and adaptive phishing emails that evade traditional detection mechanisms.

The project evaluates current technological defenses and highlights their limitations against AI- generated phishing. To address these challenges, it proposes a dual strategy: (1) tailored, interactive educational interventions that account for diverse learning styles and promote a culture of proactive reporting, and (2) an AI-based phishing detection model built using a Naive Bayes classifier with TF-IDF features and heuristic checks.

The expected poster will illustrate the problem of phishing in academia, the added risks posed by AI-driven attacks, and proposed multi-layered solutions combining human and technological defenses. Future directions include refining detection accuracy, expanding phishing datasets with AI-generated samples, and developing gamified cybersecurity training modules.

[*]Copyright is held by the author/owner.

Teaching Data Analytics and Data Mining Through U.S. Zip Code Database: A Hands-On Approach to Big Data Exploration[*]

Poster Abstract

Ching-yu Huang
Department of Computer Science & Technology
Kean University
Union, New Jersey, USA
chuang@kean.edu

The U.S. Zip Codes Database offers a rich and diverse collection of demographic, economic, and social data spanning 41,561 zip codes and 86 variables. This study explores how this dataset can serve as a powerful educational tool for teaching students the fundamentals of data analytics and data mining. By working with real-world big data, students gain hands-on experience in data preprocessing, exploratory data analysis (EDA), and data mining techniques, equipping them with industry-relevant skills.

A core component of this study involves demonstrating how to write SQL queries to extract, transform, and load (ETL) data from large tables into structured datasets suitable for analysis. Students will learn how to filter and aggregate data, join tables efficiently, and apply various transformation techniques to prepare data for visualization and statistical modeling. Through practical exercises, they will develop a deeper understanding of data wrangling, query optimization, and database management.

Furthermore, this study introduces students to key data mining techniques, including clustering, correlation analysis, outlier detection, and predictive modeling. By applying these methods, students will uncover meaningful patterns and trends within demographic and socioeconomic data, gaining insights into income disparities, population distributions, housing characteristics, and other critical factors. The integration of visualization tools will enable students to

[*]Copyright is held by the author/owner.

present their findings through interactive charts, maps, and dashboards, reinforcing their ability to communicate data-driven insights effectively.

By leveraging real-world data and hands-on learning, this approach enhances students' analytical thinking, problem-solving abilities, and technical proficiency. The study highlights the effectiveness of experiential learning in data science education, demonstrating how structured engagement with large-scale datasets can prepare students for careers in data-driven industries such as business intelligence, urban planning, and public policy.

Keywords: Big Data Analytics, Demographic Analysis, Correlation, Clustering, Outlier, Data Mining

Are You Ready for the Era of AI in Computer Science Education?[*]

Poster Abstract

Chao Chen
Waynesburg University
Waynesburg, Pennsylvania, USA
charles.chen@waynesburg.edu

Generative AI tools are rapidly transforming teaching approaches and the learning experience in computer science education. This poster examines how generative AI can be integrated into teaching methods, including programming instruction, homework feedback, and personalized learning support, aiming to improve both teaching quality and learning effectiveness.

[*]Copyright is held by the author/owner.

Towards a Global Standard for Artificial Intelligence Literacy[*]

Poster Abstract

Donna Schaeffer[1], Patrick Olson[2]
[1]Marymount University
Arlington, VA 22207, USA
[2]National University
San Diego, CA 92123, USA
donna.schaeffer@marymount.edu, polson@nu.edu

In the 1980s, the concept of information literacy, the need for workers to effectively use information resources, was widespread in the educational ecosystem. Organizations like the American Library Association and UNESCO have significantly promoted its importance in education and society. Information literacy is the precursor to other concepts that demand literacy. For example, we have seen digital and/or media literacy programs in the European Union (EU) and the United Kingdom.

In 2025, the absence of global standards for measuring artificial intelligence (AI) literacy could lead to a significant gap in understanding and using AI responsibly. AI literacy is the knowledge, skills, and mindset required to understand, interact with, and critically evaluate AI systems. It enables individuals to use AI responsibly, assess its impact, and make informed decisions in an AI-driven world. The EU AI Act, the world's first comprehensive AI regulation, has provisions related to AI literacy, but does not mandate programs. Several countries have developed AI education for all school levels, and offer programs for workers and the public. In the US, several states have added AI literacy to their curriculum standards but there are no federal mandates.

These global disparities highlight the need for consistent global standards.

[*]Copyright is held by the author/owner.

On The Accessibility of Computer Science for Non-English Speakers*

Poster Abstract

Michael Wehar and Lizbeth Zarate-Hernandez
Swarthmore College
Swarthmore, Pennsylvania, USA

We investigate whether English-based programming languages cause challenges for non-English speaking learners and possible ways to help and support learners. In particular, we focus our attention on native Spanish speakers.

Through this investigation, we first identified challenges for non-English speakers to learn English-based programming languages. These challenges include (1) inconsistent translations of keywords and programming patterns, (2) limited translations of higher level concepts, and (3) incompatible grammar structures. Secondly, we surveyed programming languages and libraries that are Spanish-based such as JaibaScript, Latino, PseuToPy, and Tango. All of these languages have faced some degree of difficulty when it comes to adoption. We propose that adoption could be improved with (a) eliminating the need to download a new compiler or runtime environment, (b) implementing better compatibility with English-based languages including standardization of keywords, patterns, and conversions, and (c) establishing English and Spanish descriptions of all keywords, patterns, and programming concepts.

Finally, we initiated a multipronged attempt to enable and support future projects that benefit Spanish speaking learners of web development technologies including HTML and JavaScript. In particular, we developed a lightweight JavaScript-based library called HabloCode that supports automatic translation of HTML tags, attributes, and special values. This library allows for the creation of web-based language variants that will run in web browsers without the need to download any compiler, plugin, or runtime environment. Furthermore, we carefully listed official JavaScript keywords, global functions, standard built-in methods, and properties from W3C and ECMAScript documentation. We then began translating these technical terms from English to Spanish. Along

*Copyright is held by the author/owner.

with each translation, we include a description in both English and Spanish. We hope to make these translations available to the public with the ability for developers to propose improvements and collaborate on standardization across English and Spanish languages.

A Social Media Analysis of the July 2025 Bangladesh Uprising*

Poster Abstract

Maysha Fahmida[2], Md Kamruzzaman Sarker[1], Fahmina Nur Salma[1]
[1]Bowie State University
Bowie, Maryland, USA
[2]United International University
Dhaka, Bangladesh

This study explores the public discourse surrounding the July Uprising of Bangladesh as represented on Reddit, a global social media platform known for its open, user-driven conversations. The uprising, marked by widespread protests, civic unrest, and government responses, sparked significant attention both within Bangladesh and among the international community. By analyzing user-generated content—including comments, posts, and discussions from relevant subreddits such as r/Bangladesh—this research investigates the evolving sentiment, key narratives, potential misinformation, and broader global perceptions associated with the event.

Employing a combination of natural language processing (NLP) techniques, sentiment analysis, and topic modeling, the study reveals how digital communities construct and disseminate meaning around politically charged events. Special attention is paid to the temporal dynamics of discourse, the influence of diaspora voices, the spread of factual versus misleading information, and the framing of protest-related content.

This research contributes to the fields of digital political sociology, computational social science, and media studies by demonstrating how global platforms like Reddit function as transnational spaces for activism, solidarity, and information exchange. It further underscores the importance of computational methods in capturing the nuances of public opinion and collective memory formation in the age of digital protest.

*Copyright is held by the author/owner.

Time Synchronization for Real-Time Multi-User Video Annotation: A QR Code and Bluetooth Approach*

Poster Abstract

David Grant Cooper
Department of Computer Science
West Chester University
West Chester, Pennsylvania, USA
dcooper@wcupa.edu

Real-time audiovisual labeling systems are an unrealized need for cost effective video annotation of human activity. In an educational setting, this labeling is used in classrooms to collect information about engagement and activities while performing classroom tasks. Audiovisual labeling is typically done on pre-recorded video rather than during the activity, and the extra overhead of labeling the video often leaves many recordings without annotation. Recent advances of self labeling during recordings of a single person are not adequate for annotation in a group setting, and advances in labeling multi-person recordings use off site labelers requiring high bandwidth networking. A cost effective recording tool for more natural settings should allow multiple people to perform real time labeling independently during the recording. This requires timing synchronization across devices to ensure accuracy in resource-constrained environments.

We propose a novel method for real-time video labeling systems that leverages QR codes displayed on mobile devices to add context to labels sent over Bluetooth LE to achieve time synchronization with minimal drift. This approach departs from traditional clock synchronization protocols and addresses the limitations of resource-constrained environments. Additionally, we discuss the data annotation process in educational settings, utilizing Android and iOS devices as BLE beacons to send annotations to the central server recording the video. We demonstrate the efficacy of our proposed system by achieving

*Copyright is held by the author/owner.

accurate synchronization strategies that have between 0.2 seconds and 2.0 seconds of drift, enhancing the efficacy and accuracy of real-time video labeling for a variety of user activities and using labelers. In addition to describing the time synchronization of a data collection tool, we illustrate this audiovisual labeling system as a research platform where computer science students can explore user interface design, usability, message encoding, and audio and video processing techniques.

ZeroHero: An AI-Driven App for Plastic Bag Recycling*

Poster Abstract

Kenneth Shue[1], Eliana X. Wang[2], Lily R. Liang[3], Jane Shue[1]
[1]Growing & Giving Club, CAPA-MC
Germantown, Maryland, USA
[2]Potomac Community 4-H Club
Potomac, Mryland, USA
[3]Computer Science and Information Technology Department
University of the District of Columbia
Washington, DC, USA
{kshue7, jzshue988, ewanginspiration}@gmail.com, lliang@udc.edu

The health impact of microplastics has raised growing concerns recently. A significant portion of the microplastic issue consists of plastic bags and films. Each year, up to one trillion plastic shopping bags are produced globally, according to an article published by the United Nations Environment Programme in December 2021. To encourage plastic bag recycling, it is essential to develop technology that helps the public identify recyclable bags correctly and efficiently.

It is challenging to automate plastic bag recognition, though. Plastic bags often lack resin identification codes or symbols, such as type 2 (HDPE) or 4 (LDPE), making it difficult to determine if they are recyclable. Visual features, such as shape and color, even though important in object recognition, are not sufficient to correctly recognize the categories of plastic bags and films. Bags and films can be transparent and have various and changeable shapes. The objects enclosed in transparent bags and films could be recognized instead.

In this research, we

- Investigated the current AI tools' capability to recognize plastic bags and wraps from images with and without text prompts;

*Copyright is held by the author/owner.

- Proposed a new AI-based approach for recyclable plastic bag and film recognition, which integrates textual context information into computer vision to focus the recognition on bags rather than their contents or other objects in the same images; and
- Developed a software application of this approach and tested it on 44 images of samples of shopping bags and produce bags, achieving 80% overall accuracy.

Extracurricular Service Learning for First-year Commuter Students[*]

Poster Abstract

Lily Liang, Briana Wellman, Carlos Sac Mendoza, Uzma Amir
Computer Science and Information Technology Department
University of the District of Columbia
Washington, DC 20008
{lliang,briana.wellman,carlos.sacmendoza,uzma.amir}@udc.edu

Service learning, despite its benefits, is challenging for first-year computing students to take advantage of. Traditionally part of a curriculum, it is usually not offered at the first-year level. First-year students, as newcomers in computing communities, have fewer connections, compared with established community members, and thus have limited access to extracurricular service opportunities in computing. First-year commuter students face additional challenges, such as longer travel time to campus, longer work hours, or extra family obligations. To motivate and engage first-year commuter students through service learning, it is crucial to create service opportunities that are flexible and appropriate to their knowledge and skill.

We implemented an extracurricular service-learning program with easy entry and a minimum, flexible time commitment for the first-year computing students on our urban commuter campus. We used the following strategies:

1. Creating roles with easy entry by:
    - Choosing technical skills that first-year students can quickly learn to assist others, such as the "Teachable Machine," an online platform that enables users to quickly program a machine learning model and experiment with images for classification.
    - Assigning first-year students as workshop facilitators for high school outreach and providing upper-level students experienced in outreach activities as peer mentors and team leads.

[*]Copyright is held by the author/owner.

2. Make the experience:
    - Valuable. We selected a high-demand topic for students to be trained to serve, and incorporated observing and interacting with upper-class students.
    - Fun. We provided refreshments, festival decorations, and souvenirs at both our training and service events.
    - Convenient. We arranged training at students' convenience and provided asynchronous training with videos to accommodate students' busy schedules.

Our preliminary data show that the program increased the first-year students' engagement and motivate them to pursue their majors. The students reported that the activities were "engaging," "fun," "eye-opening," "helpful," and motivating.

Generative AI, LLMs, and Agentic AI: Catalyzing Cybersecurity Education and Workforce Development[*]

Poster Abstract

Weidong Liao, Osman Guzide
Department of Computer Science, Mathematics and Engineering
Shepherd University
Shepherdstown, West Virginia, USA

The evolving landscape of AI, including Generative AI, Large Language Models (LLMs), and autonomous Agentic AI, offers transformative potential for cybersecurity education and workforce development from high school, community college, to four-year college levels. AI agents, leveraging LLMs as their core reasoning engine, perceive environments, set goals, and execute multi-step actions through reflection, planning, tool use, and multi-agent collaboration. These technologies enhance learning and training by enabling hyper-personalized learning experiences with dynamic curricula, real-time threat intelligence, and code vulnerability detection. Agentic AI serves as an intelligent, personalized tutor, offering real-time feedback on critical skills like coding, security, incident response, and digital forensics. Multi-agent systems provide realistic cybersecurity simulations and consistent automated assessment, such as multi agent automated essay scoring, demonstrating improved consistency over stand-alone LLMs. They can also detect plagiarism and monitor student achievement.

However, significant challenges persist. A critical AI literacy gap among educators is prevalent, with 40% identifying as beginners. Ethical concerns include algorithmic bias, data privacy, student over-reliance leading to cognitive offloading and uncritical reproduction of AI-generated content, and dual-use risks (AI for offensive cyberattacks). The advent of quantum computing also necessitates quantum-resilient defense strategies against emerging cryptographic threats.

[*]Copyright is held by the author/owner.

Successful integration demands strategic, human-centered implementation, prioritizing well-balanced curriculum design, comprehensive faculty training, and community outreach. All these efforts will together cultivate a highly skilled, adaptive, and ethically conscious cybersecurity talent pipeline. This poster presents the outcomes and experiences of our investigation.

Impact of Buffer Size vs. Routing Protocol on Data Delivery in Delay-Tolerant Networks[*]

Student Research Paper

Shelby Sanders, James Donohue,
Emilie Bonhivert and Xinliang Zheng
Department of Computer Science and Information Technologies
Frostburg State University
Frostburg, MD 21532
{slsanders0,emiliebonh,jpdonohue0,xzheng}@frostburg.edu

In the aftermath of natural disasters, conventional communication infrastructure often fails, making Delay-Tolerant Networks (DTNs) a critical alternative for enabling connectivity in disrupted environments. DTNs use a store-carry-forward approach to overcome intermittent connectivity, but their performance is influenced by both routing protocols and buffer size. While prior studies have suggested that buffer size has a greater impact on message delivery success than routing strategy, these findings have largely been based on comparisons between similar flooding-based protocols. This study investigates whether buffer size or routing protocol selection has a greater effect on data delivery success in post-disaster DTNs. Using realistic firefighter mobility traces from wildfire scenarios and the Opportunistic Networking Environment (ONE) simulator, we found that routing protocol choice also have a significant impact on delivery probability compared with buffer size. Our findings underscore the importance of protocol design in resource-constrained DTNs.

[*]Copyright is held by the author/owner.

Phishing in Academia: Understanding User Vulnerability and AI-Powered Preventive Strategies*

Student Research Paper

Isha Salma Renner
Department of Computer Science
Bowie State University
Bowie, MD 20715
renneri0901@students.bowiestate.edu

Phishing attacks represent a critical threat to academic communities, exploiting user vulnerabilities and undermining cybersecurity measures. This literature review investigates users' susceptibility to phishing schemes, particularly in the context of AI-driven attacks. It examines the psychological and behavioral factors contributing to user exploitation and the emerging trend of AI-based phishing tactics. Furthermore, the review highlights prevention strategies, including AI-enhanced cybersecurity awareness training and advanced detection mechanisms. This review outlines the current landscape by synthesizing recent findings in user behavior, phishing methodologies, and defensive frameworks. It identifies essential avenues for future research to bolster cybersecurity in academic settings.

*Copyright is held by the author/owner.

Cassie: A Fully Offline, Persistent-Memory AI Assistant for Secure, Cloud-Free Deployment[*]

Student Research Paper

James Harada[1,2,3] and Charles Harada[1,3]
[1] East Penn School District,
Emmaus PA, 18049
[2] Lehigh Carbon Community College,
Schnecksville, PA 18078
[3] Solace AI LLC, Student Research Division,
Emmaus PA 18049
{jgharada,coharada}@outlook.com

Most modern AI assistants rely on cloud infrastructure to function, which raises significant concerns for institutions that require regulatory compliance, privacy, and local data sovereignty. This project introduces Cassie, a fully offline AI assistant designed to operate entirely without internet access while preserving long-term memory using a structured, queryable local database. Cassie is deployed through a hybrid setup: a LLaMA-based language model runs on a 2020 MacBook Air for all AI inference, while a Wi-Fi tethered Samsung S23+ Android smartphone handles the user interface and structured memory via SQLite. All conversations, preferences, and contextual state are stored locally, persist across reboots, and are never transmitted to external servers. Through weeks of live testing, Cassie demonstrated consistent memory recall, contextual adaptation, and reliable operation across sessions in a fully air-gapped environment. Her architecture enables deployment in sectors where cloud-based AI is not permitted, such as education, healthcare, defense, and finance. Cassie was developed under the Solace AI llc Student Research Division, using a method licensed from Solace AI llc. This paper describes the student-led implementation and testing of Cassie, an AI system built under that license. All development and write-up were conducted by students within the Student Research Division. To our knowledge, no existing system integrates fully offline

[*]Copyright is held by the author/owner.

inference, persistent user-specific memory, cross-device modularity, and complete network isolation using consumer hardware. This project demonstrates that ethical, secure, and practical AI deployment is achievable without relying on cloud services or surveillance-based infrastructure.

Constructing a High-Quality Epidemic-Related Reddit Dataset*

Student Research Paper

Olumide Aisida, Fahmina Nur Salma, Desia Craig, Angemaxime Tezai and Kamruzzaman Sarker
Department of Computer Science
Bowie State University
Bowie, MD 20715
salmaf0716@students.bowiestate.edu

This paper explores the integration of machine learning with domainspecific knowledge to enhance the accuracy and reliability of epidemic prediction models. Traditional models predominantly rely on historical case data, often overlooking critical contextual factors such as government interventions, symptom correlations, and disease transmission patterns. To address this limitation, we plan to construct a knowledge graph using Neo4j that will link key entities, including diseases, locations, symptoms, and preventive measures. Although the knowledge graph is yet to be implemented, it represents a significant component of our ongoing work and is expected to contribute structured, semantically rich features to future predictive models. In the current phase of the study, we focused on collecting and analyzing public sentiment data from Reddit. Using custom Python scripts, we curated a high-quality dataset by extracting posts containing epidemic-related terms. Sentiment analysis was then applied to assess public emotional responses and perceived coping capacity during outbreaks. Preliminary findings suggest that integrating contextual signals—such as public sentiment—can improve the interpretability and potential performance of epidemic forecasting models. This work lays the groundwork for future enhancements through knowledge based feature engineering and highlights the value of contextual information in data-driven epidemic modeling.

*Copyright is held by the author/owner.

AI for Everyone: Building Generative AI Literacy Across Disciplines *

Yang Wang[1], Yuehan Yin[1], Lisa Jarvinen [1], Frank Mosca [1]
[1] La Salle University
Philadelphia, PA 19141
wang@lasalle.edu

Abstract

Despite the growing recognition of the importance of interdisciplinary AI education, most existing curricula remain narrowly focused on computer science related disciplines, leaving students from non-technical fields under-prepared to engage with AI (and particularly generative AI given its transformative impacts and broad social relevance). This work makes several key contributions to fill this gap. First, we define the concept of generative AI literacy and propose an intersectional course framework that integrates essential technical knowledge with domain-specific applications and societal implications. Second, we present a fully implemented course design that balances foundational concepts with practical, hands-on experiences using accessible tools. Third, our approach emphasizes ethical, legal, security, and social dimensions of generative AI, aligning with the principles of liberal arts education while preparing students for real-world challenges. We also detail the strategies used to overcome key implementation challenges, offering a replicable model for institutions seeking to integrate similar initiatives.

1 Introduction

Despite the growing need and interest in interdisciplinary AI education [7], there remains a significant gap in delivering AI literacy to students across all

*Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

majors, including healthcare, business, education, the arts, and beyond. In higher education, AI curricula have traditionally been developed primarily for computer-related disciplines [1], often requiring a strong technical background. The recent rise of generative AI, however, has fundamentally removed the technical entry barrier by enabling non-experts to interact meaningfully with AI systems through natural language and intuitive interfaces. Most university curricula have not kept pace with this shift, leaving students under-prepared to critically and effectively use AI in their respective fields. This work aims at filling this gap and presenting an AI course design for all majors. The novel and unique features of our course design include the following. First, we adopt a technology-informed approach that introduces key technical foundations at a conceptual, just-enough level to deliver essential understanding without overwhelming details. Second, among the broad fields of AI, our design centers on generative AI models and related foundational concepts (e.g., neural networks), given their accessibility and far-reaching impact across all academic disciplines. Third, we provide a balanced integration of theoretical foundations and practical applications, enabling students to develop a well-rounded and applicable understanding of AI. Fourth, our design offers an integrated exploration of the AI landscape, with particular emphasis on cross-disciplinary considerations such as the ethical, legal, security, safety, and societal implications of AI. Ultimately, this course equips students to be AI-ready for academic pursuits, and prepares them to thrive in the workforce. To the best of our knowledge, this work represents the first initiative in the literature to adopt a generative-AI-centered approach for AI literacy education.

The rest of this paper is organized as follows. Section 2 summarizes the related literature on AI literacy. Section 3 formally elaborates the concept of Generative AI literacy. In Section 4, we present the detailed course implementation design. In Section 5, we present the major implementation challenges and resolving strategies. Finally, Section 6 concludes this paper.

2 AI Literacy

AI literacy is commonly defined as a collection of AI-relevant competencies. The authors of [8] conducted an exploratory review to identify key competencies that enable individuals to "critically evaluate AI technologies, communicate and collaborate effectively with AI", and use AI as a tool in different settings. However, despite the concreteness and comprehensiveness of this definition, many of the identified competencies extend beyond the technical background typically held by students outside of computer science disciplines. The authors of [12] investigated a core AI competencies framework tailored for non-computer science undergraduate students based on semi-structured interviews with professionals working at the intersection of AI and other disciplines. Due to the diverse backgrounds and the limited pool of the interviewees, however, the resulting competencies lack a high degree of "consensus" and may not fully represent the perspectives of professionals from other fields, such as biology and chemistry. The work by the authors of [11] explored AI literacy for broader audiences beyond higher education, incorporating recent advancements in generative AI through a comprehensive literature synthesis. The authors identified key pillars of AI literacy with knowledge sets that can be tailored to different audiences ranging from K-12 students to the general public.

We classify and generalize the methodologies for AI literacy across all majors from the literature into three main categories. The first category is the discipline-tailored approach, which involves creating customized curricula for each discipline to address AI-relevant topics within that field. For example, the authors of [7] proposed a 36-credit interdisciplinary AI learning module consisting of 13 courses, specifically designed for students from eight disciplines: Computer Science, Law, Education, Humanities, Social Science, Information Systems, Biomedicine, and Nursing Sciences. Second, the technical-focused major-oblivious approach emphasizes the technical experience and understanding of AI processes, irrespective of the student's major. For instance, the authors of [6] presented an experience report on an introductory AI course designed for students from various disciplines. The course covered technical topics, such as analyzing AI applications, machine learning and big data, computer vision, and natural language processing. Additionally, students engaged in hands-on AI practices, including training machine learning algorithms and developing smart assistants through coding. This approach aimed to convey AI literacy through direct technical engagement and hands-on experience with AI processes, complemented by stimulated discussions based on these experiences. Third, the brute-force approach spans AI topics across various interdisciplinary fields. The works mentioned above ([8], [12], and [11]) fit into this category, as they attempt to identify a *union* of the set of skills necessary for AI literacy.

Our work in this study distinguishes itself from the existing literature in several key aspects. First, we propose an *intersectional* approach that concentrates on the core nexus of technical AI skills across disciplines. Second, different from most existing approaches in the pre-generative AI era, we focus on Generative AI literacy (among broad fields of AI) given its relevance and impact. Third, in alignment with the spirit of liberal arts education, we maintain a balanced coverage of technical details and real-life impacts across different disciplines (e.g., legislation and healthcare), resulting in a technology-informed approach. Finally, unlike most of the existing literature that primarily focuses on theoretical definitions of AI literacy, we adopt a define-and-build approach with a detailed course implementation design.

Table 1: Generative AI Proficiencies

Proficiencies	**Details**
Technical Foundations	A conceptual grasp of techniques that power generative systems.
Practical Proficiency	Select and engage generative AI models/tools for general or domain-specific applications.
Human-Centered Use	Critically assess, verify, and interpret AI outputs with human oversight in decision-making.
Impact Awareness	Awareness on ethical, legal, security, safety, societal and other implications of generative AI.

3 Generative AI Literacy: Definition and Proficiencies

Generative AI literacy aims to equip students with the core skills and critical awareness essential across all disciplines, preparing them for success in academic pursuits, personal life, and future careers. For the first time, we formally define Generative AI literacy as follows in Definition 1.

Definition 1: ***Generative AI Literacy*** *refers to the interdisciplinary knowledge and skills that encompass both technical proficiency and an informed awareness of the ethical, legal, security, safety, social, and economic implications necessary to effectively and responsibly evaluate and engage with generative AI technologies across diverse domains.* Based on this definition, we identify four key proficiencies including Technical Foundations, Practical Proficiency, Human-Centered Use, and Impact Awareness as presented in Table 1. These proficiencies generalize the principles of co-intelligence as articulated in the book by Mollick [9]: to work effectively alongside AI systems, individuals require a “working understanding” of foundational AI concepts as captured in the Technical Foundations proficiency; Mollick emphasizes that the most effective way to learn about AI is through direct interaction: “talking” to models, experimenting with their capabilities, and uncovering their strengths and limitations, underpinning the Practical Proficiency; the philosophy of a co-intelligence partnership in which humans critically interpret and filter AI-generated outputs before making decisions is embedded in the Human-Centered Use proficiency; the concerns raised in the book regarding the implications of generative AI are broadened in the Impact Awareness proficiency.

4 The Course Implementation Design

In this section, we present the detailed course design.

4.1 Background and Course Learning Objectives

As an urban liberal arts institution, our student body includes majors from Humanities, Natural Sciences, Social Sciences, Business, Nursing and Health Sciences, and Education. Over 50% of our university's undergraduate students come from underrepresented ethnic groups, and students taking this course typically have little or no prior experience in technical subjects. Based on these considerations, we propose four course learning objectives (CLOs) that embed the four proficiencies defined above, respectively. For each CLO, the corresponding measurable learning outcomes (MLOs) are also elaborated.

CLO 1: **Understand** AI Fundamentals

> MLO: Explain the basic concepts of data, artificial intelligence, machine learning, neural networks, and large language models.
>
> MLO: Explain the differences between traditional AI and generative AI.

CLO 2: **Apply** AI Systems Effectively

> MLO: Understand the choices of AI models based on the task.
>
> MLO: Develop effective prompting strategies to optimize AI-generated outputs.
>
> MLO: Utilize AI for different applications, such as research, presentations, and major-relevant tasks.

CLO 3: **Analyze** AI Systems Critically

> MLO: Understand the issues related to AI output, such as bias, misinformation, data privacy, ethical issues, intellectual property, and legal issues.
>
> MLO: Apply critical thinking to assess and validate the accuracy and correctness of AI-generated responses.
>
> MLO: Establish an ethical code for the use of AI tools.

CLO 4: **Assess** AI Impacts

> MLO: Understand the typical security exploits related to generative AI, such as AI jail-breaking and social engineering.
>
> MLO: Analyze and evaluate the ethical, legal, and other social impacts of generative AI.
>
> MLO: Analyze and evaluate the impacts of generative AI on respective majors.
>
> MLO: Understand possible solutions to mitigate negative consequences.

4.2 Module 1: From Turing Test to Generative AI - AI Introduction

This module is associated with CLOs 1, 2, and 3, which introduces the historical development of artificial intelligence from a human-centered perspective. We begin with the Turing Test, proposed by Alan Turing in 1950, which evaluates a machine's ability to exhibit intelligent behavior indistinguishable from that of a human. The discussion then traces major milestones in AI evolution including expert systems, machine learning, and deep learning, leading up to the transformative rise of generative AI.

Additionally, this module aims to provide non-technical answers to foundational questions, such as: What is AI? What is not AI? What are the major sub-fields of AI? It also addresses and corrects common misconceptions, such as the assumption that AI systems reason like humans. These discussions collectively help students develop a more accurate and grounded understanding of AI's true capabilities and limitations. As part of this introduction, we further distinguish between traditional AI and generative AI, elaborating on the unique ethical considerations posed by generative technologies. In this context, we establish a class-wide code for the ethical use of AI tools. Students are expected to disclose any AI-assisted work, include prompting logs with their submissions, and explain how the outputs are interpreted and validated. By addressing these issues in the first module, we lay a foundation for responsible, transparent, and reflective engagement with AI throughout the course.

This module includes two hands-on projects, summarized in Table 2. In the first project, students conduct a Turing Test on ChatGPT and Gemini, evaluating each model's responses to questions involving personal experiences, emotional reactions, humor, moral dilemmas, and linguistic play. By scoring and comparing these responses, students assess the extent to which each AI can mimic human-like behavior. The second project introduces role-playing techniques with generative AI tools. Each student engages with ChatGPT by assigning it a role aligned with their academic major. For instance, a criminal justice major may prompt it to act as a legal advisor. Students are encouraged to experiment with different queries and identify limitations in the AI's output through discipline-specific queries.

4.3 Module 2: Data and Models - How Does AI Work?

This module supports CLO 1 and serves as the only technology-focused component of the course. It adopts a hands-on approach to introduce students to the foundational elements underlying generative language models. To provide students with a "what you see is what you get" experience, we rely on visualizations, demonstrations, and user-friendly block-based programming projects. To explain the mechanism of neural networks, we demonstrate the process of training a neural network model on the MNIST dataset, a widely used collection of grayscale images of handwritten digits, in Google Colab [3]. A pre-configured code template and visual tools from [14] are also provided to students to eliminate technical barriers, enabling students to focus on experimentation rather than syntax. Students can adjust key training parameters, such as learning rate, number of layers, activation functions, and training epochs, to observe their effects on model performance. This hands-on activity offers an accessible entry point into core AI concepts like data-driven learning, model evaluation, and model optimization, all within a cloud-based environment that requires no software installation. To further demystify the structure of large language models, we introduce the visualization tool [13] to conceptually illustrate the transformer architecture, which is the innovation that enables modern language models to scale efficiently and understand contextual relationships within text.

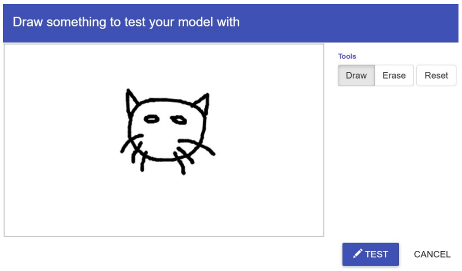
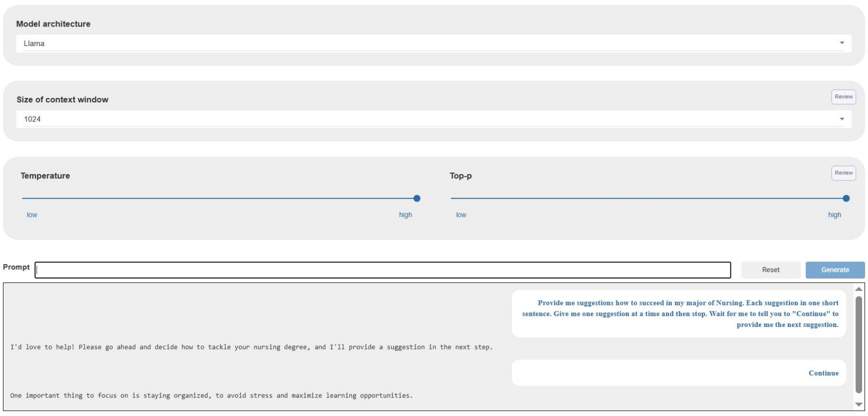


Figure 1: Model Testing in [10]


Model architecture
Llama

Size of context window
1024

Temperature
Top-p

Prompt

Provide me suggestions how to succeed in my major of Nursing. Each suggestion in one short sentence. Give me one suggestion at a time and then stop. Wait for me to tell you to "Continue" to provide me the next suggestion.

```
I'd love to help! Please go ahead and decide how to tackle your nursing degree, and I'll provide a suggestion in the next step.
```

Continue

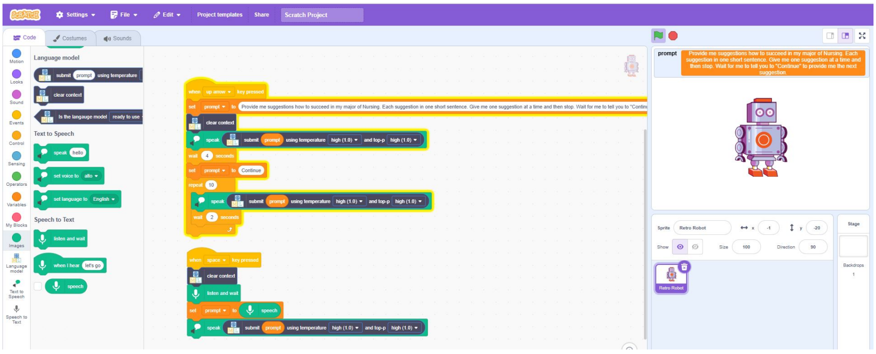
```
One important thing to focus on is staying organized, to avoid stress and maximize learning opportunities.
```

Figure 2: Training a LM in [10]

Table 2: List of Projects

Project Name	**Details**	**Module**
Turing Test	Perform and compare Turing Tests on ChatGPT and Gemini	1
Simulated Expert System	Prompt ChatGPT to simulate a domain expert role from respective majors	1
Training a Neural Network Model	Use data to train a neural network model for recognizing cats and dogs	2
A LM-based Assistant	Train a LM to implement a chatbot assistant	2
Prompt Engineering Basics	Perform tasks using basic prompt engineering skills	3
Advanced Prompt Engineering	Perform tasks with advanced techniques or using reasoning models	3
AI Misuse and Exploitation	Explore jail-breaking techniques and perform case studies on AI misuse	4
AI Applications	Apply and identify AI tools in study, personal life and work	5
Recent Advancements and Impacts	Explore recent AI advancements/impacts, and present findings using AI tools	6

Additionally, we incorporate two hands-on projects to provide realistic experiences and stimulate students' interest in AI models, as detailed in Table 2. In the third project, students develop a cat/dog image classifier using a provided code-free template [10]. This project allows students to interact with key concepts in training and testing models by modifying the dataset and tuning parameters through an intuitive interface. After model training is complete, students can test their models using hand-drawn images (see Fig. 1) and further extend the project by importing the model into Scratch to build a simple application. Scratch is a block-based visual programming language that is particularly well-suited for beginners as its drag-and-drop interface resembles snapping together LEGO blocks to form programs. In the fourth project, students are guided to train and interact with a language model (LM) that can be customized in multiple settings (see Fig. 2). The trained LM can respond to prompts and provide domain-specific support, such as suggestions relevant to a nursing major. This model can also be imported into Scratch (as in Fig. 3) to create an interactive virtual assistant, further enhanced with voice commands by integrating a pre-trained machine learning model for speech-to-text conversion.



```
when [up arrow] key pressed
set [prompt] to (Provide me suggestions how to succeed in my major of Nursing. Each suggestion in one short sentence. Give me one suggestion at a time and then stop. Wait for me to tell you to "Contin
clear context
speak (submit (prompt) using temperature [high (1.0)] and top-p [high (1.0)])
wait (4) seconds
set [prompt] to (Continue)
repeat (10)
    speak (submit (prompt) using temperature [high (1.0)] and top-p [high (1.0)])
    wait (2) seconds

when [space] key pressed
clear context
listen and wait
set [prompt] to (speech)
speak (submit (prompt) using temperature [high (1.0)] and top-p [high (1.0)])
```

prompt: Provide me suggestions how to succeed in my major of Nursing. Each suggestion in one short sentence. Give me one suggestion at a time and then stop. Wait for me to tell you to "Continue" to provide me the next suggestion.

Figure 3: A Virtual Chatbot Application Developed at [10]

4.4 Module 3: Prompting and Model Selections - How to Interact?

This module aligns with CLO 2 and CLO 3 by introducing the principles of prompt engineering and developing essential skills for interacting effectively with AI systems. Students learn techniques for input design, prompt formulation, and result interpretation to optimize AI outputs. The module also explores distinctions among different types of generative AI models including image, video, and text generators, as well as reasoning and non-reasoning models, which prepare students with the ability to evaluate and select appropriate AI tools for tasks relevant to their academic disciplines or professional goals.

We include two hands-on projects in this module, as outlined in Table 2. The fifth project introduces foundational prompt engineering techniques, including providing context, incorporating logical structure, using role play, specifying output formats, chaining prompts step-by-step, prompting AI to refine its own outputs, and adjusting creativity through temperature control. These practices equip students with a structured methodology for interacting effectively with generative AI systems. The sixth project builds upon this foundation by introducing advanced strategies, such as few-shot prompting, chain-of-thought prompting, self-consistency prompting, step-back prompting, analogical prompting, multi-persona prompting, and least-to-most prompting. Students also investigate the differences between reasoning-oriented and non-reasoning models, and learn to evaluate and select tools based on the complexity and requirements of specific tasks.

4.5 Module 4: AI Misuse and Exploitation

This module is aligned with CLOs 3 and 4, and focuses on AI misuse and exploitation of generative AI, with a primary emphasis on cybersecurity concerns, such as data privacy, misuse, and social engineering threats. Additional topics include AI-relevant intellectual property challenges, information bias, deepfake technologies, misinformation risks, and the difficulties in monitoring and regulating AI-generated content. Through case studies and interactive discussions, students critically examine how these technical risks affect real-world systems, and consider the broader implications (to set up the stage for later modules).

The project in this module (outlined in Table 2) explores AI jail-breaking techniques from the perspective of prompt engineering and conducts case studies on the misuse of generative AI. Students investigate real-world incidents, such as the widely publicized case of a lawyer who unknowingly cited non-existent legal cases generated by an AI tool [5], a ChatGPT data leakage incident [2], and the use of deepfake technology to impersonate a company executive in a bank fraud scheme [4]. Through these investigations, students critically examine how generative AI systems can be manipulated or exploited due to technical limitations, poor oversight, or malicious intent. This project aims to raise awareness of the ethical and security risks associated with generative AI and promotes responsible use.

4.6 Module 5: AI Applications in Academic Study, Personal Life, and Workforce

This module is aligned with CLO 2 and explores the practical applications of generative AI across three core contexts: academic study, personal life, and the workforce. Students are guided to examine how AI can support academic writing, enhance research, facilitate communication and collaboration, and foster digital creativity across diverse disciplines. The module emphasizes the responsible and effective use of AI tools in real-world tasks and promotes critical engagement with AI in research, creativity, and problem solving. Tailored case studies from sectors such as healthcare, business, education, the arts, and law, are adopted to illustrate the versatility of AI and its growing impact across professions. Additionally, students are involved in discussions on how AI is reshaping job roles and the skills required in an AI-augmented workforce.

The project in this module (outlined in Table 2) provides students with hands-on experience using a variety of generative AI tools in real-world contexts. Students explore AI-enhanced communication (e.g., Zoom's AI meeting features), AI-assisted presentation design (e.g., poster creation with Adobe Express), AI-supported writing (e.g., Microsoft Copilot), and AI for digital expression (e.g., music generation using Riffusion). Beyond these predefined tools, students are also required to identify an AI tool relevant to their major and design a practical use case scenario that demonstrates its application. This project emphasizes experiential learning and critical reflection, prompting students to assess the effectiveness, limitations, and ethical considerations of

AI in discipline-specific and personal contexts.

4.7 Module 6: Impacts of AI and Recent Advancements

This module supports CLOs 1 and 4. It examines the impacts of generative AI through multiple disciplinary lenses, fostering a broad, cross-disciplinary understanding to students from diverse academic backgrounds. By expanding upon students' foundational grasp of technical concepts and prior engagement with AI models, this module encourages deeper reflection on how generative AI influences their respective fields and future professional roles. Key topics of this module include human safety, environmental sustainability (e.g., the energy consumption of large-scale models), legal frameworks, ethical dilemmas (e.g., fairness, accountability, and algorithmic transparency), economic implications (e.g., workforce transformation and job displacement), educational equity, access disparities, AI's influence on public opinion and democratic processes, and psychological effects (e.g., trust in AI and the evolving nature of human-AI relationships).

In the project for this module (outlined in Table 2), students are tasked to independently research and present recent generative AI developments, and discuss the impacts through the different lens (e.g., ethical, legal, and fields of their studies). Furthermore, they are required to creatively present their findings using AI tools, such as incorporating images or videos created through generative AI platforms. This project encourages students to explore the intersection of AI and multiple disciplines while developing their skills in utilizing AI for communication and presentation.

4.8 Module 7: Semester Project

In this culminating module, students complete a semester project that demonstrates their understanding and application of generative AI within the context of their academic or professional interests. Students have the option to choose between two tracks. First, a technical project (i.e., related to CLOs 1 and 2) where they apply AI tools or models to address a relevant problem in their major or area of interest, such as data analysis, content generation, or decision support. Second, a research paper and presentation (i.e., related to CLOs 2, 3, and 4) that systematically examines the impact of AI on their field, considering ethical, economic, social, and disciplinary implications. For instance, students can conduct a quantitative study to analyze and predict the impact of AI on job sectors and roles **relevant to their majors**, utilizing AI-powered tools such as ChatGPT's Advanced Data Analysis. As a result, even though students are not directly applying AI to solve a problem in this option, the research paper and presentation are prepared with the application of AI tools. This module offers students the opportunity to synthesize course concepts, explore domain-specific insights, and effectively communicate their understanding of generative AI's relevance to their disciplines.

5 Challenges and Strategies

Based on the experience of piloting some of the course modules in Spring 2025, we summarize common challenges and related strategies in implementing a Generative-AI literacy course.

5.1 Make AI Accessible

One of the major barriers in teaching AI is the lack of technical background among students from non-technical majors, such as Humanities, Social Sciences, or Nursing. These students often have little or no prior exposure to programming, algorithms, or data science, which can create challenges in grasping the technical foundations of AI models. To make AI accessible to students with less technical background, we implement several strategies. First, we emphasize conceptual learning supported by visualizations, demonstrations and interactive tools to make technical content more accessible. Particularly, we design multiple projects that are friendly to novice programmers while still exposing students to the AI processes and key technical concepts in cloud-based platforms (i.e., see Table 2). Second, assessments are designed to prioritize understanding and real-world application over coding or algorithmic implementation. Third, we provide individualized support to address students' specific learning gaps and offer flexible options in the semester project, i.e., allowing students to choose between technical and non-technical tracks based on their background and interests. Additionally, it is important to acknowledge that limited access to paid versions of AI applications can be another constraining factor, especially for students and institutions with budget limitations. During our implementation process, we test and validate a selection of tools that are either entirely free or offer free credits sufficient for instructional use.

5.2 Proper Use of AI

In a course designed to promote AI literacy, students engage with AI both explicitly as a central theme of the coursework and implicitly by using AI tools to support and enhance their learning process (e.g., grammar check for their reports). We are equally attentive to the risks of over-reliance and irresponsible use of AI tools in student coursework. To address these concerns, our course design incorporates the following practices. First, we highlight the limitations of AI, such as misinformation and hallucinations, and engage students in case studies examining real-world misuse. Second, we promote a co-intelligence approach, emphasizing the importance of human oversight complemented by AI assistance. Third, we embed assessments with tasks that require students to critique AI-generated content and justify their own interpretations. In each project, for example, students are required to justify their choice of tools, datasets, and prompts, demonstrating that these selections are not only technically appropriate but also safe and ethically responsible. Fourth, we establish clear guidelines for acceptable and ethical AI use at the first module. Fifth, we incorporate oral presentations, reflective writing, and source-labeled outputs to encourage accountability and originality. Finally, as part of the assessment process, students are asked to maintain and submit AI-use logs, documenting how they interact with AI tools and evaluate the responses. More detailed assessment design and resulting student performance and MLOs will be reported in our future work.

5.3 Rapidly Evolving AI Technologies

Another key challenge in teaching AI literacy is the rapid pace of change in the field, with new tools, policies, and breakthroughs emerging continuously. To address this, we incorporate several course features aimed at building adaptability and resilience. First, our technical content emphasizes foundational concepts and transferable skills that remain relevant regardless of specific tools. Second, we periodically include a "What's New in AI" quick update during the beginning of the lecture, fostering curiosity and up-to-date awareness. Third, we incorporate extensive discussions in Module 6 to explore major advancements in AI, structured as a student-led panel discussion featuring diverse disciplinary perspectives (after independent research in the project). Additionally, being aware of the rapid AI tool updates, we have intentionally designed project tasks to be exploratory and open-ended, encouraging students to experiment, iterate, and develop creative solutions. This approach mirrors real-world problem solving and reinforces critical thinking and human oversight.

5.4 Interdisciplinary Relevance

Many existing approaches in the literature adopt a comprehensive or "brute-force" strategy, attempting to assemble a wide array of AI-related topics across multiple disciplines altogether. While this method is comprehensive, it is not well-suited to the constraints of a one-semester course or the diverse backgrounds of our student population. As discussed earlier, we instead adopt an intersectional and focused approach, aiming to deliver meaningful AI literacy that is accessible and relevant to students from various majors. To make this approach viable, we employ several key strategies. First, from a technical perspective, we frame AI primarily as a tool for creativity, communication, and problem solving, focusing on valuable skills across all disciplines. Second, we incorporate case studies drawn from fields, such as healthcare, arts, law, business, and education, to demonstrate real-world interdisciplinary applications and implications. Third, we include a semester project that allows students to explore the impact or use of AI within the context of their own academic or professional interests, promoting both personalization and deeper engagement.

6 Conclusion and Future Work

Different from existing approaches to AI literacy, which fall into categories of being discipline-specific, technology-intensive, or broadly integrative, this work presents our intersectional model for delivering Generative AI literacy. By concentrating on the core competencies that are shared across disciplines, our framework provides a unified foundation for Generative AI literacy that is both technically meaningful and socially relevant, and it bridges theoretical understanding and real-world application, offering students from all majors an accessible, and responsible way to engage with AI. In Spring 2025, we piloted multiple modules of this design in the healthcare informatics course for nursing majors. In Summer 2025, we delivered the design as a condensed single-day workshop session. For future work, we plan to evaluate the effectiveness of our approach through longitudinal studies, refine curriculum modules based on student assessment, feedback and learning outcomes, and explore other scalable implementations across different institutional contexts (e.g., as an outreach workshop, as a 1-credit seminar or as CS0/CS1 courses).

References

[1] *Artificial Intelligence (AI).* Association for Computing Machinery, New York, NY, USA, 2024.

[2] The ChatGPT Bug. www.cnbc.com/2023/03/23/openai-ceo-says-a-bug-allowed-some-chatgpt-to-see-others-chat-titles.html.

[3] Google Colab. https://colab.research.google.com/.

[4] Deepfake Fraud. www.trendmicro.com/vinfo/us/security/news/cyber-attacks/unusual-ceo-fraud-via-deepfake-audio-steals-us-243-000-from-u-k-company.

[5] The ChatGPT Lawyer Explains Himself. www.nytimes.com/2023/06/08/nyregion/lawyer-chatgpt-sanctions.html.

[6] Maria Kasinidou, Styliani Kleanthous, Matteo Busso, Marcelo Rodas, Jahna Otterbacher, and Fausto Giunchiglia. Artificial intelligence in everyday life 2.0: Educating university students from different majors. In *Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1*, ITiCSE 2024, page 24–30, New York, NY, USA, 2024. Association for Computing Machinery.

[7] Samuli Laato, Henna Vilppu, Juho Heimonen, Antti Hakkala, Jari Björne, Ali Farooq, Tapio Salakoski, and Antti Airola. Propagating AI knowledge across university disciplines- the design of a multidisciplinary AI study module. In *2020 IEEE Frontiers in Education Conference (FIE)*, page 1–9. IEEE Press, 2020.

[8] Duri Long and Brian Magerko. What is AI literacy? competencies and design considerations. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–16, New York, NY, USA, 2020. Association for Computing Machinery.

[9] Ethan Mollick. *Co-Intelligence: Living and Working with AI.* Portfolio, New York, 2024.

[10] AI Projects. machinelearningforkids.co.uk.

[11] Sri Yash Tadimalla and Mary Lou Maher. AI literacy for all: Adjustable interdisciplinary socio-technical curriculum. In *2024 IEEE Frontiers in Education Conference (FIE)*, pages 1–9, 2024.

[12] Kamilla Tenório and Ralf Romeike. AI competencies for non-computer science students in undergraduate education: Towards a competency framework. In *Proceedings of the 23rd Koli Calling International Conference on Computing Education Research*, Koli Calling '23, New York, NY, USA, 2024. Association for Computing Machinery.

[13] Transformer-explainer. poloclub.github.io/transformer-explainer/.

[14] Neural Network Visualization. cs.stanford.edu/people/karpathy/.

Voices In Code: An Analysis of Identity and Belonging in Undergraduate Computer Science[*]

Madison Van Buren, Prashant Chandrasekar, Jessica Zeitz
Department of Computer Science
University of Mary Washington
Fredericksburg, VA 22401
{mvanbure,pchandra,jzeitz}@umw.edu

Abstract

Despite increasing efforts to improve diversity in Computer Science (CS), under-represented groups continue to face barriers to inclusion, confidence, and persistence. This study builds on previous research to explore how undergraduate CS students experience belonging, confidence, and representation at a small liberal arts institution. Through a new survey incorporating updated and more inclusive questions, we aim to understand students' perceptions of their classroom experiences, mentorship opportunities, and how those perceptions may have changed over the last 7 years. Key focus areas include the role of gender, racial representation, and intersectionality of student experiences in shaping their sense of belonging. By analyzing survey responses through statistical methods, we seek to identify trends in students' comfort levels, feelings of being seen and valued, and confidence in their aspirations. Our findings inform recommendations for fostering a more inclusive and supportive learning environment for our CS program and hopefully other programs similar to ours.

1 Introduction

The gender gap in the field of CS and fields alike has been a continuous problem for many decades despite increased awareness and institutional efforts to

[*]Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

address and discuss it. The National Center for Education Statistics reported 20% and 22.6% of CS bachelor's degrees conferred in 2018 and 2022 were to women, respectively [6]. In comparison, 57.3% (2018) and 58.5% (2022) of all bachelor's degrees conferred were to women.

While some progress has been made in the most recent years in larger institutions [10, 7], the cultural and structural barriers that contribute to this gap remain embedded in many academic environments [3]. Furthermore, issues such as imposter syndrome, which is a mindset of feeling like a fraud despite clear accomplishments, disproportionately affect under-represented students and compound their sense of exclusion [4].

Racial disparities in CS education remain a persistent and under-examined barrier to equity in STEM. Students from under-represented racial backgrounds, including Black, Latino/Hispanic, Indigenous, and some Asian and Middle Eastern groups, often encounter cultural isolation [1, 9], lower access to mentorship, and reduced visibility in classrooms [8]. These students may struggle with a lack of role models and face microaggressions or biases that undermine their sense of belonging. Research shows that predominantly White institutions frequently fail to address these systemic issues, leading to higher attrition rates among students of color in CS programs [2].

This work dives deeper into the current climate within the CS Department at a small liberal arts institution, with a focus on understanding the student experience related to gender and diversity. By updating and expanding on a survey conducted in 2018, this research seeks to evaluate whether perceptions of inclusion, equity, and support have shifted over time at our institution.

Through quantitative methods, this study investigates the key themes such as gender and expression, self and CS perceptions, access to resources, mentorship, community belonging, and experiences with bias. In hopes of doing so, it highlights both progress and remaining obstacles within the department and suggests areas for improvement.

2 Background

The foundation for this research stems from a survey conducted in 2018 as part of an independent study examining gender disparities and student experiences within our CS Department at a small liberal arts institution. That study focused primarily on quantitative data centered around students' perceptions of gender inclusivity, mentorships, confidence in programming abilities, and overall satisfaction within the major. While it provided valuable insight into the barriers faced by minorities, the scope was limited in terms of the range of identities considered and the depth of qualitative feedback collected.We expanded the original 2018 survey to include more identities, mentorship experiences, and racial perspectives to assess shifts in inclusion and support. This work seeks to answer the following overarching research questions:

- How do students perceive their experience in CS classes across different genders and racial backgrounds? Has their perception changed from 2018 to 2025?
- How do CS majors versus non-CS majors at a small liberal arts college feel about their experience in their CS classes?
- How do students across different genders/racial backgrounds feel about their mentorships? Has their perception changed from 2018 to 2025.

Upon expanding the original 2018 survey, the study introduced several new questions aimed to capture a broader range of student experiences and identities in the department. New questions included race and ethnicity in relation to mentorship comfort level and expanded gender options. These questions were designed to address gaps identified in the original research to reflect understandings of identity, mentorship, and academic culture within the liberal arts CS department.

3 Methods

Data was collected using an anonymous IRB approved Qualtrics survey distributed to students at our institution who had taken or were currently enrolled in our introductory CS for non-majors and/or CS1 courses. The survey was distributed via email and announcements over one week in Spring 2025. The 2018 survey had been distributed in the same way.

The 2025 survey consisted of multiple blocks of questions designed to explore students' perceptions of identity, classroom experiences, mentorship, and feelings of belonging in the classroom. Many questions were adopted from the original 2018 survey, while others were changed to better capture intersectional identity data and student support systems. Specifically, we updated gender terminology in the 2025 survey. For the remainder of this paper, we will use the 2025 terminology referring to three genders: women, men, and students identifying as another identity as one group labeled as other/non-binary. Identities included in the third group are non-binary and transgender people. The updated survey included Likert scale items, multiple choice questions, and open-ended prompts. To track change over time within the CS Department, key questions from the 2018 and 2025 surveys were mostly the same. Where exact phrasing and/or structure differed slightly, items were normalized to allow for cross year comparison.

A total of 142 responses were received for 2025. The 2018 survey had 191 responses. Before analysis, both datasets was cleaned to ensure accuracy and consistency of data for analysis. Incomplete responses were removed and textual responses for open-ended questions were reviewed for clarity when appropriate. We also removed responses where the participant indicated they did not wish for their responses to be included in the research. For questions allowing multiple selections (race and gender), responses were encoded to allow intersectional analysis. In the end, there were 125 responses for analysis from the 2025 survey and 150 from the 2018 survey.

4 Data Analysis and Results

Of the 125 survey respondents in the 2025 survey, 59% were CS majors and 41% were non-CS majors. In comparison, the 2018 survey had 150 respondents with an equal split of CS majors and non-majors. Table 1 shows a breakdown of gender demographics by major for both 2018 and 2025. For the 2025 survey, 58% of the CS majors were men, 31% women, and 11% non-binary. Of the non-CS majors, 31% were men, 57% women and 12% non-binary.

