The Journal of Computing Sciences in Colleges

Papers of the 34th Annual CCSC Rocky Mountain Conference

October 17th-18th, 2025
Utah Valley University
Orem, UT

Abbas Attarwala, Editor
California State University, Chico

Pam Smallwood, Regional Editor
Regis University

Volume 41, Number 2

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.

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.

Table of Contents

The Consortium for Computing Sciences in Colleges Board of Directors **7**

CCSC National Partners **9**

Welcome to the 2025 CCSC Rocky Mountain Conference **10**

Regional Committees — 2025 CCSC Rocky Mountain Region **11**

Transformations in Computer Science Programs to Integrate Quantum-AI **12**
Bhaskar Sinha, Mohammad Amin, and Pradip Peter Dey, National University

How to Install and Use T-Pot Honeypot VM in a Defensive Cybersecurity Course **16**
Mohamed Lotfy, Utah Valley University

Curd: A Curriculum Design Tool **19**
Peter Aldous, Utah Valley University

A Game Design Approach to CS2: Take Two **29**
Daniel Cliburn, Jacob Bejarano, Sai Sravan Meduri, and Srikar Podicheti, University of the Pacific

AI as a Study Aid: Using Artificial Intelligence to Learn Difficult Concepts in Computer Science **38**
Heather Jackson and Victor Clincy, Kennesaw State University

Teaching AI Ethics through a Reading Project: A Structured and Reflective Approach **44**
Jingsai Liang, Westminster University and Xi Chen, Utah Valley University

A Comparative Study of Architectural Patterns in Flutter: A Case Study and Teaching Framework for Undergraduate Mobile Development **53**
Madhav Neupane and Liqiang Zhang, Indiana University South Bend

Engaging with the Wiphala: A Code-Generation Hardened CS1 Final **65**

Calvin Deutschbein and Shouvik Ahmed Antu, Willamette University

Using LLMs for the Browse Task: An Exploratory Document Clustering **76**

Daniel McDonald and John E. Anderson, Utah Valley University

Artificial Intelligence in Cybersecurity: Applications, Challenges, and Future Directions **87**

Diego Sigaran, Jeff Richards, Bryce Shepard, Xi Chen, and Imtiaz Parvez, Utah Valley University; Kazi Arafat, Atish Dipankar University of Science & Technology; Iftekhar Hossain, Dhaka University of Engineering & Technology

A Case for Conveying Design Rationale in Curricular Computing Reports **105**

Richard Blumenthal, Regis University

Breaking Silos: An Agile Approach to Interdisciplinary Software Engineering Pedagogy **120**

Lynn Roy Thackeray and Susan L Thackeray, Utah Valley University

Best Practices for Inclusive Teaching in Computer Science **134**

Thyago Mota, Metropolitan State University of Denver

Reducing Math Anxiety Through MATLAB: A Case Study in Mechatronics Engineering Technology Education **147**

Rawan Al-Nsour, Utah Valley University

Cyber Students' Perceptions of Specifications Grading **156**

Mathew J. Heath Van Horn and Christopher Warner, Embry-Riddle Aeronautical University

Prompt-Driven Cybersecurity Education: Exploring LLM-Based Simulation of Security Operations Center Activities **166**

Basil Hamdan, Utah Valley University

Generative AI and its Use in Defensive Cybersecurity Courses **177**

Mohamed Lotfy, Utah Valley University and Ed Lindoo, Regis University

Vast, Low-cost Honeypot Architecture **195**
David Charles Loper and Prakash Ranganathan, University of North Dakota and Jenny Valentine Nehring, Utah Valley University

From Confusion to Clarity in Introductory Computer Science through Intelligent Discussion Board Systems **211**
Tirth Shah, Saloni Tilekar, Kalpit Sharma, and Abbas Attarwala, California State University, Chico and Ed Lindoo, Regis University

Voice-Driven Conversational Mixed Reality for Linear Algebra in Computer Science Education Using an AI Avatar Tutor **223**
Sheldon Henriques and Abbas Attarwala, California State University, Chico and Ed Lindoo, Regis University

Reviewers — 2025 CCSC Rocky Mountain Conference **237**

The Consortium for Computing Sciences in Colleges Board of Directors

Following is a listing of the contact information for the members of the Board of Directors and the Officers of the Consortium for Computing Sciences in Colleges (along with the years of expiration of their terms), as well as members serving CCSC:

Bryan Dixon, President (2026), bcdixon@csuchico.edu, Computer Science Department, California State University Chico, Chico, CA 95929.

Shereen Khoja, Vice President/President-Elect (2026), shereen@pacificu.edu, Computer Science, Pacific University, Forest Grove, OR 97116.

Abbas Attarwala, Publications Chair (2027), aattarwala@csuchico.edu, Department of Computer Science, California State University Chico, Chico, CA 95929.

Ed Lindoo, Treasurer (2026), elindoo@regis.edu, Anderson College of Business and Computing, Regis University, Denver, CO 80221.

Haiyan Cheng, Membership Secretary (2028), hcheng@willamette.edu, Department of Computer Science, Willamette University, Salem, OR 97301.

Judy Mullins, Central Plains Representative (2026), mullinsj@umkc.edu, University of Missouri-Kansas City, Kansas City, MO (retired).

Michael Flinn, Eastern Representative (2026), mflinn@frostburg.edu, Department of Computer Science & Information Technologies, Frostburg State University, Frostburg, MD 21532.

Gabe Ferrer, Midsouth Representative (2028), ferrer@hendrix.edu, Department of Computer Science, Hendrix College, Conway, AR 72032.

David Largent, Midwest Representative (2026), dllargent@bsu.edu, Department of Computer Science, Ball State University, Muncie, IN 47306.

Michael Gousie, Northeastern Representative (2028), gousie_mike@wheatoncollege.edu, Computer Science Department, Wheaton College, Norton, MA 02766.

Ben Tribelhorn, Northwestern Representative (2027), tribelhb@up.edu, School of Engineering, University of Portland, Portland, OR 97203.

Mohamed Lotfy, Rocky Mountain Representative (2028), MohamedL@uvu.edu, Information Systems & Technology Department, College of Engineering & Technology, Utah Valley University, Orem, UT 84058.

Mika Morgan, South Central Representative (2027), mika.morgan@msutexas.edu, Department