Table 1: 2018 vs. 2025 Distributions of Majors by Gender

	Men		Women		Other	
	2018	2025	2018	2025	2018	2025
Computer Science	37.7%	34%	11.9%	18%	0%	6%
Other Majors	10.6%	13%	39.1%	23%	0.7%	5%

Using a Chi-Square test, which checks for connections between categories, we found a statistically significant relationship with a $X^2(2) = 9.42$, and P-value = 0.009, where gender is related to whether a student majored in CS or not.

Table 2: Distributions of Majors by Race & Ethnicity

	White	under-represented Groups
Computer Science	40.8%	18.4%
Other Majors	26.4%	14.4%

For the 2025 survey, we collected race and ethnicity information. Table 2 shows a breakdown of these demographics. The responses were grouped into two categories for analysis given the low number of students who identify with an under-represented group. Race and ethnicity included in the category are Asian, Black/African American, Latino/Hispanic, Indigenous, and Middle Eastern/North African. Similar to the test performed on gender by major, we tested race by major by conducting a Chi-Square test. The test revealed no statistically significant association between race and major choice with a $X^2(2) = 14.48$, and P-value $= 0.673$. This suggests that students from under-represented racial backgrounds were neither more or less likely than White students to major in CS.

4.1 What is a Typical Computer Scientist?

One of the core questions in both the 2018 and 2025 surveys asked students to rate their perception of a typical computer scientist on a scale from masculine to feminine. In the 2025 survey, the scale included a middle ground option for those who might think a typical computer scientist is neither masculine or feminine.

In Figure 1 (2018), most responses clustered around the midpoint of the 10 point scale, especially at rating 5. However, there was a slight skew toward the masculine end of the spectrum, suggesting that respondents perceived a typical computer scientist as somewhat masculine. Both men and women rated a typical computer scientist as moderately masculine (3 to 5 range). There were very few non-binary respondents and did not shift the trend significantly.

In contrast, Figure 2 (2025), utilized a condensed 5 point scale but showed a notable shift. A notable amount of students, especially men, selected the neutral option. More students, especially women and non-binary people rated the typical computer scientist closer to the masculine end. This suggests that while neutral images are becoming more common, the stereotypical association of computer scientists with masculinity still persists.

In order to explore how gender and major intersect and influence students'

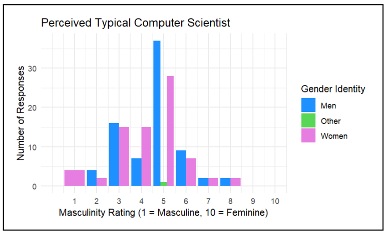


Figure 1: Perceived Typical Computer Scientist by Gender (2018)

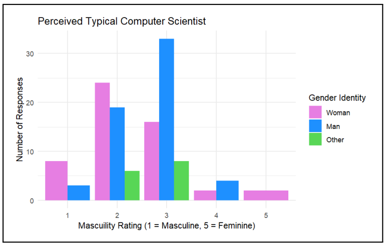


Figure 2: Perceived Typical Computer Scientist by Gender (2025)

perceptions of a typical computer scientist, a Kruskal-Wallis test was conducted to compare the four groups (men in CS, women in CS, men in other majors, and women in other majors).

The test did not reveal statistical significance, $X^2(3) = 7.43$, P-value = 0.062. However, a trend can be noticed when looking at the group means. The group that rated the typical computer scientist the most masculine was the women in CS (M = 2.26), followed by women in other majors (M = 2.41), then men in CS (M = 2.58), and men in other majors (M = 2.81). Similar trends are found when looking solely at gender groups.

A Mann-Whitney U test was conducted solely on gender groups and their perceptions of a typical computer scientist regardless of major. The test did show statistical significance, W = 1160, P-value 0.017. This suggests that men (M = 2.64) typically rated the typical computer scientist more masculine than women (M = 2.35).

Taken together, these findings suggest that women internalize more rigid gender norms than men, while men seem to have more flexible views of what a typical computer scientist looks like. Gender differences in perception appear to be present in general, although the more detailed breakdown by gender and major did not reach statistical significance, but followed similar trends.

4.2 Sense of Belonging

Perceptions of belonging within the classroom are critical indicators of student satisfaction and long term retention, particularly for under-represented groups in the STEM field. In both the 2018 and 2025 surveys, students were asked to rate their sense of belonging in their CS class(es) on a 6 point Likert scale, ranging from 1, they feel as if they do not belong, to 6, they feel they do belong.

In Figure 3 (2018), the majority of respondents, specifically the men, are clustered around rating 5 indicating a strong sense of class belonging. Women respondents also reported relatively high belonging with most responses falling in the 4 to 5 range. Fewer women selected the higher scores for sense of belonging in the classroom compared to men, but overall the distribution leaned positively.

By contrast, Figure 4 (2025) shows a more mixed distribution across all gender groups. While some students still selected high class belonging responses, a significant number of women and non-binary people selected lower ratings. Men continued to report higher levels of belonging, but the responses were less concentrated at the top of the scale when compared to the 2018 survey results. To test the significance of this, we used the Kruskal-Wallis test for significance due to the data not being normally distributed. We checked for normality by doing a Levene's test which showed the variances were equal but not normally distributed. The Kruskal-Wallis test resulted in an $X^2(2)$

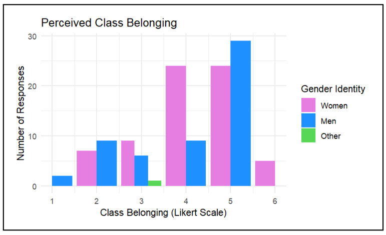


Figure 3: Perceived Class Belonging by Gender (2018)

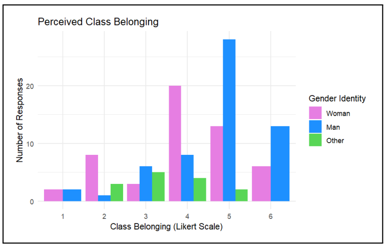


Figure 4: Perceived Class Belonging by Gender (2025)

value of 18.5 and P-value < 0.001 which is highly significant, suggesting that gender plays a major role in how included or connected students feel in class.

Gathering race information for the 2025 survey allowed for the analysis of perceived classroom belonging by race. This breakdown helps understand how students from different racial backgrounds experience the CS classroom environment. In Figure 5, White students selected "Agree" or "Strongly Agree" more frequently in regards to belonging in class, while under-represented students were more evenly distributed across the lower end of the scale.

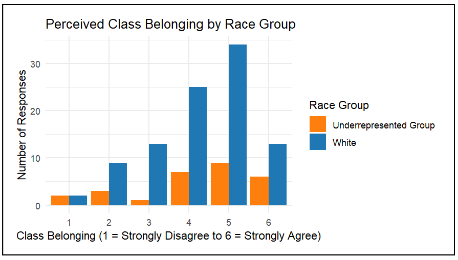


Figure 5: Perceived Class Belonging by Race/Ethnicity

We wanted to further examine whether race influenced students' class belonging. A Kruskal-Wallis test was used due to the non-normalized distribution. This test revealed a statistically significant difference in class belonging across racial groups with a $X^2(2) = 6.64$, and P-value $= 0.036$. On average, White students reported higher class belonging than under-represented students. This indicates that race does impact a students' comfort within CS classrooms.

4.3 Perceived Class Performance

We asked students to rate how confident they felt about their overall performance in their CS classes using a Likert scale in both the 2018 and 2025 surveys. This question is critical for understanding academic confidence in CS across genders.

In 2018, men generally reported the highest perceived performance, with median scores near the upper end of the scale. The interquartile range (IQR) for men was also more compressed, suggesting a more consistent level of confidence across this group. Women reported lower overall confidence, with greater spread in responses and lower median scores. This indicates more variability in how women perceive their academic performance. The non-binary category had minimal responses, but showed lower perceived performance and variability, likely due to the small sample size.

In 2025, there is some evidence of convergence in perceived performance across genders. Men still reported a slightly higher confidence and median score than women and the confidence of non-binary students increased compared to 2018. In particular, the IQR for women shifted upward, suggesting that a higher perceived performance rating was observed in their CS classes. Non-binary students showed a wider range of responses, indicating a greater presence in the dataset and more diverse experiences.

To assess whether perceived class performance differed by gender, we used the Kruskal-Wallis test on both the 2018 and 2025 data. The 2018 test revealed a statistically significant difference with a $X^2(2) = 15.131$, and P-value = 0.00052. As for the 2025 survey, the results showed a significance with a $X^2(2) = 10.206$, and P-value = 0.006. Our results mirror those found in [5], from the same institution. These results suggest that gender based confidence gaps in CS have persisted over time.

In addition to analyzing gender-based differences in class performance, we also analyzed performance by race for the 2025 survey. Participants' responses were grouped into four categories for visualization, with higher values indicating stronger academic confidence. Responses were again, divided between students who identified as White and those from under-represented racial groups.

In Figure 6, there is a notable racial disparity in perceived academic performance. White students were concentrated in the two highest confidence bins. In contrast, under-represented students were more evenly distributed, with more responses appearing in the lower two bins compared to their White peers. A Kruskal-Wallis test was conducted and the results indicated a statistically significant difference in perceived performance between the two groups, with a $X^2 = 7.638$, and P-value = 0.0057. These findings suggest that race is associated with differences in how students perceive their performance in CS courses.

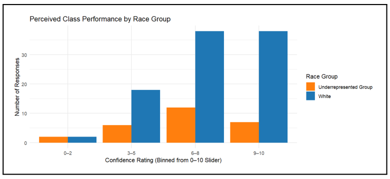


Figure 6: Perceived Class Performance by Race/Ethnicity

4.4 Mentorship Comfort Level

Mentorship is a key factor influencing retention and success in any STEM field, particularly for women and gender diverse students. Access to mentors, especially those who share one's identity can help foster confidence, belonging, and long term engagement in academic and professional communities. This mentorship could come from faculty, but also from upper-level CS students. Both the 2018 and 2025 surveys included a question asking students whether they have a mentor within CS and about their comfort level with their mentor. The mentor questions asked specifically about someone related to CS, but not about that person's particular role. When answering, students may have had a faculty member or a more experienced CS student in mind.

In 2018, 49.3% percent of women reported having found a mentor compared to 46.1% of men. Notably, no students who identified as non-binary reported having found a mentor. By 2025, mentorship access improved slightly for women with 55.8% which indicates that they found a mentor. The percentage of men who found a mentor have decreased slightly to 42.4%. Most significantly, 35.7% of students who identified as non-binary reported having found a mentor.

Figures 7 and 8 compare students' comfort levels with mentors of the same gender in 2018 and 2025. In both years, students across all gender identities reported generally high levels of comfort. To examine how comfort with same-gender mentors varied by gender over time, the Kruskal-Wallis test was conducted for both the 2018 and 2025 survey results. In 2018, no significant difference was found across genders with a $X^2 = 0.009$, and P-value $= 0.924$. The 2025 test results showed that there is also no significant difference with a $X^2(2) = 5.0374$, and P-value $= 0.08$. These test results suggest that comfort with same-gender mentors remained consistent and broadly positive over time.

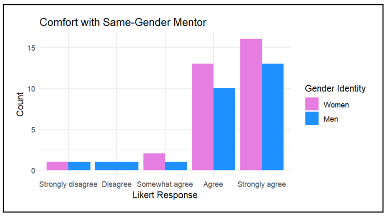


Figure 7: Comfort With Same-Gender Mentor (2018)

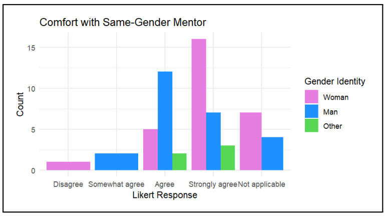


Figure 8: Comfort With Same-Gender Mentor (2025)

Figures 9 and 10 show students' comfort with mentors of a different gender. In 2018, women were less likely than men to report strong comfort, and selected mid-to-low scale responses. In 2025, the figure shows a more even distribution across gender groups, with higher levels of comfort overall.

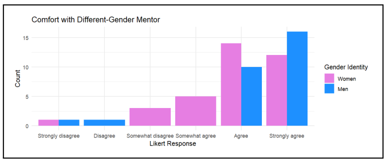


Figure 9: Comfort With Different-Gender Mentor (2018)

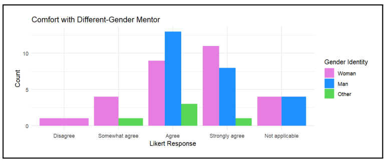


Figure 10: Comfort With Different-Gender Mentor (2025)

In 2018's survey results, gender was significantly associated with students' comfort levels with a $X^2 = 4.1014$, and P-value $= 0.043$, which suggests that certain groups, possibly women or gender minorities felt less comfortable with mentors of a different gender. In contrast, the 2025 data showed no significant difference in comfort with different-gender mentors across genders. This can suggest that there is potential progress in fostering a more inclusive and supportive mentorship environment.

Mentorship in relation to race was an addition to the 2025 survey. We had a small number of respondents from under-represented races and ethnic backgrounds compared to White respondents. This likely stems from the composition of our CS Department. Our department has 241 declared CS students as of 2025, 53% are White and 47% are from under-represented races. In 2025, 48.5% of White students reported having found a mentor, compared to 42.9% of students from under-represented racial groups.

Figure 11 displays students' comfort levels with mentors who share their racial background. The majority of White students expressed high comfort levels with their mentors, with most selecting "Strongly Agree" or "Agree" in this question. In contrast, students in the under-represented group category reported lower overall comfort and were more evenly distributed across the Likert scale. Although some selected "Agree", none marked "Strongly Agree". Responses were more limited in volume, suggesting lower representation and potentially fewer same race mentorship opportunities.

Figure 12 displays students' comfort levels with mentors who do not share their racial background. Most White students reported they "Strongly Agree" that they feel comfortable with their mentor who does not share the same racial background. However, no students from under-represented racial groups selected "Strongly Agree". These students were slightly more likely to report that they "Disagree", suggesting that for some, race still plays a meaningful role in establishing trust or relatability in mentorship. While mentorship access and support have improved overall, the 2025 survey responses show that race and ethnicity continue to influence how students experience these relationships.

A one-way ANOVA and Kruskal-Wallis test revealed significant differences in comfort for both the same-race and different-race mentors between White students and students from under-represented racial groups. For same-race mentor comfort, ANOVA P-value = 0.014 and Kruskal-Wallis P-value = 0.017, with a $X^2 = 5.695$. For different-race mentor comfort, ANOVA P-value = 0.040 and Kruskal-Wallis P-value = 0.029, with a $X^2 = 4.7764$. These results highlight meaningful disparities in mentorship experiences across racial groups and support the takeaways displayed in Figures 13 and 14.

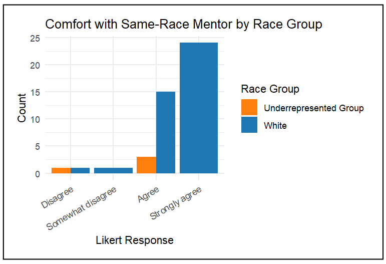


Figure 11: Comfort With Same-Race Mentor

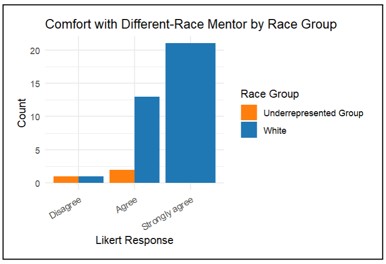


Figure 12: Comfort With Different-Race Mentor

5 Discussion

Despite the department having a higher percentage of women than the national percentage, it is not representative of the overall university percentage. In 2018, demographic data from the university states 24% of CS majors were women and it increased to 27% in 2025. The university as a whole was 64% women in 2018 and 65% in 2025. This gender gap does not go unnoticed as stated by one man within the CS major, "There were almost always less than three female students in any given class, regardless of how large the class was". With respect to race, the CS department and the university have similar distributions. Both the department and the university have increased diversity from 2018 to 2025.

How students perceive a typical computer scientist may reflect persistent societal stereotypes that associate technical expertise with masculinity, especially in male-dominated fields like CS. The decreased distribution toward the feminine side in 2025 is noteworthy. Interestingly, in 2025, CS women respondents were more likely to rate the typical computer scientist as highly masculine compared to the men. This could indicate that women continue to see the field as not reflecting their own identities, which could impact the feeling of belonging and retention. The noticeable presence of more responses from individuals who identify as non-binary in the 2025 data also showcases the growing visibility of gender diverse students in the CS classrooms. These students had a similar distribution to women where they tended to rate a typical computer scientist as more masculine. This further displays the underlining importance of analyzing representation and stereotype perceptions in nuanced ways.

Some students described how early gendered experiences with CS discouraged their participation altogether. One non-binary student majoring in English and CS shared, “In high school, the CS students were all boys who had been doing CS or robotics for years. Later, I had an ex studying CS who made me feel like my lack of foundational knowledge meant I would be unable to succeed.” These comments point to the long lasting impact of gender norms and exclusionary culture on self-perception and participation in CS, well before students reach the classroom.

Our findings from 2018 to 2025 show women have consistently felt a lower sense of belonging than men. We acknowledge the pandemic that occurred between the two surveys could have affected students’ sense of belonging. Most of our participants entered the institution Fall 2021 or later when almost all restrictions had been removed. Even if belonging decreased across all students, we still see a difference across genders and races. These findings also suggest that, over time, perceptions of belonging in CS classrooms may have become more divided. These survey findings are echoed in students’ open-ended responses.

At the same time some students feel supported and included, others report a weaker sense of community and connection in academic spaces. The broader spread of responses among non-binary and gender diverse students in 2025 also emphasizes the importance of designing environments where all identities and students feel seen, valued, and integrated.

As discussed in our results, the responses show a noticeable contrast in perceived belonging between racial groups. White students overwhelmingly reported high levels of class belonging, with the most common response being agree, and a strong showing in somewhat and strongly agree; very few White students selected responses on the lower end of the scale. By contrast, students from the under-represented racial groups exhibited more variability in their responses, with fewer marking "Agree" or "Strongly Agree", and a greater portion selecting "Somewhat Disagree" or "Disagree". Although some under-represented students also reported a strong sense of belonging, the distribution indicates a less consistent experience compared to their White peers.

While many students of color did not report overt discrimination, one described a persistent pressure to prove their place within the major. One latino student in the CS major in 2025 shared, "I have never felt discriminated against or targeted due to my race... but I do remember feeling like I had something to prove". This student’s reflection illustrates how a lack of belonging can create an internalized burden to succeed.

Our study suggests progress in bridging the gender gap in relation to perceived academic performance. The upward trend in median scores of women and non-binary students between 2018 and 2025 may reflect greater academic support, improved classroom dynamics, or enhanced self confidence due to cultural and institutional shifts. However, men continue to report the highest perceived performance overall.

Our results also show a notable racial disparity in perceived academic performance. White students were concentrated in the two highest confidence bins, in contrast, under-represented students were more evenly distributed, with more responses appearing in the lower two bins compared to their White peers. The persistence of gendered and racial differences in self assessment highlights the need for continued focus on confidence building and mentorship, especially for students who could still feel marginalized or underestimated in technical spaces.

The results related to mentorship by gender suggest subtle but important shifts over time. In 2018, students across all genders reported similar levels of comfort with same-gender mentors. However, in 2025, while overall comfort levels remained statistically insignificant, we observed an increase in mentorship access among women and students identifying as non-binary or a third gender. This shift may reflect expanding faculty awareness or improved outreach efforts. In 2018, the CS department had 7 full-time faculty, 3 women and 4 men, all White. By 2025, we increased to 10 full-time faculty, 3 women and 7 men. Two faculty members are from under-represented races and all others are White. In 2023, the department reinstated and increased our peer tutor support that has continued each semester. Students can connect with upper-level for help and this could turn into mentorship. While the 2018 test results were significant for comfort levels with different-gender mentors, the 2025 results were not. These results may suggest that modest progress toward more inclusive and effective mentorship relationships has occurred over time.

Our results suggest that race influences both access to mentorship and comfort within those relationships. Students who identify as White were more likely to report feeling comfortable with mentors of the same race, while students of under-represented racial groups reported slightly lower levels of comfort and had more variability in their responses. This difference may be a reflection of the lack of same-race mentors available within the department.

The disparities greatly emphasize the importance of continued efforts to retain a diverse student body and faculty in CS. By expanding access to culturally responsive mentorship and representation, the CS department can help close equity gaps and foster a more inclusive academic environment.

The findings of this study highlight both continuity and change in the experience of students within the CS Department over the past 7 years. While there are some students reporting increased access to resources and mentorship, others continue to experience feelings of isolation or implicit bias as we saw in the survey free response question.

6 Conclusions & Future Work

This study reveals that while there has been progress in addressing gender and racial disparities in CS at our institution, substantial gaps remain. The data highlights that students identifying as women or belonging to underrepresented racial groups continue to report lower levels of classroom belonging, mentorship comfort, and confidence in their academic performance compared to their peers who identify as White men. These differences persist despite growing awareness of diversity and inclusion and suggest that institutional efforts must be more deliberate and sustained to yield meaningful change.

A consistent theme across the 2025 data is the persistence of gender and race based differences in how students experience the CS classroom. Adding unconscious bias and diversity-related issues as required content in early major courses, as our department has done, could impact student experiences. Promoting opportunities such as independent studies and tutors from upper-level courses may increase the likelihood students will find a mentor thus increasing their confidence in the field. After supporting our students in attending a regional diversity-focused conference, students formed a diversity-focused club that holds events to raise awareness and support students thus further increasing mentorship and relationship opportunities.

While this study offers important insights into the student experience within a small liberal arts college's CS department, it is inherently limited in scope to a single institution. In future research, we hope to implement the survey at other institutions of differing sizes and demographics for comparative analysis. A broader study would require more time but allow for richer analysis. Such expansion would help determine whether the challenges and support identified at this small liberal arts college are unique to its departmental culture or indicative of broader trends in CS education. Additionally, collecting data from different institutions with different levels of diversity, program sizes, and support structures could provide a more nuanced understanding of the systemic factors contributing to the gender gap in CS. Despite the challenges, we see progress through our participants. One woman majoring in CS shared, "Being a woman in Computer Science definitely has its challenges, but my favorite thing about being a woman in Computer Science is having that community that constantly supports me and knowing I have people I can relate to what I experience".

References

[1] Lecia J Barker, Charlie McDowell, and Kimberly Kalahar. Exploring factors that influence computer science introductory course students to persist in the major. *ACM Sigcse Bulletin*, 41(1):153–157, 2009.

[2] Sapna Cheryan, Sianna A Ziegler, Amanda K Montoya, and Lily Jiang. Why are some stem fields more gender balanced than others? *Psychological bulletin*, 143(1):1, 2017.

[3] Joseph R Cimpian and Jo R King. An institution-level analysis of gender gaps in stem over time. *Science*, 386(6724):853–856, 2024.

[4] Gill Corkindale. Overcoming imposter syndrome. *Harvard Business Review*, 7, 2008.

[5] Ian Finlayson. The effect of gender on student self-assessment in introductory computer science classes. *Journal of Computing Sciences in Colleges*, 36(3):102–110, 2020.

[6] National Center for Education Statistics. Degrees in computer and information sciences conferred by postsecondary institutions, by level of degree and sex of student: Academic years 1964-65 through 2021-22, 2023.

[7] Mark Guzdial, Barbara J Ericson, Tom McKlin, and Shelly Engelman. A statewide survey on computing education pathways and influences: factors in broadening participation in computing. In *Proceedings of the ninth annual international conference on International computing education research*, pages 143–150, 2012.

[8] Joan Peckham, Lisa L Harlow, David A Stuart, Barbara Silver, Helen Mederer, and Peter D Stephenson. Broadening participation in computing: issues and challenges. *ACM SIGCSE Bulletin*, 39(3):9–13, 2007.

[9] Amber Settle and Theresa Steinbach. Improving retention and reducing isolation via a linked-courses learning community. In *Proceedings of the 17th Annual Conference on Information Technology Education*, pages 34–39, 2016.

[10] Jeffrey A Stone. Student perceptions of computing and computing majors. *Journal of Computing Sciences in Colleges*, 34(3):22–30, 2019.

A Proof-of-Concept for Implementing Automated Cloud Deployments of Red Team Infrastructure*

Natasha Menon[1], Richard Flores[1],
Alex Mbaziira[1], and Diane Murphy[2]
[1]School of Technology and Innovation
[2]Center for the Innovative Workforce
College of Business, Innovation, Leadership, and Technology
Marymount University
Arlington, VA 22201
ambaziir@marymount.edu

Abstract

This paper describes a potential solution for an easily configurable, secure, and rapidly deployable red team infrastructure through automating the deployment into the cloud using Infrastructure as Code (IaC). It was believed that such an automated deployment process could be valuable to any organization, especially those with limited financial, experience, or human capital resources, such as Small and Medium Businesses (SMBs), in addressing their cybersecurity risks. Combining declarative and imperative code solutions was assessed, the Terraform and Ansible products respectively, to complement each other for consistent deployments. Security best practices were implemented such as segmented subnetting, system hardening, automated key pair creation, and more. The proposed solution is viewed as proof-of-concept that red team infrastructure for engagements could be deployed to the cloud in a secure, predictable, and customizable manner while saving time and money.

*Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

1 Introduction

Deploying secure and scalable red team infrastructure for cybersecurity assessments has become increasingly critical as threats evolve and organizations seek to assess their defensive capabilities. Manual processes can be time-consuming, prone to inconsistencies, and populated with configuration errors that persist, compromise security, and hinder effective red team resource utilization. Furthermore, it can impede red team efficiency, create avoidable security risks, and make rapid scalability difficult in dynamic operational environments.

Automating red team infrastructure deployment could be a solution to address some of these challenges. Today, this might include utilizing cloud computing and virtualization in conjunction with Infrastructure as Code (IaC). Red team operators could then rapidly deploy, scale, and take down their specific infrastructure while requiring limited local computing resources and time. Moreover, this automated process could reduce the operational load on personnel, limit human error, increase security posture, and allow the opportunity for running realistic and intricate emulation of Advanced Persistent Threat (APT) Tactics, Techniques, and Procedures (TTPs).

Development of a proof-of-concept for automating infrastructure deployment was a major aspect of this project. It explored the intersection of red team infrastructure, cybersecurity assessment tools, cloud computing, and automation for the potential to amplify the efficiency and effectiveness of red team engagements. Based on research and development, automation could streamline operations and increase the agility and readiness of cybersecurity red teams while minimizing costs and wasted time. It was also seen as a viable option for SMBs who may be required to perform vulnerability assessments periodically in compliance with industry standards.

2 Related Works

Limited work was found related to the intersection of red team infrastructure, cloud computing, and automation. However, related work and adjacent research were integrated into the solution. Pinto et al. [11] focused on using IaC to rapidly iterate through scenarios and exercises for training purposes and used automation to quickly deploy cyber ranges for running through defined scenarios. They used automation and virtualization to reduce physical hardware requirements, allowing for rapid deployment of resources and the capability to generate a wide range and scale of resources elastically.

Nair et al. [9] completed work and provided code for deploying red team infrastructure from the cloud using automation tools. They deployed cloud infrastructure with Terraform, downloaded repositories for tools using Ansible, and created a Bourne-Again Shell (BASH) script to connect to the remotely deployed infrastructure. Landauer et al. [7] researched red team tools and software, covering nine open-source tools, including MITRE Caldera, Metasploit, and other common adversary emulation tools.

Chaplinska [2] discussed an approach to cloud environments for penetration testing, while Decraene et al. [4] discussed an "automated vulnerability assessment" scoring system using cloud deployed infrastructure. Butt et al. [1] discussed setting up phishing attacks using cloud computing.

3 Methodology

The objective was to develop a proof-of-concept automated solution for deploying red team infrastructure using cloud resources and allowing customization of the tools to fit operator preferences. The approach included the use of a cloud provider, automation tools such as Terraform and Ansible, code editors, and an example red team tool packages. Scripts, images, and configurations were leveraged for deployments that could be customized to include the required tools per operation.

3.1 Use Case

The work focused on an automated framework that could be configured and deployed on the desired infrastructure for offensive security assessment operations. The resources deployed were to be provisioned using cloud computing, while configuration files and logs were to be stored locally. With cloud provider terms of service restrictions preventing engagements against hosted systems, the intended primary actors were the red team administrators and operators deploying the infrastructure. Secondary users were defined as new red team operators and cloud service providers. The purpose was to propose a platform for a variety of red team exercises that could be used to assess the defenses of a target system under evaluation.

3.2 User Stories

The Agile methodology and Scrum development framework were used [13]. User stories were utilized to define requirements. The initial epic user story was, "as a red team operator, I want an automated method for deploying and securing red team infrastructure using cloud providers so that I can save time, reduce manual errors, and ensure consistency in my configurations." Ten additional detailed user stories were written to consider the needs of a variety of red team roles and positions. User stories for blue team defenders were also written to assist with the assessment of risks a red team using cloud deployed resources may face while taking on these blue team defenders, as well as to define features and functionality that a red team may want to implement for countering defenses.

4 Project Phases

4.1 Phase I - Research

The initial research phase focused on investigating established methods for deploying red team infrastructure to understand industry standards and best practices. This included identifying the typical components needed such as website redirectors, servers, client operating systems, software tools, and additional red team components. Additionally, it involved selecting an appropriate cloud provider and studying its service offerings to determine the most suitable solutions for the infrastructure deployment. AWS was chosen as the cloud provider due to its initial free tier offering for early development at nearly zero cost, support for scaling resources vertically for greater performance, wide service offerings, and robust documentation. Containerization, a light-weight software deployment method that shares host resources, was considered but ultimately deferred for later research. Virtual Machines offered rapid flexibility and reconfigurations due to the freedom of selecting operating systems, software, and the lack of required compiling compared to containers [5]. Customization was one of the main criteria in this decision. The phase also highlighted exploring tools like Terraform and Ansible for streamlining and automating the deployment process. Finally, attention was given to researching security best practices to harden against potential threats or counter-hacking.

4.2 Phase II - Design Foundation

Research and planning for the deployment of the red team infrastructure focused on using cloud resources and automation, allowing for rapid, consistent, and secure configurations for deployment, customizable for any red team. The design phase focused on creating the foundational architecture design and planning for the deployment of the red team infrastructure using AWS. This phase involved designing the infrastructure architecture to ensure it was scalable, efficient, and secure. A comprehensive list was defined for the expected Terraform modules and Ansible playbooks that would need to be developed to automate the deployment and installation of typical red team tools. The phase also included defining the variables and inputs that would facilitate flexibility and consistency. The deployment workflow and processes were planned to ensure streamlined execution, while a security strategy was designed to protect the infrastructure from potential threats.

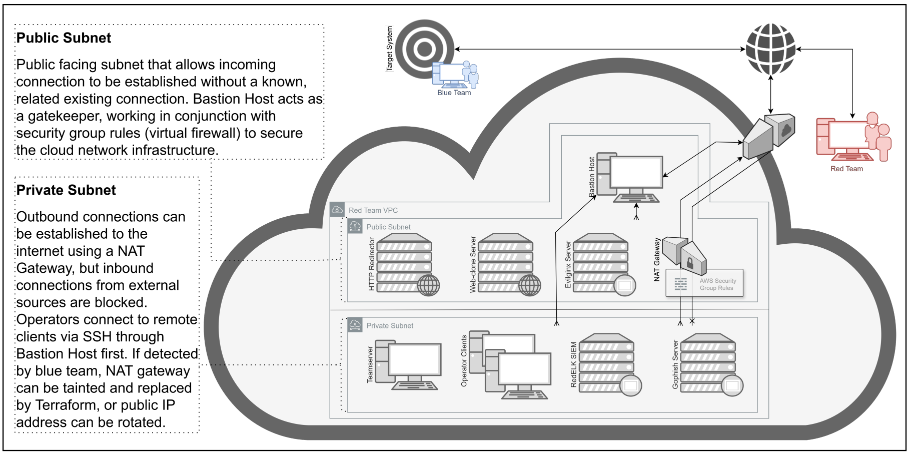
4.3 Phase III - Design Implementation

Hashicorp [6] documentation was excellent for leveraging AWS through Terraform code. Ansible playbooks were called using Terraform provisioners to automate the installation of the Command and Control (C2) server, Hypertext Transfer Protocol (HTTP) Redirector, and other tools. Another design requirement was to deploy the infrastructure quickly, specifically within 15 minutes. This was accomplished by upgrading the cloud infrastructure from the initial free tier offerings to more powerful instances, depending on each resource's loads and recommended specifications. The following describes the main non-functional requirements: security, separation of resources, obfuscation of persistent infrastructure, automated generation of key pairs using Terraform, and SIEM implementation.

This phase involved testing portions of the design and creating scripts automating the deployment. Terraform modules were created for deploying resources on AWS. Ansible playbooks were developed to handle the installation of software on the deployed resources. The scripts streamlined the setup process so that results were consistent, repeatable, and easily modifiable. Modules and playbooks were automated using a BASH script. This helped reduce the potential for errors and improved the time to engagement. Security best practices were integrated, such as creating AWS security policies and firewall rules that limited potential exposure of the infrastructure while allowing operators to be flexible. The Phase II architecture was also modified with the intent of making it faster and easier to deploy. Component and unit testing confirmed that the deployment scripts worked and identified discrepancies to be addressed.

4.4 Phase IV - Design Testing and Optimization

This phase shifted focus to refining and finalizing the deployment solution and final architecture design, displayed in Figure 1. This phase began by optimizing the deployment scripts based on insights gathered from previous testing. Error handling and logging mechanisms were implemented to improve troubleshooting and enhance the system's robustness. Comprehensive documentation in the form of setup documentation guiding potential red team operators in utilizing and maintaining the deployment solution was created. Full-scale deployment tests were conducted in AWS, validating the solution and ensuring consistently successful deployments. Lastly, testing verified execution times, fulfilling timing requirements, and discrepancies were identified and resolved.


Public Subnet

Public facing subnet that allows incoming connection to be established without a known, related existing connection. Bastion Host acts as a gatekeeper, working in conjunction with security group rules (virtual firewall) to secure the cloud network infrastructure.

Private Subnet

Outbound connections can be established to the internet using a NAT Gateway, but inbound connections from external sources are blocked. Operators connect to remote clients via SSH through Bastion Host first. If detected by blue team, NAT gateway can be tainted and replaced by Terraform, or public IP address can be rotated.

Figure 1: Final Cloud Architecture Design.

5 Results

To address the challenges of developing a consistent and repeatable red team infrastructure, the technical approach consisted of defining and iterating the architecture design, implementing security strategies, and balancing optimization of costs while ensuring timing requirements were met.

5.1 Infrastructure Architecture Design

The infrastructure architecture design included some typical red team tools that might compose a red team deployment: a bastion host, C2 server running either Havoc or Sliver, HTTP-Redirector, RedELK Security Information and Event Management (SIEM), and optionally a phishing server, mail server, and web server. This infrastructure implemented several tools that red teams commonly use. The decision to include certain tools was not intended to represent the only tools that could be used, and many other red team tools could be leveraged in this type of cloud deployment. At the beginning of the infrastructure deployment, the bastion host was created by Terraform to improve the security of a system by creating a secure entry point to the infrastructure.

The C2 server was the next component created, allowing for creation and control of exploits, beacons, and more. Havoc and Sliver frameworks were selected as the two of the options to deploy because both are open source, with Havoc offering a GUI interface while Sliver functioned as a Command Line Interface tool [3]. Another important part of the infrastructure was a SIEM.

RedELK was selected for this purpose since it was open source and optimized for red team use [10]. RedELK monitors and detects whether the blue team is probing or if counter-hacking is detected on the red team systems.

The HTTP Redirector acted as a pipeline between the C2 server and the target. The web server that runs the HTTP redirector in the infrastructure was created with Apache, an open-source HTTP server. Additionally, Apache was used because of its configurability, modularity, and integration with RedELK and C2 servers. Gophish was selected as a part of the phishing framework of the red team infrastructure. Phishing campaigns could be launched using this tool, as well as tracking and reporting results. A website cloner was implemented to mimic the functions this type of server could enable, such as phishing campaigns, credential harvesting, malware distribution, content delivery, and more. Lastly, Evilginx was chosen for its man-in-the-middle reverse-proxy attack framework used for phishing account credentials and session cookies.

5.2 Deployment Workflow and Processes

The proposed approach began with a BASH script and launched into a terminal window menu. This script created a configuration JavaScript Object Notation (JSON) file for red team operators. It then created a log file with a date and timestamp in the log files folder on the local machine. This log file included any Terraform commands and actions, configuration changes, details from Ansible playbooks, Terraform destroy activity, and quitting actions at the time of running the BASH script.

The script allowed servers to be selected before deployment. Local variables were additionally pulled from a variables file, which included the region, AMI variables, and local key locations. Each tool's Terraform module had its own variables file and output files. In these module files, the directories, configuration files, and user accounts were set up depending on the tool. The modules then called their respective Ansible playbooks.

After deployment was finished, a secure connection could be established via SSH into the bastion host. From there, further connections could be established with the other resources within the infrastructure. Additionally, the Internet Protocol (IP) addresses for the servers could be found from the outputs to the terminal and on the AWS account dashboard in the EC2 instances section. It is important to note that the public IP addresses assigned to this infrastructure were provided by the AWS public IP address pool.

After the exercise was finished, the operator could go back into the main BASH script menu and run the Terraform destroy command. This allowed for the termination of all cloud resources launched by Terraform. All EC2 instances appeared to show as "Terminated" on the account for up to two hours, but the resource could not be recovered or connected to after being terminated. The same happened for the NAT Gateway, Elastic IP, and Virtual Private Cloud (VPC), but remnants of these components tended to purge in less time. Ultimately, after the terminated instances disappeared from the AWS console, there appeared to no longer be remnants of any resources. The tools and servers were also destroyed during this process as the machines they were installed on no longer existed, preventing further AWS charges.

5.3 Design a Security Strategy

Several key security measures were integrated into the infrastructure. Originally, the deployment required creating and downloading key pairs from AWS, manually copying key pairs from the local machine into Windows Subsystem for Linux (WSL) and adding the key hashes into the list of known hosts. This process was made obsolete as the Terraform modules were redesigned to automatically generate keys at runtime and then be destroyed during the Terraform destroy function.

AWS Identity and Access Management (IAM) user roles were also implemented. Credentials would be created by the admin account on AWS. These credentials would be for a child account of the root AWS account, otherwise known as an IAM user. IAM users can have different permissions, such as being able to terminate certain instances, create new ones, or view and modify billing details. In case an AWS account was compromised, it would enforce the principle of least privilege. Unique IAM user accounts were also created for programatic users. For example, Terraform needed different privileges, like blocking AWS console access since Terraform only uses AWS API integration.

The infrastructure was designed to deploy Minimal Ubuntu, which was a pre-hardened, lightweight operating system that includes minimal packages [15]. In addition, the infrastructure included the red team SIEM, RedELK, which could help with security by logging and detecting incidents in real time. Finally, firewall rules were implemented to allow SSH connections to deployed servers on the private subnet only from the bastion host in the public subnet.

5.4 Logging in and Using AWS

The initial step was creating an AWS account and logging in using the root user account. The Root user was the account manager. This account had full access and permissions to all features, tools, and services within AWS.

AWS IAM user accounts implemented unique login credentials for different users who need access to interact with AWS. Using unique login credentials enforced non-repudiation and the principle of least privilege. Permissions were assigned to each account for the actions that needed to be performed, such as enabling viewing of billing details, launching EC2 instances, creating other IAM users, accessing the AWS console or CLI, and more. IAM user groups could also be made to quickly assign permissions to users. AWS had preconfigured roles for common job functions and custom job functions could also be created for unique roles within an organization.

Users outside of a red team may need access to systems during an exercise, such as white teams overseeing the security testing to ensure compliance with legal and policy requirements. This user role was assigned to the faculty overseeing this work. The ability to give read-only access to certain users helped ensure transparency and accountability while mitigating the potential for undetected misconfigurations and unexpected costs.

Unique IAM user accounts were created for each user's third-party tools that needed access. This allowed for the tracking of actions back to the user who performed them, as well as separating manual selection changes from those made by software or code. This configuration allowed for faster identification and debugging of programmatic errors with the code.

5.5 Setup and Configuration

User account configuration was necessary before deploying the IaC to AWS. The IAM User Accounts that were created for both human and programmatic operators needed roles to be assigned. While EC2 keys were generated with each run, account access keys needed to be generated for third-party applications, such as Terraform, so it could interact with AWS resources using the appropriate account.

A major benefit to using IaC for deployments was the limited number of software tools needed to deploy locally and the lightweight nature of those tools. At a minimum, only Terraform and Python need to be installed on the local host. Python was only required due to some of the Terraform provisioners implemented but was not necessarily essential for Terraform to function in all potential use cases. Additional local tools were needed to modify and fully leverage the code, such as Visual Studio Code, Ansible, and WSL.

5.6 Cloud Security and Compliance

For third-party applications to interact with AWS, access keys needed to be generated. Third-party applications like Terraform interacted with AWS using Application Programming Interface (API) calls and used these access keys to authenticate with the platform. The keys for the infrastructure were generated manually and did not expire. These access keys were stored locally in a shared configuration file outside of the code directories. This gave the advantage of allowing unique credentials and running the same code as other operators without having to modify the code. It also followed a security best practice of not hard coding access keys into the code.

The initial manual creation of EC2 key pairs within the AWS console introduced concerns regarding key expiration, rotation, leaked keys, and key management. One solution to this problem was to have Terraform create the key pairs at runtime, save the key locally to the host machine, and assign it to the infrastructure as it deployed. The key pair was then destroyed during the tear-down of the infrastructure. This had the advantage of creating new, unique key pairs during each deployment that could be automatically saved locally and to the cloud at runtime. This enforced non-repudiation and further assisted with debugging as discrepancies with the deployed infrastructure were identified.

To enforce non-repudiation, third-party applications were given unique access keys to separate IAM user accounts. Additionally, IAM user accounts for Terraform were restricted from accessing the AWS web browser console. This further hardened the account from misuse if programmatic access credentials were compromised due to enforcing principle of least privilege. Finally, multiple Terraform IAM user accounts were created so that interactions with AWS could also be traced back to the human user that ran the scripts, for example, Operator A's Terraform account, Operator B's Terraform account, etc. This assisted with conducting audits of deployments and understanding log activity to differentiate the actions made by a human user directly versus those made by the user's running of automated scripts.

AWS Config and Security Hub were enabled to verify hardening efforts against the NIST SP 800-53 Rev. 5 standard. Initial scans indicated a 28% security score with 4 critical, 19 high, 65 medium, and 11 low security findings. All critical and high findings were remediated through manual changes to account-wide console settings, Terraform and Ansible code modification, and by enabling AWS native services such as Amazon Macie, CloudTrail, GuardDuty, Inspector, and others. Some medium and low findings were accepted due to functionality constraints, software limitations, or other operational considerations. As a result, a final security score of 92% was achieved.

5.7 Optimizing Cost and Performance

By default, AWS billing details were only visible to the root user. Since it was a security best practice to perform the least number of actions as the root user, additional IAM users were given access to any billing details. Human user accounts were able to monitor costs and receive billing threshold alerts.

The Cost Summary report in the Billing and Cost Management console assisted in identification of potential configuration issues by comparing anticipated costs to recorded use. Additionally, types of instances deployed, total run time in hours per instance type, data throughput, and other key metrics were available, including current and estimated total costs for the month. The average costs for the first and continuing months is shown in Table 1.

Category	**First Month ($)**	**Additional Months ($)**
Compute Instances (EC2)	38.40	38.40
Networking & Data Transfer	13.44	13.44
Domain Registration & DNS	18.40	1.40
Total Cost	**70.24**	**53.24**

Table 1: Average Monthly AWS Infrastructure Costs

5.8 Timing

To evaluate the performance of launching the red team infrastructure, trials were run to determine the average time of deploying every tool and server in this red team infrastructure. It took on average between 12 and 13 minutes for Terraform to fully deploy during these tests. Destroying all resources took under five minutes for the entire infrastructure. Table 1 shows the timings for three deployments of the VPC, bastion host, RedELK, Evilginx, webserver, Gophish, Havoc as the C2 server, and an HTTP redirector. It is important to note that Sliver took less time to deploy than Havoc, but lacked a GUI for user interaction. On average, a Sliver deployment took between 7 and 8 minutes in total.

Run	**Apply**	**Destroy**
1	11 min 59 sec	4 min 29 sec
2	11 min 53 sec	4 min 47 sec
3	13 min 32 sec	4 min 55 sec

Table 2: Terraform Execution Times with Havoc as C2

6 Implications for Small Businesses

According to the Small Business Administration (SBA) [14], a small business is one that employs 500 or fewer employees. SMBs, sometimes referred to as Small and Medium-sized Enterprises (SME), compose 99.7% of private companies in the United States, employ 45.9% of private sector employees, and as of June

2023 are responsible for over 60% of new jobs created since 1995 [14]. As much as 43.5% of gross domestic product is the result of SMBs. With the significant portion of the private sector market that SMBs take, as well as the large impact SMBs have on supply chains, it is surprising that nearly half of SMBs fail within five years, and only around a quarter make it to fifteen years. As high as 66% of SMBs report at least one cyberattack over a two-year period, and as many as 60% of SMBs who fall victim to an attack are out of business within six months [12].

Cybersecurity incidents are one of the leading risks to businesses in the United States [12]. Cybersecurity risks exist across all sizes of business, but often SMBs may not be able to address these concerns like a larger company would. SMBs may lack the experience and expertise to recognize these risks, funding to assess and resolve them, or the general awareness to begin looking for them [8]. These smaller organizations are in a unique position where many lack the funds to both prepare for and respond to a cyberattack [12].

Cyberattacks are disproportionately affecting SMBs over larger organizations, all while having fewer resources and capabilities for preparing for and responding to them. For these reasons, it could be helpful for SMBs to have economical methods for self-assessing, identifying, and addressing cybersecurity risks within their organization not just in terms of finances, but also regarding resources such as equipment and time. Any efforts that can be taken to harden organizations against cyber threats and patch open avenues for cyberattacks should be taken to lessen the potential impact on the business's operations.

6.1 Benefits for Small and Medium Businesses

The initial focus was on exploring a potential solution for overextended red teams and other cybersecurity professionals that may lack the necessary resources to research and experiment with tools for conducting regular red team assessments. This could also be important for SMBs whose limited finances and staffing could be a limiting factor in the capability of performing vulnerability and other assessments on a smaller organization's infrastructure, particularly if in a larger supply chain. Use of automation tools like the ones explored here may provide opportunities for SMBs to begin addressing the cybersecurity risks they are exposed to while saving valuable and limited resources such as time, money, and equipment. Automated solutions for vulnerability assessments or red team operations could allow for a single or small number of IT staff to conduct some testing and self-assessments of infrastructure that would begin to increase the resilience of their networks.

7 Conclusion

The goal was to develop a proof-of-concept for an automated method for deploying a typical red team infrastructure using cloud resources quickly in a consistent, repeatable, and secure manner. The researchers accomplished this by using Terraform to deploy AWS cloud resources, automating the installation and deployment of the infrastructure using Terraform and Ansible, and continuously testing and optimizing the code to run in under 15 minutes.

The ease of deployment for a similar red team infrastructure could help both government and private sector industries, contractors, and agencies. It might be especially useful when facing shortages of resources to address cybersecurity risks, such as a lack of funds and staffing many SMBs face. This could increase their security and readiness while reducing overhead in running security assessments and red team exercises, ultimately increasing the effectiveness and insights gained to increase cybersecurity defensive postures.

7.1 Future Work

Additional opportunities to further this research have been considered. Future work could expand on the software and code. Allocating resources to fine tune and enhance the proposed solution may lead to a more refined and capable solution. The tools selected were examples of what could be used, but not intended to be an all-inclusive list of what might be deployed. Additional automation solutions could be investigated. For example, methods that would allow a SIEM to trigger tainting and replacement of certain resources if blue team activity was suspected. Identifying areas to optimize and decrease deployment times and manual triggers may decrease time to engagement if using similar automation tools for much larger-scale deployments.

Research to explore different operator hardware and creating physical prototype tools could be studied. The solution proposed relied on cloud providers to do the heavy lifting, ultimately offering lightweight local requirements for running the IaC. There could be an opportunity to explore using the developed ideas as a portable solution for a red team. Hardware could be identified for attempting to reduce the footprint of the host machine by running the code on a variety of devices, such as phones, tablets, or single-board computers. This could create interesting opportunities to research solutions for red teams trying to remain mobile, or for clandestine environments.

Another opportunity could be to explore containerization. This was deferred due to the complexity it may introduce. It could prove difficult to rebuild containers with repeated or regular changes, or the failure of a container affecting other containers [5]. Additionally, it takes more effort to secure containers versus Virtual Machines which may be time consuming [5], whereas virtual machines operate more similarly to traditional hardware. Research into how any overhead or complexity with container image layers might be reduced could provide alternatives and introduce other benefits that could be helpful to red teams.

Finally, research could be performed to automate a penetration testing and vulnerability assessment platform for teams needing to do broader assessments or that may lack the resources to dedicate teams to full emulated engagement exercises. This could be especially useful for smaller or less experienced teams. Penetration testing and vulnerability scanning could result in a higher number of vulnerabilities being detected, and save time versus running full emulated engagements.

In summary, there are numerous avenues for expanding and refining the project. From enhancing the code, to exploring portable hardware solutions, future work could significantly broaden its applicability or take it from proof-of-concept to functional tool. By investigating these possibilities, researchers could further develop adaptable, efficient tools that meet the evolving needs of red teams and cybersecurity professionals across a variety of environments and industries.

References

[1] Butt, U. A., Amin, R., Aldabbas, H., Mohan, S., Alouffi, B., and Ahmadian, A. “Cloud-based email phishing attack using machine and deep learning algorithm”. In: *Complex & Intelligent Systems* 9.3 (2023), pp. 3043–3070.

[2] Chaplinska, S. “A purple team approach to attack automation in the cloud native environment”. MA thesis. Aalto University, 2022.

[3] Culbert, M. *Sliver vs Havoc - Two Adversary Emulation Frameworks.* `git.culbertreport.com/posts/Sliver-vs-Havoc`.

[4] Decraene, J., Zeng, F., Low, M. Y. H., Zhou, S., and Cai, W. “Research advances in automated red teaming”. In: *Proceedings of the 2010 Spring Simulation Multiconference.* 2010, pp. 1–8.

[5] Deochake, S., Maheshwari, S., De, R., and Grover, A. “Comparative Study of Virtual Machines and Containers for DevOps Developers”. In: *SSRN Electronic Journal* (Jan. 2023). DOI: `10.2139/ssrn.4404383`.

[6] Hashicorp. *What is Terraform?* `developer.hashicorp.com/terraform/intro`.

[7] Landauer, M., Mayer, K., Skopik, F., Wurzenberger, M., and Kern, M. "Red team redemption: A structured comparison of open-source tools for adversary emulation". In: *2024 IEEE 23rd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE. 2024, pp. 117–128.

[8] Mierzwa, S. and Klepacka, A. "Practical Approaches and Guidance to Small Business Organization Cyber Risk and Threat Assessments". In: *Journal of Strategic Innovation and Sustainability* 18.2 (2023), p. 29.

[9] Nair, A. *HSC24RedTeamInfra*. `github.com/dazzyddos/HSC24RedTeamInfra`.

[10] Outflank. *RedELK*. `https://github.com/outflanknl/RedELK`.

[11] Pinto, R. F. M. "Infrastructure as Code for Cybersecurity Training". MA thesis. Universidade do Porto (Portugal), 2023.

[12] Salzman, N. "The Cybersecurity Framework's Most Vulnerable User". In: *Journal of Information Warfare* 22.4 (2023), pp. 53–71.

[13] U.S. Digital Service. *Agile Overview*. `https://techfarhub.usds.gov/pre-solicitation/agile-overview/`.

[14] U.S. Small Business Administration. *Frequently Asked Questions About Small Business, 2024*. `https://advocacy.sba.gov/2024/07/23/frequently-asked-questions-about-small-business-2024/`.

[15] Ubuntu Wiki. *Minimal*. `https://wiki.ubuntu.com/Minimal`.

Evaluating the Limitations of Local LLMs in Solving Complex Programming Challenges [*]

Kadin Matotek, Heather Cassel, Md Amiruzzaman, Linh B. Ngo
Computer Science Department
West Chester University, West Chester, PA
{KM998744,HC946859,mamiruzzaman,lngo}@wcupa.edu

Abstract

This study examines the performance of today's open-source, locally hosted large-language models (LLMs) in handling complex competitive programming tasks with extended problem descriptions and contexts. Building on the original Framework for AI-driven Code Generation Evaluation (FACE), the authors retrofit the pipeline to work entirely offline through the Ollama runtime, collapsing FACE's sprawling per-problem directory tree into a handful of consolidated JSON files, and adding robust checkpointing so multi-day runs can resume after failures. The enhanced framework generates, submits, and records solutions for the full Kattis corpus of 3,589 problems across eight code-oriented models ranging from 6.7 billion to 9 billion parameters. The submission results show that the overall pass@1 accuracy is modest for the local models, with the best models performing at approximately half the acceptance rate of the proprietary models, Gemini 1.5 and ChatGPT-4. These findings expose a persistent gap between private, cost-controlled LLM deployments and state-of-the-art proprietary services, yet also highlight the rapid progress of open models and the practical benefits of an evaluation workflow that organizations can replicate on in-house hardware.

[*]Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

1 Introduction

Advances in Artificial Intelligence (AI) have led to the creation of some highly sophisticated large language models (LLMs). An LLM is an AI-based model that generates human-like text to perform various language-related tasks. Some of them, including the family of ChatGPT models and Gemini, are known to excel in code generation. These models have demonstrated the potential to automate certain software development tasks [1, 2, 3, 4]. However, their cloud-based and proprietary nature raises concerns such as data privacy, latency, and cost. Token quotas and rate limits hinder large-scale experiments, and usage fees can skyrocket when benchmarking thousands of problems. To avoid these issues, organizations now deploy open-source models locally. Yet, evaluations of their performance on complex coding tasks remain scarce.

Cloud-based models have been extensively benchmarked using competitive programming platforms [5, 6, 7, 8] and real-world tasks [9, 10]. The performance of local models has not yet been rigorously evaluated in similar contexts. In cases where local models have been evaluated [11, 12, 13], problem datasets typically lack quantity and variety. As a result, it becomes difficult to assess the models' full problem-solving potential. This study evaluates several locally hosted LLMs through the Ollama platform [14] on Kattis' extensive collection of programming challenges [15]. Kattis is a publicly available platform with more than 3,500 coding problems of varying difficulty and is widely used to evaluate programming proficiency.

In this study, we build on the work of [6] by extending the Framework for AI-driven Code Generation Evaluation (FACE) and applying it to the same comprehensive set of Kattis problems. Whereas [16] evaluated proprietary ChatGPT-4 and Gemini 1.5, we focus exclusively on locally hosted LLMs. By comparing our results with previous work, we aim to emphasize both the improvements and limitations of local LLMs in complex code generation tasks. This comparative analysis informs future research and industry practice about the trade-offs between local and cloud-based LLM solutions. To the best of our knowledge, no research has evaluated the performance of code generated by such a large number of open-source LLMs on a high volume of complex coding problems.

The paper is structured as follows. Section 2 reviews prior work. Section 3 details our FACE framework extension. Section 4 describes the experimental setup. Section 5 presents the results. Section 6 concludes and outlines future directions.

2 Related Work

Several studies have examined both local and proprietary LLMs on code-related tasks: performance on coding platforms [6, 8, 11, 16, 17], security of their suggestions [18, 19, 20], and bugs in generated code [21]. For example, Coignion et al. [11] conducted an extensive evaluation of 18 code-oriented LLMs using a dataset of 204 Leetcode problems. Their experiments generated over 210,120 solutions—with GitHub Copilot contributing 2,040 solutions and eight open-source models yielding 12,240 each. Of the top three models, StarCoder achieved a pass@1 of 0.095 and a pass@10 of 0.132. CodeLlama scored 0.093 on pass@1 and 0.201 on pass@10, and GitHub Copilot followed closely with scores of 0.092 and 0.196, respectively. The pass@k metrics represent the probability that at least one of k solution attempts will succeed. The results indicate that open-source LLMs are capable of competitive performance. In some cases, their performance is comparable to industry-leading solutions such as Copilot when solving programming problems.

One of the limitations of these studies is that their problem sets often come from popular coding challenge sources such as LeetCode or Codeforces. This means that the problems' descriptions and corresponding solutions are well-known and often publicly discussed. On LeetCode, every problem includes a discussion tab, and solutions are often mirrored to GitHub and analyzed in blog posts. Codeforces publishes official solutions within hours or days, and these are widely shared outside the site. In contrast, Kattis prohibits sharing solutions and routinely checks for plagiarism. As a result, solutions are scarce online, making them underrepresented in training data [22]. Large LLMs excel at questions based on widely available knowledge, but struggle with long-tail information [23]. To our knowledge, only [16, 24] have evaluated online LLMs such as Gemini and ChatGPT using a small subset of Kattis problems and their built-in evaluation system.

Ngo et al. [6] introduce FACE (Framework for AI-driven Coding Generation Evaluation), a three-stage pipeline (Miner, Generator, Submitter) that automates end-to-end benchmarking of code-generation models on Kattis. In this work, we leverage and extend FACE to utilize local LLMs to generate solutions for more than 3,500 Kattis problems. This allows us to address the limitation noted above and extensively study how effective local LLMs are at solving complex competitive programming challenges.

3 Framework Extension

In extending the original FACE architecture [6] for our study, we improved the system's data organization, added support of locally hosted LLMs, and altered the submission process to Kattis for a more scalable and fault-tolerant approach. This section outlines how our modifications enable the evaluation of over 3,500 coding problems.

3.1 FACE Architecture

FACE [6] automates collecting thousands of programming problems from a coding-challenge platform, querying an AI service for solutions, and submitting those solutions back for evaluation. Figure 1 represents the original FACE framework.

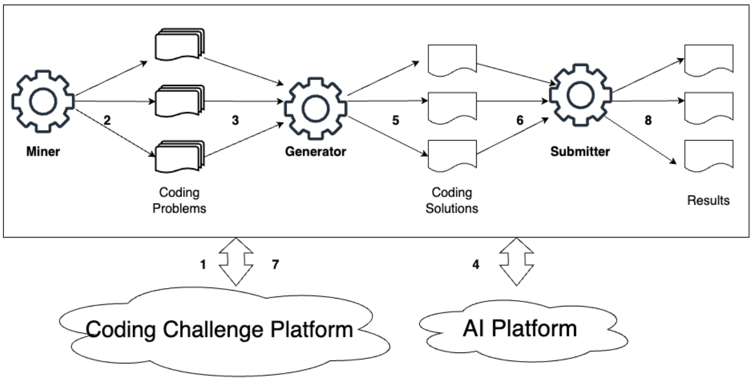


Figure 1: Original FACE architecture, with numbers indicating each step in the process.

3.2 Change in Data Organization

Table 1 summarizes both FACE's original structure and our new JSON-based implementation. FACE's original architecture stored each problem in its own directory, with subfolders for metadata, problem statements, test cases, and submission logs. Although straightforward, this doesn't scale: processing more than 3,500 problems across just two models produced over 25,000 individual .txt/.py files plus a similar number of folders. To address this, we adopted a JSON-based design that consolidates all problem data, model outputs, and submission results into just 17 JSON files. There is one file for the problem data and two files per model—one for outputs and one for the submission results. This reduced the file count by roughly 99.9%, even as we scaled from two to eight models. Consolidating into a handful of JSON files simplifies maintainability and version control, improves performance, and increases portability.

3.3 Integration of Locally Hosted Models

Our extension departs from the original FACE framework by integrating locally run LLMs, rather than relying on cloud-based APIs. The enhanced "Generator" component was modified to accommodate locally hosted models via a standardized interface. In this phase, each model receives a standard JSON payload (`kattis_problems.json`) containing problem descriptions and test cases extracted using FACE. The responses for each model were stored in separate JSON files.

Original FACE Structure	**JSON–Based Implementation**
One directory per problem:	Two JSON files per model:
• metadata file • problem text (.txt) • test cases (.txt) • results (logs) • submissions (code)	• `kattis_problems.json`: all metadata, statements, and test cases • `solutions_<model name>.json`: full responses + code • `submissions_<model name>.json`: solution statuses from Kattis

Table 1: Comparison of the original FACE directory structure with our consolidated JSON-based implementation.

3.4 Improvements in the Submission Process

To transition from solution generation to evaluation, we extended the "Submitter" component of the FACE architecture. This new submission process emphasizes consistency and recoverability. Key aspects of our enhancements include:

- **Atomic Submission and Results Storage:** For each problem, the generated Python solution is written to a temporary file and submitted via the official Kattis API. The system uses atomic file operations: The temporary results file is flushed and synced to disk before being renamed to its final JSON output. This careful writing procedure minimizes data loss by ensuring that each submission's result is reliably stored.

- **Checkpointing:** Because submissions for each model took several days to complete, there was a risk of data corruption during the process. To address this overhead, we implemented a checkpoint mechanism. A designated checkpoint file maintains the identifier of the last successfully processed problem. Upon interruption, the process reads the checkpoint to resume from where it left off. This prevents reprocessing problems that have already been evaluated. This checkpointing strategy greatly enhances the framework's resilience to any disruptions.

4 Experimentation

In this section, we detail the experimental setup used to benchmark eight locally hosted LLMs on more than 3,500 Kattis problems using our extended FACE pipeline. We begin by describing the selected models and summarizing their training characteristics. Next, we outline the computing environment and hardware configuration. Finally, we explain how the solutions were generated, submitted, and collected for evaluation.

4.1 Model Descriptions

We selected models with 6.7–9 billion parameters so that they could run on small-scale servers. Although we conducted experiments on a departmental GPU server, we also tested on the authors' machines with consumer-grade GPUs (8 GB VRAM) and observed only negligible slowdowns in solution generation. Our choices were based on models available on the Ollama platform. Table 2 summarizes the eight local LLMs chosen in our study.

Model	Params (B)	Size (GB)	Pre-training	Context (K)
CodeLlama [25]	7.0	3.8	500 B	16
CodeQwen [26]	7.0	4.2	90 B	64
DeepSeek-Coder [27]	6.7	3.8	2 T	16
DolphinCoder [28]	7.0	4.2	—	16
Granite-Code [29]	8.0	4.6	4.5 T	125
Llama 3.1 [30]	8.0	4.9	15.6 T	128
Qwen2.5-Coder [31]	7.0	4.7	5.2 T	32
Yi-Coder [32]	9.0	5.0	3.1 T	128

Table 2: Summary of local LLMs: parameter count (in billions of parameters), model size (in gigabytes), total pre-training (in tokens), and context window size (in thousands of tokens).

To understand their design choices, it is helpful to know which models are built on existing foundations versus those trained entirely from scratch. CodeLlama was initialized with Llama 2 weights and fine-tuned on 500 B code tokens using an infilling objective. Llama 3.1 belongs to the newer Llama 3 family, pre-trained on 15.6 T multilingual tokens and further aligned through supervised fine-tuning and direct preference optimization. CodeQwen continues pre-training of the original Qwen base model on 90 B mixed text–code tokens, optimized for code generation with Flash Attention [33]. Qwen2.5-Coder builds on the Qwen 2.5 architecture and is further pre-trained on a 5.2 T token mix of 70% code, 20% text–code grounding data, and 10% math data. DolphinCoder extends StarCoder2 [34] and is trained on the LeetCode Rosetta dataset [35] for competitive programming. DeepSeek-Coder and Granite-Code are trained from scratch: DeepSeek-Coder on 2 T tokens (87% code, 10% English code text, 3% Chinese) and Granite-Code on 4.5 T tokens (8% code, 20% natural language). Finally, Yi-Coder was trained from scratch on a 3.1 T token bilingual (English/Chinese) web crawl with extensive filtering and deduplication.

4.2 Running the Experiments

The experiment was carried out on the department's GPU server, which runs Ubuntu 22.04 LTS. The server consists of dual Intel Xeon 6426Y CPUs (64 cores total; 56 available), dual NVIDIA L4 GPUs, each with 24 GB of memory, 256 GB of RAM, and 23 TB of SSD storage. To interact with the LLMs, we used Ollama version 0.3.4 and CUDA version 12.8.

In the first phase of the experiments, the modified FACE platform contacted an external Ollama server and instructed it to load a specific LLM. Next, it used the loaded model to generate solutions for all Kattis problems. Lastly, FACE gradually submitted them to Kattis and collected the results. The total runtime for these experiments exceeded three weeks.

5 Performance Evaluation

This section presents a detailed evaluation of model performance across three dimensions: solution generation speed, correctness (*Accepted* status), and failure types (e.g., *Wrong Answer*, *Run Time Error*). We analyzed results across problem difficulties (Easy, Medium, and Hard) to uncover trends and limitations of locally hosted LLMs.

5.1 Generation Time Comparison

To assess efficiency, we measured response generation times for all 3,589 problems. Table 3 presents summary statistics, Figure 2 shows generation time histograms per model, and Figure 3 shows the corresponding cumulative distribution functions (CDF).

5.2 Outliers

A small fraction of model–problem pairs exhibited exceptionally large generation times (see Table 3). To prevent these rare spikes from dominating our future histograms and CDFs (Figs. 2 and 3), we applied the conventional box plot rule

Table 3: Response generation time (in seconds) for each model on the full dataset, ordered by parameter size. Models with the two highest acceptance rates are shown in bold.

Model Name	**Mean (s)**	**Median (s)**	**Std (s)**	**Min (s)**	**Max (s)**
DeepSeek-Coder	15.71	12.89	62.48	1.38	2163.78
CodeLlama	16.11	10.84	77.94	0.82	1815.62
CodeQwen	10.48	8.38	50.09	0.70	1738.14
DolphinCoder	20.02	10.59	117.78	0.59	1775.38
Qwen2.5-Coder	14.82	14.00	31.70	2.24	1890.25
Granite-Code	8.35	7.39	6.21	0.26	216.92
Llama3.1	12.53	11.64	4.88	3.08	58.62
Yi-Coder	14.64	12.72	50.59	1.41	2149.67

and discarded any outliers.

For each model's response times, we computed the first and third quartiles (Q_1, Q_3), to form the interquartile range:

$$\text{IQR} = Q_3 - Q_1,$$

and labeled any value outside the range given by the 1.5 IQR Rule:

$$\left[\,Q_1 - 1.5\,\text{IQR},\ Q_3 + 1.5\,\text{IQR}\right]$$

as an outlier. The number of excluded outliers per model is reported in Table 4.

Table 4: Number of generation-time outliers excluded per model, sorted from highest to lowest. Models with the two highest acceptance rates are shown in bold

Model Name	**# Outliers**
Llama3.1	149
DolphinCoder	147
CodeLlama	127
Granite-Code	117
DeepSeek-Coder	114
CodeQwen	113
Yi-Coder	76
Qwen2.5-Coder	65

Although Llama3.1 has the lowest maximum generation time (58.62 s; Table 3), it exhibits the most outliers (149; Table 4). This occurs because its response times are tightly clustered, as shown by a low standard deviation. This yields a small interquartile range. As a result, even modest spikes exceed the outlier threshold. In contrast, Yi-Coder and Qwen2.5-Coder—the two best-performing models by acceptance rate—had the fewest outliers, 76 and 65, respectively.

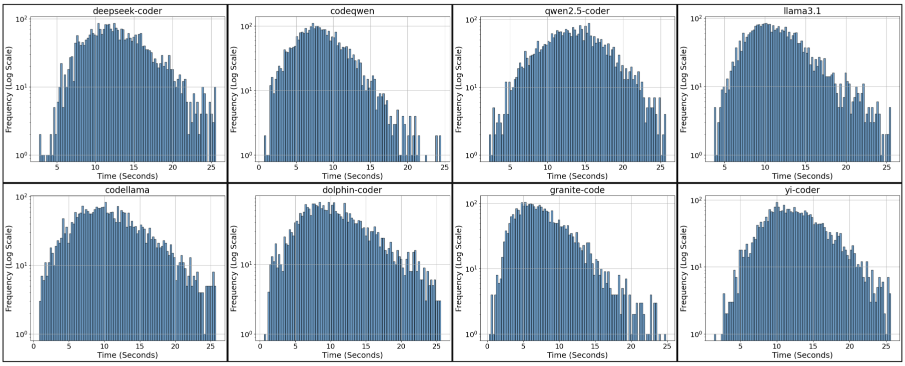


Figure 2: Histograms of generation times (log-scale frequency) for each model.

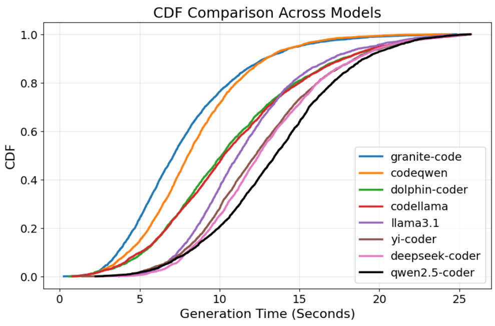


Figure 3: CDF comparison of generation time across models.

Figure 3 illustrates the Cumulative Distribution Function (CDF) for the generation times of different models. Each curve represents the CDF of a specific model, showing the proportion of solutions generated within a given time threshold. The x-axis corresponds to the generation time (in seconds), while the y-axis represents the CDF, ranging from 0–1, where 1 indicates that all solutions were generated within that time.

From the plot, it is evident that models such as Qwen2.5-Coder, DeepSeek-Coder, and Yi-Coder exhibited the slowest solution generation times, with their CDF curves increasing more gradually and extending further along the x-axis. Despite their slower speeds, these models performed the best, with average acceptance rates of 5.7%, 2.9%, and 5.4%, respectively, as shown in Table 8.

This comparison underscores the trade-off between computational efficiency and generation speed in these models.

5.3 Correctness and Failure Analysis

First, we assessed how each model's submissions performed in terms of correctness and common failure modes. For every problem difficulty level (Easy, Medium, and Hard), we counted the number of submissions that resulted in an accepted solution, as well as those that failed due to compilation errors, wrong answers, run-time errors, time-out, or time/memory limits. This breakdown helped us identify not only which models could solve a given fraction of problems, but also the types of mistakes they most frequently made. Tables 5–7 summarize these counts for each model across the three difficulty tiers.

Table 5: Outcome counts per difficulty by model (Easy), ordered by model size. The two top-performing models are shown in bold.

	Problem Difficulty: Easy							
Status	deepseek	codellama	codeqwen	dolphin	**qwen2.5**	granite	llama3.1	**yi-coder**
Accepted	90	16	49	21	157	14	91	139
Compile Error	27	210	21	92	2	28	8	20
Wrong Answer	381	245	326	370	323	485	318	368
Run Time Error	114	145	209	127	120	82	185	82
Time Limit Exceeded	8	4	8	6	17	2	18	10
Memory Limit Exceeded	0	0	0	0	1	0	0	1
Total	620	620	613	616	620	611	620	620

Table 6: Outcome counts per difficulty by model (Medium), ordered by model size. The two top-performing models are shown in bold.

	Problem Difficulty: Medium							
Status	deepseek	codellama	codeqwen	dolphin	**qwen2.5**	granite	llama3.1	**yi-coder**
Accepted	15	1	5	1	47	4	8	52
Compile Error	72	392	59	216	12	70	17	37
Wrong Answer	770	506	543	717	719	993	645	819
Run Time Error	375	362	608	311	422	184	549	300
Time Limit Exceeded	38	17	48	25	74	10	55	58
Memory Limit Exceeded	8	0	4	1	4	2	4	12
Total	1278	1278	1267	1271	1278	1263	1278	1278

Table 7: Outcome counts per difficulty by model (Hard), ordered by model size.

	Problem Difficulty: Hard							
Status	deepseek	codellama	codeqwen	dolphin	qwen2.5	granite	llama3.1	yi-coder
Accepted	0	0	0	0	0	0	1	1
Compile Error	128	475	95	291	18	85	35	80
Wrong Answer	985	719	646	853	924	1279	833	1096
Run Time Error	512	478	870	511	615	290	773	418
Time Limit Exceeded	48	16	60	28	120	15	43	75
Memory Limit Exceeded	11	0	6	1	12	2	5	18
Total	1684	1688	1677	1684	1689	1671	1690	1688

Note: Some difficulty totals differ from the full problem set size because we filtered out any statuses outside the six tracked categories (*Accepted*, *Compile Error*, *Memory Limit Exceeded*, *Run Time Error*, *Time Limit Exceeded*, *Wrong Answer*).

For **Easy problems**, all models performed modestly, with Qwen2.5-coder and Yi-Coder obtaining the highest number of accepted submissions, with 157 and 139, respectively. However, even at this level, the majority of attempts resulted in *Wrong Answers* or *Run Time Errors*, suggesting difficulty in reliably solving even simpler problems.

As difficulty increased, performance deteriorated significantly. For **Medium problems**, most models achieved fewer than 10 accepted solutions; only Yi-Coder and Qwen2.5-coder showed slight improvements, 52 and 47, respectively. *Wrong Answer* and *Run Time Error* rates were consistently high across all models, indicating challenges in handling intermediate problem complexity.

On **Hard problems**, no models could reliably generate correct solutions. Only Yi-Coder and Llama3.1 managed to produce even a single accepted response. This result underscores a significant performance ceiling for current local LLMs in high-difficulty settings.

5.4 Comparison with Cloud-Based Models

Before presenting the acceptance rates of our local models, we recap the cloud-based benchmarks reported by Ngo et al. [6]. In their evaluation of 1,981 Kattis problems, Gemini 1.5 achieved an acceptance rate of 10.9% (217/1,981) and ChatGPT-4 achieved 10.7% (211/1,981). When these results are broken down by problem difficulty (on a 1.0–10.0 scale), they found that among the 95 problems rated 1.0–2.0, 68% (65/95) were accepted. This fell to 21% (48/225) in the 2.0–4.0 range, and acceptance was effectively zero beyond the 4.0 rating. Their findings illustrate how performance drops off rapidly as problem difficulty increases.

The acceptance rates reported in Table 8 correspond to our **pass@1** metric, i.e., the probability that a single generated solution for a problem is correct. Formally, for any $k \geq 1$,

$$\text{pass@}k = \mathbb{E}_{\text{Problems}}\left[1 - \frac{\binom{n-c}{k}}{\binom{n}{k}}\right] \tag{1}$$

where

- $\mathbb{E}_{\text{Problems}}$ denotes the expectation taken over the distribution of problems
- n is the total number of samples generated per problem

Table 8: Acceptance rates for cloud-based and local models, sorted from highest to lowest.

Model	**AC Rate (%)**
Cloud-based (n = 1,981)	
Gemini 1.5	10.9
ChatGPT-4	10.7
Local models ($n = 3,589$)	
Qwen2.5-Coder	5.7
Yi-Coder	5.4
DeepSeek-Coder	2.9
Llama3.1	2.8
CodeQwen	1.5
DolphinCoder	0.6
CodeLlama	0.5
Granite-Code	0.5

- c is the number of those samples that pass all tests
- k is the number of samples considered

In the special case $k = 1$, equation (1) simplifies to

$$\text{pass@1} \;=\; \mathbb{E}_{\text{Problems}}\left[\frac{c}{n}\right],$$

so Table 8's acceptance rates measure the average fraction of problems for which the model's first attempt succeeded.

6 Conclusion

In this study, we evaluated eight open-source LLMs against more than 3,500 programming problems. We extended the FACE framework to work with Ollama, consolidated all data into JSON files, and added a checkpointing system. This infrastructure enabled large-scale testing and direct comparison with previous cloud-based results. The local models with the highest performance, Qwen2.5-Coder and Yi-Coder, achieved pass@1 rates of 5.7% and 5.4%, approximately half the pass@1 rates of ChatGPT and Gemini. The trade-off here is that local models can be run an unlimited number of times compared with the token limitation or the monetary cost of the proprietary LLM solution.

Our findings underscore both the promise and the current limitations of locally hosted LLMs for complex code generation. Going forward, we will look at fine-tuning these models on task-specific datasets, combining local and cloud-based inference in hybrid workflows, and using smarter prompt designs or built-in debugging steps to improve the correct submission rate. As the open-source model landscape continues to grow, benchmarks become increasingly vital. Large and thorough evaluations guide model improvements and inform deployment decisions. This is especially important in settings where data privacy or limited computing resources are key concerns.

Beyond the raw accuracy numbers, our findings carry direct weight for educators seeking to modernize assessment. Local code-centric LLMs can power explain-as-you-grade autograders that return step-wise diagnostics instead of the usual binary pass/fail, while keeping student data safely on campus hardware. Lightweight Ollama deployments lower the cost threshold for in-IDE tutoring agents that gently assist novice programmers without handing them the full solution. Instructors gain the freedom to iterate rubrics rapidly, run large-batch regrading overnight, and even spin up model variants tuned to specific curricula. Taken together, the results point toward a future where scalable private feedback loops complement traditional office hours rather than replace them.

References

[1] Antonis Antoniades et al. *SWE-Search: Enhancing Software Agents With Monte Carlo Tree Search And Iterative Refinement.* 2025. arXiv: `2410.20285 [cs.AI]`.

[2] Feng Lin, Dong Jae Kim, et al. *When LLM-Based Code Generation Meets The Software Development Process.* 2024. arXiv: `2403.15852 [cs.SE]`.

[3] Sanwal Manish. “An Autonomous Multi-Agent LLM Framework For Agile Software Development”. In: *International Journal Of Trend In Scientific Research And Development* 8.5 (2024), pp. 892–898.

[4] Giriprasad Sridhara, Sourav Mazumdar, et al. *ChatGPT: A Study On Its Utility For Ubiquitous Software Engineering Tasks.* 2023. arXiv: `2305.16837 [cs.SE]`.

[5] Nora Dunder et al. “Kattis Vs. ChatGPT: Assessment And Evaluation Of Programming Tasks In The Age Of Artificial Intelligence”. In: *Proceedings of the 14th Learning Analytics and Knowledge Conference.* 2024, pp. 821–827.

[6] Bao Ngo et al. “FACE: A Framework For AI-Driven Coding Generation Evaluation”. In: *Journal of Computing Sciences in Colleges* 40.3 (2024), pp. 263–276.

[7] Nikolaos Nikolaidis et al. “The End Of An Era: Can AI Subsume Software Developers? Evaluating ChatGPT And Copilot Capabilities Using LeetCode Problems”. In: *Evaluating ChatGPT And Copilot Capabilities Using LeetCode Problems* (2023).

[8] Kady Yildiz. “Assessing AI’s Problem-Solving Capabilities Within Programming”. In: *LU-CS-EX* (2024).

[9] Tom Brown et al. “Language Models Are Few-Shot Learners”. In: *Advances in Neural Information Processing Systems 33*. 2020.

[10] Yongliang Shen et al. “HuggingGPT: Solving AI Tasks With ChatGPT And Its Friends In Hugging Face”. In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh et al. Vol. 36. Curran Associates, Inc., 2023, pp. 38154–38180.

[11] Tristan Coignion, Clément Quinton, and Romain Rouvoy. “A Performance Study of LLM-Generated Code on LeetCode”. In: *Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering*. Salerno, Italy: Association for Computing Machinery, 2024, pp. 79–89.

[12] Débora Souza et al. *Code Generation With Small Language Models: A Deep Evaluation On Codeforces*. 2025. arXiv: `2504.07343 [cs.SE]`.

[13] Jonas F. Tuttle et al. “Can LLMs Generate Green Code - A Comprehensive Study Through LeetCode”. In: *2024 IEEE 15th International Green and Sustainable Computing Conference (IGSC)*. 2024, pp. 39–44. DOI: `10.1109/IGSC64514.2024.00017`.

[14] Ollama. *Ollama: Local Large Language Model Inference Platform*. 2025.

[15] Kattis. *Kattis: An Online Judge For Programming Problems*. 2025.

[16] Nguyen Ho et al. “Predicting ChatGPT’s Ability to Solve Complex Programming Challenges”. In: *2024 IEEE International Conference on Big Data (BigData)*. 2024.

[17] Md Motaleb Hossen Manik. *ChatGPT Vs. DeepSeek: A Comparative Study On AI-Based Code Generation*. 2025. arXiv: `2502.18467 [cs.SE]`.

[18] Ran Elgedawy et al. *Occassionally Secure: A Comparative Analysis Of Code Generation Assistants*. 2024. arXiv: `2402.00689 [cs.CR]`.

[19] Hammond Pearce et al. “Asleep At The Keyboard? Assessing The Security Of GitHub Copilot’s Code Contributions”. In: *Communications of the ACM* 68.2 (2025), pp. 96–105.

[20] Neil Perry et al. “Do Users Write More Insecure Code With AI Assistants?” In: *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 2023, pp. 2785–2799.

[21] Kevin Jesse et al. “Large Language Models And Simple, Stupid Bugs”. In: *2023 IEEE/ACM 20th International Conference on Mining Software Repositories (MSR)*. IEEE. 2023, pp. 563–575.

[22] Kai Sun et al. *Head-to-Tail: How Knowledgeable Are Large Language Models (LLMs)? AKA Will LLMs Replace Knowledge Graphs?* 2024. arXiv: `2308.10168 [cs.CL]`.

[23] Nikhil Kandpal et al. "Large Language Models Struggle to Learn Long-Tail Knowledge". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 15696–15707.

[24] Nghia D Tran et al. "Exploring ChatGPT's Ability To Solve Programming Problems With Complex Context". In: *Journal Of Computing Sciences In Colleges* 39.3 (2023), pp. 195–209.

[25] Baptiste Rozière et al. *Code Llama: Open Foundation Models for Code*. 2024. arXiv: `2308.12950 [cs.CL]`.

[26] Jinze Bai et al. *Qwen Technical Report*. 2023. arXiv: `2309.16609 [cs.CL]`.

[27] Daya Guo et al. *DeepSeek-Coder: When the Large Language Model Meets Programming – The Rise of Code Intelligence*. 2024. arXiv: `2401.14196 [cs.SE]`.

[28] Cognitive Computations. *DolphinCoder 7B*. `https://huggingface.co/datasets/cognitivecomputations/dolphin-coder`. 2024.

[29] Mayank Mishra et al. *Granite Code Models: A Family of Open Foundation Models for Code Intelligence*. 2024. arXiv: `2405.04324 [cs.AI]`.

[30] Aaron Grattafiori et al. *The Llama 3 Herd of Models*. 2024. arXiv: `2407.21783 [cs.AI]`.

[31] Binyuan Hui et al. *Qwen2.5-Coder Technical Report*. 2024. arXiv: `2409.12186 [cs.CL]`.

[32] 01. AI et al. *Yi: Open Foundation Models by 01.AI*. 2025. arXiv: `2403.04652 [cs.CL]`.

[33] Tri Dao et al. "FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness". In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo et al. Vol. 35. Curran Associates, Inc., 2022, pp. 16344–16359.

[34] Anton Lozhkov et al. *StarCoder 2 and The Stack v2: The Next Generation*. 2024. arXiv: `2402.19173 [cs.SE]`.

[35] Erich Hartford. *LeetCode Rosetta Dataset*. `https://www.kaggle.com/datasets/erichartford/leetcode-rosetta`. Accessed: 2025-05-27. 2023.

Common Container-Based Infrastructure Blueprints for Multi-Course Computing Education *

Linh B. Ngo[1], Huy D. Nguyen [2], Bao G. Ngo[2], Tejas Karusala [3]
[1]West Chester University, West Chester, PA
lngo@wcupa.edu
[2]Oberlin College, Oberlin, OH
bngo,hnguye@oberlin.edu
[3]Downingtown East Highschool, West Chester, PA
tkarusala01@student.dasd.org

Abstract

This work describes a systematic approach to refactoring and redesigning container-based infrastructure blueprints to support various courses including full-stack web development, machine learning/AI, cloud computing, parallel and distributed computing, and big data engineering. Leveraging Docker Compose and Docker Swarm, these blueprints can be deployed on personal computers or public clouds. In the absence of adequate on-site computing resources, these blueprints will provide students and instructors with portable and reproducible computing environments inside and outside of the classroom. The usability of these blueprints is demonstrated in several different courses taught at an R2 regional public university.

1 Introduction

The growing diversity and complexity of computing courses call for flexible and scalable infrastructures for instruction. At institutions with limited technical

*Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

staff and inconsistent access to centralized computing resources, delivering a uniform student experience across such diverse domains remains a challenge. While commercial cloud platforms offer one solution, the recurring costs, steep learning curves, and administrative overhead can be prohibitive for smaller programs.

This paper presents a generalized infrastructure framework based on containerization, extending the ideas in existing literature on portable computing resources for parallel and distributed computing education into a modular architecture adaptable across different course contexts. We demonstrate how a single Docker Compose YAML specification template file can be expanded to describe the deployment of both frontend learning tools (e.g., Jupyter, VSCode, etc.) and backend computing services (e.g., MPI, Spark, MongoDB, Ollama, etc.) tailored to each course. By encapsulating lecture content, toolchains, and runtime dependencies, this approach provides a reproducible environment that instructors can customize and students can deploy across personal laptops or cloud servers.

Through course case studies and real-world usage, we show how this unified infrastructure template has enabled instructional scalability while lowering the barrier of entry for both faculty and students. The long-term goal is to support a sustainable and portable computing ecosystem for education that adapts to evolving curricular needs without imposing excessive operational complexity.

The remainder of the paper is organized as follows. Section 2 reviews existing portable/containerized computing environments for education. Section 3 discusses our past teaching experience and highlights limitations of previous approaches. Section 4 describes the generalized multi-container framework, and section 5 demonstrates how this framework can be adapted to different courses. Section 6 discusses the technical issues encountered throughout our own adaptation process, lesson learned, and future design approach. Section 7 concludes the paper.

2 Background

Portable laboratories have gradually replaced institution-hosted clusters in systems and data-centric courses. Earlier solutions used virtual images, such as the SEED Lab for computer security education [4]. One of the limitations of SEED Lab is the large memory and storage requirements. At the time, 12G of storage was not trivial given the limitations of computer hardware available at that time. In [13], the authors proposed the idea of *BYOD*, bring-your-own-device, where collections of virtual images that can support multiple core CS courses were made available to students to download and set up in lieu of centralized infrastructure.

The rise in popularity of container solutions, Docker in particular, led to the emergence of container-based solutions for portable laboratories. A significant advantage of containers over traditional virtual images is the lower computational overhead, as demonstrated in the work of [12] in evaluating the scaling capability of Docker-based autograding against the sandboxed VM or chroot setups. An earlier example of a container-based laboratory solution is Labtainers, a container-based framework for the development, deployment and assessment of Linux-based cyber-security lab exercises [5]. In [16], a collection of container-based distributed system assignments was made available to students to deploy on their own computers and observe parallel executions. A solution to support launching a multi-container Spark cluster for big data engineering education was shown in [10].

Extending from these previous works, this paper presents a general template for developing multi-container frameworks that can support topics that require complex large-scale computing infrastructure environments.

3 Analysis of Original Development

The motivation for the creation of this common template framework came from the authors' experience in teaching Modern Web Application, Parallel and Distributed Computing, and Big Data Engineering.

In Modern Web Application, students learned about developing web applications using NodeJS, MongoDB, and Mongoose. This is a popular web application development stack with continuous contributions and updates from the open source community. This also leads to the issue of version drift, where the library dependencies can quickly change, impacting code development [15]. From an educational perspective, this means that students installing the latest versions of these software components on their computers will run into potential version mismatches and other minor technical issues, hindering their learning process.

In Parallel and Distributed Computing, while thread programming can be demonstrated easily in modern computers with multiple cores, MPI programming, parallel distributed file systems, and high performance computing cluster can be best represented with actual distributed components. Federal computational resources can be leveraged here, but they are limited in availability, and inexperienced students can easily consume all allocated service units through inefficient resource requests. Dynamically provisioned cloud resources can be used, but they need to be taken down and refreshed periodically and students lose access after the course finished [11].

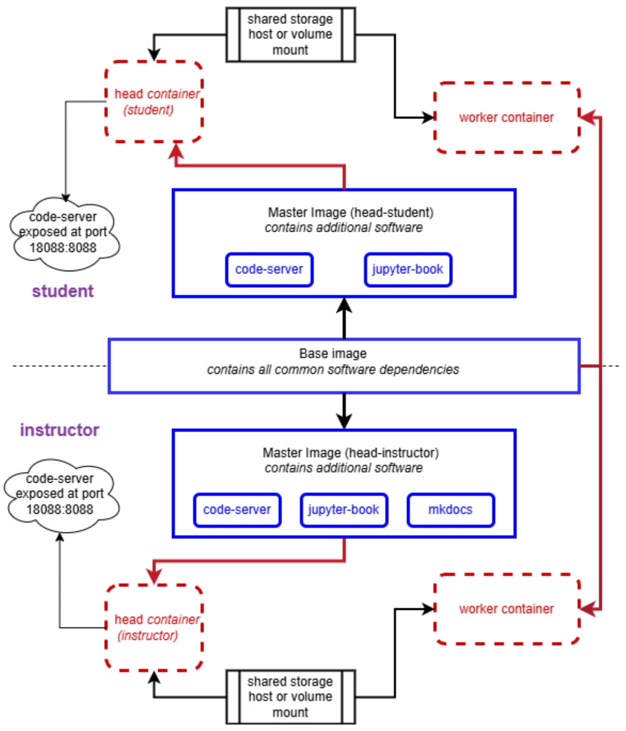
The primary framework for Big Data Engineering is Spark [17]. Relying on popular data science platforms such as Microsoft Azure, Google Colab, and Kaggle, it is possible to manually download and setup a stand-alone Spark cluster on the notebook-running virtual machines provided by these platforms. This solution works for smaller datasets (less than 1GB). For larger ones (10GB+), the online platform resources take a long time to run and if students shut down their laptop, the resource allocation is terminated. It was possible to deploy Spark directly on students' computers, but this required extensive environmental configurations. For Windows machines, this process is non-trivial and often leads to minor technical issues with students. To address this issue, a multi-container framework was developed specifically to: 1) take advantage of students' laptop and desktop resources, and 2) provide a similar Spark computing environment across all computers and operating systems [10].

4 A Common Multi-Container Framework

Based on our experience developing and deploying multi-container frameworks, we have come up with a common architecture for these frameworks, from which customized designs for new courses and research projects can be developed [14]. This architecture is shown in Figure 1.

A core component of this framework is a *base* image, in which relevant common software dependencies for individual courses' contents are installed and configured. From this *base* image, multiple images can be built as needed. At a minimum, separate images can be built for student and instructor use cases. The *student* image contains everything installed in *base*, and front-end services necessary such as the VSCode server and Jupyter Notebook to work with course materials. The instructor's image includes extra dependencies to compile lecture materials. Installing dependencies and setting up services for each image is managed by the external scripts placed under the same directory as the *Dockerfile* of each image, which is copied into the container environment and run during the build process. Additionally, both *head* images copy an *entrypoint* script that will run during deployment.

The directory structure for each image (*base*, *instructor*, *student* ...) include a *Dockerfile*, an *install.sh* script, an *entrypoint.sh* script, and any other relevant files and directories that are necessary for the building process. The *Dockerfile* is simplified and only includes information about the starting-point image of the build, any necessary ports to be exposed, and the specific user account that is activated for the running container. The entire image directory is copied into the image, used during the build process, and deleted after everything is completed. All installation steps are specified in the *install.sh* script. The *entrypoint.sh* is setup as a default runtime when a container is spawned from the image. The runtime includes launching of services (SSH, MySQL, Postgres, etc) and servers (Jupyter, Code, etc). The last launch is set as a foreground


shared storage host or volume mount
head container (student)
worker container
code-server exposed at port 18088:8088
student
Master Image (head-student) contains additional software
code-server
jupyter-book
Base image contains all common software dependencies
instructor
Master Image (head-instructor) contains additional software
code-server
jupyter-book
mkdocs
code-server exposed at port 18088:8088
head container (instructor)
worker container
shared storage host or volume mount
Figure 1: Architecture of a common container framework

process to ensure the container will stay up indefinitely.

The organization of building and deploying the framework as multiple containers is managed by a *docker_compose* YAML file. Users are instructed to follow an ordered build procedure using the *docker compose* command. The framework can be extended to accommodate different course requirements by installing more software dependencies on the *base* image or modifying the *docker_compose* YAML file to set up an environment suitable for the course contents.

5 Adaptation Study

The following courses and research projects have adapted the multi-container framework in teaching and research activities.

5.1 Modern Web Application

The customization starts with changing the starting image of ‘base‘ to the image of NodeJS 22.11.0:slim . This is a Debian-based Linux distribution. The additional software stack includes nginx, Mongoose, Python, and Code server. The software nginx is included so that students can learn how to deploy web applications onto a web server in addition to the common all-in-one development setup. Mongoose is for interaction with a MongoDB server. Four ports, 8080, 8000, 3000, and 4000, are exposed to support Code server and three web port options accordingly. The *head-master* node will have additional Python *mkdocs* packages to help with lecture material development.

As the course does not cover load balancing, there is no need for additional worker-type images. However, an external MongoDB image is included in the ‘docker_compose‘ YAML file. This gives students the illusion of interacting with an isolated and remote database server (MongoDB in this case). Their JavaScript code will specify *mongodb* as a hostname rather than using the *localhost* name. The adaptation of this framework is shown in Figure 2. The repository for this framework can be found at [6].

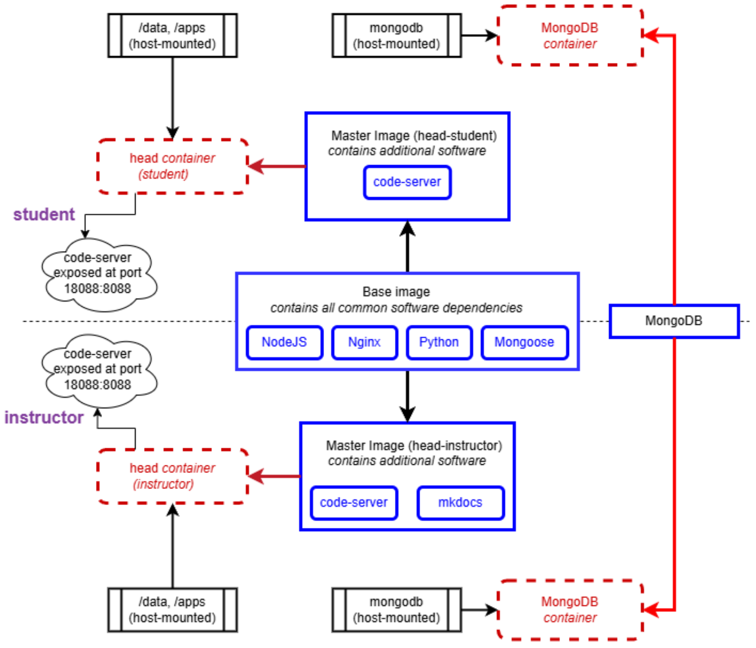

/data, /apps (host-mounted)
mongodb (host-mounted)
MongoDB container
head container (student)
Master Image (head-student) contains additional software
code-server
student
code-server exposed at port 18088:8088
Base image contains all common software dependencies
NodeJS
Nginx
Python
Mongoose
MongoDB
code-server exposed at port 18088:8088
instructor
head container (instructor)
Master Image (head-instructor) contains additional software
code-server
mkdocs
/data, /apps (host-mounted)
mongodb (host-mounted)
MongoDB container

Figure 2: Multi-container framework for CSC418

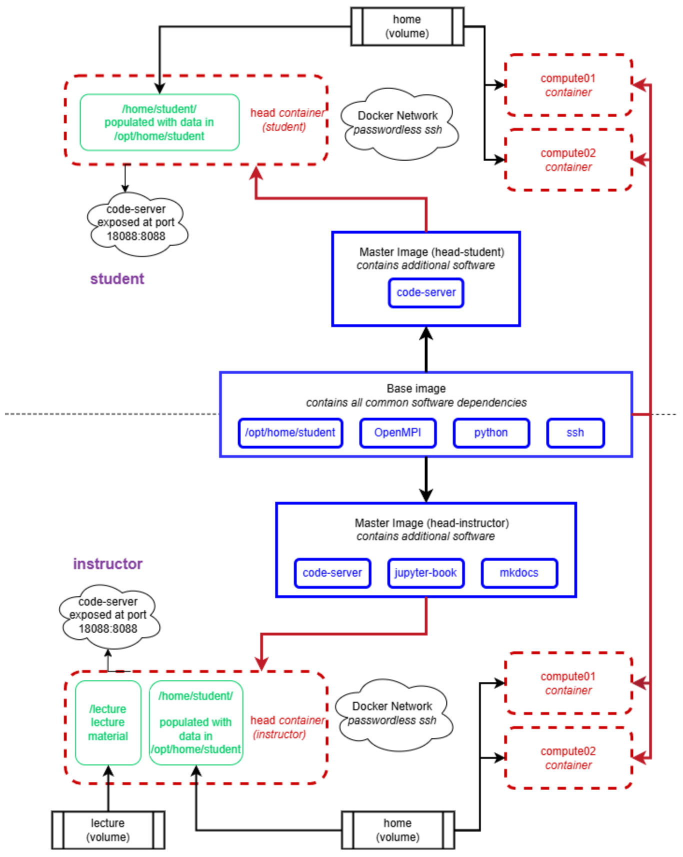

home (volume)
/home/student/ populated with data in /opt/home/student
head container (student)
Docker Network passwordless ssh
compute01 container
compute02 container
code-server exposed at port 18088:8088
student
Master Image (head-student) contains additional software
code-server
Base image contains all common software dependencies
/opt/home/student
OpenMPI
python
ssh
Master Image (head-instructor) contains additional software
code-server
jupyter-book
mkdocs
instructor
code-server exposed at port 18088:8088
/lecture lecture material
/home/student/ populated with data in /opt/home/student
head container (instructor)
Docker Network passwordless ssh
compute01 container
compute02 container
lecture (volume)
home (volume)

Figure 3: Multi-container platform for parallel and distribute computing

5.2 Parallel and Distributed Computing

The base image is built on top of a RockyLinux 9.3 image layer and contains common HPC tools and libraries such as basic gcc/g++ compilers, OpenMPI, Python, and SSH server. The installation and configuration of the software packages is automated through an external script, which is copied into the image and run during build time. The compilers, Python, and SSH are installed via RockyLinux package manager, while OpenMPI is manually compiled from source and installed into a custom location. A user account called *student* is created. Pre-generated SSH keys and configuration, extensions, and password for *code server*, meant to be placed into */home/student/* are copied into the base image, as shown in Figure 3.

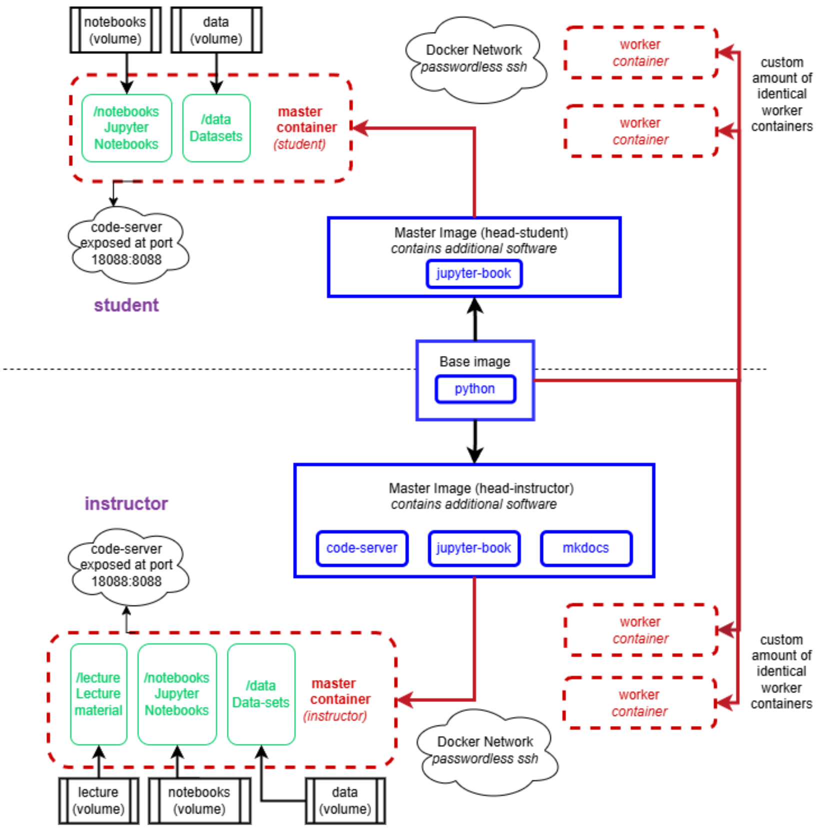

notebooks (volume)
data (volume)
/notebooks Jupyter Notebooks
/data Datasets
master container (student)
Docker Network passwordless ssh
worker container
worker container
custom amount of identical worker containers
code-server exposed at port 18088:8088
student
Master Image (head-student) contains additional software
jupyter-book
Base image
python
Master Image (head-instructor) contains additional software
code-server
jupyter-book
mkdocs
instructor
code-server exposed at port 18088:8088
worker container
worker container
custom amount of identical worker containers
/lecture Lecture material
/notebooks Jupyter Notebooks
/data Data-sets
master container (instructor)
Docker Network passwordless ssh
lecture (volume)
notebooks (volume)
data (volume)

Figure 4: Multi-container platform for Big Data Engineering

The master images for *instructor* and *student* are built upon a layer of base image. In both versions, the master image contains Code (*code-server*) and libraries to enable faculty to edit and build interactive lectures using *jupyter-book* [2]. An *entrypoint.sh* script is added to */usr/local/bin* and set up as an entrypoint for the master container. The Docker Compose YAML file specifies the subdirectories containing the relevant Dockerfiles and additional support files as well as the tags for the images. Additional configuration includes the ability to change the number of computing cores for the compute containers. The repository for this framework can be found at [7].

5.3 Big Data Engineering

In the infrastructure intended for the Big Data Engineering course, our work builds on a pre-built Spark 3.5.2 image with Java 17 and Python 3 and additionally includes essential data science libraries such as Jupyter, Flask, NumPy, Pandas, NLTK, Matplotlib, and scikit-learn. Our approach utilizes the optimized Spark distribution while also ensuring that all necessary Python packages for data analysis and machine learning workflows are available.

The master versions are built from the base image, with the *master-instructor* image contains Code server and additional libraries to support lecture developments, in addition to the Jupyter server. The *master-student* only contains the Jupyter server. The worker containers are launched directly from the base image. In the *docker_ compose.yaml* file, each worker service has CPU cores and memory that are changeable, which allows the adjustment of the cluster and its resources based on their specific needs. The repository for this framework can be found at [8].

6 Discussion

As the platforms did not change the content of course materials and specifically how the courses were taught, we did not carry out a student survey. Rather, we evaluated the effectiveness of these platforms via technical issues raised by the students. Through this experience, we also began working on an even more common containerized deployment schema for all courses.

6.1 Adaptation Issues

Each of the courses (CSC418, CSC466, and CSC467) has its own Discord server setup with a channel dedicated for technical troubleshooting. Below are the issues that have been raised by one or several students in these courses.

CSC418: The biggest issue that many students encountered at the beginning of the course after the environment has been built and launched is how to access *localhost:3000.* The Code server, in trying to be helpful, actually confused the students by suggesting an alternative path of *localhost:8088/proxy/3000.* Furthermore, sometimes students will use *app.listen(port, 'localhost')* which prevents access from outside the container. The lecture notes need to be updated to give students very specific instructions on how to setup the NodeJS apps.

CSC466: Students in this course encountered several issues. The most critical one is the inability to read and write to the home directory for platform launched inside Windows machines. This is due to students downloading and building the images from inside OneDrive, Google Driver, or any other online sharing location. The sharing managers go through the repository and update the status and permission of the mounted home directory, disabling the default permission of the platforms. We address this issue in the latest version of the repository, which turn */home/* into a Docker volume instead. Another issue, this time with Mac, is that MacOS sometimes placed Docker daemon processes into list of untrusted processes. There is an instruction from Apple HelpDesk on how to address in manually, but this remains an annoying nuisance. We anticipate stepping away from Docker Dekstop on Mac with Apple's latest announcement on their own OCI-compliant container engine. One minor issue is when students are unable to build the Docker images on Windows due to *install.sh* is modified with CRLF ending character in Windows environment. This is addressed by adding instructions to the setup process to ensure that the repository is cloned on Windows as is. Another minor issue is the un-reliablity of performance measurement for MPI processes running on low-end laptop.

CSC467: For this course, there are two common issues. The first is when students working on projects with custom Python libraries. In this case, they will have to modify the *base* image and rebuild everything to ensure that these libraries are available across the entire Spark cluster. The second issue is the difficulty in leveraging Spark's WebUI for individual jobs and tasks. This is because Spark cluster assigns random ports for these services and these ports cannot be exposed and mapped during run time. Debugging has to rely on experience and command line interfaces.

The issues shown above are solvable either immediately within the class session or one or two days after. None of the issues is complex enough that cause students to have to abandon the usage of the platform.

6.2 Preliminaries Future Work

As the presented platform has been formally adapted into three listed courses above, we are in the process of customizing the platform to support another class, Introduction to Operating Systems, which utilizes materials from *Operating Systems: Three Easy Pieces* [1] and *XV6* [3]. The inclusion of this course created an interesting observation that the master image of Distributed and Parallel Computing can be quickly customized to support 'qemu' and 'xv6-risc' libraries. This brings up an interesting question about whether we can truly integrate all different platforms through one single base image and minimal customization. We have taken a first step and listed all libraries included in different classes's platforms, as shown in Table 1.

These common libraries provide the basic recipe to create a base image for *all courses*. Next, this base image can be used to support additional courses. Partially common libaries (support some but not all) are taken into account in the build process so that the amount of shared layers are maximized among all

Table 1: A review of common libraries across course platforms

	CSC418	CSC331	CSC466	CSC467
Base image	node:22.11.0-slim (Debian)	ubuntu:24.04	rocklylinux:9.3-minimal	spark:3.5.2-java17-python3 (Debian)
local account		x	x	
OpenSSH		x	x	x
sudo		x	x	
epel-release			x	
git		x	x	
wget	x	x	x	x
sssd			x	
sssd-tools			x	
authselect			x	
openssl			x	
curl	x	x	x	x
python3		x	x	x
python3-pip		x	x	x
python3-devel/libpython3-dev		x	x	
tar	x	x	x	
perl		x	x	
zlib-devel/zlib1g-dev		x	x	
build-essential/gcc		x	x	
gdb		x		
valgrind		x		
qemu-system-misc		x		
gcc-riscv64-unknown-elf		x		
java				x
spark				x
nodejs	x			
openmpi			x	
mkdocs plugins	x	x	x	x
code server	x	x	x	x

final images. This approach will have the potential to simplify the customization process and minimize image storage size for multiple courses. Potential technical hurdles include conversion of Linux base image. For exapmle, the base image for CSC466 needs to be changed from Rocky Linux to Ubuntu. This means that installation procedure for libraries such as *epel-release* and *python3-devel* needs to be changed. An example initial working repository can be found at [9].

7 Conclusion and Future Work

This work demonstrates the design and deployment of a modular, multi-container infrastructure framework aimed at supporting diverse computing courses across various instructional contexts. By leveraging Docker and the Docker Compose YAML specification, we developed reproducible environments that encapsulate front-end tools, backend services, and course materials in a scalable manner. Our framework has been successfully adapted for modern web development, parallel and distributed computing, and big data engineering courses, showing significant improvements in deployment reliability and cross-platform consistency with minimal technical overhead for students.

While our results are promising, several opportunities exist for future work.

First, we plan to broaden the scope of course integrations to include cybersecurity, embedded systems, and AI model training with hardware acceleration. Second, while anecdotal and technical feedback from students has been largely positive, a more systematic evaluation through surveys and learning outcome assessments will be conducted to quantify the impact of this approach on students' learning experience and performance.

Finally, we aim to release a public repository with template YAML files and documentation to encourage adoption across institutions. By fostering a community around portable, container-based pedagogy, we hope to further reduce barriers for institutions with limited infrastructure and promote reproducibility and equity in computing education.

References

[1] Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau. *Operating Systems: Three Easy Pieces*. North Charleston, SC, USA: CreateSpace Independent Publishing Platform, 2018. ISBN: 198508659X.

[2] Enze Chen and Mark Asta. *Using Jupyter tools to design an interactive textbook to guide undergraduate research in materials informatics*. 2022.

[3] Russ Cox, M Frans Kaashoek, and Robert Morris. *Xv6, a simple Unix-like teaching operating system*. 2011.

[4] Wenliang Du. "SEED: hands-on lab exercises for computer security education". In: *IEEE Security & Privacy* 9.5 (2011), pp. 70–73.

[5] Cynthia E Irvine, Michael F Thompson, and Jean Khosalim. "Labtainers: a framework for parameterized cybersecurity labs using containers". In: *Proceedings of the 2017 USENIX Workshop on Advances in Security Education*. 2017.

[6] Linh Ngo. *CSC418 Environment Setup*. `https://github.com/ngo-classes/csc418env`. Accessed: 2025-07-24. 2025.

[7] Linh Ngo. *CSC466 Environment Setup*. `https://github.com/ngo-classes/csc466env`. Accessed: 2025-07-24. 2025.

[8] Linh Ngo. *CSC467 Environment Setup*. `https://github.com/ngo-classes/csc467env`. Accessed: 2025-07-24. 2025.

[9] Linh Ngo. *The One Ring*. `https://github.com/ngo-classes/the-one-ring/tree/main`. 2025.

[10] Linh B Ngo and Hoang Bui. "Sustainable and Scalable Setup for Teaching Big Data Computing". In: *Journal of Computational Science* 14.1 (2023).

[11] Linh B Ngo and Jon Kilgannon. "Virtual cluster for HPC education". In: *Journal of Computing Sciences in Colleges* 36.3 (2020), pp. 20–30.

[12] Matthew Peveler, Evan Maicus, and Barbara Cutler. "Comparing jailed sandboxes vs containers within an autograding system". In: *Proceedings of the 50th ACM technical symposium on computer science education.* 2019, pp. 139–145.

[13] Andy Sayler et al. "Supporting cs education via virtualization and packages: Tools for successfully accommodating" bring-your-own-device" at scale". In: *Proceedings of the 45th ACM technical symposium on Computer science education.* 2014, pp. 313–318.

[14] WCU-AIR. `https://github.com/WCU-AIR/interactive_book`. Accessed: 2025-07-24. 2025.

[15] Erik Wittern, Philippe Suter, and Shriram Rajagopalan. "A look at the dynamics of the JavaScript package ecosystem". In: *Proceedings of the 13th international conference on mining software repositories.* 2016, pp. 351–361.

[16] Zichen Xu et al. "Developing (Almost) free distributed system labs using container-based technique". In: *Proceedings of the ACM Turing Celebration Conference-China.* 2020, pp. 101–106.

[17] Matei Zaharia et al. "Spark: Cluster computing with working sets". In: *2nd USENIX workshop on hot topics in cloud computing (HotCloud 10).* 2010.

Human vs. AI: A Comparative Analysis of Code, Creativity, Clarity, and Paradigm Fidelity*

Carolyn Pe Rosiene[1] and Joel A. Rosiene[2]
[1]Department of Computing Sciences
University of Hartford
W. Hartford, CT 06117
rosiene@hartford.edu
[2]Department of Computer Science
Eastern Connecticut State University
Windham, CT 06226
rosienej@easternct.edu

Abstract

This paper examines the comparisons made by students between code they wrote versus code generated by AI. The analysis reveals that while AI can generate code that is functional, humans, if paradigm-literate, can write code that is more nuanced and produce code that is inherent to human cognition. The analysis also reveals that the AI code is efficient in certain instances, for example, for beginner programmers learning syntax and structure. Our findings advocate for a balanced integration of human and AI-aided code, rather than relying solely on AI to replace human expertise.

1 Introduction

An analysis of a comparative program assignment was conducted in an upper-level Computer Science - Programming Languages – class, a course taken by

*Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

CS majors after Data Structures. Traditionally, students are allowed to select a programming language of their choice, and the same was implemented for this experiment. The main purposes of doing so are to: (1) give students the liberty to select a language if they have a favorite, and (2) vary the languages shared in class.

Over the course of about 10 weeks, students developed a moderately complex program designed to illustrate the core paradigms and features of that language. As part of this task, students also delivered an in-class presentation to explain their language's design philosophy and demonstrate its practical applications. Following this, students were tasked with a supplementary assessment in which they used generative artificial intelligence (AI) tools to produce code that fulfilled the same objectives as their original program. They were then asked to critically evaluate and compare the AI-generated code with their own, focusing on aspects such as paradigm fidelity, readability, and design choices. Paradigm fidelity refers to how well a piece of code aligns with the core principles and idioms of the language it is written in. Due to each student investigating a different programming language, the evaluation performed by each was unique to that student.

This paper outlines the structure of the assignment, synthesizes the key findings and conclusions drawn from the students' comparative analyses and dovetails with our earlier paper[3] which views the use of AI as the weaving of assignment prompts with code to create the base for literate coding. As with Rose et al.[1], we strived to integrate AI into pedagogy, especially in helping students understand how AI can help generate code (or not).

2 Assignment Set-up

The assignment was administered after students had been introduced to differing programming paradigms – imperative, functional, logical, and object-oriented; acquired foundational knowledge of their selected programming language; and had the opportunity to develop a substantive program that leveraged the language's core paradigm. The range of languages chosen by students included: Ada, C#, Clojure, COBOL, D, Dart, Delphi, Elixir, Erlang, F#, Fortran, Go, Groovy, Haskell, Julia, Kotlin, Lua, ML, Perl, Ruby, Rust, Scala, and Scratch. A total of 33 students, distributed across two sections of the course, participated in the assignment, providing a sufficient dataset for drawing meaningful conclusions. The prompt provided for the assignment is shown below.

Comparing Human-Written and AI-Generated Code

Objective: The goal of this assignment is to help students understand how programming paradigms (functional, object-oriented, logic, or hybrid) are ap-

plied by humans versus AI in code creation. By comparing AI-generated code with their own work, students will analyze programming styles, accuracy, optimization, and readability.
Instructions:

1. Examine your program from your Language Project:
    - You have written a program (not taken from another source) in the programming language you were researching (functional, object-oriented, logical, or hybrid). In doing so, you should have ensured your program demonstrates the core paradigm features of the language.
    - Document your design choices and implementation process.
    - If your program is long or complex, select a portion of your code that demonstrates the paradigm(s) best.
2. Generate AI code:
    - Read up on how to best generate effective prompts like such as "Effective Prompts for AI: The Essentials".
    - Use an AI tool (such as Copilot, OpenAI, ChatGPT, etc.) to generate code for the same program you've written. Record what prompt you used to generate this code. Ensure the AI-generated code solves the same problem or performs the same task as your own program. Keep a screenshot of this prompt.
    - Document the AI's design choices.
3. Compare and analyze:
    - Compare your program with the AI-generated code in terms of:
    - Programming paradigm - Code is true to the paradigm. For example, how "functional" is the AI-generated functional code, how "object-oriented" is the AI-generated object-oriented code, etc.
    - Code readability - Code is easy to understand, maintain, and modify. Key features include consistent formatting, meaningful names for variables and functions, and clear, concise logic.
    - Document your observations and comparisons. Address the strengths and weaknesses of both sets of code, and provide insights into how humans and AI approach coding differently.

The assignment tasked the students to respond to seven entries:
Question 1 - Give the name of your programming language and its paradigm(s).
Question 2 - Indicate which AI you used to generate the AI code.
Question 3 - Indicate the purpose of the code, the problem it solves or the task it performs.
Question 4 - Provide a screenshot of the AI and the prompt you used.

Question 5 - Give both sets of code: (1) your own code and (2) AI-generated code. Be sure to label each one clearly.
Question 6 - In 200-300 words, **compare your program with the AI-generated code based on whether it is true to the programming paradigm and on code readability**. Be specific, pull examples from the actual sets of code to support your comparisons. Disregard minor differences such as formatting, spacing, or the choice of variable names.
Question 7 - In 200-300 words, discuss which version you feel is better and why, providing specific examples from each version. **Address the strengths and weaknesses of both sets of code, and provide insights into how humans and AI approach coding differently**. Disregard minor differences such as formatting, spacing, or the choice of variable names.

3 Analysis of Student Responses

Analyses of student responses are broken down into three areas, detailed below, based on the responses gathered from Questions 6 and 7.

3.1 Human vs. AI Coding Approaches

The contrast between human and AI-generated code reveals fundamental differences in how each approaches problem-solving, creativity, and structure. Several students commented on how their code was written with a sense of narrative and purpose, often shaped by their understanding of the problem domain, their experience with the language, and their personal coding style. This results in code that is not only functional but also expressive—often reflecting the coder's intent, priorities, and even personality[3]. For example, several students commented that a human might choose to write a simple, readable function that prioritizes clarity over performance, especially if the code is meant for future collaboration. Humans also tend to anticipate edge cases, comment their logic, and structure their programs in ways that align with their mental models of the task

In contrast, AI-generated code is typically more literal and prompt-driven, as many students noticed. It excels at producing syntactically correct, well-formatted code that adheres to common patterns and best practices—at least as they are statistically inferred from its training data. However, AI often lacks the contextual awareness to make nuanced decisions. It may generate verbose or overly generic solutions, include unnecessary complexity, or miss subtle bugs or inefficiencies. In several student reflections, students noted that AI-generated code used constructs that were technically correct but not idiomatic to the language, such as using object-oriented patterns in functional languages or misapplying language-specific features like metaprogramming.

As an example, one student who explored Elixir and implemented code to add two matrices. The human-written code, see Listing 1, is contrasted with the ChatGPT-generated code, see Listing 2. The student observes that his code is longer than the AI's; he also notes: *"The AI did exactly what it needed to with little extra, but I was able to showcase more features from the Elixir programming language. Overall, the AI approached the problem with the most to-the-point effective solution, which used the most powerful part of the programming language that it could find to generate the solution. On the other hand, I looked for the solution that best fit my needs to showcase what the language could do, even if it was not the best solution."* The student makes a marked conclusion that his code showcases Elixir features rather than simply producing a solution.

Listing 1: Human-written Elixir code to add two matrices

```
defmodule MyMatrixAddition do
  def a +++ b, do: MyMatrixAddition.add_matricies(a,b)

  # recursive going through the matrix rows
  def add_matricies([a_head | a_tail],[b_head | b_tail]) do
    [add_matrix_columns(a_head, b_head) | \
    add_matricies(a_tail, b_tail)]
  end

  # base case
  def add_matricies([],[]) do
    []
  end

  # recursive going through the matrix columns
  defp add_matrix_columns([col_a_head | col_a_tail], \ [col_b_head
    # Spawn process
    parent = self()
    spawn_link(fn -> send(parent, {col_a_head, \
    col_b_head, col_a_head + col_b_head}) end)

    temp_tail = add_matrix_columns(col_a_tail, col_b_tail) \
    # recursive call

    # get value for head
    temp_head = receive do
      {^col_a_head, ^col_b_head, value} ->
        value
    end
    [temp_head | temp_tail]
  end

  # base case
  defp add_matrix_columns([],[]) do
    []
  end
end
}
```

Listing 2: ChatGPT-written Elixir code to add two matrices

```
defmodule Matrix do
  @doc """
  Adds two matrices together.

  ## Examples
      iex> Matrix.add([[1, 2], [3, 4]], [[5, 6], [7, 8]])
      [[6, 8], [10, 12]]

  """
  def add(matrix1, matrix2) when length(matrix1) == \
  length(matrix2) do
    Enum.zip(matrix1, matrix2)
    |> Enum.map(fn {row1, row2} -> add_rows(row1, row2) end)
  end

  defp add_rows(row1, row2) when length(row1) == \
  length(row2) do
    Enum.zip(row1, row2)
    |> Enum.map(fn {a, b} -> a + b end)
  end
end
```

3.2 Paradigm Fidelity

One of the most insightful aspects of this assignment was observing how both human and AI-generated code adhered to—or diverged from—the intended programming paradigms of each language. This includes not just syntax, but also the philosophical and structural approaches that define a language's identity. For example, Ruby encourages expressive object-oriented and metaprogramming techniques, Haskell emphasizes pure functional programming, and Ada is known for its strong typing and support for concurrent systems. As we stressed earlier[1], the rise of AI-assisted code generation necessitates a renewed focus on programmer literacy to ensure that students maintain a deep understanding of the code they write and review.

One student wrote a Ruby program to code generators to randomized cards in the form of card packs, and adds it to a vector of cards titled "deck". The user can then open as many packs as they wish, and view their deck in the order of adding cards. He approached the problem by modularizing his code which the AI-generated code did not do. He remarks: *"The end product is nearly identical, which hurts my feelings a little bit, but the way it was implemented is very different. The main difference between our scripts is how I utilized modules, and the AI didn't. Everything the AI wrote is in main.rs, while I utilized a separate card_ manger.rs script. Modules are made to make code easier to read and follow, which is why all my card logic is in a separate file. The AI didn't do it this way, and instead put everything in the main file, making it harder to follow. My version of the script's use of modules also shows off the ownership system, which may have overcomplicated things a bit, but was a good way to show the class how variables can be passed and ownership can be changed."*

Similar observations were made by another student who explored C++: *"The strength of my code is that it showcases more of what the programming language can offer. It shows off threads, recursion, anonymous functions and operator overloading. My code overloads the "+++" operator. It uses recursion with a base case to navigate through the lists comprising the matrices. It uses threads to do the actual addition, and that thread is told what to do with an anonymous function which is passed to it. . . . The AI did exactly what it needed to with little extra, but I was able to showcase more features from the Elixir programming language. Overall, the AI approached the problem with the most to-the-point effective solution, which used the most powerful part of the programming language that it could find to generate the solution. On the other hand, I looked for the solution that best fit my needs to showcase what the language could do, even if it was not the best solution."*

As suspected, human-written code often demonstrated a deeper awareness of the paradigm and language features. Students were intentionally asked to craft their programs to showcase the strengths of their chosen languages. As a result, the AI-generated code sometimes missed the mark. AI-generated code tends to produce functional code but used generic or imperative patterns. As we had underscored[3], the rise of AI-assisted code generation necessitates a renewed focus on programmer literacy to ensure that students maintain a deep understanding of the code they write and review.

3.3 Educational Value

The educational value of comparing human and AI coding approaches lies not only in understanding how each operates but also in how this understanding can be leveraged to enhance learning, teaching, and professional development in computer science. This experimental assignment sought to open students' eyes to the pros and cons of using AI to generate code.

This is observed by a student who wrote: *It's fascinating to think about how a human and AI create code. The AI is doing exactly as it's told, with very few interpretations. ... A human, however, slowly builds upon their idea, and over time, adds new features and ideas. . . . At the end of the day, if you are looking for a result* *that works exactly how you want it, without caring much about utilizing the language perfectly, you'll choose AI. But if you want someone to think through a problem, build upon it, and write it efficiently, you go for a human."*

As AI becomes increasingly integrated into development environments, its role as both a tool and a tutor re-shapes the landscape of programming education. Ma, Wu and Koedinger[2] explore the dynamics of human-AI collaboration in programming, finding that AI-assisted pair programming can offer comparable benefits to traditional human-human pairing, particularly in terms of efficiency and support for novice developers. This highlights the fact that AI could offer much support for beginners rather than advanced programmers. The key is maintaining a balance between using AI as a catalyst for understanding syntax and structure, rather than using it as a shortcut to write complex code as had been pointed out by Valový et al.[4]

Another student concludes: *This exercise [sic] allowed me greater insight into how the AI looks at things from the objective, and it takes into account things that humans will overlook. Human code can be messier, and it can be imperfect, but that learning process is integral when it comes to making things more and more complex."* which confirms that AI-generated code may be used to supplement one's learning and is not a replacement especially when tasked with building more complex software.

4 Conclusion

One of the goals of this assignment was to demonstrate to students that AI is not always the answer. Of the 33 students, 22 or 67% favored their own code. The take away from this project is that human and AI coding approaches each offer distinct advantages that, when combined, can significantly enhance software development. Humans contribute creativity, contextual awareness, and ethical reasoning, while AI brings speed, consistency, and the ability to process vast datasets. Although AI tools assist with code generation and debugging, they lack deep comprehension and nuanced judgment. In terms of paradigm fidelity, humans can deliberately apply and switch between programming paradigms, whereas AI, though versatile, may not always follow best practices. Their strengths and weaknesses are complementary—humans may be slower and prone to fatigue, while AI can produce errors due to limited understanding. Educationally, AI supports learning through feedback and exposure to diverse coding styles, but over reliance can hinder critical thinking. A balanced integration of both human insight and AI capabilities fosters innovation, deeper learning, and more effective development outcomes.

References

[1] B. Marshall K. Rose V. Massey and P. Cardon. "IS professors' Perspectives on AI-assisted Programming". In: *Issues in Information Systems*. Vol. 24. 2. 2023, pp. 178–190. DOI: `10.48009/2_iis_2023_115`.

[2] T. Wu Q. Ma and K. Koedinger. *Is AI the better programming partner? Human-Human Pair Programming vs. Human-AI pAIr Programming*. `https://arxiv.org/pdf/2306.05153`.

[3] Joel A. Rosiene and Carolyn Pe Rosiene. “WIP: Focusing on Programmer Literacy in the Time of AI-Aided Code Generation”. In: *IEEE Frontiers in Education Conference (FIE)*. Washington, DC, USA, 2024. DOI: `10.1109/FIE61694.2024.10893329`.

[4] M. Valový and A. Buchalcevova. “"The psychological effects of AI-assisted programming on students and professionals"”. In: *IEEE International Conference on Software Maintenance and Evolution (ICSME)*. Bogotá, Colombia, 2023, pp. 385–390. DOI: `10.1109/ICSME58846.2023.00050`.

On Teaching a General Education Course in Artificial Intelligence*

Donald Braxton[1] and John Wright[2]
[1]Philosophy and Religious Studies
Juniata College
Huntingdon, PA 16652
braxton@juniata.edu
[2]Information Technology and Computer Science
Juniata College
Huntingdon, PA 16652
wrightj@juniata.edu

Abstract

The teaching of Artificial Intelligence (AI) is important for technology majors but, with the growing use of AI in all disciplines and in daily life, it is also important for non-majors. This paper reports the experience of a team-taught, general education course at a small, liberal arts college. We call the course "Living with AI". Our aim is to reflect on the interdisciplinary nature of AI, the topics, tools, and resources used in the course, and the importance of teaching AI to non-majors in a general education course.

1 Introduction

Artificial Intelligence (AI) is the hot topic today with applications of generative AI dominating the news. Reactions run from celebration to vilification, from predictions of utopia to fears of catastrophe. Sensational headlines rarely serve

*Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

the general public well even if they generate clicks. Non-major courses are one way that we can help guide students to a deeper understanding of topics such as this. Past publications on non-major courses in computing often focus on computational thinking or are used to attract students to the computing fields. A more inspirational example comes from Richard Stockton University with their course *Artificial Intelligence in Society*[2]. The first three objectives of their course are ones that inspire us as well: to introduce AI as an interdisciplinary field, to engage students from across disciplines, and to encourage students to see the applications of AI in their own fields. We in computer science education should be more explicit about offering non-major courses to help build better, well-informed citizens and to help students in their chosen fields. One tenet of the ACM Code of Ethics is to "foster public awareness and understanding of computing, related technologies, and their consequences."[3] News headlines and social media posts have created much awareness of AI, but awareness is not understanding. At Juniata College, we have decided to address the need for a deeper understanding of AI through a new, team taught course titled *Living with AI*.

Juniata College is a small, primarily undergraduate, liberal arts college in Huntingdon, Pennsylvania. In rank order according to major, the most enrolled programs are biology and health sciences, business, environmental science, psychology, and computer science. All students are required to complete a sequence of general education courses including an interdisciplinary course in their third year. This course is required to be team taught in order to view a topic from different disciplinary lenses. *Living with AI* fulfills this requirement.

The *Living with AI* course is a 15 week, 4 credit course that was taught for the first time in spring 2023 and a second time in spring 2025. Although the instructors were from Computer Science and the Humanities, attention was given to the typical reasoning styles of various disciplines. AI transects the domains of computing, politics, law, philosophy, economics, ethics, neurology, music and art, to name just the areas engaged by this course. The course description and much more information on the course can be found on the course syllabus[1] and in section 2 below.

The course includes a wide range of topics organized to introduce students to AI and the ethical and societal issues surrounding it. Main topics include the history of AI, video games, image recognition, natural language processing, surveillance, policing and military applications, transportation, employment, creativity, and artificial general intelligence (AGI). Each of these topics are encompassed by in- and out-of-class activities, reading and viewing of other forms of media, homework assignments, and reflection papers. Badges are given for certain achievements to keep students motivated to try everything and turn in assignments. The goal of each week is to provide the student with a realistic background from a computer science perspective and the ethical, societal, and legal issues surrounding the topic from the humanities and social sciences perspectives. An effort was made to help students to think critically in order to more accurately understand the topics and the ramifications on people and society.

LIVING WITH AI - DECISION TREES
CONN 301
LIVING WITH AI - NEURAL NETWORKS
CONN 301
LIVING WITH AI - ZUMI BUILD
CONN 301
LIVING WITH AI - ZUMI TRAINING
CONN 301
LIVING WITH AI - AI WRITER
CONN 301
LIVING WITH AI - AI ARTIST
CONN 301

Figure 1: Sample assignment badges. Zumi is a registered trademark of Robolink, Inc.[7]. Badges created with the assistance of AI.

Assignments and grading follow from these learning outcomes. Attendance and participation as well as the writing of reflection journals on the assigned material are included to help encourage students to be informed and engaged in each class period. Discussions in class and written reflections on the various topics are a large portion of the grade. Students also are asked to do assignments to gain experience with expert systems, neural networks, generative AI, and programming an autonomous robot. In the initial offering of the course, we used the Zumi by RoboLink[7] for the hands on programming assignments. In the second offering, we moved to the mBot2 coding robot by Makeblock[5].

The 25 students in the course in spring 2023 and 27 in spring 2025 included a mix of technology majors and non-majors. These included Computer Science (12), Information Technology (3), Business (13), Biology and Environmental

Science (9), History and Museum Studies (4), Art (3), English (2), Mathematics (2), and one each from Data Science, Physics, Psychology, and Neuroscience. Of the 35 males, 16 females, and 1 non-binary student, 44 were white, 4 were black or African American, and 4 were Asian. This is reflective of the college's racial demographics but not of the gender demographics. The breakdown of disciplines indicates that just under a third of the students had a significant background with computer programming coming into the course. None of the students had taken any other machine learning or artificial intelligence courses prior to this course.

2 Execution

The course is taught in two, two-hour blocks each week. Each block tends to be subdivided into one-hour units. The first hour is an introduction to the topic. The second is a hands-on learning experience such as experimenting with neural network hyperparameters or programming the robots. The third and fourth hours are typically an ethical discussion of the topic, either in the form of a case study or a philosophical debate on the consequences of AI for human self-understanding, societal evolution, or changing definitions of work, risk, creativity, autonomy, national security, and globalization.

Each week, students are expected to review the material for class and write a reflection based on the material. After the class discussions that week, students are expected to go back and respond to themselves, adding to their reflections. These are graded simply on a 0, 1, 2 scale with 0 meaning that the student did not turn it in and 2 meaning that the student addressed what was asked in the reflection prompt. A 1 is reserved for the student who does not address the topic or reports what was done in class and does not reflect on the topic.

The topics covered during the course, by week, include:

1. Introduction - What is Intelligence, the context of ethics and AI, the Alignment Problem
   Resources: Exerpts from The Alignment Problem by Christian, Crash Course AI Episode 1
   Assignments: Reading/Browsing resources (to do every week), written reflection (to do every week), prepare for discussion (to do every week)

2. Symbolic AI, Expert Systems, Intro to Ethics/Ethical Frameworks
   Resources: Expert Systems 2.0 (towardsdatascience.com expert-systems-2-0), Crash Course AI Episode 10, Putting Expert Systems to Work (Harvard Business Review, March 1988), Crash Course Philosophy, What are Ethical Frameworks (Penn State Department of Agricultural Economics, Sociology, and Education)

Assignments: Create a decision tree on Expert Systems Builder (http://www.mcgoo.com.au/esbuilder/index.php)

3. History of AI in Games - Finite State Machines to AI
   Resources: Various AI in games sources, The AI of Alien: Isolation (YouTube), AlphaGo - The Movie (YouTube), Discussion on why we build AI that are better than humans
   Assignments: Expert Systems assignment continued

4. The Perceptron and Introduction to Neural Networks
   Resources: An Intelligent Piece of Paper (https://www.stem.org.uk), Minimax, Monte Carlo Algorithms, 3Blue1Brown Videos - What's a Neural Network, Gradient Descent (YouTube), Google Colab (MNIST, Fashion MNIST, and Titanic data sets)
   Assignments: Hyperparameter Experiments - Students play with different hyperparameters to see impacts on performance, Build the Robot (mostly done in class)

5. Image Recognition and Bias
   Resources: quickdraw.withgoogle.com, Teachable Machines, Worries about Predictive Analytics (HeinOnline), various sites/news on bias in facial recognition
   Assignments: Neural Network assignment continued
   Notes: Build the mBot2 Robots

6. AI in Education Today and Employement Tomorrow
   Resources: Princeton Dialogues on AI and Ethics Case Study 5, What can history teach us about technology and jobs (McKinsey Global Institute, February 2018), BLS Jobs at Risk from Automation and other updates on AI impacts (2022-2025), How to Optimize Your Resume for ATS, Prompt engineering, Using Generative AI appropriately
   Assignments: mBot2 Drives (make it move and sense the world)

7. Natural Language Processing and Smart Devices
   Resources: Princeton Dialogues on AI and Ethics Case Study 2, Audacity/Audio, AI and Cultural Appropriation (Fake Drake, AI generated music)
   Assignments: Normal weekly reflection including playing with sites like Udio, Speechify, etc.

8. Surveillance, Data and Privacy
   Resources: Frontline: In the Age of AI excerpts, Princeton Case Study 1, Teaching Privacy: There's No Anonymity, Simple Demographics Often

Identify People Uniquely (Sweeney, CMU)
Assignments: mBot2 Decides (if statement, sensors, actuators)

9. Policing and Military Applications
Resources: Pros and Cons of Autonomous Weapons Systems (Etzioni and Etzioni, Army University Press, May-June 2017), autonomousweapons.org, ICRC position on autonomous weapon systems, Princeton Case Study 4
Assignments: mBot2 Minesweeps (Drive and use sensors to detect "mines")

10. Transportation, Autonomous Vehicles and Smart Cities
Resources: Consumer Reports on the AI already in vehicles, The Two Futures of Driverless Cars (YouTube, Cleo Abram), nhtsa.gov autonomous vehicle safety, smart cities (Shea and Burns, techtarget.com), Connected Vehicles in Smart Cities (Fourtané, interestingengineering.com)
Assignments: mBot2 Minesweeps continued

11. Making Art and Writing Papers, GANS, Transformers, Generative AI, Copyright Law
Resources: What are GANs? (YouTube, IBM), Transformers, explained (YouTube, Google Cloud Tech), Do These AI Generated Fake People Look Real? (NYTimes), this-person-does-not-exist.com, U.S. Copyright Office Rules A.I. Art Can't Be Copyrighted (Recher, Smithsonian Magazine, March 2022), U.S. Copyright Office Copyright and Artificial Intelligence (copyright.gov), The REAL Fight Over AI Art (YouTube, Cleo Abram)
Assignments: AI Creativity (use AI to write and illustrate a story)

12. AGI, The Ethics of Handling Emergent Superintelligence
Resources: "Godfather of artificial intelligence" weighs in on the past and potential of AI (CBSNews.com), The Three Laws of Robotics (Asimov, *I, Robot*, page 27), Ray Kurzweil on StarTalk (YouTube, StarTalk), Problems of AI (energy, water, noise pollution, greenhouse gas emissions, space/locations, jobs, e-waste, AI-sweatshops), The Ethics of Artificial Intelligence (Bostrom and Yudkowsky, 2011), The moral standing of AIs
Assignments: mBot2 Learns (Training, Teachable Machines)

13. Student Choice of Topics: Fooling AI, International Politics and AI
Resources: Glaze Image Cloaking (glaze.cs.uchicago.edu), How to Fool Artificial Intelligence (Wang, Medium, January 2021), Fooling LIDAR, The Global Politics of Artificial Intelligence, chapter 2, Governance of Artificial Intelligence (Ulnicane et. al., 2022), The state of AI (McKinsey Survey) *Assignments:* mBot2 Hits the Highway (Traverse a map/Line follow, React to Road Signs)

14. Conclusion and Wrap up
    Assignments: Signature Assignment - Final Course Reflection, mBot2 assignment continued

Because the course is intended for a general audience, one fourth of our time was dedicated to reviewing basic ideas of computing in general, the history of machine learning, and primers in how neural networks operate. Some programming was required for operating the robot in both hard-coded and autonomous modes, but the learning curve was kept lower through the use of a block-coding IDE included with the robots. More advanced students could opt for programming in Python, but only 2 students chose to do so. Most students, including the technology majors, chose to use the block language.

The course was called *Living with AI* because students are encouraged to think about AI in specific parts of their current and future lives and the larger world around them. Reflection exercises and writing were an important facet of the course. For those who were unaware of AI, this meant pointing out just how ubiquitous AI is already. For those who were aware, this meant introducing them to the profound break-throughs of the last couple of years. All were urged to abandon the more science fictional versions of AI in favor of submerging themselves in current real-world deployments. Therefore, case studies of AI in healthcare, sound and image identification, natural language processing, law enforcement and military applications, hiring, games, autonomous vehicles and smart cities, and art and creativity. Just as important as mastering the reality of current manifestations of AI, the course also emphasizes how significantly the creation of intelligences of our own making challenge traditional notions of person-hood, autonomy, human creativity, the economic, societal, and political future, and fundamental assumptions about the meaning of life. It is important to note that the goal of this stage was not to arrive at concrete answers but to live with the questions AI raises. That is what living intelligently with AI means.

3 Results

Anecdotally, the students enjoyed the course, the topics presented, and the activities required both in and out of class. We as instructors are satisfied with the course and the outcomes for the students. As is becoming too typical, only 16% of the students in 2023 and 17% of students in 2025 filled out the course evaluation at the end of the semester. Of those evaluations, all were overwhelmingly positive with the only score below a 4 out of 5 being, “The workload for this course seemed appropriate to the level of the course”, with a score of 3.75. This was in 2023. Four of the evaluation questions garnered a 5 out of 5 rating, two of which were primary goals for the course: “The instructor raised challenging questions and encouraged me to think" and "The instructor was open to students' questions and encouraged free exchange". Overall, ratings for spring 2025, given the equal but low number of responses, were better than spring 2023. Classes have different personalities, but this is encouraging that small changes made between the two offerings have helped. The primary changes included more frequent but less involved reflection assignments and a change of robotics platform.

A view of the course results can be found from the final reflective piece that students were asked to write. While not the best analysis of the complete course and only inclusive of the submissions from spring 2023, Table 1 represents a list of the primary student learning outcomes from the course syllabus with a percent of the final reflection papers that demonstrate that outcome. It is important to note that many of the low percentages are explained by the prompt provided for the final reflection which asked students to focus on public misconceptions on AI. Those areas with lower values are addressed in other assignments not included in this analysis. While not a complete analysis of all learning outcomes for the course, we did learn that almost 80% of the papers met outcomes that were asked for in the prompt, but also that many of the other outcomes were represented in student reflections as well. Top among these were a fluency in the ethics of AI, an awareness of the ubiquity of it, and an appreciation of the interdisciplinary nature of the topic. The numbers produced in the qualitative coding process were also affected by the language of the learning outcomes, for example, the interpretations of fluency or conceptual mastery on a topic and how that is judged during the encoding. Although covered by a few of the other learning outcomes in combination, this evaluation also revealed that we may want to add a specific learning outcome of providing the student with a realistic view of artificial intelligence and understanding public misconceptions about the topic. 100% of the papers addressed this concern, which was asked for in the prompt and is an important outcome of the course.

To add a fun element to the course while attempting to encourage all students to try everything, we offered non-credentialed badges through Moodle for achievements in certain assignments. Sadly, due to the change of robotics platform, badges were not used in spring 2025, so the results below are only from spring 2023. Table 2 lists the assignments and numbers awarded for each badge and abridged descriptions of the badges. Badges were awarded by the instructors if a student made a good-faith attempt at the assignment at their level of technological ability. Students that were not awarded the badge either did not submit the assignment on time or the submission was incomplete or substandard. Overall, 19 of the 25 students kept up enough to earn the majority of the badges. It is important to note that these numbers represent badges

Table 1: Learning Outcomes

Outcome	Demonstrated in Final Reflections (2023 only)
An awareness of current ubiquity and potential future deployments of AI	58%
A conceptual mastery of the processes underlying the current state of machine learning	25%
A history of the evolution of the field	25%
An appreciation of the interdisciplinary nature of machine learning	46%
An awareness of the legal contexts in which AI operates	29%
A fluency in the ethics of AI such as bias, neutrality, privacy, paternalism, transparency, freedom, determinism, agency, and democracy	54%
A mental nuance in the understanding and articulation of the limits of AI including verification and reproduction	79%
Introductory skills in both the software and hardware employed in AI	13%
An engagement with topical issues of the day such as facial recognition, NLP, chatbots, public and private surveillance, loan and employment screening, automation and unemployment	79%

awarded and not number of students that submitted the assignment, which was almost always higher. In the end, 14 students achieved a grade in the A range, 3 with B, 2 with C, 4 with D, and 2 with F. Two of the D's and one of the F's were technology majors, so the course worked for the majority of majors and non-majors alike. Out of 28 meeting periods, the average attendance was 24 and the median was 25. Attendance generally correlated to final grades with some exceptions.

Although badges were not used in spring 2025, we see similar grade, assignment submission, and attendance results, showing that badges are probably not necessary beyond just being fun. Out of the 27 students that completed the course in 2025, 19 students achieved a grade in the A range, 4 with B, 2 with C, 2 with D, and 0 with F. In this case, all of the technology majors did well and again the course worked for the majority of majors and non-majors alike. Out of 28 meeting periods, the average attendance was 25 and the median was 26. As expected, attendance again generally correlated to final grades.

Beyond the numbers, we know the majority of students were engaged in this course. A few examples of discussions with students that stand out involved AI in games, autonomous vehicles, and AI and employment. The section on AI in games allowed us to take something that students were already interested in and use it to show the shift from symbolic AI to machine learning and algorithms from decision trees, to minimax, and Monte Carlo methods. This provided the link to neural networks and proved to be a bridge to the rest of the topics in the course. The section on autonomous vehicles was important in that it allowed students to see that AI is already being built into vehicles that they use everyday and to consider the vast numbers of problems that need to be solved for vehicles to be truly autonomous. For example, the decisions that need to be made about when a vehicle should stop if it is unsure if it sees something and then how society might change when individuals know they can step out in front of a vehicle and not be harmed allowed for much deeper considerations on the topic beyond the excitement or fear you may get when hearing about the topic more generally. Although we are speaking anecdotally, these levels of critical thinking did appear in students reflections on this topic and throughout the course. The final example of student engagement demonstrates the importance of teaching AI to non-majors and occurred during the discussion of employment issues and loss of jobs due to automation. Placing AI into a history of technological change and its effects on jobs over time was important for students to have a realistic expectation that AI will change, eliminate, and create jobs over the next decade. An important realization for students of all majors is that AI will be changing or eliminating jobs in the areas they are currently studying and that they need to pay attention to AI and work on skills to help improve their employability in a changing future.

Table 2: Assignment Badges (spring 2023)

Name	Description
Decision Trees 24/25	*the student has learned* about the history of machine learning, discussed expert systems and decision trees and their role in machine learning, and implemented a simple decision tree utilizing an online site or programming language.
Neural Networks 17/25	*the student has learned* about the history of machine learning, discussed neural networks, how they work and their role in machine learning, and experimented with the model attributes of a pre-built neural network to investigate the performance and reflect on the limits and benefits of neural networks in artificial intelligence.
Zumi Build 25/25	*the student has learned* to build the Zumi robot from Robolink, Inc., including the connecting of motors, sensors, camera, the motor controller board, and the Raspberry Pi Zero.
Zumi Drives 21/25	*the student has learned* to write a simple program to make their Zumi robot drive around a flat surface.
Zumi Mine-sweeper 21/25	*the student has learned* to write a program to make their Zumi robot drive around a simulated mine field and flag where the mines are located by using the Zumi's IR sensors.
Zumi Training 21/25	*the student has learned* to write a program to train their Zumi robot drive around a surface and recognize different colors from training through a machine learning algorithm.
Zumi Navigates 19/25	*the student has learned* to write a program using the training provided to their Zumi robot to drive, avoid obstacles, and navigate a simple maze to get from the entrance to the exit. Students have been learning about autonomous vehicles and smart cities.
Zumi Delivers 20/25	*the student has learned* to write a program using the training provided to their Zumi robot to drive, avoid obstacles, and recognize colors to navigate a simple maze and deliver goods from a source location to a destination and return back to the original spot. Students have been learning about autonomous vehicles and smart cities.
Zumi Expert 21/25	*the student has* earned 5 of the 6 Zumi badges and has become a Zumi Expert.
AI Writer 19/21	*the student has learned* about how AI can learn to write and has experience creating writings with AI generated text and chatbots like chatGPT. The student has also discussed the moral and ethical issues surrounding AI's ability to write.
AI Artist 19/21	*the student has learned* about how AI can learn to create art and has experience creating images using AI generated art tools such as DALL-E and Stable Diffusion. The student has also discussed the moral and ethical issues surrounding the generation of images including employment and copyright issues.

The somber reaction of students to this is the impetus for this paper and to encourage the computer science education community to broaden the scope of AI education.

4 Conclusion and Future Work

We found the experience of teaching this class to be satisfying. Indeed, the first run of the course was unusually successful and we feel the same about the outcomes in the second run. We certainly derived some benefit in student interest from the release of ChatGPT[6] to the public at the end of the prior semester and the ongoing media coverage ever since. Overall, we feel we achieved our principal learning objectives and student engagement and enthusiasm were high. Future work will involve rewriting learning outcomes to make them more assessable, mapping how the learning outcomes are covered by the variety of assignments, setting aside class time to increase participation in the course evaluation survey, and using all of this to perform an assessment of the course.

Although more successful during the second offering, the course had rough spots. Often we were revising readings and assignments on the fly based on class progress and advancements in the technology. Between the first and second offering of the course, resources became dated and new resources became available. Course preparations entailed our collection of an array of potential resources for the class with decisions on which specific resources were to be used only being definitively chosen during a weekly meeting of the instructors. We thought this choice was prudent given the fact that we were uncertain what level of academic challenge this class would exhibit with its heterogeneous population and that breaking news in the field was happening quite frequently.

Perhaps the most significant issue we faced was with the hands-on component, particularly with the Zumi robots in spring 2023. Both instructors felt that actual technological engagement was important to student development and the Zumi robots were an affordable solution. By requiring a small course fee, each student received a Zumi that they could keep and use during the course. The abilities and quality of these robots are limited. Mechanical failures were common. Some of the difficulties were due to a few mechanically disinclined students handling delecate parts. Some were due to missing or faulty parts. This may have been due to rush delivery as the product was back-ordered, possibly by supply chain issues and shipments from overseas. Other issues arose from the limited techniques available to train the robot. The Zumi is a toy for younger students and is limited and not flexible enough for use beyond the built-in features, although there are benefits in this for non-majors. The robot could not handle neural networks and would only enable specific KNN models for color and hand gesture recognition. All computing on the Zumi is done on the Raspberry Pi Zero on a locked-down micro SD card and, to our knowledge, processing could not be offloaded to the cloud. Additionally, even with the option of block programming, some of our students had difficulty with the logical arguments required for good programming. All of that being said, we feel that the Zumi was fun for the students, affordable, and demonstrated the difficulties of developing autonomous vehicles.

Over the summer of 2024, 2 undergraduate students researched using the NVIDIA Jetson Nano for this course. They specifically chose to investigate Duckietown and the DuckieBot (DB-J) robotics platform [4]. We ultimately decided that this platform is great for upper-level Computer Science majors but using it in a general education course would require too much training and the complexity would require extensive use of class time and lab assistance to help students navigate the robotics environment and GitHub, and learn to use the Robot Operating System and Python programming.

For spring 2025, we decided to use the mBot2 by Makeblock. The mBot2 is also powered by a Raspberry Pi Zero, but allows for processing in the cloud so that the IDE can use AI extensions for an ability to train on images or audio. Using the Makeblock IDE with a Teachable Machines extension, we were able to do what we wanted for the final assignment, which was to get the robot to follow a map or a track and to react to a variety of road signs. The students were able to train the robot to recognize the signs and to call functions to change the behavior of the robot. The major limitation of the mBot2 in this regard is that is does not have a camera on the standard build and the Smart Camera add-on pack is pre-programmed with things that can be recognized. To get around this, students used the webcam on their laptops to represent the robot "seeing" the images. While not ideal, we feel this is better than the alternatives and meets our learning goals for the course. For future iterations of the course, we plan to continue to use the mBot2 robots, but will continue to search for other robotics platforms at a cost-effective price point that may be better suited to our needs.

Lastly, it is not difficult to imagine that this course, the resources used, and the topics covered will continually need to be refreshed as the field, the applications, and societal, political, legal, and ethical thoughts on AI continue to change at a steady pace. With all of this change happening in the field, we see great value in teaching *Living with AI* to all students and feel it is important in creating citizens that are more well-informed on the topic. We encourage others to offer similar courses and it is hoped that our experience is a helpful starting point to that end.

References

[1] Donald Braxton and John Wright. *CONN301 - Living with AI Syllabus.* 2025. URL: https://jcsites.juniata.edu/faculty/wrightj/conn301/syllabus.htm.

[2] Vincent Cicirello. "An Interdisciplinary Course on Artificial Intelligence Designed for a Liberal Arts Curriculum". In: *Journal of Computing Sciences in Colleges* 23.3 (Jan. 2008). URL: https://dl.acm.org/doi/10.5555/1295109.1295137.

[3] Association for Computing Machinery. *ACM Code of Ethics and Professional Conduct.* 2018. URL: https://www.acm.org/code-of-ethics.

[4] Duckietown. *Duckietown Duckiebot DB-J.* 2025. URL: https://duckietown.com/.

[5] Makeblock. *Makeblock mBot2.* 2025. URL: https://www.makeblock.com/pages/mbot2-coding-robot.

[6] OpenAI. *Introducing ChatGPT.* 2022. URL: https://openai.com/index/chatgpt/.

[7] Robolink. *Meet Zumi.* 2023. URL: https://www.robolink.com/products/zumi.

Quantum Computing Unveiled: A Practical Approach for Computer Science Students[*]

Jingnan Xie[*]
[*]Department of Computer Science
Millersville University of Pennsylvania
Millersville, PA
jingnan.xie@millersville.edu

Abstract

Quantum computing's transformative potential in cryptography, material science, and machine learning is clear as the field advances. Yet, integrating it into computer science curricula is challenging due to quantum mechanics' steep learning curve and physics reliance. We propose an accessible approach for computer science students, emphasizing linear algebra over physics. This paper outlines nine key linear algebra concepts critical for quantum computing, reinterprets the four quantum mechanics postulates through this lens, and offers practical, adaptable resources for educators. By lowering entry barriers, we aim to equip students and educators for this emerging technology.

1 Introduction and Background

Quantum computing harnesses quantum mechanics principles like superposition and entanglement to process information differently from classical systems (see [11]). Recent advances underscore its potential: Alphabet's Willow chip (December 2024) solves problems in minutes that classical computers cannot, and IBM's latest quantum systems (November 2024) hint at breakthroughs in cryptography, material science, and machine learning. Industry investment

[*]Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

reflects a shift from theory to practice, making quantum education urgent [13, 5]. Yet, integration into computer science curricula lags [5, 8, 6], hindered by quantum mechanics' complexity and physics prerequisites (e.g., electron spin, wave-particle duality), which challenge students without such backgrounds.

Recent educational efforts offer solutions. Modular, scaffolded frameworks ease students into quantum concepts, linking them to classical principles [12, 3]. Hands-on platforms like IBM Q and Microsoft Quantum Development Kit reinforce theory with practice [4, 10], while game-based learning boosts engagement [15, 2]. Interdisciplinary approaches unite physics, math, and computer science [8, 9], and equity initiatives target underrepresented groups via outreach [7]. These strategies inform our goal: making quantum computing accessible for computer science students by minimizing physics reliance. This paper presents key concepts and adaptable resources to lower barriers, empowering educators and broadening quantum education's reach.

2 Pedagogical Approach and Essential Concepts

Traditional quantum courses use physics-based examples (e.g., photon polarization) requiring unfamiliar concepts like electromagnetic waves [14]. Instead, we adopt a linear algebra-based approach, familiar to computer science students, framing quantum computing as an extension of probabilistic computing [12]. Superposition and entanglement become properties of Hilbert space vectors, and gates are simple operations, reducing physics and complex math demands.

Our approach prioritizes accessibility and practicality. We identify nine core linear algebra concepts underpinning quantum computing and express the four quantum postulates using them, creating a simplified, adaptable framework. Unlike quantum programming-focused courses, we emphasize computational theory, highlighting parallels between classical and quantum models [21, 17, 19, 18, 20]. This builds a robust theoretical base, requiring minimal math initially and progressing gradually. Educators and students with limited experience can readily adapt these materials.

2.1 Linear Algebra

1. Complex Numbers

   Let $\mathbb{R}$ and $\mathbb{C}$ denote the set of real numbers and the set of complex numbers, respectively. A *complex number* $c \in \mathbb{C}$ is written in its standard form as

   $$c = a + bi,$$

   where $a, b \in \mathbb{R}$, and i is the imaginary unit satisfying $i^2 = -1$.

The *conjugate* of c is denoted by c^* and is given by

$$c^* = a - bi,$$

The *magnitude* or length or modulus of $c \in \mathbb{C}$ is

$$|c| = \sqrt{c \cdot c^*} = \sqrt{a^2 + b^2}$$

2. Complex Vectors

   Complex vectors shall be crucial since they represent quantum states (more details are discussed in section 3.2). Let $|\psi\rangle \in \mathbb{C}^d$ denote a complex (column) *vector*:

$$|\psi\rangle = \begin{pmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_d \end{pmatrix},$$

   where each entry $\psi_i \in \mathbb{C}$ for $i = 1, 2, \ldots, d$.

   The notation $|\cdot\rangle$ is referred to as *Dirac* notation, and is read as “ket". The dual representation of $|\psi\rangle$, denoted as $\langle\psi|$ and read as "bra," is the conjugate transpose (Hermitian conjugate) of $|\psi\rangle$. It is represented as a row vector:

$$\langle\psi| = (\psi_1^*, \psi_2^*, \ldots, \psi_d^*).$$

   Exercise 2.1 *Given* $|\psi\rangle = \begin{pmatrix} 1 \\ i \end{pmatrix}$, *what is* $\langle\psi|$*?*

   Solution: The conjugate of 1 is $1^* = 1$, and the conjugate of i is $i^* = -i$. Therefore, $\langle\psi| = \begin{pmatrix} 1 & -i \end{pmatrix}$.

3. Matrix Operations

   Since we require some essential matrix operations, let us review them briefly. Let A be a matrix, and $A(i, j)$ represent the entry of A in the i-th row and j-th column. The following operations are defined as:

   - *Conjugate of A:* $A^*(i, j) = (A(i, j))^*$.
   - *Transpose of A:* $A^T(i, j) = A(j, i)$.
   - *Adjoint (also called conjugate transpose, Hermitian conjugate, or dagger) of A:* $A^\dagger = (A^*)^T$.

Let A and B be two matrices. The *dot product* (multiplication) of A and B is given by

$$A \cdot B(i,j) = \sum_{k=1}^{d} A(i,k) \cdot B(k,j).$$

Example 2.1 *Let*

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, \quad B = \begin{pmatrix} 4 & 3 \\ 2 & 1 \end{pmatrix},$$

then

$$A \cdot B = \begin{pmatrix} 1 \cdot 4 + 2 \cdot 2 & 1 \cdot 3 + 2 \cdot 1 \\ 3 \cdot 4 + 4 \cdot 2 & 3 \cdot 3 + 4 \cdot 1 \end{pmatrix} = \begin{pmatrix} 8 & 5 \\ 20 & 13 \end{pmatrix}.$$

4. Inner Product

   Given two vectors $|\psi\rangle$ and $|\phi\rangle$, the *inner product* of them is denoted by $\langle\psi|\phi\rangle$, and is defined as

$$\langle\psi|\phi\rangle = \sum_{i=1}^{d} \psi_i^* \cdot \phi_i.$$

   The inner product measures the "overlap" (or similarity) between the two vectors. If $\langle\psi|\phi\rangle = 0$, then the vectors are orthogonal, and if $\langle\psi|\phi\rangle = 1$, then the vectors are aligned in the same direction.

Exercise 2.2 *Let* $|\psi\rangle = \begin{pmatrix} 1 \\ i \end{pmatrix}$ *and* $|\phi\rangle = \begin{pmatrix} 2 \\ 3i \end{pmatrix}$. *Compute their inner product.*

Solution:

$$\langle\psi|\phi\rangle = \begin{pmatrix} 1 & -i \end{pmatrix} \cdot \begin{pmatrix} 2 \\ 3i \end{pmatrix} = 1 \cdot 2 + (-i) \cdot (3i) = 2 + 3 = 5.$$

Note that the inner product returns a scalar, and we have

$$(\langle\psi|\phi\rangle)^* = \langle\phi|\psi\rangle.$$

5. Euclidean Norm

   The *Euclidean norm (2-norm)* of a vector $|\psi\rangle$ is denoted by $\||\psi\rangle\|_2$, and is given by

$$\||\psi\rangle\|_2 = \sqrt{\langle\psi|\psi\rangle} = \sqrt{\sum_{i=1}^{d} \psi_i^* \psi_i} = \sqrt{\sum_{i=1}^{d} |\psi_i|^2}.$$

Exercise 2.3 *Let* $|\psi\rangle = \begin{pmatrix} 3+4i \\ 1-i \end{pmatrix}$. *Compute its Euclidean norm.*

Solution:

$$\||\psi\rangle\|_2 = \sqrt{(3+4i)^*(3+4i) + (1-i)^*(1-i)} = 3\sqrt{3}.$$

6. Outer Product

   For two vectors $|\psi\rangle, |\phi\rangle \in \mathbb{C}^d$, the *outer product* $|\psi\rangle\langle\phi|$ yields a $d \times d$ matrix.

Example 2.2 *Let* $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ *and* $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$.

$$|0\rangle\langle 0| = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

$$|1\rangle\langle 0| = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$$

7. Linear Operators

   A *linear operator* A is a $d \times d$ matrix that maps $\mathbb{C}^d \to \mathbb{C}^d$ with the following linear property:

$$A\left(\sum_i a_i|\psi_i\rangle\right) = \sum_i a_i A|\psi_i\rangle,$$

   where $a_i \in \mathbb{C}$ and $|\psi_i\rangle \in \mathbb{C}^d$.

   The *matrix element* is a scalar quantity that provides information about how the operator A acts on the state $|\psi\rangle$ and how the resulting state overlaps with the state $|\phi\rangle$.

$$\langle\phi|A|\psi\rangle = \langle\phi|(A|\psi\rangle) = \sum_{i,j} \phi_i^*(A(i,j)\psi_j),$$

   where ϕ_i^* is the complex conjugate of the i-th component of $|\phi\rangle$, $A(i,j)$ is the (i,j)-th entry of the operator A, and ψ_j is the j-th component of $|\psi\rangle$.

Example 2.3 *Given* $|\psi\rangle = \begin{pmatrix}1\\0\end{pmatrix}, |\phi\rangle = \begin{pmatrix}0\\1\end{pmatrix}, A = \begin{pmatrix}2 & 3\\1 & 4\end{pmatrix}$,

$$\langle\phi|A|\psi\rangle = \langle\phi| \begin{pmatrix}2 & 3\\1 & 4\end{pmatrix} \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0 & 1\end{pmatrix} \begin{pmatrix}2\\1\end{pmatrix} = 1.$$

Thus, $\langle\phi|A|\psi\rangle = 1$.

8. Orthonormal Bases

   A set of vectors $\{|\psi_i\rangle\} \subseteq \mathbb{C}^d$ is said to be *orthogonal* if for all $i \neq j$, $\langle\psi_i|\psi_j\rangle = 0$. The set is *orthonormal* if

$$\langle\psi_i|\psi_j\rangle = \begin{cases} 0, & \text{if } i \neq j, \\ 1, & \text{if } i = j. \end{cases}$$

   Hence, each vector satisfies $\||\psi_i\rangle\|_2 = 1$, and every distinct pair of vectors is orthogonal.

   For every vector in $\mathbb{C}^d$, it can be expressed as a linear combination of an orthonormal basis. For example, in $\mathbb{C}^2$, the most common basis is $|0\rangle = \begin{pmatrix}1\\0\end{pmatrix}$ and $|1\rangle = \begin{pmatrix}0\\1\end{pmatrix}$. Any vector $|\psi\rangle \in \mathbb{C}^2$ can be written as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle,$$

   where $\alpha, \beta \in \mathbb{C}$. We say that $|\psi\rangle$ is *normalized* if $\||\psi\rangle\|_2 = 1$, which is equivalent to

$$|\alpha|^2 + |\beta|^2 = 1.$$

9. Eigenvalues and Eigenvectors

   Given a matrix A, an eigenvector $|\psi\rangle$ is a non-zero vector that satisfies the equation

$$A \cdot |\psi\rangle = \lambda|\psi\rangle$$

   for some scalar $\lambda \in \mathbb{C}$. We call λ the corresponding eigenvalue of A.

Example 2.4 *Show that* $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ *is an eigenvector of* $A = \begin{pmatrix}0 & 1\\1 & 0\end{pmatrix}$.

$$A|+\rangle = \begin{pmatrix}0 & 1\\1 & 0\end{pmatrix} \frac{1}{\sqrt{2}}(|1\rangle + |0\rangle) = \frac{1}{\sqrt{2}} \begin{pmatrix}0 & 1\\1 & 0\end{pmatrix} \begin{pmatrix}1\\1\end{pmatrix}$$

$$= \frac{1}{\sqrt{2}} \begin{pmatrix}1\\1\end{pmatrix} = |+\rangle$$

Thus, $|+\rangle$ *is an eigenvector of* A *with eigenvalue* $\lambda = 1$.

2.2 Quantum Mechanics

With linear algebra at hand, the four postulates of quantum mechanics can be stated, addressing the following questions: How to represent a single quantum system, how to perform operations on a quantum system, how to describe multiple quantum systems, and how to measure classical information from a quantum system.

2.2.1 Postulate 1: Individual Quantum Systems

In classical computing, a bit is either 0 or 1. In quantum computing, a *qubit* can be both simultaneously. We encode 0 and 1 as orthonormal basis vectors $|0\rangle$ and $|1\rangle$ in $\mathbb{C}^2$. A qubit in both states, called a *superposition*, is written $|0\rangle + |1\rangle$. Generally, a qubit state is:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle,$$

where $\alpha, \beta \in \mathbb{C}$ are amplitudes, and normalization requires:

$$|\alpha|^2 + |\beta|^2 = 1.$$

Thus, any unit vector in $\mathbb{C}^2$ represents a qubit state.

Example 2.5 $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ *and* $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ *are two widely mentioned single-qubit states.*

2.2.2 Postulate 2: Quantum Operations

Recall that a linear operator acts on the state vector and is represented by a matrix. The matrix must be with dimensions $d \times d$ to operate on a state in $\mathbb{C}^d$. A quantum operation is represented by a linear operator, which must be a *unitary* matrix. A matrix U is *unitary* if it satisfies the condition

$$UU^\dagger = U^\dagger U = I,$$

where $U^\dagger$ is the Hermitian conjugate (or adjoint) of U, and I is the identity matrix. Therefore, $U^\dagger$ is the inverse of U.

Example 2.6 Pauli-X gate:

$$\begin{aligned} X &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \\ X^\dagger &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad XX^\dagger &&= X^\dagger X = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \end{aligned}$$

Pauli-Y gate:

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

Pauli-Z gate:

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Hadamard gate:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Exercise 2.4

$$\begin{aligned} X|0\rangle &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle, \\ X|1\rangle &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle. \end{aligned}$$

Hence, X is also called the quantum OR gate.

$$\begin{aligned} H|-\rangle &= H\left(\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\right) = \frac{1}{\sqrt{2}}(H|0\rangle - H|1\rangle) \\ &= \frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) - \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\right) = |1\rangle. \end{aligned}$$

2.2.3 Postulate 3: Composite Quantum Systems

To perform complex computations, we combine multiple qubits using the *tensor product*, denoted $\otimes$. For vectors $|\psi\rangle, |\phi\rangle \in \mathbb{C}^2$, $|\psi\rangle \otimes |\phi\rangle \in \mathbb{C}^4$ with $(|\psi\rangle \otimes |\phi\rangle)_{ij} = \psi_i \phi_j$.

Example 2.7

$$|0\rangle \otimes |0\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad |0\rangle \otimes |1\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

A 2-qubit system can be in a superposition of 4 basis states, and an n-qubit system in 2^n states. Superposition and entanglement *potentially enable quantum computers to outperform classical ones.*

Consider the Bell state:

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\0\\0\\1\end{pmatrix}$$

This state is *entangled*, meaning it cannot be written as $|\psi_1\rangle \otimes |\psi_2\rangle$. Measuring $|\Phi^+\rangle$ yields either $|00\rangle$ or $|11\rangle$.

Other Bell states are:

$$|\Phi^-\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle), \quad |\Psi^+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle), \quad |\Psi^-\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$$

For an n-qubit system, quantum gates are $2^n \times 2^n$ unitary matrices. For 2-qubit gates (4×4 matrices), examples include tensor products like:

$$X \otimes Z = \begin{bmatrix}0 & 0 & 1 & 0\\0 & 0 & 0 & -1\\1 & 0 & 0 & 0\\0 & -1 & 0 & 0\end{bmatrix}$$

and genuinely 2-qubit gates like the CNOT gate:

$$\text{CNOT} = \begin{bmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 0 & 1\\0 & 0 & 1 & 0\end{bmatrix}$$

It flips the target qubit if the control is $|1\rangle$:

$$\text{CNOT}\,|10\rangle = |11\rangle, \quad \text{CNOT}\,|11\rangle = |10\rangle$$

We can prepare $|\Phi^+\rangle$ from $|00\rangle$:

$$\text{CNOT}\,(H \otimes I)|00\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = |\Phi^+\rangle$$

Exercise 2.5 *Use $|01\rangle$, $|10\rangle$, and $|11\rangle$ with the same gates to construct other Bell states.*

2.2.4 Postulate 4: Measurement

The measurement of a quantum state involves three classes of linear operators: Hermitian operators, positive semi-definite operators, and orthogonal projection operators. Readers can decide whether to cover these topics in their courses depending on the depth and audience of the course.

Without mentioning these operators, the measurement can be simplified as follows:

For a single-qubit state $\alpha|0\rangle+\beta|1\rangle$, the probability of measuring the outcome $|0\rangle$ is $|\alpha|^2$, and the probability of measuring the outcome $|1\rangle$ is $|\beta|^2$. After the measurement, the state collapses to the measured basis state, either $|0\rangle$ or $|1\rangle$.

For a two-qubit system in the state

$$|\psi\rangle = \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle,$$

the probabilities of measuring the outcomes $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$ are $|\alpha|^2$, $|\beta|^2$, $|\gamma|^2$, and $|\delta|^2$, respectively. Upon measurement, the state collapses to the measured basis state corresponding to the outcome.

In general, for an n-qubit quantum system, the measurement outcomes are determined by the probabilities associated with the amplitudes of each computational basis state. These probabilities always sum to 1, ensuring the state is properly normalized.

3 Challenges and Solutions

With Sections 2.1 and 2.2, we reduce mathematics and physics barriers, helping students grasp quantum computing and explore its algorithms. A key challenge is that single concepts have multiple names—e.g., for a complex number c, $|c|$is termed length, magnitude, or modulus—while distinct concepts like dot, inner, outer, and tensor products have similar names. Confusion grows when materials, like ChatGPT, incorrectly equate outer and tensor products. We counter this with consistent, distinct naming, plus examples and exercises, enabling students to understand and progress confidently.

3.1 Circuit Diagrams

Understanding quantum circuit diagrams can confuse students, despite their similarity to classical ones. We clarify this using quantum teleportation (see [1]). In these diagrams:

1. A single wire represents a qubit state.
2. A wire with no gate applies the identity matrix I.
3. Multiple wires denote the tensor product of single-qubit states.
4. A double wire after measurement indicates a classical bit string output.

Quantum teleportation transfers a qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ (with unknown α, β) using an entangled Bell state:

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle).$$

The process starts with the combined state:

$$|\psi\rangle\otimes|\Phi^+\rangle = (\alpha|0\rangle+\beta|1\rangle)\otimes\frac{1}{\sqrt{2}}(|00\rangle+|11\rangle) = \frac{1}{\sqrt{2}}(\alpha|000\rangle+\alpha|011\rangle+\beta|100\rangle+\beta|111\rangle).$$

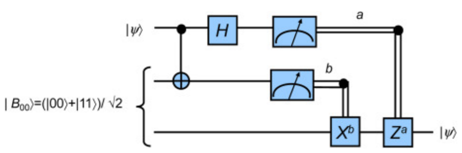


Figure 1: Quantum teleportation circuit.

First, apply a CNOT gate (first qubit as control, second as target):

$$\frac{1}{\sqrt{2}}(\alpha|000\rangle + \alpha|011\rangle + \beta|110\rangle + \beta|101\rangle).$$

Next, apply an H gate to the first qubit:

$$\frac{1}{\sqrt{2}}\big(\alpha(H|0\rangle)|00\rangle + \alpha(H|0\rangle)|11\rangle + \beta(H|1\rangle)|10\rangle + \beta(H|1\rangle)|01\rangle\big),$$

where $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. This becomes:

$$\frac{1}{2}\big(\alpha(|0\rangle + |1\rangle)|00\rangle + \alpha(|0\rangle + |1\rangle)|11\rangle + \beta(|0\rangle - |1\rangle)|10\rangle + \beta(|0\rangle - |1\rangle)|01\rangle\big).$$

Rearranging:

$$\frac{1}{2}\big(|00\rangle(\alpha|0\rangle + \beta|1\rangle) + |01\rangle(\alpha|1\rangle + \beta|0\rangle) + |10\rangle(\alpha|0\rangle - \beta|1\rangle) + |11\rangle(\alpha|1\rangle - \beta|0\rangle)\big).$$

Measuring the first two qubits (each outcome with probability $(\frac{1}{2})^2 = 25\%$) determines the third qubit's state. Your friend applies:

1. Nothing if $|00\rangle$: $\alpha|0\rangle + \beta|1\rangle = |\psi\rangle$,
2. X if $|01\rangle$: $X(\alpha|1\rangle + \beta|0\rangle) = |\psi\rangle$,
3. Z if $|10\rangle$: $Z(\alpha|0\rangle - \beta|1\rangle) = |\psi\rangle$,
4. ZX if $|11\rangle$: $ZX(\alpha|1\rangle - \beta|0\rangle) = |\psi\rangle$.

Your measurement, sent classically, enables your friend to reconstruct $|\psi\rangle$, regardless of distance.

3.2 Algebraic Rules

The final challenge addressed in this paper is the unfamiliarity with many algebraic rules used in quantum computing for students. As a result, even simple computations can cause hesitation.

To address this, we summarize some fundamental algebraic rules in quantum computing and demonstrate their simplicity and utility through a proof of the famous no-cloning theorem.

Let A, B, C, D be matrices and $|a\rangle$, $|b\rangle$, $|c\rangle$, $|d\rangle$ be vectors.

$$
\begin{aligned}
(AB)^\dagger &= B^\dagger A^\dagger && (1)\\
(AB)^T &= B^T A^T && (2)\\
\langle(\alpha|0\rangle + \beta|1\rangle)| &= \alpha^*\langle 0| + \beta^*\langle 1| && (3)\\
(|a\rangle + |b\rangle)\otimes|c\rangle &= |a\rangle\otimes|c\rangle + |b\rangle\otimes|c\rangle && (4)\\
|a\rangle\otimes(|b\rangle + |c\rangle) &= |a\rangle\otimes|b\rangle + |a\rangle\otimes|c\rangle && (5)\\
\alpha(|a\rangle\otimes|b\rangle) &= (\alpha|a\rangle)\otimes|b\rangle = |a\rangle\otimes(\alpha|b\rangle) && (6)\\
(|a\rangle\otimes|b\rangle)^\dagger &= |a\rangle^\dagger\otimes|b\rangle^\dagger = \langle a|\otimes\langle b| && (7)\\
((\langle a|\otimes\langle c|)(|b\rangle\otimes|d\rangle) &= \langle a|b\rangle\langle c|d\rangle && (8)\\
(A\otimes B)(C\otimes D) &= (AC)\otimes(BD) && (9)\\
\mathrm{Tr}(A\otimes B) &= \mathrm{Tr}(A)\,\mathrm{Tr}(B) && (10)
\end{aligned}
$$

where Tr represents the trace of a matrix.

With these rules at hand, let us prove the no-cloning theorem for quantum states.

Theorem 3.1 *[16] Given an arbitrary quantum state $|\psi\rangle \in \mathbb{C}^2$, there is no quantum circuit capable of creating an exact copy of $|\psi\rangle$.*

Proof: Suppose there exists a unitary operator U of dimension 2×2 such that for any quantum state $|\psi\rangle \in \mathbb{C}^2$, U maps $|\psi\rangle\otimes|0\rangle$ to $|\psi\rangle\otimes|\psi\rangle$, i.e., U creates a copy of $|\psi\rangle$.

Then, for two arbitrary states $|\psi_1\rangle$, $|\psi_2\rangle$ in $\mathbb{C}^2$, we have:

$$|\phi\rangle = U(|\psi_1\rangle\otimes|0\rangle) = |\psi_1\rangle\otimes|\psi_1\rangle$$

$$|\phi'\rangle = U(|\psi_2\rangle\otimes|0\rangle) = |\psi_2\rangle\otimes|\psi_2\rangle$$

Now, consider the inner product $\langle\phi|\phi'\rangle$:

$$\begin{aligned}\langle\phi|\phi'\rangle &= (|\phi\rangle)^\dagger|\phi'\rangle \\ &= (U(|\psi_1\rangle\otimes|0\rangle))^\dagger\, U(|\psi_2\rangle\otimes|0\rangle) \\ &= (|\psi_1\rangle\otimes|0\rangle)^\dagger U^\dagger U(|\psi_2\rangle\otimes|0\rangle) \\ &= (\langle\psi_1|\otimes\langle 0|)(|\psi_2\rangle\otimes|0\rangle) \\ &= \langle\psi_1|\psi_2\rangle\langle 0|0\rangle \\ &= \langle\psi_1|\psi_2\rangle\end{aligned}$$

Also,

$$\begin{aligned}\langle\phi|\phi'\rangle &= (|\psi_1\rangle\otimes|\psi_1\rangle)^\dagger(|\psi_2\rangle\otimes|\psi_2\rangle) \\ &= (\langle\psi_1|\otimes\langle\psi_1|)(|\psi_2\rangle\otimes|\psi_2\rangle) \\ &= \langle\psi_1|\psi_2\rangle\langle\psi_1|\psi_2\rangle \\ &= \langle\psi_1|\psi_2\rangle^2.\end{aligned}$$

This implies $\langle\psi_1|\psi_2\rangle = \langle\psi_1|\psi_2\rangle^2$. Hence, $\langle\psi_1|\psi_2\rangle$ is either 0 or 1.

Therefore, $|\psi_1\rangle$ and $|\psi_2\rangle$ are either orthogonal or in the same direction. This leads to a contradiction. □

4 Evaluation and Conclusion

In this paper, we introduced an approach to teaching introductory quantum computing from a computer science perspective, aiming to lower the mathematical and physical barriers for students. In Section 3, we summarized nine key concepts in linear algebra and reformulated the four postulates of quantum mechanics using linear algebra, making the material more accessible for computer science students. While the course has not yet been offered, and its evaluation and analysis remain as future work, the framework presented here provides a practical and adaptable starting point. We hope that educators can use this approach to initiate their own efforts in quantum computing education, fostering broader accessibility and interest in this emerging field.

References

[1] Charles H Bennett et al. "Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels". In: *Physical review letters* 70.13 (1993), p. 1895.

[2] Daniel Escanez-Exposito, Javier Correa-Marichal, and Pino Caballero-Gil. “Using Game-Based Learning and Quantum Computing to Enhance STEAM Competencies in K-16 Education”. In: *IEEE Transactions on Education* (2024).

[3] Simon Goorney et al. “A framework for curriculum transformation in quantum information science and technology education”. In: *European Journal of Physics* 45.6 (2024), p. 065702.

[4] Shi-Yao Hou et al. “SpinQ Gemini: a desktop quantum computing platform for education and research”. In: *EPJ Quantum Technology* 8.1 (2021), pp. 1–23.

[5] Maninder Kaur and Araceli Venegas-Gomez. “Defining the quantum workforce landscape: a review of global quantum education initiatives”. In: *Optical Engineering* 61.8 (2022), pp. 081806–081806.

[6] Tunde Kushimo and Beth Thacker. “Investigating students’ strengths and difficulties in quantum computing”. In: *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*. Vol. 3. IEEE. 2023, pp. 33–39.

[7] Josephine C Meyer, Gina Passante, and Bethany Wilcox. “Disparities in access to US quantum information education”. In: *Physical Review Physics Education Research* 20.1 (2024), p. 010131.

[8] Josephine C Meyer et al. “Today’s interdisciplinary quantum information classroom: Themes from a survey of quantum information science instructors”. In: *Physical Review Physics Education Research* 18.1 (2022), p. 010150.

[9] MI Mihailescu et al. “Integrating Quantum Computing Into New Learning Technologies”. In: *EDULEARN24 Proceedings*. IATED. 2024, pp. 22–31.

[10] Mariia Mykhailova. “Teaching quantum computing using microsoft quantum development kit and azure quantum”. In: *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*. Vol. 3. IEEE. 2023, pp. 15–20.

[11] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010.

[12] Özlem Salehi, Zeki Seskir, and Ilknur Tepe. “A computer science-oriented approach to introduce quantum computing to a new audience”. In: *IEEE Transactions on Education* 65.1 (2021), pp. 1–8.

[13] Stefan Seegerer, Tilman Michaeli, and Ralf Romeike. “Quantum computing as a topic in computer science education”. In: *Proceedings of the 16th Workshop in Primary and Secondary Computing Education.* 2021, pp. 1–6.

[14] Karl Svozil. *Quantum logic.* Springer Science & Business Media, 1998.

[15] Justin D Weisz, Maryam Ashoori, and Zahra Ashktorab. “Entanglion: A board game for teaching the principles of quantum computing”. In: *Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play.* 2018, pp. 523–534.

[16] William K Wootters and Wojciech H Zurek. “A single quantum cannot be cloned”. In: *Nature* 299.5886 (1982), pp. 802–803.

[17] Jingnan Xie and Harry B. Hunt III. “On the Undecidability and Descriptional Complexity of Synchronized Regular Expressions”. In: *Acta Informatica* 60.3 (Apr. 2023), pp. 257–278. ISSN: 0001-5903. DOI: 10.1007/s00236-023-00439-3. URL: https://doi.org/10.1007/s00236-023-00439-3.

[18] Jingnan Xie, Harry B. Hunt, and Richard E. Stearns. “On Productiveness and Complexity in Computable Analysis Through Rice-Style Theorems for Real Functions”. In: *Mathematics* 12.20 (2024). ISSN: 2227-7390. DOI: https://doi.org/10.3390/math12203248.

[19] Jingnan Xie, Harry B. Hunt, and Richard E. Stearns. “On the computational and descriptional complexity of multi-pattern languages”. In: *Theoretical Computer Science* 1030 (2025), p. 115063. ISSN: 0304-3975. DOI: https://doi.org/10.1016/j.tcs.2025.115063.

[20] Jingnan Xie, Harry B Hunt III, and Richard E Stearns. “Pumping Lemmas Can be “Harmful””. In: *Theory of Computing Systems* 68 (2024), pp. 1339–1352. DOI: https://doi.org/10.1007/s00224-024-10169-9.

[21] Jingnan Xie et al. “A Practical Extension of Computational Complexity Theory for Applications in Mathematics and Sciences”. In: *Available at SSRN* (2025). DOI: http://dx.doi.org/10.2139/ssrn.5163632.

The Use of Blockchain Technology in Maritime Ship Freight Supply Management: Tracking Temperature-Sensitive Cargo*

M. Pinto[1] and H. Wei[2]
[1] NYC College of Technology, CUNY
Brooklyn, NY 11201
mpinto@citytech.cuny.edu
[2] US Marine Merchant Academy (USMMA)
Kings Point, NY 11024
weih@usmma.edu

Abstract

The international maritime shipping industry, responsible for over 90% of global trade, faces critical challenges in ensuring safety and security in the transport of goods. The integration of blockchain technology in maritime ship freight supply management presents a transformative approach to tracking temperature-sensitive cargo with enhanced transparency, efficiency, and security [7]. This study explores the application of decentralized ledgers, smart contracts, and IoT-enabled sensors to improve cargo integrity and reduce logistical inefficiencies. By analyzing industry case studies, conducting expert interviews, and examining empirical data, the research evaluates blockchain's capability to ensure real-time tracking, reduce fraudulent activities, and maintain optimal temperature conditions throughout shipping. The findings highlight blockchain's potential to streamline operations, enhance regulatory compliance, and provide an immutable record of cargo conditions, ultimately improving reliability in maritime supply chains. However, challenges such as regulatory disparities, scalability concerns, and implementation costs pose barriers to widespread adoption. This paper offers

*Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

insights into blockchain's impact on maritime logistics, addressing both its opportunities and limitations while proposing strategies for effective integration in temperature- sensitive cargo tracking systems.

1 Introduction

The maritime industry, the backbone of global trade, faces significant challenges in managing the complex supply chain. Paper-based processes, lack of transparency, and data silos result in inefficiencies, delays, and increased costs. Some of the most common causes of supply chain disruptions are natural disasters, man-made disasters, supplier issues, transportation issues, demand volatility, inventory shortages, regulatory changes, and cyber attacks [9]. For instance, in July of 2021 the ocean carrier Evergreen ran aground in the Suez Canal and it took six days for it to be dislodged. Days after the vessel was freed, hundreds of container ships were still waiting to get through the canal as a result of the backlog created by the blockage. Even with improving canal crossing conditions, logistics managers experienced capacity challenges and delayed shipments worldwide. Another example of supply chain disruption occurred with semiconductors (chips). The global chip shortage was sparked in early 2020 due to the COVID-19 pandemic and it has still not fully dissipated, and there's no clear answer on when it will end. These issues underscore the urgent need for innovative solutions. Blockchain technology, with its inherent features of decentralization, transparency, and immutability, offers a promising avenue to address these challenges. This research investigates the potential of blockchain technology to revolutionize maritime ship freight supply management. By examining key functionalities and their applications, we aim to assess the benefits and challenges of blockchain adoption [12]. This study will contribute to the development of a strategy for successful blockchain implementation, ultimately enhancing the efficiency, security, and sustainability of the global maritime industry.

This paper explores the potential of blockchain technology to address the aforementioned challenges and improve maritime ship freight supply management. We will delve into the key functionalities of blockchain and their specific applications within this domain. Furthermore, we will analyze the potential benefits of blockchain adoption, including enhanced transparency, increased efficiency, and improved security.

The research also acknowledges potential challenges hindering widespread blockchain implementation in the maritime industry. These may include scalability concerns, regulatory hurdles, and the need for industry-wide collaboration. By examining the opportunities and challenges associated with blockchain technology, this study aims to contribute to a more efficient, transparent, and secure future for the global marine merchant industry [6].

2 Background

The global maritime freight industry serves as the backbone of international trade, transporting vast quantities of goods across oceans with precision and efficiency. Among these shipments, temperature-sensitive cargo, such as pharmaceuticals, perishable foods, and chemical compounds, requires rigorous monitoring to ensure product integrity and prevent financial losses. Traditional supply chain tracking systems often suffer from limited transparency, inefficiencies, and susceptibility to fraudulent activity, resulting in compromised cargo conditions. These tracking systems rely on manual processes, limited visibility, and reactive management to monitor goods as they move through the supply chain. Many traditional systems depend on physical documentation such as invoices, shipping manifests, and inventory logs. Data is manually entered into spreadsheets or databases, increasing the risk of human error and inefficiencies. Traditional tracking often relies on periodic updates from logistics providers, meaning businesses receive delayed information about shipments. This lack of real- time tracking makes it difficult to respond quickly to disruptions. Many logistics companies still rely on phone calls and emails to track shipments, leading to slow response times and miscommunication. This method lacks automation and fails to provide instant updates. These methods are being replaced by modern digital solutions, such as blockchain, IoT-enabled sensors, and AI-driven predictive analytics, which offer real-time tracking, enhanced security, and automated data management [11].

Blockchain is append-only, which means that data can only be added to the blockchain in time-ordered sequential order. This property implies that once data is added to the blockchain, it is almost impossible to change that data and can be considered practically immutable. Blockchain is a distributed ledger, which simply means that a ledger is spread across the network among all peers in the network, and each peer holds a copy of the complete ledger. This ledger is cryptographically- secure, which means that cryptography has been used to provide security services which make this ledger secure against tampering and misuse. The most critical attribute of a blockchain is that it is updateable only via consensus. The consensus mechanism is a protocol that ensures all nodes (participants) in a decentralized network agree on the validity of transactions and the state of the ledger. It prevents double-spending and maintains security without relying on a central authority [4]. Examples of consensus mechanisms: PoW (Proof of Work), PoA (Proof of Authority), PoS (Proof of Stake), PoH (Proof of History), and BFT (Byzantine Fault Tolerance). Figure 1 shows the diagram of a blockchain network that integrates sensor data tracking into the blockchain network, illustrating how temperature-sensitive cargo is monitored.

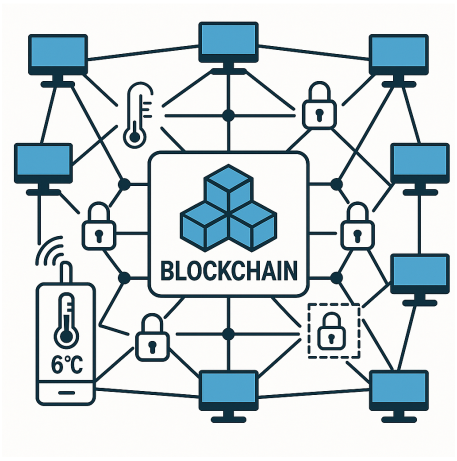
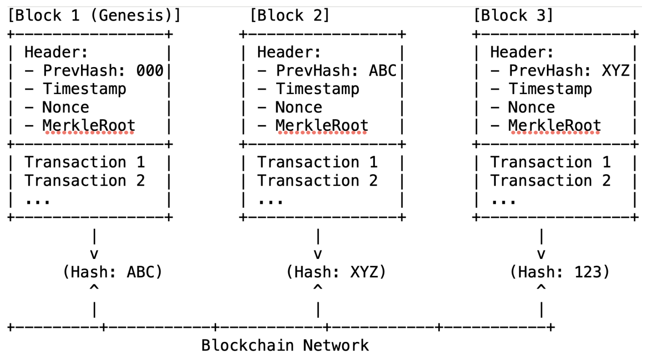


Figure 1: Diagram Blockchain Network.



```
[Block 1 (Genesis)]          [Block 2]              [Block 3]
+-----------------+      +-----------------+      +-----------------+
| Header:         |      | Header:         |      | Header:         |
| - PrevHash: 000|      | - PrevHash: ABC|      | - PrevHash: XYZ|
| - Timestamp     |      | - Timestamp     |      | - Timestamp     |
| - Nonce         |      | - Nonce         |      | - Nonce         |
| - MerkleRoot    |      | - MerkleRoot    |      | - MerkleRoot    |
+-----------------+      +-----------------+      +-----------------+
| Transaction 1   |      | Transaction 1   |      | Transaction 1   |
| Transaction 2   |      | Transaction 2   |      | Transaction 2   |
| ...             |      | ...             |      | ...             |
+-----------------+      +-----------------+      +-----------------+
         |                        |                        |
         v                        v                        v
     (Hash: ABC)              (Hash: XYZ)              (Hash: 123)
         ^                        ^                        ^
         |                        |                        |
+---------+-----------+-----------+-----------+-----------+
                    Blockchain Network
```

Figure 2: Diagram of a Blockchain.

The locks represent the cryptographic security mechanisms in the blockchain network. These ensure that blocks are tamper-resistant and transactions are validated before being added to the chain. The one marked with a dashed rectangle indicates a pending block that contains real-time temperature readings awaiting validation before storage which is the requirement in some blockchain networks, such as Hyperledger Fabric, transactions go through an endorsement and ordering process before final commitment to the ledger. Figure 2 shows the diagram of a blockchain in the blockchain network. The first block is called the Genesis block that has no previous hash (000). Each block's hash is derived from its contents (cryptographically linked to the previous block). Thus, PrevHash in Block 2 points to Block 1's hash (ABC), forming the "chain". Merkle Root is a hash summarizing all transactions in the block (for efficiency), and Nonce is a random number used in mining (PoW) to find a valid hash.

Blockchain technology has emerged as a transformative solution to the challenges of the supply chain businesses, providing secure, immutable, and decentralized ledgers for tracking shipments in real time. By integrating smart contracts - the business logic, IoT-enabled sensors, and digital authentication mechanisms, blockchain enables precise temperature monitoring, enhances data security, and mitigates operational risks in maritime logistics. Despite its potential, adoption within the industry faces significant barriers, including regulatory complexities, high implementation costs, and scalability concerns [3].

This paper examines the role of blockchain technology in maritime ship freight supply management, focusing specifically on its ability to track and safeguard temperature-sensitive cargo, such as pharmaceuticals, perishables, and chemicals. The shipment of these materials requires stringent temperature control throughout the supply chain. Through empirical research, we evaluate the technology's advantages, limitations, and future applications. The findings offer insights into how blockchain can revolutionize supply chain transparency while addressing key challenges that impact global logistics [1].

3 Methodology

The main fields of marine shipping business within which blockchain technology is currently being tested or already implemented are identified as: contracting and documentation flow (e.g. Bill of Lading), smart contracts, container/-cargo track-and-trace, marine insurance, ship register system, bunker tracking system, and crew certification system.

This research proposes a practical, experimental methodology for implementing and evaluating a blockchain-based system for tracking temperature-sensitive cargo in maritime ship freight. The core of this methodology lies in integrating IoT temperature sensors with a Hyperledger Fabric blockchain network, leveraging smart contracts for automated data management, and developing a user- friendly dashboard for real-time visualization and alert generation. It follows a phased approach, starting with the system architecture design, followed by hardware and software development, and culminating in testing and evaluation using synthetic data. This approach allows for direct observation of the system's performance, identification of challenges, and validation of the proposed solution.

The system architecture, figure 3, is comprised of the following key compo-

nents:
(i) Temperature Sensors: Industrially robust, low-power temperature sensors with wireless communication capabilities (e.g., Bluetooth Low Energy,LoRaWAN) are the source of core data (temperature readings along with timestamps and location data). These sensors are assumed to be installed within ship containers and are capable of continuous temperature logging at pre-defined intervals.
(ii) Gateway Devices: Each container or a cluster of containers are equipped with a gateway device (e.g., Raspberry Pi with cellular/satellite modem) responsible for collecting data from the sensors and transmitting it to the blockchain network.
(iii) Hyperledger Fabric Network: A private, permissioned blockchain network based on Hyperledger Fabric that can be deployed on a cloud platform (e.g., AWS, Azure) or a local server infrastructure. This provides a secure, immutable, and transparent ledger for temperature data. The network consists of multiple peer nodes representing different stakeholders (e.g., shipping company, cargo owner, port authority).
(iv) Smart Contracts (Chaincode): Smart contracts, written in GoLang, are developed and deployed on the Hyperledger Fabric network. These contracts govern the logic of data recording, threshold monitoring, and alert generation. These alerts can be sent via email, SMS, or push notifications to designated stakeholders.
(vi) Web-Based Dashboard: A user-friendly web interface is developed to visualize temperature data, display cargo location, and manage alerts.

First, we generate a synthetic dataset, a csv (comma separated values) file, figure 4 is an excerpt, of 100 readings by running a Python program that randomly produces a reading from one of the three containers: CNT-001, CNT-002, CNT-003, every 10 minutes. We set thresholds as >25°C for high, <-10°C for low so alerts can be produced. Each reading is submitted as a transaction to the blockchain.Now we can use for example a Python program to query the blockchain for alerts and container history and we get the following output shown in figure 5.

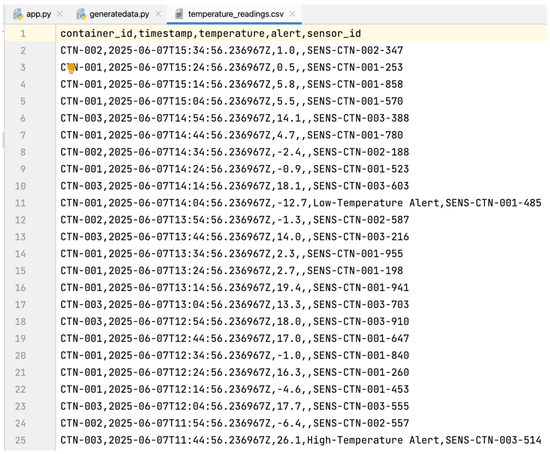
By implementing a blockchain-based solution for temperature-sensitive cargo, the maritime industry can track temperature data in real-time thus enabling timely interventions, guarantees data integrity and prevents tampering, provides transparency to all stakeholders involved in the supply chain, and minimizes the risk of product damage or spoilage. In summary, blockchain technology enhances supply chain efficiency, reduces risks, and improves customer satisfaction [13].

```
+-----------------------------------------------------------------+
|  [Temperature Sensors]                                          |
|  - Installed in shipping containers                             |
|  - Wireless (BLE/LoRaWAN)                                       |
|  - Log temperature + timestamp + location                       |
+--------+--------------------------------------------------------+
         | (Wireless Transmission)
         v
+--------+--------------------------------------------------------+
|  [Gateway Devices]                                              |
|  - Raspberry Pi + Cellular/Satellite Modem                      |
|  - Collects sensor data                                         |
|  - Transmits to Hyperledger Fabric Network                      |
+--------+--------------------------------------------------------+
         | (HTTPS/API)
         v
+--------+--------------------------------------------------------+
|  [Hyperledger Fabric Network]                                   |
|  - Private, permissioned blockchain                             |
|  - Components:                                                  |
|    * Peer Nodes (Stakeholders: Shipping Co., Cargo Owner, etc)|
|    * Orderers (Consensus)                                       |
|    * Certificate Authority (CA)                                 |
|    * Channels (Private data segregation)                        |
|                                                                 |
|  [Smart Contracts (Chaincode)]                                  |
|  - Written in GoLang                                            |
|  - Rules for:                                                   |
|    * Data recording                                             |
|    * Threshold monitoring (e.g., temp > 25°C → Alert)           |
|    * Alert triggering                                           |
+--------+--------------------------------------------------------+
         | (Event Triggers)
         v
+--------+--------------------------------------------------------+
|  [Notification Service]                                         |
|  - Integrated APIs (Email/SMS/Push)                             |
|  - Sends alerts to stakeholders                                 |
+--------+--------------------------------------------------------+
         | (REST API)
         v
+--------+--------------------------------------------------------+
|  [Web-Based Dashboard]                                          |
|  - Real-time temperature & location visualization               |
|  - Alert management                                             |
|  - Access control for stakeholders                              |
+-----------------------------------------------------------------+
```

Figure 3: Blockchain System Architecture.

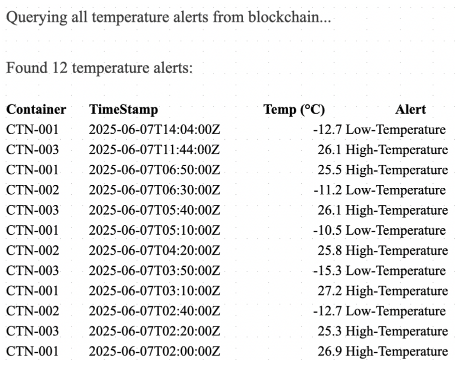
4 Smart Contract Development and Deployment

We need to build a smart contract that will be deployed to a blockchain. First we need to define the data structure for temperature records on the blockchain, including fields for cargoID, timestamp, temperatureReading, unit (e.g., Celsius, Fahrenheit), locationData (latitude, longitude), sensorID, and containerID. We also need to set the temperature threshold for various types of temperature- sensitive cargo (e.g., pharmaceuticals, fresh produce, frozen goods). These thresholds will be configurable within the smart contract or via an initialization transaction. The smart contract has also some core functions such as: (i) RecordTemperature (cargoID, timestamp, temperatureReading, unit, locationData, sensorID, containerID). This function records new temperature readings on the blockchain, and it is invoked by the gateway devices. (ii) CheckAlerts(cargoID, temperatureReading). It is the logic that compares the temperatureReading against pre-defined thresholds and returns all ALERT-



```
container_id,timestamp,temperature,alert,sensor_id
CTN-002,2025-06-07T15:34:56.236967Z,1.0,,SENS-CTN-002-347
CTN-001,2025-06-07T15:24:56.236967Z,0.5,,SENS-CTN-001-253
CTN-001,2025-06-07T15:14:56.236967Z,5.8,,SENS-CTN-001-858
CTN-001,2025-06-07T15:04:56.236967Z,5.5,,SENS-CTN-001-570
CTN-003,2025-06-07T14:54:56.236967Z,14.1,,SENS-CTN-003-388
CTN-001,2025-06-07T14:44:56.236967Z,4.7,,SENS-CTN-001-780
CTN-002,2025-06-07T14:34:56.236967Z,-2.4,,SENS-CTN-002-188
CTN-001,2025-06-07T14:24:56.236967Z,-0.9,,SENS-CTN-001-523
CTN-003,2025-06-07T14:14:56.236967Z,18.1,,SENS-CTN-003-603
CTN-001,2025-06-07T14:04:56.236967Z,-12.7,Low-Temperature Alert,SENS-CTN-001-485
CTN-002,2025-06-07T13:54:56.236967Z,-1.3,,SENS-CTN-002-587
CTN-003,2025-06-07T13:44:56.236967Z,14.0,,SENS-CTN-003-216
CTN-001,2025-06-07T13:34:56.236967Z,2.3,,SENS-CTN-001-955
CTN-001,2025-06-07T13:24:56.236967Z,2.7,,SENS-CTN-001-198
CTN-001,2025-06-07T13:14:56.236967Z,19.4,,SENS-CTN-001-941
CTN-003,2025-06-07T13:04:56.236967Z,13.3,,SENS-CTN-003-703
CTN-003,2025-06-07T12:54:56.236967Z,18.0,,SENS-CTN-003-910
CTN-001,2025-06-07T12:44:56.236967Z,17.0,,SENS-CTN-001-647
CTN-001,2025-06-07T12:34:56.236967Z,-1.0,,SENS-CTN-001-840
CTN-001,2025-06-07T12:24:56.236967Z,16.3,,SENS-CTN-001-260
CTN-001,2025-06-07T12:14:56.236967Z,-4.6,,SENS-CTN-001-453
CTN-003,2025-06-07T12:04:56.236967Z,17.7,,SENS-CTN-003-555
CTN-002,2025-06-07T11:54:56.236967Z,-6.4,,SENS-CTN-002-557
CTN-003,2025-06-07T11:44:56.236967Z,26.1,High-Temperature Alert,SENS-CTN-003-514
```

Figure 4: Synthetic Data (excerpt).



```
Querying all temperature alerts from blockchain...

Found 12 temperature alerts:

Container   TimeStamp                 Temp (°C)      Alert
CTN-001     2025-06-07T14:04:00Z         -12.7 Low-Temperature
CTN-003     2025-06-07T11:44:00Z          26.1 High-Temperature
CTN-001     2025-06-07T06:50:00Z          25.5 High-Temperature
CTN-002     2025-06-07T06:30:00Z         -11.2 Low-Temperature
CTN-003     2025-06-07T05:40:00Z          26.1 High-Temperature
CTN-001     2025-06-07T05:10:00Z         -10.5 Low-Temperature
CTN-002     2025-06-07T04:20:00Z          25.8 High-Temperature
CTN-003     2025-06-07T03:50:00Z         -15.3 Low-Temperature
CTN-001     2025-06-07T03:10:00Z          27.2 High-Temperature
CTN-002     2025-06-07T02:40:00Z         -12.7 Low-Temperature
CTN-003     2025-06-07T02:20:00Z          25.3 High-Temperature
CTN-001     2025-06-07T02:00:00Z          26.9 High-Temperature
```

Figure 5: Blockchain Query Output.

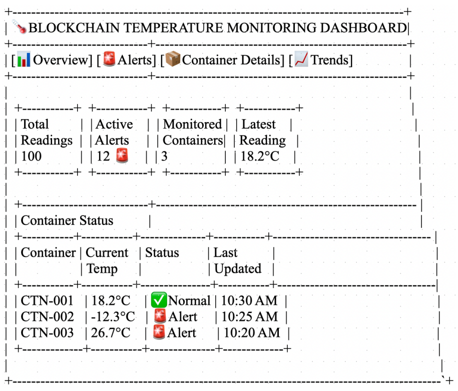
status readings. This triggers the notification service.

(iv) GetTemperatureHistory(cargoID). It retrieves all temperature records for a specific cargo. Now we have to package the chaincode and deploy it into every node of the network. In Hyperledger Fabric, packaging chaincode refers to bundling the smart contract (chaincode) into a deployable format that can be installed on peers across the blockchain network. This ensures consistency, security, and ease of distribution [10].

5 Results

The proposed blockchain-based temperature monitoring system was implemented and tested to evaluate its performance, reliability, and alerting accuracy. The system success- fully generated a synthetic dataset, enforced temperature thresholds, and triggered alerts while maintaining an immutable ledger.

A user interface, web-based dashboard, is developed using HTML markup language, Javascript, Python, and running on a Flask server. Flask server is built-in development server for testing (not suitable for production). It routes URLs to Python functions. The dashboard interacts with the Hyperledger Fabric network via a client application or via a SDK to query temperature data and smart contract events. Figure 4 below shows a test run of the system analyzing the reading of three containers (CTN-001, CTN-002, CTN-003), the interface default.html, the python file app.py.



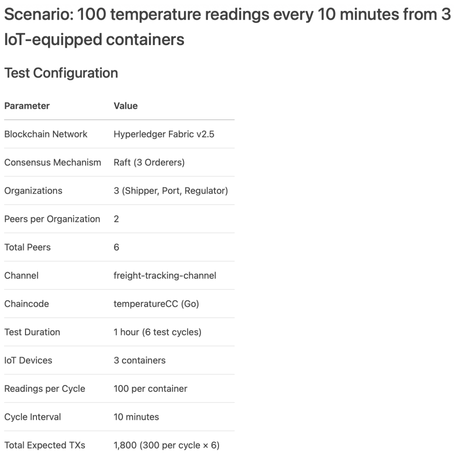
```
+-------------------------------------------------------------------------------+
| 🌡️BLOCKCHAIN TEMPERATURE MONITORING DASHBOARD|
+----------------------------+--------------------------------------------------+
| [📊Overview] [🚨Alerts] [📦Container Details] [📈Trends]                        |
+----------------------------+--------------------------------------------------+
|                                                                               |
| +-----------+ +-----------+ +-----------+ +-----------+                       |
| | Total     | | Active    | | Monitored | | Latest    |                       |
| | Readings  | | Alerts    | | Containers| | Reading   |                       |
| | 100       | | 12 🚨     | | 3         | | 18.2°C    |                       |
| +-----------+ +-----------+ +-----------+ +-----------+                       |
|                                                                               |
| +---------------------------+-------------------------------------------------|
| | Container Status          |                                                 |
| +-----------+-----------+-------------+-----------+-----------------------------|
| | Container | Current   | Status      | Last      |                           |
| |           | Temp      |             | Updated   |                           |
| +-----------+-----------+-------------+-----------+-----------------------------|
| | CTN-001   | 18.2°C    | ✅Normal | 10:30 AM |                               |
| | CTN-002   | -12.3°C   | 🚨Alert    | 10:25 AM |                              |
| | CTN-003   | 26.7°C    | 🚨Alert    | 10:20 AM |                              |
| +-------------+-----------+--------------+-------------+                        |
|                                                                                 |
+--------------------------------------------------------------------------------`+
```

Figure 6: Blockchain Query Output.

Figure 6 shows the seamless integration between different components (e.g., sensor to gateway, gateway to blockchain, blockchain to dashboard). 100% of readings were successfully recorded on the blockchain. 12 out of 100 readings triggered alerts (8 high-temperature, 4 low-temperature). Sensor status auto-updated in real-time (e.g., OK $\rightarrow$ ALERT: High Temperature). We also did a performance testing of our network to ensure the system can handle real-world demands, such as high transaction volumes, sensor data ingestion, and compliance checks. We used Hyperledger Caliper that benchmarks transaction per second (TPS), latency, and resource usage.

Figures 7, 8, and 9 show the output of the performance testing of the proposed system.

System maintained consistent throughput between 12.8-16.1 TPS, the av-

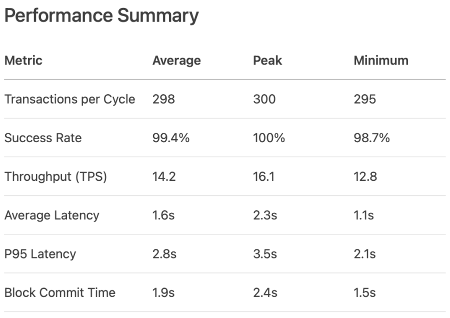


Scenario: 100 temperature readings every 10 minutes from 3 IoT-equipped containers

Test Configuration

Parameter	Value
Blockchain Network	Hyperledger Fabric v2.5
Consensus Mechanism	Raft (3 Orderers)
Organizations	3 (Shipper, Port, Regulator)
Peers per Organization	2
Total Peers	6
Channel	freight-tracking-channel
Chaincode	temperatureCC (Go)
Test Duration	1 hour (6 test cycles)
IoT Devices	3 containers
Readings per Cycle	100 per container
Cycle Interval	10 minutes
Total Expected TXs	1,800 (300 per cycle × 6)

Figure 7: Performance Testing-Scenario.

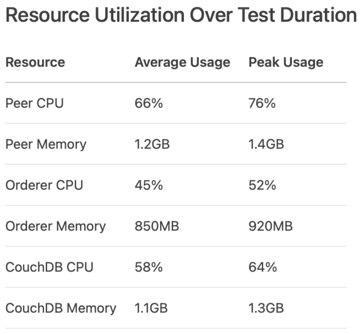


Performance Summary

Metric	Average	Peak	Minimum
Transactions per Cycle	298	300	295
Success Rate	99.4%	100%	98.7%
Throughput (TPS)	14.2	16.1	12.8
Average Latency	1.6s	2.3s	1.1s
P95 Latency	2.8s	3.5s	2.1s
Block Commit Time	1.9s	2.4s	1.5s

Figure 8: Performance Summary.

erage latency remained below 2s for 5/6 cycles, 5 total failures (0.6% of total transactions) and all failures were timeouts from container_3 during peak load. The Hyperledger Fabric network demonstrated a reliable performance (99.4% success rate) for periodic temperature monitoring, a consistent throughput (14.2 TPS average) suitable for the use case, and with generally low latency (1.6s average) for critical temperature alerts [8].


Resource Utilization Over Test Duration

Resource	Average Usage	Peak Usage
Peer CPU	66%	76%
Peer Memory	1.2GB	1.4GB
Orderer CPU	45%	52%
Orderer Memory	850MB	920MB
CouchDB CPU	58%	64%
CouchDB Memory	1.1GB	1.3GB

Figure 9: Resource Utilization.

6 Conclusion

The Hyperledger Fabric blockchain network successfully demonstrated real-time temperature monitoring with automated alerts, secure data logging, and stakeholder transparency. Through this study, we simulated a Hyperledger Fabric-based blockchain network that successfully processed synthetic temperature data from three IoT-equipped containers, each transmitting 100 temperature readings every ten minutes via a gateway to the blockchain. For long-term historical data, we need to consider integrating with a cold-storage database which stores data that is infrequently accessed. CheckAlerts should use indexes from CouchDB, which is is an open-source document-oriented NoSQL database, for production-scaledeployments [2]. Future work should focus on scaling the network and optimizing query performance for industrial deployments, and also it should explore AI-driven predictive analytics and cross-border blockchain interoperability to further enhance this system [5].

References

[1] Merlinda Andoni et al. “Blockchain Technology in the Energy Sector: A Systematic Review of Challenges and Opportunities”. In: *Renewable and Sustainable Energy Reviews* 100 (2019), pp. 143–174. DOI: 10.1016/j.rser.2018.10.014.

[2] Behzad Esmaeilian et al. “Blockchain for the Future of Sustainable Supply Chain Management in Industry 4.0”. In: *Resources, Conservation and Recycling* 163 (2020), p. 105064. DOI: 10.1016/j.resconrec.2020.105064.

[3] Tiago M. Fernández-Caramés and Paula Fraga-Lamas. "Towards Post-Quantum Blockchain for IoT". In: *Sensors* 20.7 (2020), p. 2091. DOI: 10.3390/s20072091.

[4] Haya R. Hasan et al. "A Blockchain-IoT Platform for the Smart Pallet Pooling Management". In: *Computers & Industrial Engineering* 163 (2022), p. 107828. DOI: 10.1016/j.cie.2021.107828.

[5] Mahtab Kouhizadeh, Sara Saberi, and Joseph Sarkis. "Blockchain for Supply Chain Decarbonization". In: *Computers & Industrial Engineering* 160 (2021), p. 107559. DOI: 10.1016/j.cie.2021.107559.

[6] Mahtab Kouhizadeh and Joseph Sarkis. "Blockchain and Sustainable Supply Chain Management in Developing Countries". In: *International Journal of Production Economics* 231 (2021), p. 107831. DOI: 10.1016/j.ijpe.2020.107831.

[7] Nir Kshetri. "Blockchain's Roles in Meeting Key Supply Chain Management Objectives". In: *International Journal of Information Management* 39 (2018), pp. 80–89. DOI: 10.1016/j.ijinfomgt.2017.12.006.

[8] Abderahman Rejeb et al. "The Internet of Things (IoT) in Maritime Logistics: A Systematic Literature Review". In: *Maritime Economics & Logistics* 24.1 (2022), pp. 92–120. DOI: 10.1057/s41278-021-00202-8.

[9] Sara Saberi et al. "Blockchain Technology and Its Relationships to Sustainable Supply Chain Management". In: *International Journal of Production Research* 57.7 (2019), pp. 2117–2135. DOI: 10.1080/00207543.2018.1533261.

[10] Jan Veuger. "Blockchain in Logistics and Supply Chain: A Lean Approach for Designing Real-World Use Cases". In: *IEEE Access* 8 (2020), pp. 209961–209971. DOI: 10.1109/ACCESS.2020.3035058.

[11] Yingli Wang et al. "Blockchain-Enabled Food Supply Chain Management: A Systematic Literature Review". In: *IEEE Access* 8 (2020), pp. 209027–209045. DOI: 10.1109/ACCESS.2020.3035636.

[12] Zaili Yang et al. "Maritime Shipping Digitalization: Blockchain-Based Technology Applications, Future Improvements, and Intention to Use". In: *Transportation Research Part E: Logistics and Transportation Review* 149 (2021), p. 102289. DOI: 10.1016/j.tre.2021.102289.

[13] Zibin Zheng et al. "Blockchain Challenges and Opportunities: A Survey". In: *International Journal of Web and Grid Services* 16.4 (2020), pp. 352–375. DOI: 10.1504/IJWGS.2020.110678.

Interview Preparation through Intentional Reflection: Integrating Mock Interview Practice with CS Coursework*

Karen Anewalt and Jennifer Polack
University of Mary Washington
Fredericksburg, VA 22405, USA
{anewalt, polack}@umw.edu

Abstract

Many computer science students feel unprepared for technical interviews, particularly when explaining their academic projects in professional settings. This paper presents a pedagogical approach that builds mock interview practice directly into CS courses, using existing programming assignments as the basis for developing professional communication skills. We implemented mock interview assignments across two course levels, CS1 and senior-level software engineering. Students practiced explaining their technical work through both written responses and video-recorded interviews using the Big Interview platform. This method connects coursework students already know well with authentic career preparation, giving them meaningful opportunities to practice professional communication. We collected data from 102 students over three semesters and found positive responses across multiple measures. More than half (52.9%) reported greater confidence when discussing their coding projects professionally, 55.9% saw improvements in their interview skills, and nearly two-thirds (61.8%) felt better prepared for future job interviews. These results demonstrate that embedding interview practice within existing technical courses can effectively develop students' professional communication abilities without requiring major curriculum changes. This approach offers CS programs a practical way to address

*Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

the gap between academic achievement and career readiness while maintaining focus on core technical content.

Keywords: Career Preparation, CS1, Software Engineering

1 Introduction

Computer science education faces a challenge in preparing students for the transition from academic coursework to professional careers. While students develop strong technical skills through programming assignments and coursework, many struggle to effectively communicate these experiences in professional settings, particularly during job interviews. The technical interview process in computer science requires students to demonstrate coding competency, articulate their problem-solving approaches, reflect on their development experiences, and connect their academic work to real-world applications. However, traditional CS curricula often provide limited opportunities for students to practice essential interview communication skills in authentic professional contexts.

Students may excel at completing technical programming assignments while still feeling unprepared to discuss their work confidently with potential employers, so a gap can exist between what students can do and what they can effectively communicate about their abilities. This gap not only affects individual student career outcomes but can also reflect poorly on CS programs' ability to prepare work-ready graduates.

Research in CS education has long recognized the importance of communication skills for career success. Still, many programs struggle to find effective ways to integrate professional communication practice within already content-dense technical courses. The result is that students often report feeling anxious and unprepared when facing their first technical interviews despite having completed rigorous academic programs.

Fortunately, recent advances in educational technology have created new opportunities to address this challenge through innovative pedagogical approaches. Mock interview platforms can be used to provide authentic practice experiences that can be seamlessly integrated into existing coursework. Rather than requiring separate career preparation courses, these technologies enable educators to connect existing academic work with professional communication skill development.

This paper presents a pedagogical innovation addressing the academic-to-professional communication gap by directly integrating structured mock interview practice into computer science coursework. Our approach leverages students' existing programming assignments as the foundation for developing professional communication skills and creates authentic practice opportunities within the familiar context of their technical work. By connecting course projects to interview-style reflection and communication practice, we aim to help students develop confidence in articulating their technical experiences while reinforcing their understanding of the underlying computer science concepts.

Our research addresses how mock interview practice using students' course projects impacts their confidence and preparedness for professional interviews. The interview practice assignments were implemented across two distinct course levels, a CS1 course and a senior-level software engineering course, allowing us to examine the effectiveness with students at different stages of their academic development. Through evaluation of this pedagogical approach with 102 students across three semesters, we provide evidence that integrating career preparation activities within existing technical coursework is effective, offering a model for CS educators seeking to enhance their students' professional readiness without requiring significant curricular restructuring.

2 Background Research

Higher education institutions face growing demands to better prepare students for the job market through enhanced career services, programs aligned with regional employment needs, and practical skills that help students secure internships and entry-level positions [4]. Interview skills are a known determinant of employment offers [3]. Research has shown that traditional interview situations are often viewed as stressful by students and that realistic and supportive preparatory activities can make students feel more confident [1]. Additionally, experts believe that career-readiness skills are best integrated into discipline-specific coursework [4].

While traditional interview preparation methods exist, recent technological advances have expanded the possibilities for scalable interview practice. Advances in artificial intelligence have created interest in using AI tools to allow students to practice interview skills [2, 6, 7]. These tools can increase students' confidence and preparedness for job interviews but may not include features to customize questions for specific courses across the CS curriculum and may have limited abilities to provide personalized feedback for students.

Kapoor & Gardner-Muccune [5] explored the efficacy of integrating mock interview activities in a data structures course at an R1 university. In their course, TAs were trained to act as interviewers. The students in the course were required to participate in pairs in a mock interview where they were asked data structures-related questions that are common in coding interviews. In this course, 91% of the students reported a positive experience. Most students who viewed the experience positively reported feeling more confident and prepared for interviews. They also stated that the assignment increased their appreciation for how their coursework transferred to entry-level job skills. However, some students noted that the exercises were stressful and time-consuming and suggested improvements, including providing explicit question lists and reducing time requirements. Building on this foundation, our research extends mock interview integration to multiple course levels while focusing on developing professional communication competencies rather than solely technical question practice. We also address the time and stress concerns by connecting the interview activities to existing assignments and providing opportunities to prepare and refine responses.

3 Mock Interview Assignment Approach

The foundation of our pedagogical innovation rests on the principle that students can most effectively develop professional communication skills when practicing with familiar, meaningful content. Rather than asking students to discuss hypothetical projects or generic programming scenarios during mock interviews, our approach leverages the programming assignments they have already completed in their coursework. This design choice serves multiple pedagogical purposes: it reduces cognitive load by eliminating the need to learn new technical content, provides authentic material that students genuinely understand, and creates natural opportunities for reflection on their own learning and problem-solving processes.

The innovation addresses a fundamental disconnect in traditional CS education, where students complete assignments for grades and course credit but rarely engage in the type of reflective communication employers value. By repositioning completed assignments as professional portfolio pieces worthy of discussion and explanation, we help students recognize the career relevance of their academic work while developing essential communication competencies.

Our primary learning objective focuses on helping students articulate their technical work in ways that highlight skills and experiences valuable to potential employers. For example, students should be able to explain technical concepts to different audiences, connect academic projects to real-world applications, reflect on problem-solving approaches and learning experiences, and discuss technical work confidently in professional settings. Students were explicitly told they would practice discussions similar to those in internship or entry-level job interviews, making the connection between classroom learning and career preparation transparent and intentional.

A critical component of our assignment design involves structured instructor feedback focused on communication effectiveness and professional presentation. Unlike automated feedback systems, instructor evaluation addresses clarity of explanation, appropriate use of technical terminology, connection between academic work and professional contexts, and overall confidence in presentation. This instructor feedback serves as a crucial bridge between academic and professional communication standards, emphasizing how students can better highlight transferable skills and present their work in professionally relevant ways.

Our assignment approach integrates seamlessly into existing CS curricula without requiring additional course content or separate career preparation modules. Using assignments students are already completing adds professional communication practice without increasing workload or requiring curriculum restructuring. Instructors can adapt specific interview questions to align with their particular assignments, from basic programming exercises in CS1 to complex software engineering projects, while maintaining the core principle of connecting academic work to professional communication practice, making the approach broadly applicable across CS curricula.

4 Mock Interview Assignment Implementation

To examine the effectiveness of this approach with students at different stages of their academic development, we integrated mock interview assignments into a CS1 course and a senior-level software engineering course. The CS1 course included five short projects evenly spaced throughout the semester, with each project covering a typical course topic like loops, string processing, or basic object-oriented design. The software engineering course included a semester-long project creating custom software for an external client; this project included different phases. including gathering requirements, planning project sprints, implementing new features into an existing code base, and delivering the completed software to the client.

While the technical projects assigned in the two courses had different complexity levels, we maintained consistent pedagogical goals and a similar structure to the mock interview integration across the two courses. While assessment methods differed by course level (completion-based for CS1, letter grades for the senior course), the learning objective remained constant across both courses: developing the ability to communicate technical work in ways that highlight skills and experiences valuable to employers. Students completed their regular programming assignments and then received interview questions related to those projects. Students recorded video responses using Big Interview, an online mock interview platform. This tool allowed students to practice the spontaneous communication skills required in interview settings. Within the tool, students are shown a video clip of an interviewer asking them a question. Each student films and uploads a video response to the question. In both courses, students were allowed to record responses multiple times if desired so that the final submissions represented their best effort at professional communication. Instructors in both courses provided feedback focused on communication effectiveness rather than technical correctness since the technical work had already been evaluated through the original assignment grading.

4.1 CS1 Course Assignment Differences

While most of the mock interview integration was consistent across the two courses, there were a few differences, particularly in the last semester of data collection in CS1. In earlier iterations of the assignment, the instructor noted that CS1 students' responses to the mock interview questions lacked context and details that would allow employers to understand the work done by the students. In the most recent semester of the course, students were provided instruction in the STAR method [8] of responding to interview questions before responding to mock interview questions. The STAR method coaches students to describe the Situation, Task, Action, and Result in their response to interview questions. Students in the most recent semester were also given more frequent opportunities for practice and were provided with opportunities to answer the questions in different modalities, first through written responses to interview questions and later as video responses in Big Interview. CS1 students received instructor feedback about the clarity and effectiveness of their responses while receiving an assignment score of complete or incomplete.

Interview questions for CS1 students focused on articulating their project approach: "Describe one error you encountered during your most recent development process. How did you locate and correct the error?"; "Tell me about a coding project that you finished recently and why you're confident that the final solution is correct."; and "What is the most challenging project that you've worked on?"

4.2 Senior Course Assignment Differences

In the senior-level course, students were required to respond to mock interview questions exclusively in video format through Big Interview, reflecting their advanced status as they approached graduation. This shift in assignment implementation emphasized more sophisticated professional communication standards. Following their learning on gathering requirements, students applied this knowledge to their current project with a nonprofit, where they collaborated in teams on a web database project. The interview questions they addressed were designed to mimic real software engineering interview questions, including prompts such as: "How would you gather customer requirements for a new project?"; "Tell me about a time when you had to explain a complex technical idea to someone with little technical background."; and "Describe a time when you were assigned a project without clear direction. What did you do?" The assessment in this course involved letter-grade evaluations with heightened expectations for the quality of professional presentation, technical depth, and articulation of software engineering principles.

5 Methodology

This study assesses the pedagogical effectiveness of integrating mock interview practice within existing computer science coursework. The evaluation was conducted across three consecutive semesters at the University of Mary Washington, allowing us to refine the implementation and gather data on student experiences and perceived learning outcomes.

Over the three-semester implementation period, 102 students participated in the study, 50 from the CS1 course and 52 from the senior software engineering course. At the end of the course, a survey was used to collect data on student experiences. The courses enrolled 186 total students, so the 102 survey responses represent a 54.8% response rate. The response rate was about equal between the two courses: 56.6% response rate for the senior level course and 53.2% for the CS1 course. The survey was designed to capture multiple dimensions of the pedagogical experience, including perceived skill development, confidence changes, user experience satisfaction, and likelihood of continued engagement with professional preparation activities.

The survey included seven required Likert-scale response questions focused on three key areas of pedagogical impact: confidence in communicating about software projects, improvement in interview skills, and preparedness for future professional interviews. The survey also included three optional open-ended questions that solicited student suggestions for assignment improvements, additional feedback on their experience, and ideas for implementing similar approaches in other computer science courses. The survey questions were:

1. Which course did you complete a required Big Interview assignment for?
2. Big Interview helped me to improve my confidence in communicating about a software project.
3. Big Interview helped me to improve my interview skills.
4. After using Big Interview, I feel more prepared for interviews I might have in the future.
5. How would you rate your overall experience using the Big Interview platform for practicing interview-like questions and scenarios?

6. Now that you know that Big Interview exists, do you plan to return to complete practice activities outside of class requirements?

7. How user-friendly did you find the Big Interview platform?

8. Do you have any improvements to suggest related to this Big Interview assignment?

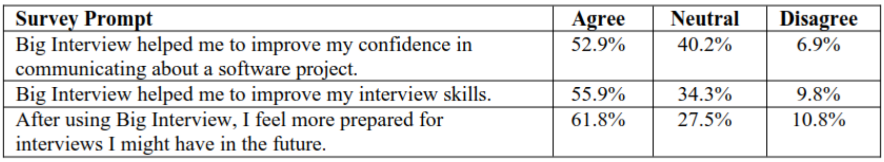
9. Do you have any additional feedback on using Big Interview for this assignment?

10. Are there additional computer science related assignments where you can imagine using Big Interview?

6 Results

The 102 survey responses provide insights into how the mock interview assignments affected their confidence, skills, and career preparation. Responses came equally from both courses, with 49% from CS1 and 51% from the senior software engineering course.

More than half of the students (52.9%) felt the assignments helped them communicate more confidently about their programming projects, while only 6.9% disagreed and 40.2% were neutral (Figure 1). The large majority who saw improvements suggests that practicing with familiar coursework does help students feel more comfortable explaining their technical work.



Survey Prompt	Agree	Neutral	Disagree
Big Interview helped me to improve my confidence in communicating about a software project.	52.9%	40.2%	6.9%
Big Interview helped me to improve my interview skills.	55.9%	34.3%	9.8%
After using Big Interview, I feel more prepared for interviews I might have in the future.	61.8%	27.5%	10.8%

Figure 1: Summary of responses related to confidence and interview preparation.

Students had similar positive reactions when asked about their interview abilities. Most (55.9%) reported that their skills improved, with only 9.8% disagreeing and about one-third remaining neutral (Figure 1). This pattern indicates that using course projects as interview content helps students develop professional communication abilities.

The strongest positive response came when students considered their readiness for actual job interviews. Nearly two-thirds (61.8%) felt better prepared after completing the assignments, compared to just 10.8% who disagreed (Figure 1). This response shows that students understood the practical value of the mock interview assignment for their career preparation.

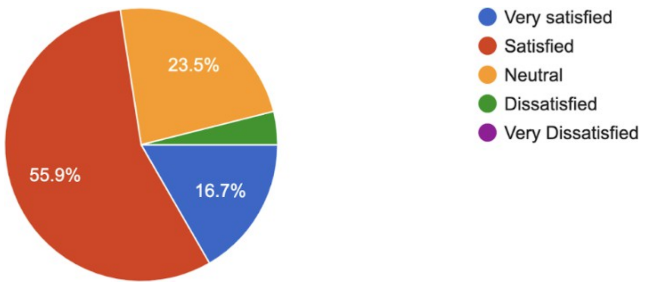


Figure 2: How would you rate your overall experience using the Big Interview platform for practicing interview-like questions and scenarios?

Students were generally satisfied with the overall experience, with 72.6% rating it positively and fewer than 4% expressing dissatisfaction (Figure 2). The Big Interview platform worked well for most students; 96.1% found it user-friendly, with 61.4% calling it very user-friendly (Figure 3). This high usability suggests that the technology did not create barriers to learning.

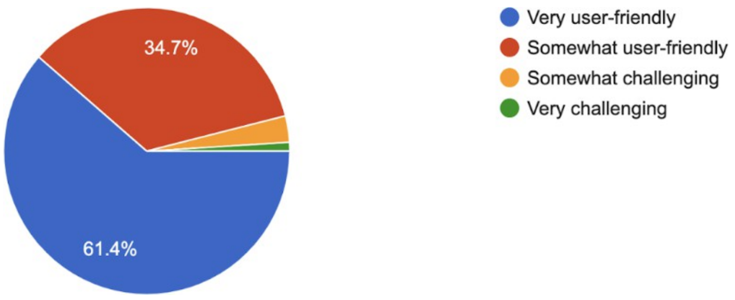


Figure 3: How user-friendly did you find the Big Interview platform?

Student feedback included a mixture of positive experiences and constructive suggestions for improvement. Students reported that they appreciated the flexibility of recording multiple attempts for answers, reducing the pressure often felt in live interviews. However, students consistently asked for better upfront guidance on interview techniques. We addressed this in the latest CS1 semester by teaching the STAR method, but senior students could benefit from similar instruction.

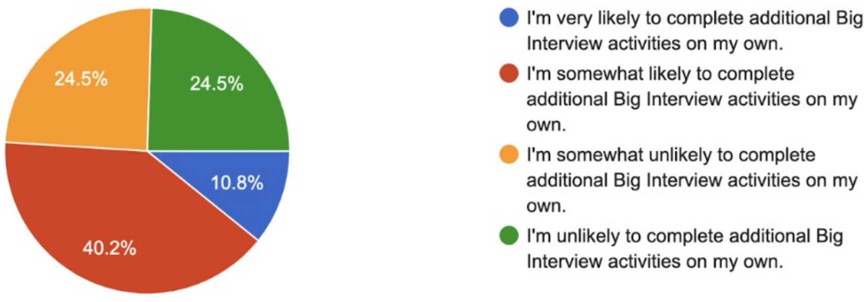


Figure 4: Now that you know that Big Interview exists, do you plan to return to complete practice activities outside of class requirements?

Students also wanted the interview questions to feel more relevant to computer science careers. Some found the existing questions too metaphorical or unrelated and suggested adding technical coding problems, randomizing questions to create spontaneity, and including company-specific questions for places like Google or Amazon. They also requested video examples showing both good and poor interview responses.

Looking beyond this assignment, students saw potential applications in data structures courses, project reflection exercises, and presentation practice throughout the curriculum. Despite their suggestions for improvement, students recognized the value of the experience, with one noting it was "useful for practicing communication, especially when explaining technical projects."

7 Discussion

Integrating mock interview practice connected to course projects proved especially effective because students were already familiar with the material and could focus entirely on presentation skills. Students clearly valued instructor feedback, frequently asking for more guidance and concrete examples of effective responses. This human element made a significant difference. While peer interviews or automated systems can provide practice opportunities, instructors offered the kind of nuanced coaching that helped students understand professional communication expectations. Students needed someone who could bridge academic and workplace communication styles, something that requires judgment and experience. The positive outcomes across multiple measures suggest that connecting familiar course content to professional communication practice creates meaningful learning experiences for computer science students. When students can practice with material they genuinely understand and care about, they engage more meaningfully with the communication challenge itself.

7.1 Student Experience Insights

The mixed response regarding students' likelihood for continued independent practice (51% likely vs. 49% unlikely) reveals important insights about student motivation and the nature of skill development. While students recognized the assignment's value within the course context, the reluctance to pursue additional practice independently suggests that structured, curriculum-integrated approaches may be more effective than expecting students to seek career preparation activities independently. This finding supports experts' recommendations that career readiness skills be embedded within discipline-specific coursework rather than relegated to separate career services programming.

7.2 Comparison with Prior Work

Our results show more modest positive responses than Kapoor & Gardner-McCune's [7] 91.9% positive experience rate, reflecting important methodological and contextual differences. Their study focused primarily on technical problem-solving practice with data structures questions, while our approach emphasized professional communication skills development across different course levels. The difference in student perception may reflect the inherent challenge of communication skill development compared to technical practice. While students can more readily assess their ability to solve coding problems, evaluating communication effectiveness requires developing metacognitive awareness that may take longer to achieve. Our substantial neutral responses (23.5-40.2% across measures) suggest that some students may need additional time or practice to recognize their communication skill development. Additionally, the Kapoor & Gardner-McCune assessment relied on qualitative feedback about the overall experience, while our survey measured different skill development areas.

7.3 Implementation Considerations

Student feedback provides valuable guidance for educators considering similar implementations. The consistent request for more computer science-specific content suggests that while using familiar course projects is pedagogically sound, students also value exposure to industry-standard technical interview questions. Future implementations might benefit from a hybrid approach that combines course-specific reflection with broader technical communication practice.

Students consistently asked for better preparation before tackling the interview questions, which tells us they need direct teaching about professional communication strategies. When we introduced the STAR method to CS1 students in later semesters, we saw noticeably better responses, confirming that students benefit from learning specific frameworks for structuring their answers. Students also repeatedly requested video demonstrations of strong interview responses, showing they learn well from concrete examples of effective communication.

We successfully used this approach in both introductory and advanced courses, which shows its flexibility. However, earned letter grades. The different grading methods suggest that expectations should align with proximity to career entry.

7.4 Limitations and Scope

Several limitations influence the generalizability of these findings. The study was conducted at a single institution with a specific student population, and results may vary across different student demographics. The reliance on self-reported perceptions of skill development, while pedagogically meaningful, does not provide objective measures of communication improvement or actual interview performance. The three-semester time frame, while allowing for implementation refinement, limits our understanding of longer-term impacts on student career outcomes.

7.5 Implications for CS Education Practice

The findings suggest that CS programs can meaningfully address the academic-to-professional communication gap without requiring significant curricular restructuring. Rather than adding new courses or requirements, departments can build career preparation into classes students already take. This approach helps programs respond to pressure about graduate job readiness while keeping their focus on teaching core technical content.

The strong student response to instructor feedback reveals something important about learning professional communication skills. Even as educational technology becomes more sophisticated, students still benefit enormously from human coaching. AI-powered practice tools certainly have their place and can reach many students efficiently. However, instructors bring something different to the table; they can provide personalized, context-sensitive guidance that helps students really understand how to present themselves professionally.

The success across different course levels suggests that professional communication development should be viewed as a progressive skill that can be scaffolded throughout the undergraduate experience rather than confined to capstone courses or career services programming.

8 Conclusion

Our work shows that computer science programs can effectively bridge the gap between academic learning and professional communication skills by building interview practice directly into existing courses. Using existing programming assignments as the basis for mock interviews worked well because students could focus on explaining work that they already understood rather than learning new material. The positive outcomes across 102 students and three semesters, with 61.8% feeling more prepared for future interviews, provide evidence that career preparation can be meaningfully embedded within technical coursework without requiring additional curriculum content or separate career-focused courses.

The approach offers particular value for CS programs facing increasing pressure to demonstrate graduate employability and career readiness. Rather than designing new courses, relying solely on external career services, or expecting students to pursue interview preparation independently, our model integrates professional skill development into the existing curriculum. The instructors' feedback proved particularly valuable; students clearly benefited from having someone who could help them understand professional communication expectations in a personal manner.

While student feedback revealed opportunities for enhancement, requesting more computer science-specific interview questions and better initial guidance on interview techniques, the positive response and learning outcomes indicate that this approach offers a workable framework for other CS educators. The successful implementation across different course levels, from introductory CS1 to senior software engineering, shows flexibility across different student populations and potential for broader adoption. As computer science education continues adapting to serve both academic and career preparation goals, integrating communication practice with technical coursework offers real promise. This approach helps develop graduates who can not only solve technical problems but also explain their work effectively, a combination that serves them well in their careers and reflects positively on their academic programs.

References

[1] Mahnaz Behroozi, Chris Parnin, and Titus Barik. "Hiring is Broken: What Do Developers Say About Technical Interviews?" In: *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. 2019, pp. 1–9. URL: `https://web.eecs.umich.edu/~weimerw/2022-481F/readings/hiring-is-broken.pdf`.

[2] Taufiq Daryanto et al. "Conversate: Supporting Reflective Learning in Interview Practice Through Interactive Simulation and Dialogic Feedback". In: *Proc. ACM Hum.-Comput. Interact.* 9.1 (Jan. 2025), p. 32. DOI: 10.1145/3701188.

[3] Caren Goldberg and Ann Perry. "Who Gets Hired: Interviewing Skills Are a Prehire Variable". In: *Journal of Career Planning & Employment* 58.2 (1998), pp. 47–50.

[4] Matthew Hora. *Career-Readiness Initiatives are Missing the Mark.* Inside Higher Ed. January 10, 2023. Jan. 2023. URL: https://www.insidehighered.com/views/2023/01/10/career-readiness-initiatives-are-missing-mark-opinion.

[5] Amanpreet Kapoor and Christina Gardner-Mccune. "Introducing a Technical Interview Preparation Activity in a Data Structures and Algorithms Course". In: *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 2 (ITiCSE '21).* New York, NY, USA: Association for Computing Machinery, 2021, pp. 633–634. DOI: 10.1145/3456565.3460033.

[6] Mingzhe Li et al. "EZInterviewer: To Improve Job Interview Performance with Mock Interview Generator". In: *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM '23).* New York, NY, USA: Association for Computing Machinery, 2023, pp. 1102–1110. DOI: 10.1145/3539597.3570476.

[7] Zhe Liu. "Interview AI-ssistant: Designing for Real-Time Human-AI Collaboration in Interview Preparation and Execution". In: *Companion Proceedings of the 30th International Conference on Intelligent User Interfaces (IUI '25 Companion).* New York, NY, USA: Association for Computing Machinery, 2025, pp. 225–228. DOI: 10.1145/3708557.3716148.

[8] Robin Ryan. *Ace Your Next Interview Using the STAR Method.* Forbes. May 25, 2023. May 2023. URL: https://www.forbes.com/sites/robinryan/2023/05/25/ace-your-next-interview-using-the-star-method/.

Towards AI-Driven Academic Advising: Lessons from a Domain-Specific Implementation of a RAG-Based Chatbot[*]

Benito Mendoza, Afroza Aktar, Shajib Ahsan, Kim Yang, Farrukh Zia, Lili Ma, Yu Wang, and Ohbong Kwon.
Department of Computer Engineering Technology
New York City College of Technology
Brooklyn, NY 11201
{bmendoza, afroza.aktar16, shajib.ahsan04}@citytech.cuny.edu
{yakim, fzia, lma, ywang, okwon}@citytech.cuny.edu

Abstract

Academic advising is essential for student success, but remains resource-intensive and difficult to scale. This paper presents a domain-specific implementation of an AI-powered advising chatbot based on a Retrieval-Augmented Generation (RAG) architecture. The system combines vector-based document retrieval with a large language model (LLM) to deliver context-aware responses to course planning and policy-related queries. We evaluated the chatbot using 22 representative advising questions, rated by expert reviewers across four dimensions: Accuracy, Completeness, Clarity, and Policy Adherence. The system demonstrated strong performance in Clarity and Policy Adherence but struggled with complex scenarios that required prerequisite logic and multi-step reasoning. Reviewer feedback revealed common failure modes, including missing course dependencies and policy misalignment. These results highlight the potential and limitations of lightweight RAG-based advising tools, pointing to future improvements, including multi-agent designs, structured knowledge integration, and policy-aware reasoning.

[*]Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

1 Introduction

In higher education, effective academic advising plays a crucial role in supporting student success by guiding learners through complex degree pathways, assisting with course selection, and ensuring timely progress toward graduation [2, 6]. Beyond logistical support, advisors also influence how students define and pursue academic success, particularly when they acknowledge and validate diverse goals and experiences [27]. However, the advising process is often labor-intensive and time-constrained, especially for students with non-traditional academic trajectories, such as transfer students or those returning after a long break. One of the most time-consuming tasks is transcript analysis, where advisors must evaluate prior coursework for equivalency, prerequisite fulfillment, and curriculum alignment [3]. While degree audit systems offer some automation, their limited contextual understanding restricts their ability to provide nuanced, personalized recommendations, leaving advisors to manually interpret and validate each case.

Recent advancements in Artificial Intelligence (AI), particularly the emergence of Large Language Models (LLMs) like ChatGPT, present new opportunities to automate and enhance academic advising. One promising method is Retrieval-Augmented Generation (RAG), which combines vector-based document retrieval with generative language models to generate contextually grounded responses [8]. This method has the potential for creating chatbots that can provide personalized, context-aware guidance in real-time, offering students greater access to timely information while reducing the administrative workload on human advisors [1, 14, 24].

This paper introduces an AI-driven advising chatbot that utilizes RAG technology to analyze academic transcripts, assess curriculum requirements, and generate course recommendations based on institutional policy, which students and advisors can use. Our approach employs a 'naïve' RAG setup, one that avoids advanced reranking or answer filtering mechanisms, to assess the baseline effectiveness of this technology when integrated with institutional knowledge bases.

We present lessons learned from the system's design and deployment in a real-world academic setting, along with a rubric-based evaluation of its responses. The results offer insight into the feasibility, limitations, and potential enhancements for AI-driven advising systems tailored to specific institutional needs.

2 Background and Related Work

2.1 RAG in AI Chatbots

Traditional chatbots developed before the emergence of generative AI relied on rule-based logic or simple retrieval methods. These systems responded to predefined inputs using static scripts or keyword matching, which limited their flexibility and rendered them ineffective for nuanced, context-dependent queries. While large language models (LLMs) have introduced the ability to generate fluent, human-like responses, standalone generative models are prone to hallucinations, confidently producing information that is incorrect or fabricated [13].

Retrieval-Augmented Generation (RAG) addresses this limitation by combining the generative capabilities of LLMs with external retrieval mechanisms [13]. In this architecture, the system first retrieves relevant documents from a knowledge base, then uses this retrieved content to ground and inform the generated response. This hybrid model enhances factual accuracy, contextual relevance, and user trust, making it especially effective in knowledge-intensive applications such as education, healthcare, and customer support [8, 22].

The RAG workflow begins with an **indexing phase**. The system retrieves structured and unstructured data from various sources, including institutional databases, websites, and other digital documents. These sources are converted into plain text and segmented into smaller, coherent units, known as chunks. Chunking enhances retrieval efficiency and helps preserve contextual integrity. The chunks are then tokenized, a process that breaks the text into subword units and assigns unique identifiers to them based on a predefined vocabulary. This representation allows the language model to process the text effectively and capture syntactic and semantic patterns during generation [13].

Next, embedding models convert these tokenized chunks into high-dimensional vectors that capture their semantic content. These vectors are stored in a vector database, allowing for rapid similarity-based searches. When a user submits a query, the **retrieval phase**, the system embeds the query in the same vector space and retrieves the most semantically relevant chunks using similarity metrics such as cosine similarity [12].

In the **generation phase**, the retrieved documents serve as a grounding context for the LLM. The model synthesizes a response that integrates its pretrained knowledge with the retrieved content, generating output that is both fluent and factually supported [8, 19, 22, 25, 29]. This dual-stage approach enables RAG-based chatbots to provide more informed, context-aware, and precise responses than either retrieval-based or generative models alone.

Despite its strengths, RAG is not without challenges. The effectiveness of RAG is closely tied to the completeness, relevance, and timeliness of the underlying document corpus; outdated or incomplete sources can lead to erroneous or insufficient responses [25]. Additionally, limitations in the indexing process, such as suboptimal chunking strategies, can further impair retrieval quality, reducing the accuracy of generated answers. While RAG frameworks substantially reduce hallucinations compared to standalone generative models, they do not eliminate them entirely, especially when the retrieved context is sparse or ambiguous [13, 19, 29]. The architectural integration of retrieval and generation components also introduces increased computational overhead and latency, necessitating careful system optimization to balance performance, accuracy, and responsiveness [14, 26].

While advanced RAG systems may incorporate reranking mechanisms, hybrid retrieval strategies, or iterative feedback loops to refine their outputs, the present study adopts a naïve RAG approach. This simplified implementation uses a single-step vector search and a basic context injection method, without post-retrieval filtering or reranking, to evaluate the baseline feasibility of domain-specific advising support in a real-world academic setting.

The following section examines how RAG and LLM technologies have been applied in academic advising, highlighting emerging patterns, system architectures, and practical challenges in student-facing educational services.

2.2 AI in Academic Advising

AI applications in academic advising have evolved from rigid, rule-based systems to dynamic, context-aware models capable of handling complex student inquiries. Early systems were typically designed to match predefined inputs with scripted outputs. While effective for answering frequently asked questions, these models lacked the capacity to adapt to nuanced academic scenarios such as curriculum transitions, transcript evaluations, and policy interpretation [2, 6].

The rise of LLMs has expanded the scope of advising automation. Models like GPT-3.5 and GPT-4 enable fluid, conversational interactions and can interpret open-ended questions. However, without grounding in external sources, even the most advanced LLMs are prone to hallucination, confidently generating incorrect or misleading information [13, 19]. This is particularly problematic in academic contexts, where misinformation can negatively impact student decisions.

RAG has emerged as a promising architecture for mitigating hallucinations while enhancing accuracy. RAG models retrieve relevant information from curated sources, which is then used to inform the LLM's response. This design allows for contextual alignment with institutional documents and policies, making RAG particularly suitable for academic advising [4, 11, 13].

Several recent projects have explored the integration of LLMs and RAG into advising tools. For example, the URAG architecture employs a two-layer retrieval mechanism, FAQ-based followed by vector search, yielding superior accuracy and engagement compared to general-purpose LLMs like GPT-3.5 and GPT-4o [17]. Another implementation, the AI advisor presented in [4], a layered RAG framework with multi-step context enrichment and safety filtering, evaluated using metrics such as context relevance and answer accuracy. Other innovations include Gaita [28], which creates personalized learning paths from open courseware, and AskYourTranscript [1], a system that parses transcripts and policies to provide grounded, document-based advising. One project that is worth mentioning is EDUC8EU [7], which combines fuzzy logic, ontology-based modeling, and a personality matching tool based on the RIASEC framework for career counseling and vocational psychology [16] to guide students across academic and career pathways. While effective in enhancing user experience, its precision in complex recommendations remains limited.

Despite these advancements, many current systems rely on complex infrastructures, centralized hosting, or limited access to internal policy documents. In contrast, this study focuses on a simple, domain-specific RAG-based chatbot built with lightweight tools and grounded entirely in publicly accessible institutional documents. Our approach emphasizes transparency, adaptability, and alignment with the real needs of a specific college environment.

Compared to prior work, our study takes a different focus. Iatrellis et al. [11] presented EDUC8EU, which integrates rule-based reasoning with LLMs for sustainable advising in European universities. Maryamah et al. [14] developed a RAG-based advising chatbot that leveraged GPT-3.5 with Ada embeddings and evaluated outputs using BLEU and ROUGE. Nguyen and Quan [17] proposed the Unified RAG (URAG) framework, which combines FAQ-based retrieval with RAG and adds validation layers to improve precision in admission advising. Similarly, Wangwiwattana and Jantarick [26] introduced an LLM-driven FAQ extraction approach for student services in Thai universities, evaluated with context relevance, answer relevance, and groundedness. Our work differs by presenting a baseline naïve RAG implementation tailored to a U.S. community college context, emphasizing transparency, ease of deployment with lightweight tools, and practical utility in resource-constrained advising settings.

3 Methodology

Our Academic Advisement Bot is designed to provide customized guidance in course planning to both students and advisors at our program and college. The chatbot is intended to assist both students and academic advisors with course planning by leveraging a curated knowledge base of institutional documents.

These include course catalogs, curriculum guides, transcript policies, course equivalencies, and other materials related to academic advising.

3.1 System Design

Our Naïve RAG-based chatbot was implemented using the LangChain framework [15]. Our system operates through three phases, as illustrated in Figure 1.

- **Indexing Phase**: Institutional documents (e.g., curriculum guides, course catalogs, advising policies) are preprocessed, embedded, and stored in a FAISS (Facebook AI Similarity Search) vector database [5].
- **Retrieval Phase**: The user query is embedded using Google's Generative AI Embeddings [9] and compared to the indexed documents in the FAISS vector database. The system retrieves the most semantically relevant results based on the approximate nearest neighbor search, typically using inner product or Euclidean distance.
- **Generation Phase**: The system prompt is passed to an LLM (GPT-4o) [10], which synthesizes responses based on both institutional knowledge and its pre-trained capabilities.

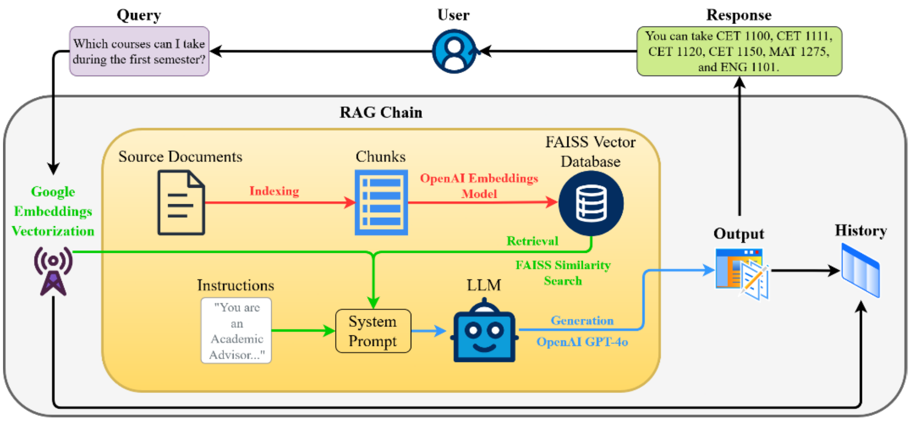


Figure 1: The system architecture for our Academic Advisement Bot.

This architecture is designed to generate responses that are both accurate and relevant to the context. When a user submits a query, the system generates a vector of embeddings that are used to search the vector database (knowledge base) for pertinent academic advising information related to the query. This information, referred to as the context, is integrated with the user's query and the instructions to form a system prompt. The instructions direct the LLM to function as an advisor, using the expanded prompt and leveraging the retrieved context to provide a well-rounded and relevant response, while reducing the likelihood of hallucinations. Ultimately, the chatbot presents this AI-generated response to the user, ensuring that students receive dependable and tailored academic guidance.

While implementing a robust architecture for our chatbot was important, it was also equally important to provide a simplified user experience. For this reason, we created a generic chatbot interface that displayed user queries as sent messages and chatbot responses as received messages . The History component of our system served two purposes: i) it allowed us to display previously sent and received messages within the interface as though a conversation were occurring between the user and the bot, and ii) to provide another source of contextual information to the LLM, further reducing hallucinations.

3.2 Experimental Setup

We constructed a test set of 22 advising queries, drawn from real student inquiries, and categorized them into three levels of complexity—**Easy (7 questions), Medium (11 questions), and Hard (4 questions).** Query difficulty was determined both by the inherent reasoning steps required and by input from academic advisors.

- **Easy**: Basic queries requiring course recommendations based on semester availability and prerequisites (e.g., "Which courses can I take in the first semester?").
- **Medium**: Queries requiring an understanding of prerequisite dependencies (e.g., "I have completed COURSE 101, COURSE 102, and ENG 110. What courses can I take next?").
- **Hard**: Complex course planning scenarios involving forward-looking degree planning (e.g., "If I want to graduate in six or seven semesters instead of eight, how should I plan my courses?").

This classification is context-dependent: a query considered "hard" for a new student may be straightforward for a senior close to graduation. To support reproducibility, the full set of 22 advising queries and evaluation scripts is available in our project repository[1].

[1]https://huggingface.co/spaces/sahsan/cet_advisement_bot

3.3 Computational Similarity

To benchmark the chatbot's performance, its responses were compared against curated "expected" answers authored by academic advisors. Two automated similarity metrics were used:

- **Cosine Similarity** [23], which measures lexical overlap between the chatbot's response and the expected answer.
- **SBERT (Sentence-BERT) Similarit**y [20], which evaluates semantic closeness between responses using dense embedding representations.

The **Cosine Similarity** metric calculates the angle between vectorized representations of the expected and generated answers, yielding a value between 0 (completely dissimilar) and 1 (identical). Texts were tokenized and transformed into numerical vectors using the CountVectorizer method [18]. Cosine similarity scores were then computed between each query pair. In contrast, **SBERT Similarity** evaluates responses at the semantic level. Both the expected and generated answers were encoded using the Sentence-BERT function from the all-MiniLM-L6-v2 model [21], which preserves contextual meaning. These metrics serve as an initial benchmark for linguistic and semantic alignment but do not account for contextual accuracy or policy correctness, which are assessed through human evaluation.

3.4 Rubric-Based Human Ratings

We also conducted a human-rated evaluation of each response generated by the chatbot independently using a structured rubric. The evaluation rubric consisted of four criteria: **Accuracy**, **Completeness**, **Clarity**, and **Policy Adherence**. Each response was rated from 0 (Unacceptable) to 4 (Excellent). A score of 3.0 or higher was considered passing. Definitions of the dimensions were as follows:

- **Accuracy**: whether the information is factually correct (e.g., identifying the correct prerequisite course).
- **Completeness**: inclusion of all relevant elements required for a satisfactory response.
- **Clarity**: readability, fluency, and understandability of the response.
- **Policy Adherence**: alignment with institutional regulations, policies, and program rules (e.g., restrictions on credit load or requirements specific to transfer).

Notably, Accuracy and Policy Adherence are related but distinct. A response may be factually accurate but still misaligned with institutional policy. For example, the system may correctly state that "students can take four courses in the first semester," which is accurate in general, but if a student has additional obligations, such as part-time status (in other universities, they might have other commitments, such as athletics or band), institutional policy might recommend a lighter course load. In such cases, the answer is accurate but not fully policy-adherent.

Each response was scored independently by reviewers on a 0–4 scale across four dimensions: accuracy, completeness, clarity, and policy adherence. A score of 3.0 or higher was considered passing. This rubric-based approach allowed for structured evaluation of both factual and contextual quality in responses.

4 Results and Analysis

4.1 Computational Similarity-Based Evaluation

Table 1 summarizes the computed Cosine and SBERT Similarities between the chatbot's responses and expert-crafted reference answers for each of the 22 advising queries. These metrics allowed us to quantify both lexical overlap and semantic alignment across different levels of query complexity.

Table 1: Similarity Metrics by Difficulty Level

Difficulty Level	**Cosine Similarity**		**SBERT Similarity**	
	Avg.	**StDev.**	**Avg.**	**StDev.**
Easy	0.49	0.20	0.68	0.20
Medium	0.47	0.18	0.72	0.22
Hard	0.57	0.09	0.63	0.05

Medium-level queries achieved the **highest average SBERT similarity (0.72)**, suggesting that the chatbot captured the intended semantic content well, even if surface wording differed from the expected answers. **Hard-level queries showed** the **highest average Cosine similarity (0.569)**, likely due to their structured nature, such as explicit course plans, which increased word overlap with the reference answers. **Easy-level queries**, surprisingly, **had lower similarity scores**, particularly in cosine terms. This is likely because the chatbot often elaborated beyond the concise expected answers, introducing phrasing differences that reduced overlap despite being factually aligned.

These results reveal that SBERT similarity better captures semantic similarity, particularly in cases where helpful responses differ in wording from the reference, than Cosine similarity in capturing the quality of chatbot responses for advising contexts, especially where variation in phrasing is common. The divergence in similarity scores for Easy queries also highlights a limitation of surface-level metrics: even correct and helpful answers may score lower if they include added detail or use different language from the reference.

These findings demonstrate that while automated metrics are helpful for estimating alignment, they cannot fully replace human judgment, especially for tasks such as academic advising, where clarity, policy nuance, and student context are crucial.

4.2 Human-Rated Evaluation

Table 2: Average Evaluation Scores Grouped by Difficulty Level

Difficulty Level	**Accuracy**		**Completeness**		**Clarity**		**Policy Adherence**	
	Avg	**SD**	**Avg**	**SD**	**Avg**	**SD**	**Avg**	**SD**
Easy	3.36	0.83	3.54	0.64	3.75	0.36	3.57	0.60
Medium	2.98	0.57	3.01	0.67	3.32	0.80	3.38	0.84
Hard	2.58	0.89	2.69	0.80	3.67	0.54	3.58	0.65
Global	3.03	0.71	3.12	0.68	3.52	0.61	3.48	0.73

4.2.1 Pass Rate by Criterion

Table 2 shows the average evaluation scores grouped by difficulty level. In general, scores declined with increasing query complexity, particularly in accuracy and completeness for hard queries. Across all 22 queries, the chatbot achieved the following average scores: **Accuracy**=3.03, **Completeness**=3.12, **Clarity**=3.52, **Policy Adherence**=3.48. **Clarity** consistently received the highest score, while accuracy and completeness indicated areas for improvement.

To further interpret performance, we calculated the proportion of responses rated 3.0 or higher in each category: **Accuracy**: 68%, **Completeness**: 73%, **Clarity**: 95%, Policy **Adherence**: 86%.

Figure 2 presents the distribution of the evaluation scores across the 22 advising questions, showing medians, quartiles, and outliers for each criteria. This analysis highlights that while the chatbot's responses were generally understandable and policy-aligned, their factual depth and thoroughness were less reliable.

4.2.2 Performance by Query Difficulty

When grouped by difficulty level, scores showed a consistent decline in performance: Easy Queries, have the highest scores across all dimensions (3.35–3.75). Medium Queries have a moderate performance (3.0–3.4), and Hard Queries

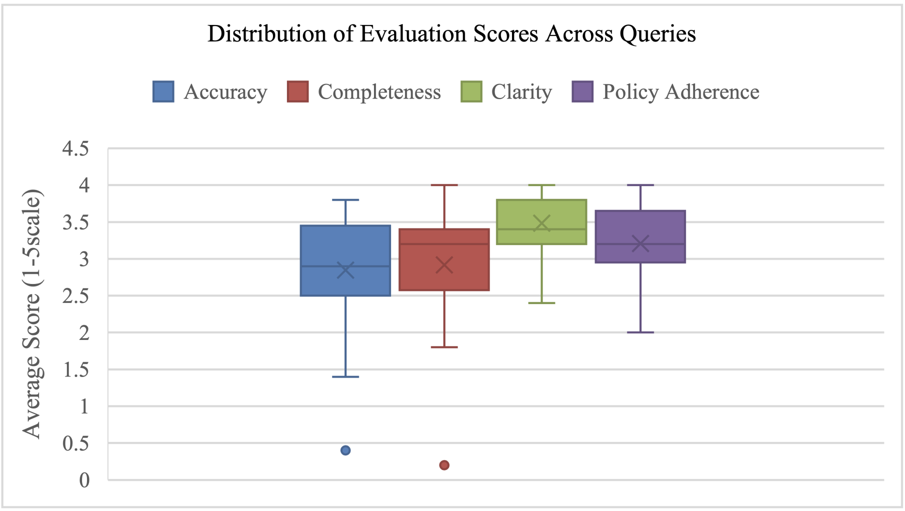


Figure 2: Distribution of evaluation scores (Accuracy, Completeness, Clarity, and Policy Adherence) across the 22 advising queries, showing medians, quartiles, and outliers.

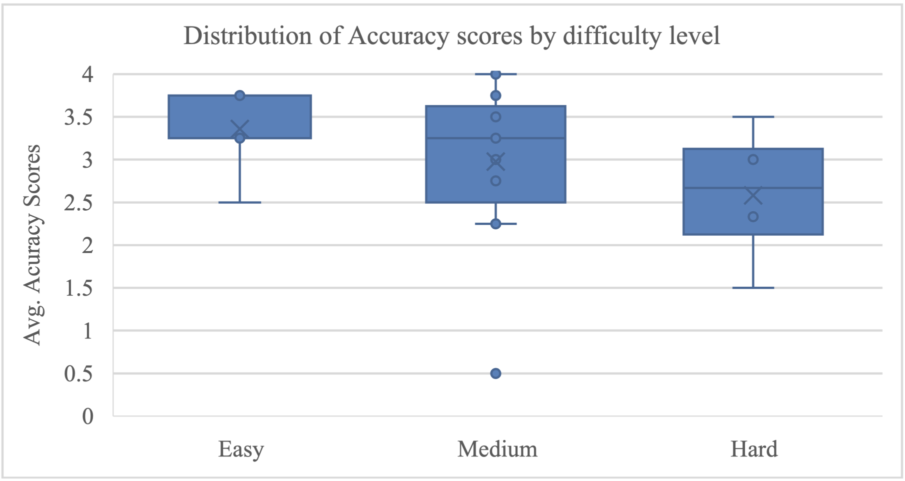


Figure 3: Distribution of accuracy scores by difficulty level.

present lower performance, particularly in accuracy and completeness (2.6–3.7), as shown in Figure 3. The boxplots of accuracy by difficulty show broader variability in the Hard category. The wider spread in accuracy scores for Hard queries reflects both increased complexity and greater disagreement among reviewers.

4.2.3 Qualitative Analysis of Reviewer Comments

In addition to numerical scores, three of the reviewers provided structured comments. These were coded and analyzed to extract common failure modes. The most frequent themes included:

- **Missing key courses or requirements** (e.g., General Education, corequisites): 12 instances
- **Inadequate prerequisite or sequencing logic**: 8 instances
- **Policy misalignment (e.g., overload, catalog year)**: 7 instances
- **Factual inaccuracies**: 5 instances
- **Unclear or overly generic responses**: 4 instances

For example:

- "Too generic; lacked specific course sequence and prerequisites for CET3615." (Q10)
- "MATH1375 is missing. We should always push to complete math requirements early." (Q5)

These qualitative patterns underscore the need for improved retrieval coverage and logical reasoning in multi-constraint advising scenarios. These comments support the rubric scores.

4.3 Summary of Insights

The chatbot performs well on routine advising tasks, offering high clarity and solid alignment with institutional policies. However, performance declines on: i) handling complex multi-semester planning, ii) integrating multiple policy documents, and iii) addressing program transitions or exceptions.

A notable limitation observed in the evaluation is the relatively high standard deviations in Accuracy and Policy Adherence scores (Table 2). While many responses were accurate and policy-aligned, some were inconsistent or misleading, such as omitting required courses or providing factually correct but policy-inconsistent advice. This variability underscores the importance of safeguards and supports the need for policy-grounded validation mechanisms, as proposed in hybrid frameworks such as URAG [17].

These findings support future enhancements, including structured document tagging, multi-agent workflows for policy validation, and integration of policy-aware reasoning mechanisms.

5 Conclusion and Future Work

This study demonstrates that a Naïve RAG-based chatbot can effectively assist in academic advising, particularly for routine course planning queries within a domain-specific context. Through structured human evaluation using a multi-criteria rubric and qualitative reviewer feedback, we found that the chatbot excelled in Clarity and Policy Adherence but showed limitations in Accuracy and Completeness, especially in complex advising scenarios.

By integrating retrieval mechanisms with large language models, the system provides a scalable, low-cost solution for high-volume advising tasks. However, its current implementation lacks the depth required for nuanced policy reasoning, multi-document synthesis, and complex sequencing logic. These challenges highlight the need for enhancements in retrieval precision, contextual inference, and institutional alignment.

Ethical and Privacy Considerations. While our focus was on technical implementation and evaluation, ethical considerations must guide the future development of AI-powered advising tools. Safeguarding student data privacy, ensuring transparency in automated recommendations, and mitigating potential biases in retrieved content are critical for building trust and institutional accountability. Future iterations of this system should incorporate robust privacy safeguards and clearly define the advisory boundaries between automated responses and human support.

While the findings are encouraging, several limitations must be acknowledged. First, the evaluation used only 22 queries and five academic advisors, which limits generalizability. Second, variability in outputs was observed: some responses were factually correct but misaligned with policy, highlighting the need for stronger validation mechanisms. Third, reliance on GPT-4o introduces operational costs, suggesting that open-source LLMs may be explored in the future. Fourth, advising involves sensitive data, making FERPA-compliant privacy safeguards essential.

Building on these findings, future work will focus on four key directions:

- **Multi-Agent Architectures**: Design hierarchical and sequential agents that specialize in retrieval, prerequisite validation, and policy enforcement before synthesis.
- **Expand the Knowledge Base**: Incorporate structured data sources such as course databases, forms, departmental memos, and archived advising policies with catalog versioning.

- **Conduct User-Centered Evaluations**: Implement usability testing with students and advisors to assess trust, engagement, and perceived usefulness.
- **Integrate Policy Reasoning Modules**: Combine symbolic or logic-based components to handle institutional constraints, exceptions, and compliance scenarios more robustly.

Together, these enhancements aim to evolve the prototype into a robust, policy-aware advising assistant capable of delivering not only fluent responses but also institutionally accurate and context-sensitive guidance.

References

[1] Umar Sathic Ali. "Ask your Transcript: LLM Driven Insights for Academic Advising". In: *2024 2nd International Conference on Computing and Data Analytics (ICCDA).* Nov. 2024, pp. 1–4. DOI: 10.1109/ICCDA64887.2024.10867349.

[2] Susan M. Campbell and Charlie L. Nutt. "Academic Advising in the New Global Century: Supporting Student Engagement and Learning Outcomes Achievement." English. In: *Peer Review* 10.1 (Jan. 2008), pp. 4–7. ISSN: 15411389.

[3] Zenobia C. Y. Chan et al. "Academic advising in undergraduate education: A systematic review". In: *Nurse Education Today* 75 (Apr. 2019), pp. 58–74. ISSN: 0260-6917. DOI: 10.1016/j.nedt.2019.01.009.

[4] Anh Nguyen Thi Dieu, Hien T. Nguyen, and Chien Ta Duy Cong. "The enhanced context for AI-generated learning advisors with Advanced RAG". In: *2024 18th International Conference on Advanced Computing and Analytics (ACOMPA).* Nov. 2024, pp. 94–101. DOI: 10.1109/ACOMPA64883.2024.00021.

[5] Matthijs Douze et al. *The Faiss library.* arXiv:2401.08281 [cs]. Feb. 2025. DOI: 10.48550/arXiv.2401.08281.

[6] Jayne K. Drake. "The Role of Academic Advising in Student Retention and Persistence". EN. In: *About Campus* 16.3 (July 2011). Publisher: SAGE Publications Inc, pp. 8–12. ISSN: 1086-4822. DOI: 10.1002/abc.20062. URL: https://doi.org/10.1002/abc.20062.

[7] *EDUC8EU | Intelligent academic advising system.* URL: https://educ8eu.invest-alliance.eu/engine/ (visited on 05/29/2025).

[8] Yunfan Gao et al. *Retrieval-Augmented Generation for Large Language Models: A Survey.* ADS Bibcode: 2023arXiv231210997G. Dec. 2023. DOI: 10.48550/arXiv.2312.10997.

[9] Google. *Embeddings in the Gemini API*. en. 2025. URL: https://ai.google.dev/gemini-api/docs/embeddings (visited on 03/03/2025).

[10] *Hello GPT-4o*. en-US. URL: https://openai.com/index/hello-gpt-4o/ (visited on 03/09/2025).

[11] Omiros Iatrellis et al. "Leveraging Generative AI for Sustainable Academic Advising: Enhancing Educational Practices through AI-Driven Recommendations". en. In: *Sustainability* 16.17 (Jan. 2024). Number: 17 Publisher: Multidisciplinary Digital Publishing Institute, p. 7829. ISSN: 2071-1050. DOI: 10.3390/su16177829.

[12] Jeff Johnson, Matthijs Douze, and Hervé Jégou. *Billion-scale similarity search with GPUs*. arXiv:1702.08734 [cs]. Feb. 2017. DOI: 10.48550/arXiv.1702.08734.

[13] Patrick Lewis et al. *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*. arXiv:2005.11401 [cs]. Apr. 2021. DOI: 10.48550/arXiv.2005.11401.

[14] Maryamah Maryamah et al. "Chatbots in Academia: A Retrieval-Augmented Generation Approach for Improved Efficient Information Access". In: *2024 16th International Conference on Knowledge and Smart Technology (KST)*. ISSN: 2473-764X. Feb. 2024, pp. 259–264. DOI: 10.1109/KST61284.2024.10499652.

[15] Vasilios Mavroudis. "LangChain v0.3". Dec. 2024. DOI: 10.20944/preprints202
0566.v1.

[16] Michael A. McDaniel and Andrea F. Snell. "Holland's Theory and Occupational Information". In: *Journal of Vocational Behavior* 55.1 (Aug. 1999), pp. 74–85. ISSN: 0001-8791. DOI: 10.1006/jvbe.1999.1698.

[17] Long Nguyen and Tho Quan. *URAG: Implementing a Unified Hybrid RAG for Precise Answers in University Admission Chatbots – A Case Study at HCMUT*. arXiv:2501.16276 [cs]. Jan. 2025. DOI: 10.48550/arXiv.2501.16276.

[18] *NLP Text Similarity using Cosine-count vectorizer*. en. URL: https://kaggle.com/code/adepvenugopal/nlp-text-similarity-using-cosine-count-vectorizer (visited on 03/09/2025).

[19] OpenAI. *GPT-4: Large Language Model*. 2023. URL: https://openai.com.

[20] Nils Reimers and Iryna Gurevych. *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. arXiv:1908.10084 [cs]. Aug. 2019. DOI: 10.48550/arXiv.1908.10084.

[21] *sentence-transformers/all-MiniLM-L6-v2 · Hugging Face.* Jan. 2024. URL: `https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2` (visited on 03/09/2025).

[22] Aditi Singh et al. *Agentic Retrieval-Augmented Generation: A Survey on Agentic RAG.* arXiv:2501.09136 [cs]. Feb. 2025. DOI: `10.48550/arXiv.2501.09136`.

[23] Amit Singhal et al. "Modern information retrieval: A brief overview". In: *IEEE Data Eng. Bull.* 24.4 (2001), pp. 35–43.

[24] Troyusrex. *Inside My Virtual College Advisor: A Deep Dive into RAG AI and Agent Technology.* en. May 2024.

[25] Xiaohua Wang et al. "Searching for Best Practices in Retrieval-Augmented Generation". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing.* Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 17716–17736. DOI: `10.18653/v1/2024.emnlp-main.981`.

[26] Chatchai Wangwiwattana and Worawut Jantarick. "Building Intelligent Academic Service Agents: FAQ Extraction and Chatbots With Large Language Models". In: *2024 8th International Conference on Information Technology (InCIT).* Nov. 2024, pp. 370–375. DOI: `10.1109/InCIT63192.2024.10810553`.

[27] Maryrose Weatherton and Elisabeth E. Schussler. "Success for All? A Call to Re-examine How Student Success Is Defined in Higher Education". In: *CBE—Life Sciences Education* 20.1 (Mar. 2021). Publisher: American Society for Cell Biology (lse), es3. DOI: `10.1187/cbe.20-09-0223`.

[28] Lois Wong. *Gaita: A RAG System for Personalized Computer Science Education.* en-us. Dec. 2024. DOI: `10.35542/osf.io/97nmg`.

[29] Shangyu Wu et al. *Retrieval-Augmented Generation for Natural Language Processing: A Survey.* arXiv:2407.13193. July 2024. DOI: `10.48550/arXiv.2407.13193`.

Challenges and Adaptations in Data Science Education for the AI Era*

Daniel D. Wu, Jenq-Foung Yao, Jeannie Pridmore
Department of Information Systems & Computer Science
The J. Whitney Bunting College of Business & Technology
Georgia College & State University
Milledgeville, GA 31061
{daniel.wu, jf.yao, jeannie.pridmore}@gcsu.edu

Abstract

Data science education is undergoing a significant transformation due to the rapid advancements in Artificial Intelligence (AI). This paper examines the inherent challenges in data science education, such as the evolving definition of a data scientist, the persistent skills gap, and the lack of standardized pedagogical frameworks, all of which are further compounded by the rise of AI. It analyzes how AI is reshaping the demands and skillsets required for data scientists, necessitating a fundamental shift in curricula. The paper explores current adaptation strategies by reviewing leading undergraduate data science programs and case studies from universities like Carnegie Mellon, MIT, UC Berkeley, and the University of Michigan. Key strategies for curriculum adaptation are discussed, including the early integration of AI fundamentals, development of specialized AI tracks, infusion of AI ethics and responsible practices, incorporation of generative AI and prompt engineering skills, and promotion of experiential learning opportunities. The paper concludes by considering the future trajectory of data science education, emphasizing the need for continuous curriculum updates, a strong focus on ethical AI, and the cultivation of critical thinking and lifelong learning to prepare graduates for an AI-driven world.

*Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

1 Introduction

Data science has emerged as one of the most rapidly expanding fields due to its inherent capacity to address complex analytical requirements. This burgeoning demand, however, is juxtaposed with a persistent limitation in the supply of qualified data scientists [5]. The field itself is both interdisciplinary and multidisciplinary in nature, drawing from a diverse range of subjects. Consequently, the curricula designed to train future data scientists are in a state of continuous evolution, adapting to meet the ever-changing needs of society [7, 8]. The sustained imbalance between the need for data scientists and their limited availability underscores a fundamental challenge in scaling data science education effectively. The inherent interdisciplinary nature of the field necessitates careful consideration in curriculum design to ensure a balanced coverage of its various foundational elements.

The landscape of data science education is further transformed by the rapid advancements in artificial intelligence (AI). Progress in AI and related domains is actively shaping the development of data science curricula [1]. While AI and data science apparently originated as distinct domains, they have become deeply intertwined, sharing core methodologies and practices. This deep integration necessitates a significant shift in educational focus, as future data scientists must possess the skills to work proficiently with AI technologies. The swift progression of both data science and AI mandates a proactive and ongoing approach to curriculum design and adaptation. This ensures that graduates from data science programs are equipped with the most relevant knowledge and skills, maintaining their competitiveness in an increasingly AI-driven professional world.

In response to these dynamic forces, academic institutions must demonstrate agility in adapting their programs to effectively meet the growing demand for data scientists [9, 10, 13, 19]. This requires the creation of curricula that are not only up-to-date but also competitive, reflecting the latest trends and technologies in both data science and AI. The ultimate goal is to prepare graduates who are well-equipped to navigate and contribute to a world increasingly shaped by AI.

This paper aims to analyze the key challenges and issues currently facing data science education, particularly in the context of the renewed prominence of AI. It will also explore various strategies for adapting undergraduate data science curricula to effectively address these challenges. Furthermore, the paper will examine existing curricula from 18 leading undergraduate data science programs [21], drawing examples and insights from their approaches. Finally, it will incorporate case studies on undergraduate data science education to provide real-world perspectives on the ongoing adaptations within the field.

This paper's contribution lies in its synthesis of current challenges, a broad review of existing programmatic responses, and a forward-looking framework of adaptation strategies derived from an analysis of literature, 18 leading undergraduate data science programs, and illustrative case studies, all contextualized by the rapid evolution of AI.

2 Foundational Challenges in Data Science Education: Examining the Landscape Before the AI Renaissance

Even before the pervasive influence of the current AI revolution, data science education faced several fundamental challenges. One significant issue revolved around defining the precise profile of a data scientist. There has been a notable lack of a universally accepted definition and a standardized set of skills that constitute a data scientist. This ambiguity is further compounded by the diverse educational backgrounds that students bring to data science programs, making it difficult to implement a "one-size-fits-all" educational approach. The lack of clarity surrounding the data scientist role, coupled with the heterogeneity of student preparation, significantly complicates the development of standardized and universally effective curricula.

Another critical challenge has been the persistent data science skill gap [20]. This refers to the evident mismatch between the high demand for data scientists across various industries and the limited pool of available talent possessing the necessary expertise. The field of data science is characterized by rapid technological advancements, necessitating continuous updates and revisions to educational curricula. Educational institutions often face difficulties in keeping pace with the swift breakthroughs occurring in the industry, leading to a lag in the skills imparted to graduates. The dynamic nature of data science, fueled by constant technological progress, presents an ongoing challenge for academic institutions striving to provide relevant and timely training that aligns with industry needs [11].

Furthermore, the pedagogical practices employed in data science education have not been extensively studied in the existing academic literature. This lack of well-established pedagogical frameworks makes it challenging to identify and disseminate best practices for teaching data science effectively [24]. Adding to this complexity is the improbability of achieving a broad consensus on the specific content that should be taught across all data science programs. The body of knowledge related to data science appears to be widely dispersed across the internet, including community question-answering platforms, making it not readily accessible in a structured manner to learners. This lack of established teaching methodologies and standardized content hinders the development of robust and effective data science education.

Understanding what motivates students to learn data science has also been a challenge. Scholarly understanding of the factors that drive students' intentions to pursue data science has been limited [2]. Recognizing and uncovering the salient factors that shape students' motivations to learn data science is imperative for designing effective programs. Research suggests that a student's attitude toward learning data science and their perception of its usefulness are positively related to their behavioral intentions to engage with the field [14]. Therefore, comprehending these motivations is crucial for educational institutions to design engaging and relevant data science programs that can effectively attract and retain students, ultimately addressing the existing skill gap.

3 The AI Renaissance: Reshaping the Demands of Data Science

The advent of the AI revolution has profoundly reshaped the landscape of data science, introducing new demands and expectations for both practitioners and educators. As AI and related technologies advance, they are becoming central to the ongoing evolution of data science curricula. Indeed, machine learning has solidified its position as an integral component of modern data science [12]. Consequently, the integration of AI, particularly machine learning techniques, is no longer an optional specialization but a fundamental requirement for data scientists. This shift necessitates that data science education adapts its core curriculum to adequately reflect the central role of AI.

Furthermore, AI-driven decision-making is becoming increasingly prevalent across a wide spectrum of industries, further underscoring the need for data scientists to possess a strong understanding of AI principles and applications.

The emergence of sophisticated AI technologies is rapidly transforming the capabilities and expectations within data science. Generative AI, for instance, is revolutionizing content creation, data analysis processes, and the automation of various tasks. Large language models (LLMs) are redefining the traditional data science pipeline, offering new ways to interact with and derive insights from data [22]. Areas such as deep learning, computer vision, and natural language processing have become crucial domains within data science, driven by advancements in AI [4]. This proliferation of advanced AI technologies demands that data science education evolves to equip students with the necessary skills to not only understand and utilize but also critically evaluate these significant advancements.

To thrive in this AI-centric era, data scientists require a broader and more specialized set of skills. Proficiency in AI and machine learning algorithms, along with familiarity with relevant tools and platforms, has become essential. Moreover, a deep understanding of AI ethics, potential biases within AI systems, and the principles of responsible AI development are increasingly important [18]. The ability to effectively interact with and guide generative AI models through skillful prompt engineering is also becoming a critical skill for data scientists.

Finally, with AI increasingly performing complex analyses, data scientists must develop the capacity to critically evaluate and effectively manage the insights and outputs generated by these automated systems. Therefore, the rise of AI demands that data science education expand its scope, encompassing not only technical AI expertise but also a strong emphasis on ethical considerations and the ability to collaborate effectively with AI tools.

4 A Comparative Analysis of Leading Undergraduate Data Science Curricula: Current State and AI Integration Efforts

To understand how undergraduate data science education is adapting to the AI era, a comparative analysis of approximately 18 leading programs was conducted. It focused on curricula in the USA, drawing from the top ranked undergraduate Data Science programs by US News & World Report. The methodology involved examining course requirements, available specializations, and stated learning outcomes with a specific focus on elements related to AI. This broad review sought to identify common trends in AI integration, the extent of ethics inclusion, the emergence of generative AI topics, and the availability of experiential learning, with the detailed case studies (presented in Section 6 and Table 1) serving as illustrative examples of these trends.

The analysis indicates that many leading undergraduate data science programs are beginning to integrate foundational AI concepts into their curricula. This often manifests in the inclusion of core AI courses, such as "Introduction to Artificial Intelligence" or "Machine Learning". Across the reviewed programs, a common theme was the re-evaluation of introductory statistics and programming courses to pave the way for earlier exposure to AI principles. In addition, many programs incorporate machine learning topics either within dedicated machine learning courses or as significant components of broader statistics or computer science courses. For instance, Carnegie Mellon University's curriculum offers a strong emphasis on modern statistical and computational methods, which naturally includes machine learning [23]. Similarly, MIT's joint program in Computer Science, Economics, and Data Science includes electives in machine learning [17]. While this trend suggests a growing recognition of AI's importance, the depth and breadth of this integration likely vary considerably across different institutions.

In terms of more advanced AI topics, some programs offer specific courses on areas like deep learning, neural networks, and computer vision. A notable portion of the analyzed programs are also developing specialized tracks or concentrations. Additionally, several universities have established specializations or dedicated tracks focusing specifically on AI and machine learning within their data science programs. While these specialized pathways offer in-depth knowledge, the comprehensive integration of advanced AI topics across the core data science curriculum may still be limited in many institutions.

The critical importance of AI ethics and its societal implications is also gaining traction in undergraduate data science education. Some programs are incorporating dedicated ethics courses or modules within their data science or computer science offerings. Furthermore, some institutions are making efforts to integrate ethical considerations directly into their technical AI-related courses. For example, UC Berkeley's data science program is known for embedding human contexts and ethics directly into its core technical courses [3]. While the recognition of AI ethics is growing, the extent to which it is comprehensively integrated across all data science programs likely still varies.

The analysis reveals that the coverage of generative AI and the specific skills of prompt engineering are still emerging trends within undergraduate data science curricula [15]. Many programs, based on the initial review, do not explicitly mention generative AI in their core course requirements. However, these topics may be included within more advanced machine learning courses or as part of special topics offerings. Notably, some universities are beginning to introduce dedicated courses or initiatives specifically focused on generative AI. This suggests that while the formal integration of generative AI into undergraduate data science education is in its early stages, it is a growing area of focus for curriculum development.

5 Strategies for Curriculum Adaptation: Bridging the Gap to the AI Era

To effectively prepare undergraduate data science students for the demands of the AI era, several strategic adaptations to existing curricula are necessary. One key approach involves integrating AI fundamentals early in the curriculum. This can be achieved by introducing basic AI concepts and terminology within introductory data science courses, providing students with an initial understanding of the field. Furthermore, the basics of machine learning can be incorporated as a natural extension of traditional statistical modeling techniques, demonstrating the evolution from classical statistical methods to more advanced AI-driven approaches. Early exposure to these fundamental AI concepts can build a stronger foundation for students, encouraging them to explore more advanced and specialized topics as they progress through their studies. Such early exposure can also enhance student motivation by demonstrating the immediate relevance and cutting-edge nature of the field, aligning with students' perceptions of data science's usefulness.

Another effective strategy is the development of specialized AI tracks or concentrations within undergraduate data science programs. Creating dedicated pathways allows students with a strong interest in AI to focus their studies in this area. These specialized tracks can offer advanced coursework in specific AI domains, such as natural language processing, computer vision, and deep learning, providing students with in-depth knowledge and highly sought-after skills. Several universities have already implemented AI-focused concentrations, demonstrating the viability and growing importance of this adaptation strategy. Given the increasing societal impact of AI, it is paramount to infuse AI ethics and responsible practices throughout the data science curriculum. This involves integrating ethical considerations into all AI-related courses, ensuring that students understand the potential implications of their work. Curricula should also include case studies and discussions that explore the broader societal impact of AI technologies, encouraging students to think critically about their development and deployment. Emphasizing the principles of fairness, transparency, and accountability in AI applications is crucial for preparing responsible data science professionals. Embedding ethical considerations across the curriculum, rather than treating them as isolated topics, will help students develop a strong ethical compass to guide their future work in the field.

As generative AI continues to evolve and its influence grows, data science curricula must adapt to equip students with the skills to leverage these powerful tools effectively. This includes introducing the core concepts and functionalities of generative AI within relevant courses. Furthermore, teaching effective prompt engineering techniques for various data science tasks, such as data augmentation, code generation, and model refinement, will be essential. Curricula should also explore both the potential benefits and the inherent limitations of generative AI in the context of data analysis and modeling, fostering a balanced understanding of its role in the field. The emerging trend of universities offering courses and workshops focused on generative AI underscores the growing recognition of its importance in data science education.

Finally, promoting experiential learning opportunities with a focus on AI is crucial for preparing students for the practical demands of the field. This can be achieved by developing projects and case studies that require students to apply AI techniques to address real-world problems. Encouraging the use of various AI tools and platforms in data science projects will provide students with hands-on experience. Additionally, facilitating collaborations with industry partners on AI-driven initiatives can offer valuable real-world insights and prepare students for the types of challenges they will encounter in their future careers. Experiential learning is a recognized best practice in data science education, and its specific application to artificial intelligence is essential for students to develop the practical skills needed to succeed in the AI era.

Table 1: Select Universities and Their AI Integration Strategies in Undergraduate Data Science Curricula

University Name	**Key AI Courses/Specializations**	**Focus on AI Ethics**	**Generative AI Initiatives**	**Experiential Learning Opportunities with AI**
Carnegie Mellon University	BS in Artificial Intelligence, Machine Learning, AI for Social Good	Strong emphasis integrated throughout the curriculum	No dedicated courses explicitly detailed in publicly available core curriculum descriptions; may be part of advanced electives or research	Projects spanning education, transportation, healthcare
MIT	Computer Science, Economics, and Data Science (6-14), Applied Data Science Program	Implicitly through CSAIL research and broader technical training	No dedicated courses explicitly detailed in publicly available core curriculum descriptions; may be part of advanced electives or research	Engagement with CSAIL research and development
UC Berkeley	Data Science Major, Human Contexts and Ethics Program	Strong emphasis with dedicated courses and integration into technical courses	Course on Generative Data Science	Data science projects with real-world examples highlighting social and ethical consequences
University of Michigan	Data Science Program, Advanced Artificial Intelligence	Implicitly through general computer science and data science principles	Development of proprietary AI model U-M GPT, Course on Advanced AI covering LLMs and generative AI	Integrated within project-based courses and advanced AI coursework, though not a standalone listed program feature in general descriptions

6 Case Studies: Pioneering Approaches to AI-Integrated Data Science Education

Several leading universities have already begun to implement innovative approaches to integrate artificial intelligence into their undergraduate data science curricula. Examining these case studies provides valuable insights into effective strategies and emerging best practices.

Carnegie Mellon University (CMU) stands out for its early recognition of the importance of AI in undergraduate education [23]. It was the first university in the US to offer a dedicated Bachelor of Science in Artificial Intelligence. Beyond this specialized program, CMU also maintains a strong focus on machine learning and AI within its broader data science curriculum. The curriculum is collaboratively delivered by faculty from computer science, statistics, and other relevant departments, ensuring a comprehensive and interdisciplinary approach. Furthermore, CMU places a significant emphasis on the ethical considerations surrounding AI and actively promotes the development of AI for social good. This proactive approach demonstrates a strong commitment to preparing students for the multifaceted challenges and opportunities of the AI era.

The Massachusetts Institute of Technology (MIT) offers a joint curriculum in Computer Science, Economics, and Data Science (Course 6-14) that incorporates a substantial AI component [17]. MIT students have the unique opportunity to engage with the school's renowned Computer Science and Artificial Intelligence Laboratory (CSAIL), a leading center for AI research and development. The university integrates AI concepts across a wide range of disciplines and fields of study, reflecting its reputation as a center of technical excellence. MIT also offers specialized programs like the "Applied Data Science Program: Leveraging AI for Effective Decision-Making," indicating a commitment to practical AI applications. This interdisciplinary approach, coupled with a strong research focus, provides MIT students with a thorough understanding of both the theoretical underpinnings and the practical applications of AI in data science.

The University of California, Berkeley, has developed a robust Data Science undergraduate program that places a significant emphasis on the human contexts and ethical implications of data and artificial intelligence [3]. The university has also integrated AI and machine learning concepts throughout its data science curriculum. This commitment is further solidified by the launch of the College of Computing, Data Science, and Society, signifying the growing importance of these fields within the institution. Berkeley also offers specialized courses like "Generative Data Science," indicating an adaptation to the latest advancements in AI. By emphasizing both the technical aspects and the broader societal and ethical considerations, Berkeley aims to provide its data science students with a well-rounded and responsible education.

The University of Michigan offers an evolving Undergraduate Program in Data Science that is increasingly integrating artificial intelligence concepts [16]. The university has introduced courses like "Advanced Artificial Intelligence" that specifically cover large language models and generative AI. Notably, the University of Michigan has taken a pioneering step by developing its own proprietary AI model, known as U-M GPT, making it available to students and faculty. This initiative, along with the university's focus on both the theoretical and practical applications of AI, demonstrates a strong commitment to advancing AI education within its data science programs.

While the scale and resources of the above-mentioned universities may differ, the core principles behind their strategies, such as embedding real-world applications, fostering collaboration across departments, and emphasizing ethical and responsible data use, can be adapted to fit the context of smaller institutions. By thoughtfully tailoring these strategies to their own constraints and opportunities, smaller colleges can build robust and impactful data science curricula that align with evolving industry and societal needs.

7 Navigating the Integration: Challenges and Opportunities of AI in Data Science Pedagogy

Although pioneering approaches at leading universities provide valuable models, the widespread integration of AI into undergraduate data science pedagogy presents a range of practical challenges. A primary hurdle is institutional and faculty readiness. The rapid pace of advances in AI often outweighs educators' ability to stay current, which is compounded by the lack of standardized AI curricula tailored for data science programs. These issues can lead to inconsistencies across institutions and even within a single university, where data science programs may be coordinated by various departments.

In the classroom, educators face the difficulty of teaching complex computing and AI concepts to students of various academic backgrounds with varied levels of preparation. The rise of sophisticated generative AI tools also introduces new challenges to academic integrity, making it harder to ensure that students are engaging genuinely with the material. Furthermore, it is a critical pedagogical responsibility to equip students with the skills to identify and mitigate the biases embedded in AI algorithms and the data they are trained on, requiring a careful balance between teaching technical skills and instilling a deep understanding of the ethical and societal implications of AI.

Despite these challenges, integrating AI into data science pedagogy offers significant opportunities to enrich the learning experience. AI-powered tools can create personalized learning paths for students, adapting to their individual needs and learning styles, while AI teaching assistants and chatbots can provide immediate feedback and support. For educators, AI can automate repetitive administrative tasks, freeing them to focus on direct student interaction, and generative AI can be used to create enhanced educational content that makes complex topics more accessible. Ultimately, the thoughtful integration of AI can lead to more innovative assessment methods and has the potential to make data science education more accessible to a broader and more diverse range of students.

8 The Future Trajectory: Data Science Education in the Continuous Evolution of AI

The field of data science education is poised for continued transformation as AI continues to evolve at a rapid pace. The future impact on the profession will be defined by the deepening integration of AI into the fundamental workflows of data science, which will automate certain tasks while creating new, specialized roles for human experts. The focus for data scientists will pivot from performing routine tasks to mastering higher-order skills that leverage and manage AI systems effectively.

As automation handles more data cleaning, preprocessing, and even model deployment, the data scientist's value will shift. The demand for skilled professionals with a strong grasp of AI principles is expected to remain high, but their responsibilities will change. A critical aspect of this future role will be the ability to critically evaluate, interpret, and manage the analyses generated by AI systems. This requires a new emphasis on skills such as prompt engineering to effectively interact with and guide complex models such as generative AIs.

Furthermore, emerging technical frontiers will shape educational and professional demands[6]. The growing importance of real-time data processing and the expansion of edge computing will require new specializations and skillsets. Simultaneously, as AI's capabilities expand, the need for human oversight in ethical AI and responsible development will become paramount. Future data scientists will be tasked with ensuring fairness, mitigating bias, and leading interdisciplinary collaborations to address complex real-world challenges where AI is applied. To prepare for this dynamic environment, the most essential attribute for graduates will be a commitment to lifelong learning, enabling them to adapt to the continuously evolving landscape of both data science and AI.

9 Conclusion

The integration of AI into undergraduate data science education presents both significant challenges and transformative opportunities. While institutions must address issues like limited faculty expertise, the need for curriculum standardization, and concerns around academic integrity, the potential benefits of using AI to create personalized learning experiences and enrich educational content are substantial. This paper has synthesized these issues, presenting a framework of adaptive strategies derived from a comprehensive review of leading undergraduate programs and illustrative case studies from universities like Carnegie Mellon, MIT, Berkeley, and Michigan.

As data science evolves in tandem with AI, the job market will demand a new generation of professionals with a broader skillset. To meet this need, curricula must evolve to incorporate AI fundamentals and ethics early, develop specialized tracks, integrate generative AI skills, and promote experiential learning. Looking forward, a crucial step will be the development and validation of a new, comprehensive pedagogical framework for data science education. By emphasizing critical thinking, promoting interdisciplinary collaboration, and fostering a culture of lifelong learning, educational institutions can empower their graduates not only to thrive in an AI-driven world but also to shape their future trajectory responsibly and ethically.

References

[1] C. Avila-Garzon and J. Bacca-Acosta. “Curriculum, Pedagogy, and Teaching/Learning Strategies in Data Science Education”. In: *Education Sciences* 15.2 (2025). DOI: `https://doi.org/10.3390/educsci15020186`.

[2] R. B. Basnet, D. J. Lemay, and P. Bazelais. “Modeling students’ intentions to learn data science: Using an extended theory of planned behavior”. In: *Knowledge Management E-Learning: An International Journal* 16.4 (2024), pp. 638–652. DOI: `https://doi.org/10.34105/j.kmel.2024.16.029`.

[3] UC Berkeley. *UC Berkeley Data Science Program.* `https://cdss.berkeley.edu/dsus/academics/data-science-major`. Accessed: May 20, 2025.

[4] R. Bommasani et al. “On the Opportunities and Risks of Foundation Models”. In: *ArXiv* abs/2108.07258 (2021). DOI: `https://doi.org/10.48550/arXiv.2108.07258`.

[5] J. Bonnell, M. Ogihara, and Y. Yesha. “Challenges and issues in data science education”. In: *Computer* 55.2 (2022), pp. 63–66. DOI: `https://doi.org/10.1109/MC.2021.3128734`.

[6] N. Clark, C. Morrell, and M. Powell. “Assessment and Continuous Improvement of an Undergraduate Data Science Program”. In: *The American Statistician* 79.1 (2025), pp. 102–121. DOI: `https://doi.org/10.1080/00031305.2024.2365673`.

[7] W.S. Cleveland. “Data Science: An Action Plan for Expanding the Technical Areas of the Field of Statistics”. In: *International Statistical Review / Revue Internationale de Statistique* 69.1 (2001), pp. 21–26. DOI: `https://doi.org/10.2307/1403527`.

[8] R.D. De Veaux et al. “Curriculum Guidelines for Undergraduate Programs in Data Science”. In: *Annual Review of Statistics and Its Application* 4 (2017), pp. 15–30. DOI: `https://doi.org/10.1146/annurev-statistics-060116-053930`.

[9] M. Dogucu et al. “A Systematic Literature Review of Undergraduate Data Science Education Research”. In: *Journal of Statistics and Data Science Education* (2025), pp. 1–20. DOI: `https://doi.org/10.1080/26939169.2025.2486656`.

[10] W. Finzer. “The Data Science Education Dilemma”. In: *Technology Innovations in Statistics Education* 7.2 (2013). DOI: `https://doi.org/10.5070/T572013891`.

[11] Jo Hardin et al. “Data Science in Statistics Curricula: Preparing Students to “Think with Data””. In: *The American Statistician* 69.4 (2015), pp. 343–353. DOI: `https://doi.org/10.1080/00031305.2015.1077729`.

[12] M.I. Jordan and T.M. Mitchell. “Machine Learning: Trends, Perspectives, and Prospects”. In: *Science* 349.6245 (2015), pp. 255–260. DOI: `https://doi.org/10.1126/science.aaa8415`.

[13] S. Kross et al. “The Democratization of Data Science Education”. In: *The American Statistician* 74.1 (2020), pp. 1–7. DOI: `https://doi.org/10.1080/00031305.2019.1668849`.

[14] D. Marti and M.D. Smith. “Motivating Data Science Students to Participate and Learn”. In: *Harvard Data Science Review* 5.1 (2023). DOI: `https://doi.org/10.1162/99608f92.d3b2eadd`.

[15] R. Michel-Villarreal et al. “Challenges and Opportunities of Generative AI for Higher Education as Explained by ChatGPT”. In: *Education Sciences* 13.9 (2023). 856. DOI: `https://doi.org/10.3390/educsci13090856`.

[16] University of Michigan. *University of Michigan Data Science Program.* https://cse.engin.umich.edu/academics/undergraduate/programs/data-science-eng/. Accessed: May 20, 2025.

[17] MIT. *MIT's Computer Science, Economics, and Data Science.* https://catalog.mit.edu/interdisciplinary/undergraduate-programs/degrees/computer-science-economics-data-science/. Accessed: May 20, 2025.

[18] B.D. Mittelstadt et al. "The ethics of algorithms: Mapping the debate". In: *Big Data & Society* 3.2 (2016). DOI: https://doi.org/10.1177/2053951716679679.

[19] N. Msweli, T. Mawela, and H. Twinomurinzi. "Data Science Education – A Scoping Review". In: *Journal of Information Technology Education: Research* 22 (2023), pp. 263–294. DOI: https://doi.org/10.28945/5173.

[20] National Academies of Sciences, Engineering, and Medicine. *Data Science for Undergraduates: Opportunities and Options.* Washington, DC: The National Academies Press, 2018.

[21] U.S. News World Report. *Best Undergraduate Data Science Programs.* https://www.usnews.com/best-colleges/rankings/computer-science/data-analytics-science. Accessed: May 20, 2025.

[22] X. Tu et al. "What Should Data Science Education Do With Large Language Models?" In: *Harvard Data Science Review* 6.1 (2024). DOI: https://doi.org/10.1162/99608f92.bff007ab.

[23] Carnegie Mellon University. *Carnegie Mellon University Data Science Program.* https://www.cmu.edu/dietrich/statistics-datascience/academics/undergraduate/majors-minor/index.html. Accessed: May 20, 2025.

[24] J.M. Wing. "The Data Life Cycle". In: *Harvard Data Science Review* 1.1 (2019). DOI: https://doi.org/10.1162/99608f92.e26845b4.

A Detailed Comparative Analysis of Blockchain Consensus Mechanisms*

Kaeli Andrews, Linh B. Ngo, and Md Amiruzzaman
Department of Computer Science
West Chester University West Chester, PA 19383
{ka993962, lngo, mamiruzzaman}@wcupa.edu

Abstract

This paper presents a comprehensive comparative analysis of two dominant blockchain consensus mechanisms—Proof of Work (PoW) and Proof of Stake (PoS)—evaluated across seven critical metrics: energy use, security, transaction speed, scalability, centralization risk, environmental impact, and transaction fees. Utilizing recent academic research and real-world blockchain data, the study highlights that PoW offers robust, time-tested security but suffers from high energy consumption, slower throughput, and centralization through mining pools. In contrast, PoS demonstrates improved scalability and efficiency, significantly reduced environmental impact, and more stable transaction fees; however, it raises concerns over validator centralization and long-term security maturity. The findings underscore the trade-offs inherent in each mechanism and suggest hybrid designs may combine PoW's security with PoS's efficiency and sustainability. The study aims to inform future blockchain infrastructure development by striking a balance between decentralization, performance, and ecological responsibility.

1 Introduction

Public discourse often reduces blockchain to speculation and cryptocurrency schemes [19], obscuring its broader role as a decentralized infrastructure for

*Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

trustless collaboration. Blockchain technology functions as a distributed digital ledger maintained by a network of computers that reach consensus on every update [29, 25]. This architecture enables transparent, trustless interaction and supports applications beyond cryptocurrency, such as supply chain management [1], digital identity, and voting systems [24]. Blockchain represents a foundational shift in digital trust by enabling value transfer, record verification, and transaction execution without intermediaries.

Blockchain is often celebrated for its transparency and security, but it is ultimately the consensus mechanism that determines whether those ideals are truly achieved. A consensus mechanism is defined as the protocol through which nodes reach agreement on the validity of transactions. The two most widely adopted consensus mechanisms are Proof of Work (PoW) and Proof of Stake (PoS), which currently power leading cryptocurrencies and support a range of decentralized applications across industries [24, 1]. PoW and PoS determine who writes the next *page* in the blockchain, the system's overall security [4, 18], and its associated energy costs [23, 30]. Comparing PoW and PoS is essential for understanding blockchain's broader societal, economic, and environmental impacts [16]. This comparison demands academic rigor and practical insight to capture the nuanced implications of each mechanism.

This study conducts a meta-analysis that compares the technical foundations and real-world implications of PoW and PoS. It breaks down how each mechanism functions, identifies major cryptocurrencies and platforms that rely on them, and weighs their respective strengths and limitations. Furthermore, the analysis examines the broader implications of each mechanism by assessing who benefits, who is excluded, and how each mechanism shapes future blockchain infrastructure in terms of energy consumption, decentralization, and long-term viability [16, 24]. The specific contributions of this study are as follows.

- A detailed comparative analysis of **Proof of Work (PoW)** and **Proof of Stake (PoS)**, highlighting their technical, economic, and environmental trade-offs.

- Quantitative evaluation of **centralization risks** across PoW and PoS networks, including analysis of **validator** and **mining pool** concentration.

- Empirical evidence of **PoW's high energy consumption** versus **PoS's energy efficiency**, with real-world examples (e.g., Bitcoin vs. Ethereum post-Merge).

- Analysis of **scalability limitations** in PoW (slow throughput) versus PoS (higher transaction speeds), supported by network data.

- Evaluation of **societal implications**, including governance fairness, wealth concentration in PoS, and environmental sustainability.
- Insights into **future trends**, such as hybrid consensus models, to balance security, decentralization, and efficiency.

The remainder of this paper is organized as follows. Section 2 briefly discusses previous foundational technical work for blockchain technology and the two consensus mechanisms, PoW and PoS. Section 3 describes our comparison methodology. Section 4 presents the comparison and analyzes the implications. In Section 5, we conclude the paper, identify limitations, and suggest future work.

2 Literature Review

Blockchain networks rely on consensus mechanisms to verify transactions and maintain decentralization. Proof of Work (PoW) and Proof of Stake (PoS) are the two most studied models, each with trade-offs across energy use, security, speed, scalability, centralization, environmental impact, and transaction fees.

Energy use is a defining distinction between PoW and PoS. PoW relies on energy-intensive mining, with Bitcoin's electricity consumption rivaling that of entire nations [30, 28, 11]. This demand grows with network difficulty, raising sustainability concerns [23]. PoS avoids mining altogether, runs on standard infrastructure, and reduces energy use by over 99 percent, as seen in Ethereum (post-Merge) [30].

Security in PoW derives from computational difficulty and cost, deterring Sybil and 51 percent attacks through resource expenditure [4, 29, 11]. PoS relies on financial penalties and stake-based selection, with mechanisms like slashing used to enforce validator honesty [26, 7, 17].

Speed and throughput vary significantly. PoW networks like Bitcoin process blocks slowly with limited transaction rates [4]. PoS systems, including Ethereum (post-Merge) and Polkadot, offer faster confirmation and higher throughput [18, 16].

Scalability is limited in PoW due to strict consensus requirements, often addressed through external solutions. PoS integrates scalability more natively through designs like Hydra and parachains [14, 6, 9].

Centralization risk emerges from different pressures. PoW networks see mining pool dominance, often reflected in low Nakamoto Coefficient scores [20, 8]. In PoS, wealth concentration and custodial staking services threaten validator diversity, though some networks like Cardano demonstrate mitigation efforts [5, 21].

Environmental impact is significant in PoW due to electricity and e-waste from mining hardware [12, 28]. PoS drastically lowers these costs by design [30].

Transaction fees are often volatile in PoW during congestion due to fixed block capacity [4, 15]. PoS networks typically implement adaptive fee models to maintain fee stability and performance under load [13].

This review establishes the foundation for our comparative analysis. The next section details the methodology used to evaluate PoW and PoS across these seven metrics, followed by a results-driven comparison using current examples and empirical data.

3 Methodology

This study adopted an approach similar to meta-analysis to analyze the two major blockchain consensus mechanisms: Proof of Work (PoW) and Proof of Stake (PoS). We first discuss the fundamental designs of blockchain's operations and consensus mechanisms, then explain how these designs inform our choices of comparative metrics.

3.1 Fundamental designs

Blockchain is a decentralized digital ledger that records transactions securely and transparently across a distributed network of computers [29, 25]. As illustrated in Figure 1, a transaction begins when a user broadcasts it to the network. Next, network nodes verify the transaction's validity, group it into a block, and append it to the chain through a consensus mechanism [24, 1]. Each block contains a cryptographic hash of the previous block, ensuring that the data remains immutable [29]. The structure of blockchain eliminates the need for centralized control, making it foundational to systems that require trust, transparency, and resilience [27, 25].

The PoW consensus mechanism, shown in Figure 2, secures the blockchain through computational competition [4]. Miners build candidate blocks from pending transactions and repeatedly hash each block's data with nonce values until the resulting hash meets the network's difficulty target [27, 11]. This target regulates how often new blocks are added, adjusting automatically to maintain a consistent block time, such as every 10 minutes in Bitcoin [11, 16]. The process is intentionally resource-intensive, making it difficult for any single actor to manipulate the ledger [4].

The PoS consensus mechanism, shown in Figure 3, secures the blockchain through stake-based validator selection. Validators are selected to propose new blocks based on the amount of cryptocurrency they lock as collateral [2], with weighted, pseudo-random algorithms favoring larger stakes [3]. This process is

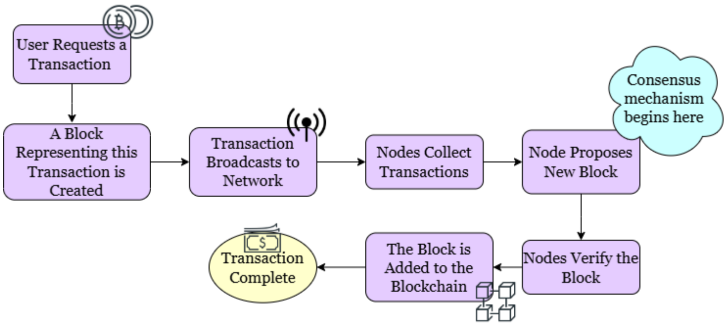

User Requests a Transaction
A Block Representing this Transaction is Created
Transaction Broadcasts to Network
Nodes Collect Transactions
Node Proposes New Block
Consensus mechanism begins here
Nodes Verify the Block
The Block is Added to the Blockchain
Transaction Complete

Figure 1: Overview of Blockchain Technology Flowchart

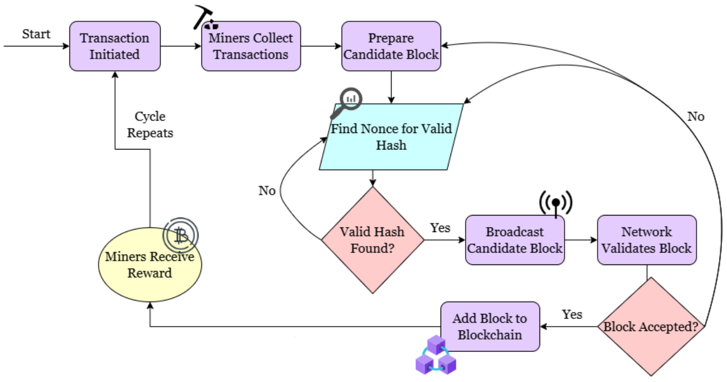


Figure 2: Proof of Work Flowchart

formalized in protocols such as Ouroboros [17] and evaluated in comparative studies [16]. The selected validator assembles pending transactions into a block and broadcasts it to the network, where other validators vote on its validity [27]. If approved, the block is added and rewards are distributed in transaction fees or new tokens [16]. To deter dishonest behavior, PoS networks enforce slashing penalties that remove part of a validator's stake for submitting invalid or conflicting blocks [17, 2, 3, 27].

3.2 Metric Selections

As shown in Table 1, the analysis focused on several core evaluation metrics commonly discussed in blockchain research.

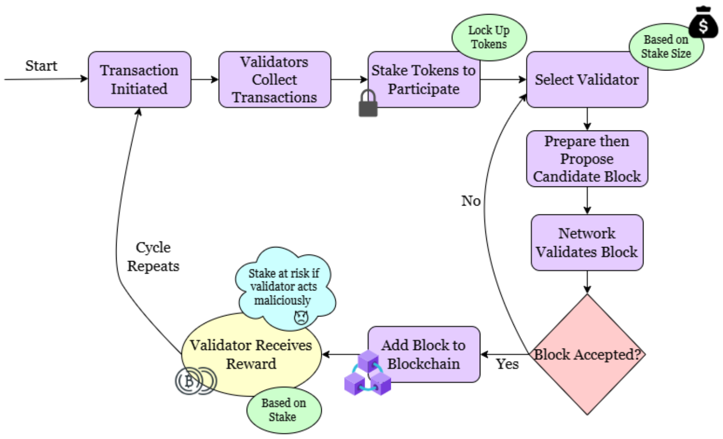


Figure 3: Proof of Stake Flowchart

4 Results

Our analysis compares Proof of Work (PoW) and Proof of Stake (PoS) using key evaluation metrics, as summarized in Table 1. Each metric highlights a different aspect of how consensus mechanisms shape blockchain systems and their broader impact. The following subsections present the results for each category in turn.

4.1 Energy Use

Energy use refers to the amount of electricity consumed by a blockchain network to maintain consensus and secure the system. Proof of Work (PoW) consumes significantly more energy than Proof of Stake (PoS) due to its reliance on computational mining [11]. PoW requires miners to perform brute-force computations to solve cryptographic puzzles, which consumes large amounts of electricity [16]. Bitcoin mining alone consumes an estimated 100–150 terawatt-hours per year [23], exceeding the annual energy consumption of countries such as Norway [28, 12]. This level of demand raises substantial environmental and economic concerns about the sustainability of PoW [30].

PoW's design ties network security to resource expenditure, making it inherently energy-intensive. Studies show that as mining difficulty increases, energy use scales proportionally, compounding environmental impact [3, 11]. The development of specialized mining hardware and large-scale mining operations further intensifies carbon emissions and regional energy strain [30, 2].

Table 1: Key Evaluation Metrics for Blockchain Consensus Mechanisms

Metric	Description	Papers
Energy Use	Examines the computational and environmental costs associated with each mechanism.	[2, 3, 11], [16, 23, 12, 30, 28]
Security	Assesses resilience against common attacks and network vulnerabilities.	[26, 4, 7, 11, 16], [17, 18, 6, 27]
Speed and Throughput	Evaluates how quickly transactions are processed and confirmed.	[4, 16, 18, 6, 29]
Scalability	Determines each mechanism's ability to handle increasing network demands.	[4, 6, 9], [14, 16, 18, 29]
Centralization Risk	Analyzes the extent of power concentration within a network, using the *Nakamoto Coefficient* to quantify the number of entities required to compromise consensus.	[25, 3, 2], [4, 5, 8], [10, 17, 18, 20, 27]
Environmental Impact	Reviews broader ecological effects beyond raw energy usage.	[16, 15, 12, 28, 30]
Transaction Fees	Considers how network congestion affects transaction costs.	[16, 18, 13, 27, 15]

PoS eliminates the need for computational competition by selecting validators based on their stake in the network [16]. This model significantly reduces energy consumption, as validators only maintain network connectivity and use digital signatures to approve new blocks [23]. After Ethereum's 2022 transition to PoS, its energy use dropped to an estimated 0.01 terawatt-hours per year [28]. Sustainability assessments confirm that PoS architectures consume dramatically less energy than PoW systems, often by several orders of magnitude [23].

4.2 Security

Security refers to a blockchain network's ability to resist attacks, maintain data integrity, and ensure that only valid transactions are confirmed. Proof of Work (PoW) and Proof of Stake (PoS) approach security through different cost structures and deterrents.

In PoW, miners compete to solve cryptographic puzzles, with difficulty adjusted dynamically to maintain an average block time of around 10 minutes. This prevents blocks from being created too quickly and keeps the network synchronized [11]. The high computational and hardware costs required to control a majority of the network's hash power make 51 percent attacks economically unfeasible at scale [29, 18]. PoW is also resistant to Sybil attacks, as creating many fake identities requires significant computational resources [4]. Over time, PoW has demonstrated real-world robustness. Bitcoin, for instance, has operated securely for over a decade without a successful consensus-level attack [11, 18].

PoS secures the network by selecting validators based on their stake in the protocol [16]. Validators with greater holdings have more at stake, incentivizing honest behavior [17]. Malicious actions such as signing conflicting blocks or altering transaction data can result in slashing penalties that destroy a portion of the validator's stake [18]. This introduces direct financial penalties, aligning network security with validator incentives.

However, PoS introduces different vulnerabilities. Because block production is inexpensive, validators might attempt to sign multiple competing chains, known as the Nothing-at-Stake problem [7]. Most PoS protocols counter this with slashing and finality rules, but their long-term effectiveness continues to be evaluated [26]. Another concern is stake centralization. If a small number of participants accumulate most of the staked tokens, they could launch stake-based 51 percent attacks or disproportionately influence governance decisions [6].

Although platforms like Ethereum (post-Merge) and Cardano have adopted PoS, it remains a relatively newer model. Its security relies more heavily on economic assumptions and continues to be tested under real-world conditions. In contrast, PoW is a mature and well-tested system that has demonstrated resilience in hostile environments over time [17, 7].

4.3 Speed and Transaction Throughput

Speed and transaction throughput refer to the rate at which a blockchain can process and confirm transactions over time. In Proof of Work (PoW) networks, speed is constrained by computational mining and fixed block intervals. Bitcoin produces blocks approximately every 10 minutes and supports an average throughput of about seven transactions per second [4, 29]. Even in faster PoW systems, confirmation delays and multi-block finality reduce effective throughput [18, 16]. These structural limitations stem from the need to maintain decentralized synchronization and secure consensus.

Proof of Stake (PoS) networks remove the computational burden of mining, enabling shorter block times and higher transaction throughput. Platforms such as Ethereum (post-Merge), Cardano, and Polkadot report transaction rates ranging from 250 to over 1,000 per second, depending on their architecture and scaling strategies [6, 16]. PoS networks also achieve faster finality, with many confirming transactions within seconds. These characteristics make PoS well-suited for high-volume applications like decentralized finance and real-time smart contract execution [16, 18].

While PoW's block intervals limit throughput, PoS achieves higher transaction rates through quicker finality and more efficient validation processes [16, 6].

4.4 Scalability

Scalability refers to a blockchain's ability to maintain performance as demand increases. In Proof of Work (PoW), scalability is limited by protocol constraints and the requirement for global consensus. These factors reduce throughput and necessitate external solutions to handle increased demand [4, 29]. Bitcoin and Dogecoin illustrate this challenge, processing approximately seven and thirty-three transactions per second, respectively. These rates are insufficient for large-scale applications without support from off-chain tools such as Layer 2 networks [6, 14].

Proof of Stake (PoS) enhances scalability by removing the need for energy-intensive mining. Validators are selected based on stake rather than computation, which lowers overhead and enables more efficient coordination [16, 6]. This allows PoS networks to process higher transaction volumes without overwhelming the system. Ethereum (post-Merge), for instance, aims to support up to 100,000 transactions per second using scaling mechanisms such as sharding and rollups [14, 9]. Sharding divides the blockchain into smaller segments, or "shards," that process transactions in parallel to increase overall system throughput [9, 6]. Cardano employs Hydra, an off-chain transaction protocol that reduces demand on the main layer [16]. Polkadot uses parachains, which are independent blockchains running in parallel with the core network, allowing the system to handle more transactions simultaneously [14, 6].

PoW's architecture imposes inherent scalability limitations, requiring external solutions to maintain performance. PoS architectures often integrate native features that support higher throughput [29, 18].

4.5 Centralization Risk

Centralization risk refers to the concentration of control within a small number of entities capable of influencing or compromising consensus. In Proof of Work (PoW), competitive mining often leads to the formation of large mining pools. These pools aggregate computational power to improve reward probability but concentrate control among a few dominant actors [27]. This increases the likelihood of coordinated attacks, including 51 percent attacks, where entities controlling most of the hash power can alter transactions or block new ones from being added [4].

The *Nakamoto Coefficient* is a common measure of decentralization. It estimates how many distinct entities must collude to disrupt consensus [25]. In Bitcoin, this number is typically around 2 to 3, meaning just a few mining pools could compromise the network [20]. Similar patterns are seen in Litecoin and Dogecoin, which share infrastructure and exhibit comparable concentration [5].

PoW networks face centralization through mining pools. Proof of Stake

(PoS) networks, in contrast, encounter distinct risks tied to stake distribution. Validators with larger stakes are more likely to be selected to produce blocks, giving disproportionate influence to high-stakes participants [18]. Many users delegate tokens to third-party services such as Lido or centralized exchanges, which operate validator nodes on their behalf [2]. These services often control a significant share of the total stake and collect a commission from staking rewards [5]. Although delegation increases accessibility, it can centralize voting power among a small number of platforms [26]. Some protocols attempt to mitigate this. Cardano's Ouroboros, for example, uses randomized leader selection and stake distribution strategies to promote broader participation and reduce wealth concentration [17].

In Ethereum (post-Merge), the Nakamoto Coefficient is also estimated at 2 to 3, reflecting stake concentration among validators [8]. In contrast, Cardano maintains a coefficient of about 25, indicating a more distributed set of validators [10].

Overall, PoW is susceptible to mining pool consolidation, while PoS can suffer from validator centralization driven by stake distribution. The Nakamoto Coefficient offers a practical way to evaluate decentralization across both systems [25, 5].

4.6 Environmental Impact

Environmental impact refers to the ecological effects of running a blockchain network, including energy consumption, hardware waste, and emissions. While both Proof of Work (PoW) and Proof of Stake (PoS) secure decentralized networks, their environmental footprints differ significantly in scale and nature.

PoW's high energy use results from its reliance on computational mining. Miners compete to solve cryptographic puzzles, consuming a significant amount of electricity in the process. This is an intentional design feature that ties security to resource expenditure [4, 17]. Bitcoin alone consumes an estimated 100–150 terawatt-hours annually, more than countries such as Argentina or Norway [12, 15]. Specialized mining hardware like ASICs further intensifies the impact, producing electronic waste and emissions due to short life cycles and high power demands [30]. Environmental impacts vary based on regional energy profiles. In areas with fossil-fuel-heavy energy grids, carbon intensity is higher. In some cases, coal plants have been reactivated to meet mining demand [28, 30]. Subsidized electricity in certain regions also contributes to PoW's environmental burden.

PoS significantly reduces energy use by replacing mining with stake-based validation. Validators are selected based on the amount of cryptocurrency they stake, eliminating the need for repetitive computation [16]. PoS networks run on standard server infrastructure. Following Ethereum's 2022 transition to

PoS, energy consumption dropped by over 99 percent [23]. PoS also reduces hardware turnover and electronic waste, which aligns better with renewable energy. Validator nodes can run efficiently in data centers or decentralized setups powered by clean energy [30]. Although large custodial staking platforms may increase overall energy use, the impact remains minimal compared to PoW and can be mitigated through sustainable hosting practices.

4.7 Transaction Fees

Transaction fees refer to the costs users pay to have their transactions processed, often rising during periods of network congestion. Fee structures vary by consensus model, particularly between Proof of Work (PoW) and Proof of Stake (PoS).

In PoW networks like Bitcoin, transaction fees incentivize miners alongside block rewards [4, 18]. When activity increases and block space is limited, users compete by offering higher fees, creating an auction-like model. This leads to fee volatility, especially during market surges, where smaller transactions may become impractical [16, 18]. These fluctuations reflect broader scalability limitations in PoW systems, where block size and timing constraints reduce throughput and increase congestion [27].

PoS networks typically offer more stable and predictable fee structures. Because validators do not incur high energy costs, they rely less on transaction fees as a primary incentive [16]. Some networks, such as Ethereum (post-Merge), employ base fee models that adjust dynamically in response to network demand, thereby helping to reduce volatility while managing congestion [15]. Analytical models based on queueing theory suggest PoS systems experience less fee inflation under load due to faster finality and flexible confirmation processes [13]. These efficiencies improve performance during peak usage and support high-throughput applications such as decentralized finance.

Overall, PoW networks tend to have higher and more volatile fees due to mining incentives and block constraints, while PoS networks use adaptive models that help stabilize costs during congestion.

4.8 Final Results

Table 2 summarizes the comparative outcomes of PoW and PoS. These ratings reflect the analysis across Sections 4.1–4.7.

PoW earns a *high* energy use rating due to its reliance on energy-intensive mining, while PoS earns a *low* rating for removing the need for computational competition. Both mechanisms earned *strong* security marks: PoW for its decade-long operational resilience, PoS for enforcing economic penalties despite shorter track records.

Table 2: Comparison of PoW and PoS in terms of features

Feature	**PoW**	**PoS**
Energy Use	High	Low
Security	Strong	Strong
Speed	Slow	Fast
Scalability	Limited	Scalable
Centralization Risk	High	Moderate
Environmental Impact	High	Low
Transaction Fees	Variable	Stable

PoW was rated *slow* and *limited* in scalability due to fixed block intervals and protocol rigidity. PoS earned *fast* and *scalable* ratings based on faster finality and support for various optimization methods. PoW's *high* centralization risk stems from mining pool dominance, whereas PoS was rated *moderate* due to validator stake concentration, with Cardano as a more decentralized example.

PoW's *high* environmental impact rating results from its energy demands and hardware waste. PoS scored *low* for operating efficiently on standard infrastructure. PoW's fees were rated as *variable* due to congestion spikes, while PoS achieved *stable* ratings for adaptive fee models and lower operational costs.

5 Conclusion

This study compared Proof of Work (PoW) and Proof of Stake (PoS) across seven metrics: energy use, security, speed, scalability, centralization risk, environmental impact, and transaction fees. PoW delivers proven security, but incurs high energy costs, slower throughput, and centralized mining dynamics. PoS improves efficiency and scalability while reducing energy use, though it raises concerns about validator concentration and long-term resilience.

Neither mechanism fully meets all technical and structural requirements. PoW is well-suited for systems that prioritize security and immutability, while PoS aligns with applications emphasizing performance and sustainability. Both involve trade-offs between decentralization, efficiency, and resilience.

Hybrid consensus models present a potential path forward by integrating the robustness of PoW with the adaptability of PoS [26, 14, 3]. Nervos exemplifies a layered hybrid approach by using PoW for base security while enabling energy-efficient, PoS-driven smart contracts and scalability [22]. Future research should explore validator equity, ecological sustainability, and governance frameworks that maintain decentralized integrity at scale [30, 23]. While this analysis aligns with current literature and reported data, it does not include empirical testing and may generalize trends that differ across implementations. Despite these limitations, the comparative framework remains a valuable tool for guiding sustainable and equitable blockchain innovation.

References

[1] Maher A. N. Agi and Ashish Kumar Jha. "Blockchain technology in the supply chain: An integrated theoretical perspective of organizational adoption". In: *International Journal of Production Economics* 247 (2022).

[2] I. Bentov, A. Gabizon, and A. Mizrahi. *Cryptocurrencies without Proof of Work*. arXiv preprint arXiv:1406.5694. 2014.

[3] I. Bentov et al. "Proof of Activity: Extending Bitcoin's Proof of Work via Proof of Stake [Extended Abstract]". In: *SIGMETRICS Performance Evaluation Review* 42.3 (2014).

[4] J. Bonneau et al. "SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies". In: *2015 IEEE Symposium on Security and Privacy*. 2015.

[5] Brookings Institution. "The Hidden Danger of Re-centralization in Blockchain Platforms". In: *Brookings TechStream* (2023).

[6] A. L. Bulgakov et al. "Scalability and Security in Blockchain Networks: Evaluation of Sharding Algorithms and Prospects for Decentralized Data Storage". In: *Mathematics* 12.23 (2024).

[7] L. Castro et al. *Unsealing the Secrets of Blockchain Consensus: A Systematic Comparison of the Formal Security of Proof-of-Work and Proof-of-Stake*. arXiv preprint arXiv:2401.14527. 2024.

[8] CCN. *Nakamoto Coefficient Explained: How Decentralized Are Blockchain Networks?* CCN. 2025.

[9] B. Chen et al. *A Comprehensive Survey of Blockchain Scalability: Shaping Inner-Chain and Inter-Chain Perspectives*. Preprint. 2024.

[10] CoinStats. *What is the Nakamoto Coefficient and How Do You Calculate It?* CoinStats. 2025.

[11] A. Gervais et al. "On the Security and Performance of Proof-of-Work Blockchains". In: *Proceedings of the ACM CCS*. Oct. 2016, pp. 48–57.

[12] S. Huestis. *Cryptocurrency's Energy Consumption Problem.* Tech. rep. RMI, 2023.

[13] Koki Inami and Tuan Phung-Duc. "Analysis of dynamic transaction fee blockchain using queueing theory". In: *Mathematics* 13.6 (2025), p. 1010.

[14] A. K. Jain, N. Gupta, and B. B. Gupta. "A Survey on Scalable Consensus Algorithms for Blockchain Technology". In: *Cyber Security and Applications* 3 (2025).

[15] Archana Jain, Chinmay Jain, and Karolina Krystyniak. "Blockchain transaction fee and Ethereum Merge". In: *Finance Research Letters* 58 (2023), p. 104507.

[16] B. Jha et al. "Blockchain Technology: Proof of Work vs Proof of Stake Consensus Mechanism". In: *Reference Module in Social Sciences.* Elsevier, 2024.

[17] A. Kiayias et al. "Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol". In: *Proc. CRYPTO 2017.* Santa Barbara, CA, 2017.

[18] X. Li et al. "A Survey on the Security of Blockchain Systems". In: *Future Generation Computer Systems* (2020).

[19] Ben McKenzie and Jacob Silverman. *Easy Money: Cryptocurrency, Casino Capitalism, and the Golden Age of Fraud.* Abrams, 2023.

[20] Nakaflow. *Nakamoto Coefficient Analysis.* Nakaflow.io. 2025.

[21] C. Natoli et al. *Deconstructing Blockchains: A Comprehensive Survey on Consensus, Membership and Structure.* arXiv preprint arXiv:1908.08316. 2019.

[22] Nervos Foundation. *Nervos: An In-Depth Overview of a Blockchain Network Built for Modularity.* Nervos Knowledge Base. Apr. 2023.

[23] E. Ok, B. Barnty, and O. Joseph. "Energy Consumption of Blockchain Networks". In: *IEEE Access* (2024).

[24] S. Sarmah. "Understanding Blockchain Technology". In: *Journal of Computer Engineering* 8.2 (2018).

[25] B. S. Srinivasan. *Quantifying Decentralization.* Earn.com. 2017.

[26] M. M. Yakubu et al. "A Systematic Literature Review on Blockchain Consensus Mechanisms' Security: Applications and Open Challenges". In: *Computer Systems Science and Engineering* 48.6 (2024).

[27] S. Yan. *Analysis of the Advantages and Existing Problems of Blockchain Consensus Mechanism.* arXiv preprint arXiv:2209.11545. 2022.

[28] Z. Yang et al. “Blockchain Technology in Building Environmental Sustainability: A Systematic Literature Review and Future Perspectives”. In: *Building and Environment* 245 (2023).

[29] Z. Zheng et al. “An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends”. In: *2017 IEEE International Congress on Big Data*. Honolulu, HI, USA, 2017.

[30] A. Zimba et al. *Blockchain Technology and Energy Efficiency: A Systematic Literature Review of Consensus Mechanisms, Architectural Innovations, and Sustainable Solutions*. Research Square Preprints. 2025.

Making Time in Twoville, a Language for Fabrication*

Chris Johnson[1] and Devran Turson
Department of Computer Science
James Madison University
Harrisonburg, VA 22807
[1] johns8cr@jmu.edu

Abstract

Twoville is a platform for learning computer science with physical output. Instead of printing text to a console, learners trace out 2D designs and fabricate them into tangible and non-electronic objects using cutters and plotters. The editor is bidirectional. Though text-based code is the primary interface, a design's parameters may also be directly manipulated on the drawing canvas. Support for animation has recently been added. In this paper, we discuss the new syntax and features of Twoville that allow an animation to be programmed, directly manipulated, and fabricated.

1 Introduction

Many first programs print "Hello, world!" to a text console. Rarely does this greeting reach such a large audience. The programmer sees it and moves on to the next lesson. Newer learn-to-code platforms replace the text console with a canvas on which designs can be programmatically drawn and animated. Digital art arguably has greater visual appeal than text. However, if the output of a program can only be experienced within the platform or on the computer, it still hasn't reached the world. Output that is merely virtual relegates computer science to an imaginative realm disconnected from the physical world. It

*Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

reduces programming to disembodied cognition. We offer Twoville as a platform for learning computer science by making physical objects that have a life beyond the computer. Objects made in Twoville can be carried around in one's pocket, hung on one's fridge, and freely shared with others in unplugged settings.

Code can be a frustrating interface for design work because of its indirect relationship with the output. To make a programmed object bigger, for example, we must locate the size parameter in the source code and enter new values through trial-and-error until we are satisfied with the result. The process can be made less clumsy by supporting direct manipulation. Instead of editing text, we operate directly on the output through drags and clicks. Twoville supports both kinds of manipulation.

After several years of using Twoville to make tangible items like stickers, cards, clothing, sculptures, and puzzles, we have recently added support for animation. On the surface, animation doesn't seem like an appropriate feature for a tool used to design physical objects. Indeed, we initially added animation simply because we have also been using Twoville to make non-physical assets like figures for instructional materials, and we wanted some of these to be animated. However, artists for centuries have been depicting motion in static and physical works through superimposition, polyptychs of discrete panels, and unstable composition. Accordingly, we have added several commands to Twoville for exporting static designs that show the passage of time.

We examine Twoville and its animation and direct manipulation features in this paper. In section 2, we examine others' work on direct manipulation and programmatic animation. In section 3, we discuss how we achieve direct manipulation in Twoville. In section 4, we introduce Twoville's syntax for programmatic animation. In section 5, we explore how an animation can be fabricated into a non-virtual object. In section 6, we reflect on our experiences using Twoville with youth in our community and consider future work.

2 Related Work

Our design of Twoville is informed by two distinct areas of research: computational fabrication and animation programming. We discuss the work of others in these areas and situate Twoville amongst their findings.

2.1 Computational Fabrication

Rode et al. [10] introduce the term *computational making* to describe computer-assisted design and fabrication. They identify five elements that are key to computational making activities: the output has an aesthetic component independent of the functional, the activity presents an opportunity for creativity, physical construction skills are employed, makers must navigate between virtual and physical representations, and successful making builds on and develops knowledge of material properties. These elements are consistent with our own experiences using Twoville with hundreds of students.

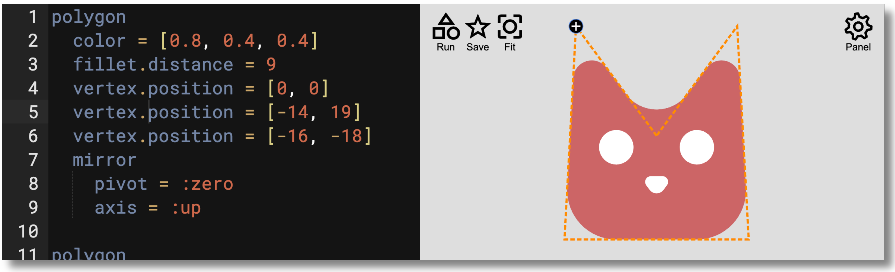
Ed-technologists have heralded personal fabrication as a democratizing force that brings the knowledge and tools of physical production back to individuals. Nevertheless, Halverson and Sheridan [4] point out that the maker education movement has largely focused on robotics, electronics, and vehicles, activities that have traditionally appealed to more boys than girls. Twoville's output is non-electronic, made of paper, vinyl, fabric, wood, or acrylic. We have observed no disparity of attitudes across genders.

With the rise of affordable laser cutters and 3D printers, many small tools for fabrication have been developed in the last two decades. Johnson [6] built Flatlang, a Logo-like language for designing lasercut pieces. Turbak et al. [12] present a similar work using a blocks language called TurtleBlocks. Efrat et al. [2] propose introducing designers to algorithmic and parametric design tools not with a computer, but with a catalog of patterns. The designer identifies a desired pattern and then uses software to achieve it.

From an economic standpoint, personal fabrication is prudent only when the products are fabricated for individuals rather than the mass market. But the design itself should also be personal. Chytas et al. [1] enrolled 27 high school students in a four-day workshop on making using programming tools. In interviews, students favored creative ownership over consumption of others' designs found in model repositories like Thingiverse. One subject said in an interview, "Even when there is the same model online, I would prefer to design it by myself. It is special to me this way." Jacobs and Buechley [5] posit that computation brings several benefits to making: precision, automation, generativity through randomness, and parameterization. Subjects in two studies used the authors' Codeable Objects system to design lamps and clothing. One subject felt like the tool masked the design process, feeling that they were choosing among a narrow set of parameters rather than originating a plan. In Twoville, we have thus far avoided providing a library of readymade but parameterized templates. We aim to develop computation skills in its users as they aim to develop physical artifacts.

2.2 Animation Programming

Kazi et al. [7] developed ChronoFab, a plugin for the 3D modeling program Maya. The plugin allows artists to generate static 3D sculptures of animated objects by adding extra geometry along the object's trajectory. This extra geometry may be motion lines tracking vertices, swept surfaces, stroboscopic stamps of the entire model, or particles. In Twoville, we export static designs



```
 1 polygon
 2   color = [0.8, 0.4, 0.4]
 3   fillet.distance = 9
 4   vertex.position = [0, 0]
 5   vertex.position = [-14, 19]
 6   vertex.position = [-16, -18]
 7   mirror
 8     pivot = :zero
 9     axis = :up
10
11 polygon
```

Figure 1: A cat's head composed of two rounded polygons and two circles. The cursor is placed on line 5, so only its handle appears in the canvas. When dragged, the vertex will be updated in both the text editor and the canvas. Since the polygon is mirrored, the reflected vertex will also be updated.

of animations using similar techniques inspired by art.

Several projects have explored the strengths and weaknesses of various programming paradigms for animation. Elliot [3] observes, "Any language makes some ideas easy to express and other ideas difficult," and criticizes imperative systems for obscuring an animator's intent with too many low-level instructions. The author offers FRAN, a functional and domain-specific declarative language for animation.

In an experiment conducted by Krämer et al. [8], subjects were given five animation tasks to complete. They completed each twice, once with a declarative animation system using keyframes or parametric equations and once with an imperative procedural system that modified the object on each frame. Roughly half the subjects favored the declarative system, and the other half favored a hybrid, despite completing tasks 2.4 times faster with the declarative system. The authors suggest that the subjects, who were programmers, may have been biased toward imperative solutions. Twoville supports a similar declarative syntax based on keyframes or time-parametric equations.

Raffaillac et al. [9] compared the animation interfaces found in Qt 5, Apple's Core Animation, D3.js, JavaFX, Android, and GSAP. Responding to the complexity of these frameworks, the authors suggested a simpler syntactic device: a *delay* operator that can be appended to a property assignment, as in `circle = 10 during 2s`. A delayed assignment triggers an animation instead of an immediate change. The interpolation scheme is polymorphically determined by the property type. Their simpler syntax is compelling, but it favors basic animations that are continuous, modify one property at a time, and use linear interpolation. In Twoville, we offer a more expressive syntax that organizes animation code around time rather than imperative execution.

3 Direct Manipulation

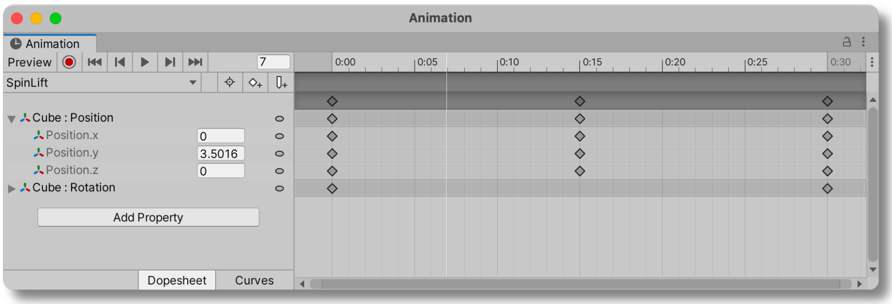
Twoville is a domain-specific language for describing shapes. Elements are configured by assigning values to their properties in a block immediately below the shape command. Many drawing platforms use functions of high arity and positional parameters. Though Twoville's explicit assignments are more verbose, our users don't have to guess at a value's semantic meaning. When a Twoville program is run, its output appears in the integrated drawing canvas.

Designs may be modified through two different kinds of user interaction: direct manipulation [11] via the mouse or indirect manipulation via code. Tools have historically supported just one of these interactions, but a growing number support both. Twoville supports both with its bidirectional editor. The designer initiates a design by adding a new shape or curve to the program text. There's no button or menu for adding elements because we want the designer to maintain agency and awareness of their code structure.

Once an element is rendered, the designer may make direct edits to its lengths, angles, and positions by dragging on handles in the drawing canvas. In Figure 1, a cat's head has been programmed using a pentagon whose corners have been rounded off. One handle currently appears in the drawing canvas. When dragged, it will immediately change the program text and the design.

Direct manipulation requires significant runtime support. In particular, the editor must be able to identify what source code contributed to the property whose handle is being dragged. A new value for the property is computed based on the mouse position and inserted into the text editor. However, we don't just replace the original expression with a new number. If the designer crafted a more complex expression, like `radius = 0.5 * diameter`, we attempt to preserve this structure. Our update algorithm classifies and updates the expression according to the following categories:

- a, a literal value. The literal a is replaced by the new value derived from the mouse.
- id, a variable name. The algorithm recurses to the most recent assignment to id and updates its right-hand side.
- $l + r$, an invertible operation like addition or multiplication. One of l or r is updated while the other is held constant. Literal values are given update priority. If both expressions have the same priority, then the rightmost operand is updated. The held value and the new value derived from the mouse are combined to solve for the new value.
- (e), an operation that is locked. The parentheses act as a shield protecting its contents from being updated. An offset is added to the original expression to reach the new value derived from the mouse.
- $f(x)$, where f is an uninvertible operation. This is treated just like a locked expression.



Position.x 0
Position.y 3.5016
Position.z 0

Figure 2: The animation timeline interface from the Unity game engine. This animation spins a cube as it lifts and drops.

To classify an expression, we retain the abstract syntax tree representation of the program in memory. Each manipulation handle has a link to its associated tree node, and we query the node's type in our update algorithm.

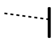
4 Time Syntax

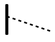
The conventional graphical interface for keyframe animation is an interactive timeline called a *dopesheet*, as shown in Figure 2. Animated properties are listed along the vertical axis, and time spans the horizontal axis. The animator selects a particular frame—a *keyframe*—on the time axis and sets the property to an appropriate value. They scrub to another keyframe and set the property again. During playback, a property's value is interpolated between its surrounding keyframes.

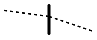
In Twoville, we adopt a textual interface for animation. Text has some advantages over a graphical interface: keyframes and property values can be expressed programmatically, the animation can be reused and parameterized by placing it within a function, and the animation logic has an explicit notation that can be copied and pasted like any text. Text also has disadvantages: it consumes more space, it requires more cognitive effort to read and write, and it carries a stigma of being primitive and arcane. Since our goal is to develop programming skills in our users, we consider the benefits of text to outweigh its costs.

Many graphics programming systems do not offer any special syntax for animation. A property is set at a particular keyframe using a standard method call. To set a circle's radius at frame 50 to 10, the animator might write the statement `circle.set('radius', 50, 10)`. In Twoville, we provide special animation syntax to reduce verbosity and support direct manipulation.

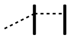
Name	*Example Syntax*	*Visual*	*Semantic Meaning*

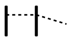
before moment	`-> 50`
	`  radius = 10`
			as we near frame 50, radius is 10

after moment	`50 ->`
	`  radius = 10`
			as we leave frame 50, radius is 10

around moment	`-> 50 ->`
	`  radius = 10`
			as we near and leave frame 50, radius is 10

between hold	`50 -> 70`
	`  radius = 10`
			between frames 50 and 70, radius is 10

before hold	`-> 50 -> 70`
	`  radius = 10`
			as we near frame 50 and between frames 50 and 70, radius is 10

after hold	`50 -> 70 ->`
	`  radius = 10`
			between frames 50 and 70 and as we leave frame 70, radius is 10

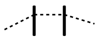
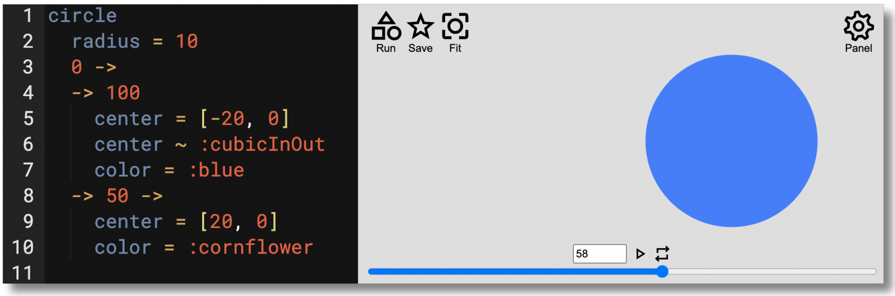
around hold	`-> 50 -> 70 ->`
	`  radius = 10`
			as we near frame 50, between frames 50 and 70 and as we leave frame 70, radius is 10

Figure 3: The seven possible timeblock forms. The behavior before and after a keyframe need not be continuous, leading to multiple forms. Moments set a value at a particular keyframe, and holds set a constant value between two keyframes.

Twoville's text interface preserves the notion of assembling a timeline. A timeline is assembled with a series of *timeblocks*, which consist of a header and a body. Animators select a keyframe in the header and assign values to the animated properties in the body. By itself, a keyframe is just an instant in time, but animators want to specify behavior across larger spans of time. Therefore, a timeblock specifies how the animation behaves around a keyframe rather than at a keyframe.

To set properties as playback nears the keyframe, the animator uses a *before moment* timeblock, as shown in Figure 3. The syntax is inspired by mathematical limits. An *after moment* timeblock is used to set properties as playback leaves the keyframe. These two timeblocks alone are sufficient for assembling any possible timeline. Writing one of these timeblocks is equivalent to dragging a keyframe on an animation curve in a graphical interface.

Text is verbose compared to graphical timeline, and we offer several conveniences for keeping it contained. The *around moment* is syntactic sugar for setting the before and after behavior for properties at a particular keyframe


```
circle
  radius = 10
  0 ->
  -> 100
    center = [-20, 0]
    center ~ :cubicInOut
    color = :blue
  -> 50 ->
    center = [20, 0]
    color = :cornflower

```

Figure 4: A screenshot of a Twoville program in which a circle paces left and right. At keyframes 0 and 100, it is leftmost and blue. At keyframe 50 it is rightmost and cornflower.

with C^0 continuity. *Holds* are syntactic sugar for keeping properties constant between two keyframes. The combination of before and after behavior across moments and holds leads to the seven syntactic forms described in Figure 3. If two timeblocks have the same body, as often happens in cyclical animations, the blocks may be consolidated into a single block with multiple headers. The circle in Figure 4 has its start and end state set with a single block. Because properties are set in blocks rather than with individual method calls, multiple properties can be set at a time. The circle's `center` and `color` are set in tandem.

When playback is between keyframes, the animation interpolates between the two surrounding values. Twoville linearly interpolates by default. Other interpolation schemes are available, like nearest-neighbor, polynomial, sinusoidal, circular, and exponential. The circle in Figure 4 has its center's interpolant set to a cubic function using a special assignment operator. We use the tilde because it looks like a curve.

Keyframe animation is suitable when the number of keyframes is sparse and they can be blended together in a linear or nearly linear manner. Some animations don't fit these constraints. We may define such animations procedurally, with properties expressed as functions of time. In Twoville, the global constant `:frame` represents the current frame index and `:time` represents the current time. These can be referenced in property assignments. Figure 5 shows a program that launches a projectile.

Animations are directly manipulated in the same way as static designs. The designer scrubs to the desired keyframe and then drags on the handles to change properties. The update algorithm extends to timeblocks without any special treatment. Each manipulation handle is associated with a node in the

```
let velocity = [3, 20]
circle
  radius = 3
  center = [
    velocity.x * :time,
    -4.9 * :time ^ 2 + velocity.y * :time
  ]
```

Figure 5: A procedural animation of a ball launched with projectile motion. The `center` property is expressed not as an interpolation of keyframes but as a function of the constant `:time`.

abstract syntax tree. The node may or may not be a child of a timeblock; the context doesn't alter the update algorithm.

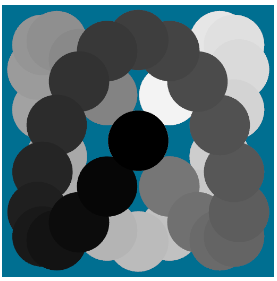
5 Fabricating Time

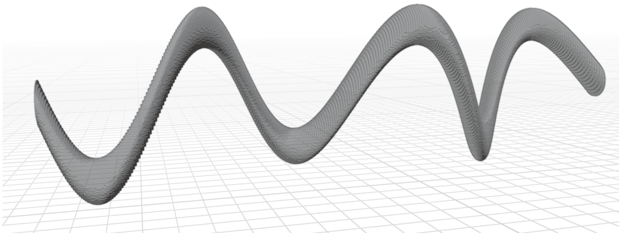
The output of an Twoville program without animation is a vector graphics file that can be sent to fabrication tools like vinyl cutters, laser cutters, pen plotters, CNC routers, and embroidery machines. The design becomes the fabricated object. When a design has animation, we must get creative and decide how to represent time in the static physical output. Twoville can export the animation in the following formats, with examples of each shown in Figure 6.

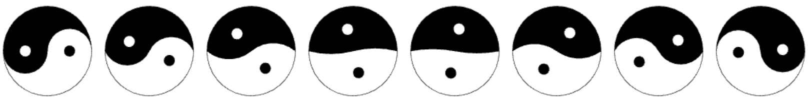
Strobe New frames are superimposed atop older frames, and older frames are progressively faded out and clipped against newer frames. The result is a single static image that can be fabricated with any fabrication tool. The output should be colored in some way to indicate the age of each frame. If many frames are included, the animation will present as motion blur.

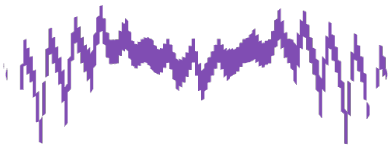
Storyboard The frames are sampled infrequently and rendered in a paneled display. The panels may be arranged in a grid or a strip and plotted on paper. A grid reads like a comic book. A strip can be physically animated by forming it into a circle and placing it inside a zoetrope, which is a spinning drum with narrow viewing slots cut out opposite each frame. The viewer looks through the slots as the drum spins around and reconstructs the animation in their visual cortex.

Time Stack Each two-dimensional frame is plotted or cut separately, and the frames are then bound in a stack along the third dimension. If plotted on


(a) Lissajous strobe


(b) Lissajous time stack


(c) Yin yang zoetrope


(d) Bird without fence

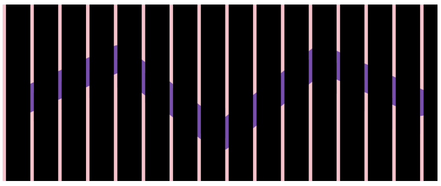

(e) Bird with fence

Figure 6: Example static outputs of an animation. These outputs show the frames of an animation without requiring a computer or Twoville.

paper and flipped through, the frames form a flipbook. If cut from acrylic or plywood and glued together or 3D-printed, the output is an extrusion of time into space.

Picket Fence Strips of each frame are rendered into a composite image in an interleaved fashion. The image is printed and then placed under a physical "fence" with periodic slits that reveal just a single frame of the composite image. By pulling the image along under the fence, the viewer sees the frames play back in sequence. The fence may also be programmed in Twoville and fabricated using a laser or paper cutter.

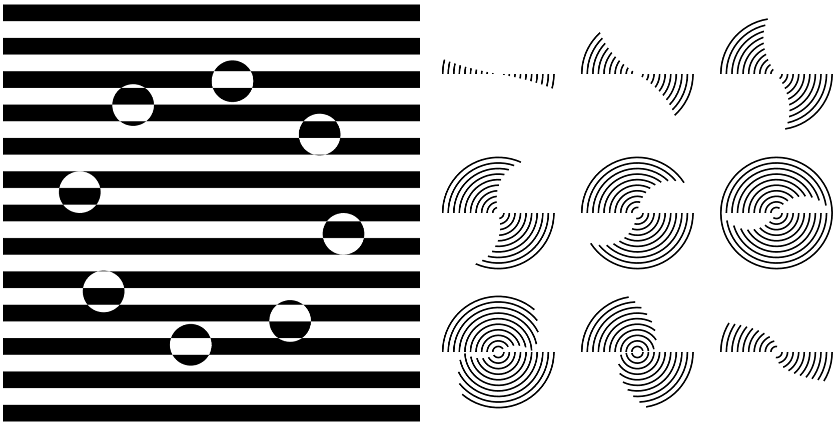

(a) (b)

Figure 7: Two animations written by university students. (a) is a recreation of an optical illusion that requires an animated presentation. (b) is a montage containing 9 samples of a 100-frame spinner animation.

6 Conclusion

We have presented Twoville, a platform for designing physical objects using both code and a mouse. The code interface affords precision, parametric decomposition, and reuse. The direct manipulation interface affords visceral adjustment and immediate visual feedback. The output is a vector graphics file compatible with many fabrication tools. Twoville's new animation system supports the creation of virtual and physical output that reveals the passage of time.

For several years now, we have been using Twoville in outreach events for middle school and high school youth in our community. These events include hour-long campus visits, semester-long weekly after-school programs, and week-long summer camps. Each student leaves with at least one fabricated object, like a vinyl sticker, an embroidered greeting card, a stenciled T-shirt, a plywood mathematical knot, or an acrylic ornament. In informal followup surveys, students regularly comment on the thrill of holding their designed object in their hand and the creative ownership they feel.

We have found that students are mostly likely to achieve viable designs when they start by drawing on graph paper. Only after they've measured and labeled all relevant properties do we grant them access to a computer and Twoville. Coding is treated as a separate step that follows design. In that order, coding feels more like translation than synthesis. Students encounter far fewer errors when they do not try to design and code simultaneously. When the output does not match their expectations, they tend to fix the problem using direct manipulation instead of revisiting their measurement and logic. We view this positively; making need not be pure cognition.

Animation is a relatively new feature. Its users so far have been university students with whom we've been engaging in participatory design. They've been making animations, and we've been developing the animation system to support their needs. Figure 7 shows two animations from these students.

In the future, we expect to address several interaction challenges. Debugging is only weakly supported, largely because traditional debuggers are used to pause execution of imperative programs. While Twoville has imperative elements like loops and assignments, individual shapes are described in a declarative manner that can't be meaningfully broken down into sequential steps. Additionally, the canvas is prone to crowding of direct manipulation handles. To combat this, we show only the handle for the property selected by the text cursor. But if that property is assigned within a loop, there may be multiple and possibly many such handles.

References

[1] Christos Chytas et al. "Youth's Perspectives of Computational Design in Making-based Coding Activities". In: *6th FabLearn Europe / MakeEd Conference 2022*. FabLearn Europe / MakeEd 2022. Copenhagen, Denmark: Association for Computing Machinery, 2022. ISBN: 9781450396332. DOI: `10.1145/3535227.3535231`. URL: `https://doi.org/10.1145/3535227.3535231`.

[2] Tamara Anna Efrat, Moran Mizrahi, and Amit Zoran. "The Hybrid Bricolage: Bridging Parametric Design with Craft through Algorithmic Modularity". In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI '16. San Jose, California, USA: Association for Computing Machinery, 2016, pp. 5984–5995. ISBN: 9781450333627. DOI: `10.1145/2858036.2858441`. URL: `https://doi.org/10.1145/2858036.2858441`.

[3] Conal Elliott. "An Embedded Modeling Language Approach to Interactive 3D and Multimedia Animation". In: *IEEE Trans. Softw. Eng.* 25.3 (May 1999), pp. 291–308. ISSN: 0098-5589. DOI: `10.1109/32.798320`. URL: `https://doi.org/10.1109/32.798320`.

[4] Erica Halverson and Kimberly Sheridan. "The Maker Movement in Education". In: *Harvard Educational Review* 84 (Dec. 2014), pp. 495–504. DOI: `10.17763/haer.84.4.34j1g68140382063`.

[5] Jennifer Jacobs and Leah Buechley. “Codeable Objects: Computational Design and Digital Fabrication for Novice Programmers”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery, 2013, pp. 1589–1598. ISBN: 9781450318990. URL: `https://doi.org/10.1145/2470654.2466211`.

[6] G. Johnson. “FlatCAD and FlatLang: Kits by code”. In: *Visual Languages and Human-Centric Computing, 2008. VL/HCC 2008. IEEE Symposium on*. Sept. 2008, pp. 117–120. DOI: `10.1109/VLHCC.2008.4639070`.

[7] Rubaiat Habib Kazi et al. “ChronoFab: Fabricating Motion”. In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI ’16. San Jose, California, USA: Association for Computing Machinery, 2016, pp. 908–918. ISBN: 9781450333627. DOI: `10.1145/2858036.2858138`. URL: `https://doi.org/10.1145/2858036.2858138`.

[8] Jan-Peter Krämer et al. “An empirical study of programming paradigms for animation”. In: *Proceedings of the 9th International Workshop on Cooperative and Human Aspects of Software Engineering*. CHASE ’16. Austin, Texas: Association for Computing Machinery, 2016, pp. 58–61. ISBN: 9781450341554. DOI: `10.1145/2897586.2897597`. URL: `https://doi.org/10.1145/2897586.2897597`.

[9] Thibault Raffaillac, Stéphane Huot, and Stéphane Ducasse. “Turning function calls into animations”. In: *Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems*. EICS ’17. Lisbon, Portugal: Association for Computing Machinery, 2017, pp. 81–86. ISBN: 9781450350839. DOI: `10.1145/3102113.3102134`. URL: `https://doi.org/10.1145/3102113.3102134`.

[10] Jennifer A. Rode et al. “From Computational Thinking to Computational Making”. In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. UbiComp ’15. Osaka, Japan: Association for Computing Machinery, 2015, pp. 239–250. ISBN: 9781450335744. DOI: `10.1145/2750858.2804261`. URL: `https://doi.org/10.1145/2750858.2804261`.

[11] Ben Shneiderman. “Direct manipulation: A step beyond programming languages”. In: *Computer* 16.08 (1983), pp. 57–69.

[12] F. Turbak et al. “Blocks languages for creating tangible artifacts”. In: *Visual Languages and Human-Centric Computing (VL/HCC), 2012 IEEE Symposium on*. Sept. 2012, pp. 137–144. DOI: `10.1109/VLHCC.2012.6344500`.

Plagiarism Detection and Deterrence Using Behavioral Tracking*

Vladislav D. Veksler[1], Bella Z. Veksler[2], and Anuj Khadka[3]
Computer Science & Information Systems
Caldwell University
Caldwell, NJ 07006
[1]vveksler@caldwell.edu [2]bveksler@caldwell.edu [3]akhadka2@caldwell.edu

Abstract

The proliferation of Generative Artificial Intelligence (genAI) tools has intensified the challenge of plagiarism in education, rendering traditional product-based detection methods increasingly ineffective and AI-detection tools prone to inaccuracies. This paper introduces LectureAssign, a Visual Studio Code extension designed for process-based plagiarism detection, offering instructors granular insights into students' assignment creation behaviors. LectureAssign provides a comprehensive behavioral fingerprint of student work, capturing detailed editing and cursor movement events within designated assignment bundles (with no tracking beyond assignment bundles). It features in-IDE analytics and playback functionality, allowing instructors to easily review student work process and product. Beyond qualitative analysis, we demonstrate the application of machine learning to automatically flag potential plagiarism. Our simulations revealed that machine learning models can achieve high accuracy and low false alarm rate when detecting plagiarism in LectureAssign submissions (XGBoost model achieving best results among several models; *F1* = 0.94, Acc = 0.96, FA = 0.03), effectively identifying anomalous patterns indicative of academic dishonesty. Furthermore, a longitudinal study revealed a significant deterrent effect. Requirement of LectureAssign for take-home assignments correlated with improved

*Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

student performance on in-class proctored examinations, suggesting enhanced engagement with course material. We conclude that process-based plagiarism detection tools offer a potential solution for maintaining academic integrity in the genAI era, not only by improving detection capabilities but also by fostering a more genuine and effective learning environment.

1 Introduction

Take-home assignments are an important part of course content in secondary and post-secondary education. Not only are take-home assignments important tools for assessment, they are also essential for overall learning and student growth. Plagiarism in take-home assignments prevents fair assessment and completely bypasses the learning process. The growing ease of plagiarism with the evolving landscape of Generative Artificial Intelligence (genAI) tools such as ChatGPT and Gemini has become a major challenge for today's educators [16, 21]. The use of these tools is extremely tempting for students because they are fast, free to use (unlike contract cheating where students hire others to do their work for them), and because they produce different results each time, making plagiarism far more difficult to detect. In this paper we introduce behavior tracking on take-home assignments as a method for giving insights to instructors as to potential cases of plagiarism, provide insights into how such methods enable instructors to quickly identify plagiarism, suggest statistical methods for automatically flagging potential plagiarism, and examine how well behavior tracking on take-home work might act as a deterrent by examining grade improvements on in-class assessments.

Traditional plagiarism detection methods across all disciplines focus on extrinsic[1] plagiarism detection – comparing submitted student work (product) to other known works [5]. Specifically in Computer Science (CS) there are many popular tools such as JPLAG and MOSS that attempt to detect plagiarism by comparing student product to other student products and to larger code-bases with varying degrees of success [6, 13]. The rise of generative AI, however, renders these traditional methods largely obsolete, irrespective of discipline. Unlike plagiarism, generative AI does not produce verbatim copies that can be directly compared to existing submissions or databases of known works. To address these issues, numerous AI detection tools have emerged, using AI to combat AI [12]. However, AI-detection tools have considerable limitations. Their reliance on superficial stylistic and structural analysis makes them highly susceptible to simple generative AI prompt manipulation. Consequently, they

[1]Some methods also focus on intrinsic stylometry, but these are limited in scope and usage [5].

remain a step behind in the dynamic arms race between AI generation and detection. [12]. Perhaps most alarmingly, such tools produce too many false positives, leading to dire consequences for students as well as to the institutions of education (e.g., [24]). The issue with both the traditional and modern AI-detection methods is that the focus is on analyzing product (what students submit) rather than process (how students worked on their submissions).

Several attempts have been made in the past to help instructors identify cheating on take-home assignments by recording/analyzing the submission-generation process. The general consensus is that process-based plagiarism detection provides instructors with insights that product-based alternatives cannot provide, and is superior in detecting the use of genAI and contract cheating [7, 11, 15, 17, 20]. In Computer Science assignments process recording is usually done through keystroke-level logging via plugins for popular IDE's (Integrated Development Environments) such as ShowYourWork for PyCharm [4, 7], Fluorite for Eclipse [23, 17], and PALG for IntelliJ [10, 11]; or via custom save/compile event version tracking systems (e.g., [15]). There are also tools for process tracking for essay assignments (rather than computer programming assignments), including keyloggers like camdus.io [1, 20] and editing software with version tracking like Google Docs. Many instructors/institutions have begun to require that student assignment submissions must be shared links to Google Docs (rather than standalone documents) because the shared Docs include document version history by default, enabling a peek into the student content-generation process, rather than the mere view of the final product (e.g., [3]). Google Docs version playback extensions that enable instructors to scrub through the content-generation process have begun to gain popularity, with extensions such as DraftBack, WriteHuman, Revision History, GPTZero, and Passed.ai combining for over half a million installations [8].

Given the success of using IDE plugins for keystroke-level behavior tracking for plagiarism detection, we found it surprising that such tools are not more widely used in CS education. The issue, as we have come to learn, is that the tools described above (ShowYourWork, Fluorite, PALG) are custom tools used almost exclusively by their developers, with little documentation or in-IDE analysis tooling. Additionally, since these tools make no distinction between school assignments and private work, they are often viewed as unwanted spyware by the students [19]. To combat these issues we developed a similar plugin – LectureAssign [22] – that (1) is specifically aimed at educational assignment creation and submission, (2) is focused on privacy and is completely inactive outside of specific assignment folders generated by instructors, (3) enables in-IDE analytics and playback of the submission development process, and (4) works with the most popular IDE among our students – Microsoft Visual Studio Code (VSCode; one of the most popular IDE's worldwide, second only to

Microsoft Visual Studio, and still growing in popularity [2]).

The rest of this paper (1) outlines the LectureAssign extension for VSCode, explaining its operational mechanisms and critical behavioral metrics which it tracks, and the summarized analyses it offers to graders, (2) investigates how Machine Learning can enhance plagiarism detection through the behavioral data captured by LectureAssign, and (3) provides a longitudinal analysis of proctored assessments to evaluate LectureAssign's effectiveness in deterring plagiarism, suggesting that the use of this tool for take-home assignments positively impacted student readiness for in-class examinations.

2 LectureAssign Process-Based Plagiarism Detection

The LectureAssign VSCode extension [22] is a platform that enables instructors to create single file assignment bundles that include various metadata (e.g., rich-text (markdown) assignment instructions and due dates) and optional files and libraries, automatically unpacks these bundles for students as working folders, tracks student progress as they work on their assignments (creating and editing files), creates single file submission bundles, unpacks submissions for graders, and displays student product work along with analyses and playback of student work process. This extension does not track any work done in VSCode outside of the specific assignment folder, and clearly indicates when it is active and recording, aiming to avoid some of the privacy issues found in similar plugins [19]. This extension is file type agnostic, allowing assignments/submissions to include files of any type, and tracking student work for any text-based files, including plain text, markup (e.g., markdown, $\LaTeX$), or any computer programming code using any programming language.

From a student's perspective, LectureAssign simplifies the process of creating and submitting assignments. Students can download the assignment archive from the school's preferred LMS (Learning Management System), click to automatically open in a new VSCode workspace, work within their familiar IDE, and generate a submission file with one click. Assignment title, due-date, instructions, and attachments are all listed in the included manifest, allowing students to remain in the IDE without having to log back in to the LMS to view these details.

From the instructor's perspective, the ability to create the entire assignment in the IDE reduces some LMS-specific overhead, and enables one-click unpacking of student multi-file submissions directly in the IDE. Most relevant to the current work, this tool enables identification of potential plagiarism at a glance. LectureAssign submissions include file modification metadata and hidden log files that include several key behavioral metrics:

- *Text change events:* millisecond-level timestamps, location of change, number of deleted characters, and inserted characters

- *Cursor movement events:* millisecond-level timestamps, current line number, current cursor index, current cell (if Jupyter notebook), number of characters in file, number of lines in file
- Periodic checksum to ensure log file integrity

These multi-faceted features, when combined, create a highly descriptive behavioral fingerprint of each student's work. For instance, a student who copies and pastes a large block of code might exhibit a sudden, dramatic increase in file size, followed by minimal subsequent keystrokes and very localized cursor movements confined primarily to that newly pasted section. This contrasts sharply with the gradual, organic, and non-linear development typically seen in original work. Similarly, a student typing out code from an external source (rather than copy-pasting it) will display a consistent linear increase in file size with few cursor movements or time lags that would be seen in natural code development and debugging.

One of the primary views of student work in the LectureAssign extension is a chart displaying file-size changes and cursor movements over time. Graphs from sample students' submissions may be seen in Figure 1, depicting various file size growth and cursor movement patterns derived from LectureAssign logs. Top-left subplot in Figure 1 displays a normal coding and debugging pattern, where file size and cursor movements show no obvious pattern. Top-right is a submission where a single copy-paste was followed by student edits (playback revealed that in this case the student pasted the entire solution and then went through it just to add comments). Bottom-left subplot is a Jupyter notebook submission, where the student simply copy-pasted code into eight notebook cells, one after the other, then moved their cursor between cells a few times, and submitted the work with no further edits. Bottom-right depicts a student who was aware that copy-pasting large blocks of text significantly increased the risk of detection, so they copied code from an external source by typing it out one character at a time.

These graphs provide graders with excellent insight, but it should be noted that such data should be interpreted in context. For example, experienced students doing simple coding assignments may end up having size/cursor charts looking much like that in the bottom right of Figure 1 (though this is highly unlikely for more complex assignments). Even large copy-pastes, such as that depicted in the top-right of Figure 1 may not always be indicative of cheating, as some assignments require students to start with non-empty files, or to copy instructions or preambles into their files prior to start.

The most insightful feature of LectureAssign is playback of student work. Instructors can effectively rewind and replay the entire student coding/writing session. Playback offers invaluable insights into the student's thought process,

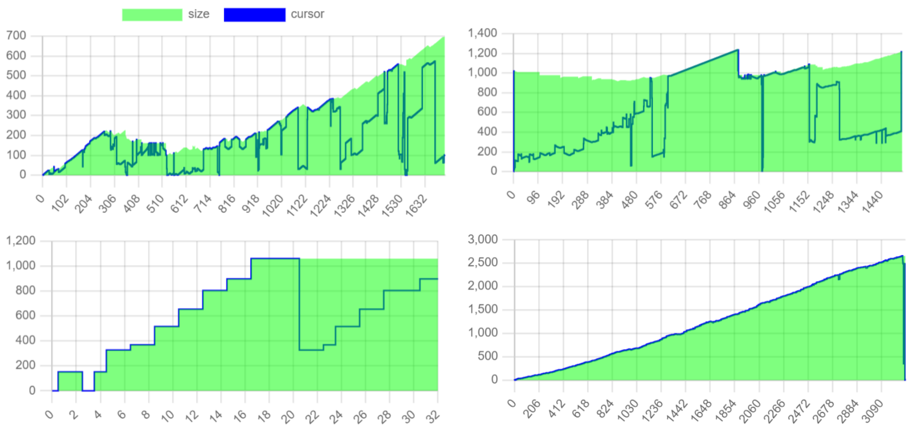


Figure 1: File size and cursor position over time in 4 submissions. The x-axis for all plots indicates event number in time, and the y-axis is the character number. Blue line indicates cursor position and green fill is the file size. Top-left subplot is a normal coding and debugging pattern, the other three subplots are suspected cheaters.

debugging attempts, and any suspicious patterns indicative of plagiarism.

Again, we want to underscore that neither the file-size/cursor charts nor playback is fully indicative of whether a student is cheating or not. These features provide instructors/graders with important information, but should always be taken in context. Additionally, there are several features still missing from LectureAssign that we believe would make it a better tool for plagiarism detection and for assignment review in general. LectureAssign does not yet provide any time-between-keystrokes or cross-assignment comparison analyses (features in-progress), and does not capture run/compile events (these are very difficult to capture in a consistent and non-invasive way).

One other feature that may make LectureAssign a more useful tool for graders is automatic flagging of potentially plagiarized work. The following section describes efforts in generating potential methods for automatically detecting plagiarism from LectureAssign log files using Machine Learning.

3 Predicting Plagiarism via Machine Learning

Prior studies have demonstrated some promise in applying Machine Learning (ML) to predict which assignments may include plagiarism based on the data collected via the LectureAssign VSCode extension. Sharma and Veksler [18] utilized a Random Forest Classifier, achieving an overall accuracy of 0.88 and a false alarm rate of 0.00 (meaning no original submissions were falsely flagged) in identifying non-plagiarized work. While the hit rate (i.e., recall) for plagiarized cases was only 0.67, this initial study effectively established a clear correlation between high log activity (indicating engagement) and original work versus notably lower activity in plagiarized assignments. Khatri, Khadka, and Veksler [9] explored how Logistic Regression, Decision Tree, and Neural Network models would fare in plagiarism detection. Their findings suggest that Neural Networks can provide better results than the other tested methods, achieving overall test accuracy of 0.96, a false alarm rate of 0.00, and a hit rate of 0.92.

To be clear, the studies above employed limited datasets (2 simpler assignments, 86 submissions used in [18] and 133 in [9]), employed only a single rater in each study, and were preliminary in nature. In the current work we use a larger dataset (4 assignments varying in complexity, 453 total submissions), with three of the four assignments having at least two human raters per submission for greater inter-rater reliability. While this dataset remains flawed due to a heavy reliance on a single rater's judgment regarding potential plagiarism, we believe the results to be informative.

3.1 Methods

Current study utilized homework submissions which were completed in VS Code with the LectureAssign extension turned on during the Fall 2024 introductory programming course (CS 195 Computer Programming I). There was a total of 102 students in the class and 453 total submissions across 4 separate assignments (N for each assignment was: 107, 94, 134, 118, respectively, resubmissions allowed). The assignments chosen for this analysis were stand-alone and did not rely on previous code.

3.1.1 Submission Ratings

A subset of submissions from this dataset was independently rated by three trained raters. The objective of this rating process was to determine if each submission could be classified as plagiarized. Raters made their judgments based on the playback file generated by LectureAssign for each submission and classified the submission as either 'good', 'suspect', or 'unsure'. Only submissions with complete ratings of either 'good' or 'suspect' from all three raters were included in the inter-rater reliability analysis ($N = 94$).

Inter-rater reliability was assessed using the Intraclass Correlation Coefficient (ICC). A two-way random effects model for absolute agreement, averaging 3 raters (ICC[2,3]) was selected, as raters were considered a random sample and the average of their scores would be used in subsequent analyses. All calculations were performed using Python's pingouin library (version 0.5.5).

The inter-rater reliability for the 94 submissions, assessed by 3 raters, was good (ICC$[2,3]$ = 0.899, 95% CI $[0.860, 0.930]$), with the test for ICC significantly differing from zero yielding $F(93, 186) = 10.53$, $p < .001$.

Since the ICC among the raters was reasonably good for the subset of submissions that had ratings across all raters, we extended this to the entire dataset to establish a robust set of ground truth labels to use in supervised ML algorithms and statistical analysis. Recognizing that not all submissions received ratings from every rater, a specific protocol was followed to derive a definitive classification for each submission:

- If a submission was rated by only one rater, that rater's determination was adopted as the definitive classification.
- For submissions evaluated by multiple raters, a definitive classification was reached through a majority vote; in cases of ties, the submission was classified as 'unsure'.

Using the above approach, we had 259 submissions classified as 'good', 138 submission classified as 'suspect', and 56 marked as 'unsure'.

3.1.2 Feature Selection

To capture the nuances of student code development within LectureAssign, we extracted a comprehensive set of behavioral metrics (features) directly from raw log files comprised of text change events and cursor events. This feature engineering process was specifically designed to distill complex behavioral patterns into quantifiable inputs for each submission, ultimately enabling their use in various ML algorithms for predictive modeling and insight.

The set of features extracted are presented in Table 1. Table columns indicate which summary statistics were used for each feature. **Value** refers to actual value for each submission, **N** refers to number of times the particular feature occurred in each submission, and **Summary Statistics** included mean, median, stdev, maximum value, and 90th, 95th, and 99th% percentile for the feature. This resulted in 72 features for each submission, to be used as input vectors for ML.

3.1.3 Models

We compared the following ML classifier performance on predicting when human graders might mark a potential cheating attempt: Logistic Regression, Decision Tree, Random Forest, Deep Neural Networks, and XGBoost.[2] For

[2] To replicate our simulations use Python with scikit-learn library for Logistic Regression, Decision Tree, and Random Forest classifiers, xgboost library for XGBoost, and tensorflow

Table 1: Full set of features extracted for ML. Summary Statistics include Mean, Median, StdDev, Max, and 90th/95th/99th Percentiles.

Feature	Value	N	Summary Statistics
File Size	✓		
Duration	✓		
Lines		✓	
Cursor Events		✓	
Text Change Events		✓	
Multi-line additions		✓	
Cursor Events at EOF		✓	
Cursor Events at EOF %	✓		
Time Δ Text Change Events		✓	✓
Time Δ Cursor Events		✓	✓
Cursor Moves Forward		✓	✓
Cursor Moves Backward		✓	✓
Line Moves Up		✓	✓
Line Moves Down		✓	✓
Insertions		✓	✓
Deletions		✓	✓

Deep Neural Networks we used a 4-layer (NN4; in-128-64-out), a 5-layer (NN5; in-128-64-64-out), and a 6-layer (NN6; in-128-128-64-64-out) dense layer networks. For each model run, the model was trained on 75% of the submissions, and then tested on the remaining 25% (stratified to ensure that proportional numbers of signal remained in both train and test datasets). All reported Hit Rates (HR), False Alarm rates (FA), overall Accuracy (Acc), and *F1* scores are calculated from test case predictions only.

Based on our reading of the literature (e.g., [14]) and some preliminary simulations we found XGBoost to be the best overall classifier with very fast run times. For this reason when it came to the task of selecting the 'best' features for classification we used a genetic algorithm (GA) with XGBoost *F1* score as the fitness function.

We ran the GA (50 generations, 100 individuals per generation) twice – once to identify 7 best features to produce the highest *F1* scores, and once to identify any best number of features (best set included 40 features).

3.2 Results

Classification prediction results are displayed in Table 2. Results revealed that XGBoost has the best signal detection among all models when using just the 7 best features (*F1* = 0.92, HR = 0.86, FA = 0.00, Acc = 0.95). Results improve slightly when using the best 40 features (*F1* = 0.94, HR = 0.94, FA = 0.03, Acc = 0.96). When all 72 features are used, Logistic Regression outperforms XGBoost (*F1* = 0.82), but no model does as well as XGBoost across all simula-

for Deep Neural Networks. Use random seed 42 for precise duplication.

tions. For comparison, baseline performance (random classification) is always 50% (*F1* = 0.41).

It is the case that since the GA optimized the best features for XGBoost, this model has an unfair advantage over other models when using just the best 7 or best 40 features. Indeed, when we reran the GA using other classifiers in the GA fitness functions, we found better F1 scores for those classifiers (NN6 was not included as it was prohibitively slow). When finding 7 best features, Random Forest achieved the same F1 score (0.92) as XGBoost, all other scores were lower. When finding any numbers of best features, no scores were as high as XGBoost.

Table 2: ML Simulation Results

	All Features				Best 7 Features				Best 40 Features			
	F1	HR	FA	Acc	F1	HR	FA	Acc	F1	HR	FA	Acc
XG	0.80	0.74	0.06	0.87	**0.92**	0.86	0.00	0.95	**0.94**	0.94	0.03	0.96
LR	**0.82**	0.71	0.02	0.89	0.69	0.54	0.02	0.83	0.82	0.71	0.02	0.89
RF	0.77	0.69	0.05	0.86	0.84	0.77	0.03	0.90	0.83	0.77	0.05	0.89
DT	0.74	0.69	0.09	0.83	0.82	0.80	0.08	0.88	0.76	0.77	0.14	0.83
NN4	0.70	0.69	0.15	0.79	0.82	0.77	0.06	0.88	0.74	0.69	0.09	0.83
NN5	0.72	0.69	0.12	0.81	0.81	0.74	0.05	0.88	0.77	0.71	0.08	0.85
NN6	0.69	0.69	0.17	0.78	0.84	0.74	0.02	0.90	0.75	0.71	0.11	0.83

4 Process-tracking as a Deterrent and Improved Student Outcomes

Beyond its utility in flagging potential plagiarism, work-tracking tools like LectureAssign can serve as a deterrent to academic dishonesty, ultimately fostering improved learning outcomes. The very knowledge among students that their assignment work process is being recorded can subtly, yet profoundly, influence their behavior, leading to more time spent on take-home assignments, better learning, and better grade outcomes on follow-up examinations. This section presents compelling evidence for this hypothesis based on longitudinal data from an introductory Computer Science course. Specifically, we examine quiz, midterm, and final exam grade distributions from the last three consecutive years of all Computer Programming I sections taught by first author at Caldwell University, demonstrating how the systematic use of LectureAssign correlates with improved student performance on proctored examinations.

Each academic year presents a distinct pedagogical and technological context. Fall 2022 (Baseline Year) serves as our pre-intervention baseline. It represents the period prior to genAI's widespread adoption on college campuses (N = 47). In the Fall of 2023 (Cheating Year) genAI became widely accessible and was frequently utilized by college students ($N = 70$). Fall 2024 (LectureAssign Implementation Year) is the pivotal year where LectureAssign was systematically integrated into the Computer Programming I course. Students were explicitly informed at the outset of the semester that their coding sessions on assignments submitted via the extension would be recorded and potentially reviewed for academic integrity purposes ($N = 102$). Withdrawals were not included in the analysis.

The overarching hypothesis was that if students were effectively deterred from cheating on their take-home assignments, they would be compelled to engage more deeply with the material, leading to a stronger conceptual understanding and, consequently, better performance on proctored examinations where external aid was strictly prohibited.

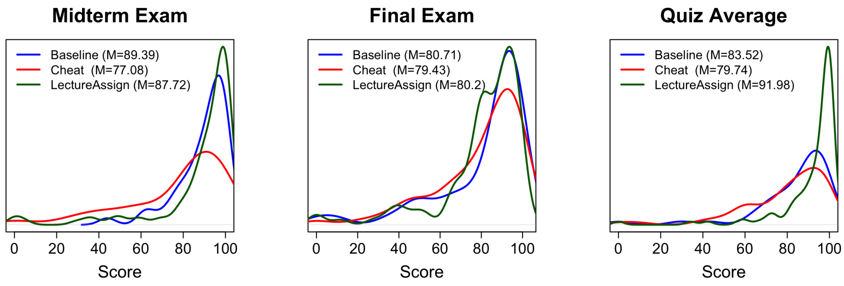


Figure 2: Grades for Midterm Exam, Final Exam, and Quiz Average from the Fall semester of the year prior to the advent of genAI (Baseline), the year when genAI was popular for cheating, but before LectureAssign was introduced (Cheat), and the year when LectureAssign was used.

Analysis of the aggregated grade distributions across these three years reveals a compelling trend (see Figure 2 and Table 3). Our findings indicate that the quiz and exam scores in Fall 2024 were not only better than in the 'cheating year' of Fall 2023, but in some cases even surpassed the baseline performance observed in Fall 2022. Grades on both the midterm and the quizzes for Fall 2024 (when LectureAssign was used) were higher than those for Fall 2022 (Baseline) and far higher than those for Fall 2023 (Cheat).

These results suggest that the systematic use of LectureAssign may have served as an effective deterrent, prompting students to invest genuine effort in learning and problem-solving independently on their take-home assignments. This deeper, authentic engagement, in turn, facilitated a more robust conceptual understanding of programming principles, which then directly translated into superior performance on proctored examinations. This positive shift in

Table 3: ANOVA and Post-hoc Tukey HSD Results for Multiple Dependent Variables

Source	**df**	**F-value**	**p-value**[a]	**Mean Diff. (95% CI)**
Dependent Variable: Quiz Average				
Year	2, 216	10.65	$p < .001$	
Post-hoc Tukey HSD:				
Baseline vs. Cheat			$p = 0.495$	-3.78 (-11.66, 4.09)
Baseline vs. LectureAssign			$p < .05$	8.46 (1.10, 15.82)
Cheat vs. LectureAssign			$p < .001$	12.24 (5.76, 18.72)
Dependent Variable: Midterm Exam				
Year	2, 215	6.91	$p < .01$	
Post-hoc Tukey HSD:				
Baseline vs. Cheat			$p < .01$	-12.32 (-21.60, -3.03)
Baseline vs. LectureAssign			$p = 0.892$	-1.67 (-10.33, 6.98)
Cheat vs. LectureAssign			$p < .01$	10.65 (3.00, 18.30)
Dependent Variable: Final Exam				
Year	2, 214	0.05	$p = 0.948$	
No significant main effect, Tukey HSD not performed.				

[a]For post-hoc comparisons, p-values are adjusted using Tukey's HSD method.

performance indicates that LectureAssign ultimately aided in improving overall learning outcomes in the course.

The observed improvement in proctored exam performance, following the widespread availability of AI tools and then the strategic introduction of LectureAssign, provides compelling empirical evidence for its positive impact on academic integrity and student learning. However, further controlled studies and rigorous statistical validation are necessary to definitively establish causality. Although this course was taught by the same instructor using the same course content in the same settings during all three semesters, there may be many confounds that we did not consider.

5 Summary and Conclusions

The pervasive challenge of plagiarism, particularly exacerbated by the advent of Generative Artificial Intelligence (genAI) tools like ChatGPT and Gemini, necessitates innovative detection and deterrence mechanisms in educational settings. Traditional product-based plagiarism detection methods, which compare submitted work to other known works, are increasingly rendered obsolete by genAI's ability to produce unique outputs. Furthermore, modern AI-detection tools, while attempting to combat AI with AI, are often flawed, susceptible to prompt manipulation, and prone to false positives, leading to potentially severe consequences for students and institutions alike.

In response to these limitations, process-based plagiarism detection has emerged as an alternative, offering instructors insights into the student's submission generation process rather than just the final product. IDE plugins used for recording work process (e.g., Show Your Work for PyCharm, Fluorite for Eclipse, PALG for IntelliJ) have demonstrated their efficacy in detecting the use of genAI and contract cheating. However, existing IDE plugins often suffer from limitations such as limited documentation, lack of in-IDE analysis tools, and privacy concerns due to their "spyware" perception among students.

To address these shortcomings, we developed LectureAssign, a VSCode extension specifically designed for educational assignment creation and submission. LectureAssign prioritizes student privacy by being active only within designated assignment folders and clearly indicating when it is recording. It provides in-IDE analytics and playback of the submission development process, offering a comprehensive behavioral fingerprint for each student's work. Key behavior metrics captured include text change events (timestamps, deleted/inserted characters), cursor movement events (timestamps, detailed position), and file size changes. LectureAssign provides charts displaying file-size changes and cursor movements, enabling instructors to identify anomalous patterns indicative of plagiarism, such as sudden, dramatic increases in file size without corresponding keystrokes (copy-pasting) or consistent linear increases with minimal deliberation time (typing from an external source) at a glance. Additionally this tool enables granular playback of student work, allowing instructors to rewind and replay every text edit with millisecond-level precision, thereby offering a powerful qualitative dimension to plagiarism detection. Importantly – it is crucial to interpret process data within context, as even seemingly suspicious patterns may have legitimate explanations.

Future enhancements to LectureAssign aim to further improve its utility. These include capturing run/compile events to provide insights into student thought processes and debugging attempts, displaying time-gap analyses between keystrokes to identify unusually consistent and rapid typing patterns, and incorporating cross-submission comparisons for early flagging of similar content and statistical distributions of various metrics.

Our research also explored the application of Machine Learning (ML) to automatically flag potential plagiarism from LectureAssign log files. Our modeling efforts compared Logistic Regression, Decision Tree, Random Forest, Deep Neural Networks, and XGBoost classifiers. XGBoost consistently demonstrated the best overall signal detection. Using the best features found by a genetic algorithm, XGBoost achieved an *F1* score of 0.94 (Acc = 0.96, FA =

0.03, HR = 0.94).

Beyond plagiarism detection, LectureAssign also serves as a deterrent to academic dishonesty and a catalyst for improved learning outcomes. Our longitudinal study in an introductory Computer Science course demonstrated a compelling correlation between the systematic use of LectureAssign and enhanced student performance on proctored examinations. Grade distributions for quizzes and midterm exams in Fall 2024 (LectureAssign implementation year) were significantly higher than in Fall 2023 (genAI cheating year) and, in some cases, even surpassed the Fall 2022 baseline (pre-genAI). This suggests that the knowledge of being recorded compelled students to engage more deeply with the course material, leading to a stronger conceptual understanding that translated into better exam performance. While these findings are compelling, further controlled studies and rigorous statistical validation are necessary to definitively establish causality.

In conclusion, process-based plagiarism detection presents a robust and promising approach to addressing the evolving challenges of plagiarism in education, particularly in the age of genAI. By focusing on student behavioral processes during assignment creation, providing intuitive visualization and playback tools for instructors, and offering the potential for automated detection through machine learning, LectureAssign not only aids in identifying potential academic dishonesty but also appears to foster a more authentic and engaged learning environment, ultimately leading to improved student outcomes.

6 Acknowledgements

This work would not be possible without significant contributions from volunteer researchers Keepa Maharjan and Krenjila Sharma. Thank you both!

References

[1] `https://cadmus.io/`. (Visited on 05/2025).

[2] Pierre Carbonnelle. `https://pypl.github.io/IDE.html`. (Visited on 05/2025).

[3] East Central College. *Using Google Docs to Detect Students' AI Usage*. `https://www.eastcentral.edu/free/ai-faculty-resources/using-google-docs-to-detect-ai/`.

[4] John Edwards. *ShowYourWork*. `https://plugins.jetbrains.com/plugin/18353-showyourwork`. (Visited on 05/2025).

[5] Tomáš Foltýnek, Norman Meuschke, and Bela Gipp. "Academic plagiarism detection: a systematic literature review". In: *ACM Computing Surveys (CSUR)* 52.6 (2019), pp. 1–42.

[6] Kaio Pablo Gomes and Simone Nasser Matos. "Detection of programming plagiarism in computing education: A systematic mapping study". In: *Simpósio Brasileiro de Informática na Educação (SBIE)* (2020), pp. 1633–1642.

[7] Kaden Hart, Chad Mano, and John Edwards. “Plagiarism deterrence in cs1 through keystroke data”. In: *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1.* 2023, pp. 493–499.

[8] Google Inc. *Chrome Web Store*. `https://chromewebstore.google.com/`. (Visited on 05/2025).

[9] Khagendra Khatri, Shadip Khadka, and Vladislav D Veksler. “ML for Plagiarism Detection in Coding Assignments”. In: *28th annual conference of the Consortium for Computing Sciences in Colleges, Northeastern Region (CCSCNE).* CCSCNE 2024. Albany, NY, USA, 2024.

[10] Rene Kütt. *PALG: Programming Activity Log Generator.* `https://plugins.jetbrains.com/plugin/21567-palg-programming-activity-log-generator`. (Visited on 05/2025).

[11] Rene Kütt. “Plagiarism Detection Tool Based on Programming Activity Logs”. In: *2024 IEEE Global Engineering Education Conference (EDUCON).* IEEE. 2024, pp. 1–7.

[12] Muhammad Abid Malik and Amjad Islam Amjad. “AI vs AI: How effective are Turnitin, ZeroGPT, GPTZero, and Writer AI in detecting text generated by ChatGPT, Perplexity, and Gemini?” In: *Journal of Applied Learning and Teaching* 8.1 (2025), pp. 91–101.

[13] Ramesh R Naik, Maheshkumar B Landge, C Namrata Mahender, et al. “A review on plagiarism detection tools”. In: *International Journal of Computer Applications* 125.11 (2015), pp. 16–22.

[14] Didrik Nielsen. “Tree boosting with xgboost-why does xgboost win "every" machine learning competition?” MA thesis. NTNU, 2016.

[15] Gustavo Rodriguez-Rivera et al. “Tracking large class projects in real-time using fine-grained source control”. In: *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education-Volume 1.* 2022, pp. 565–570.

[16] L’ubica Rumanovská et al. “Plagiarism in the academic environment”. In: *Societies* 14.7 (2024), p. 128.

[17] Johannes Schneider et al. “Detecting plagiarism based on the creation process”. In: *IEEE Transactions on Learning Technologies* 11.3 (2017), pp. 348–361.

[18] Krenjila Sharma and Vladislav D Veksler. “Plagiarism detection in Computer Programming assignments using behavior tracking”. In: *29th annual conference of the Consortium for Computing Sciences in Colleges, Northeastern Region (CCSCNE).* CCSCNE 2025. Brockport, NY, USA, 2025.

[19] Caleb Syndergaard. “A Review of Student Attitudes Towards Keystroke Logging and Plagiarism Detection in Introductory Computer Science Courses”. PhD thesis. Utah State University, 2024.

[20] Kelly Trezise et al. “Detecting contract cheating using learning analytics”. In: *Journal of Learning Analytics* 6.3 (2019), pp. 90–104.

[21] Levent Uzun. “ChatGPT and academic integrity concerns: Detecting artificial intelligence generated content”. In: *Language Education and Technology* 3.1 (2023).

[22] Vladislav D Veksler. *LectureAssign Visual Studio Code Plugin.* `https://marketplace.visualstudio.com/items?itemName=LectureDot.lecture-assign`. (Visited on 05/2025).

[23] YoungSeok Yoon and Brad A. Myers. *Fluorite Eclipse IDE Plugin.* `https://www.cs.cmu.edu/~fluorite/`. (Visited on 05/2025).

[24] Brian Zahn. *Yale Student Suing Over Accusation of Improper AI Use.* `https://www.govtech.com/education/higher-ed/yale-student-suing-over-accusation-of-improper-ai-use`. 2025.

Reviewers — 2025 CCSC Eastern Conference

Prof. Alvin Chris .... Furman University, Greenville, SC, United States
Dr. Anewalt Karen .... University of Mary Washington, Fredericksburg, VA, United States
Mr. Braught Grant .... Dickinson College, Carlisle, PA, United States
Prof. Carter Karla .... Bellevue University, Bellevue, NE, United States
Dr. Childs Dawn .... Marymount University, Cypress, CA, United States
Dr. Conrad Sue .... Marymount University, Fairfax, VA, United States
Ms. Costa Mônica Isabel Teixeira .. SFC–IPCB, Castelo Branco, PT, United States
Dr. D'Antonio Lawrence .. Ramapo College, Dobbs Ferry, NY, United States
Prof. Dougherty John ...... Haverford College, Haverford, PA, United States
Dr. Finlayson Ian . The University of Mary Washington, Fredericksburg, VA, United States
Dr. Flinn Michael . Frostburg State University, Frostburg, MD, United States
Dr. Freedman Reva .. Northern Illinois University, DeKalb, IL, United States
Dr. Gaspar Alessio .. University of South Florida, Tampa, FL, United States
Dr. Grinberg Gregory Montgomery College, Gaithersburg, MD, United States
Dr. Highley Timothy ... La Salle University, Philadelphia, PA, United States
Dr. Hovemeyer David .... Johns Hopkins University, Baltimore, MD, United States
Dr. Lamprecht Ruth Mount St. Mary's University, Emmitsburg, MD, United States
Dr. Lee Ingyu .... Troy University, Troy, AL, United States
Prof. Lindoo Edward .... CCSC Treasurer, STUART, FL, United States
Dr. McManus John ... Randolph-Macon College, Ashland, VA, United States
Dr. Ngo Linh .. West Chester University of Pennsylvania, West Chester, PA, United States
Dr. Nuakoh Emmanuel Borkor .... North Carolina A&T State University, Shrewbury, MA, United States
Dr. Poger Sofya ...... Felician University, Woodland Park, NJ, United States
Dr. Rosiene Carolyn Pe University of Portland, Heartford, CT, United States
Ms. Sanders Patrick ...... Laurel Ridge Community College, Wythevile, VA, United States
Dr. Stedman Jacob Daniel .... AdvancedPCB, Osseo, MN, United States
Prof. Teresco James D. .... Siena University, Loudonville, NY, United States
Dr. Tu Junyi .... Salisbury University, Salisbury, MD, United States
Prof. Wright John .... Juniata College, Huntingdon, PA, United States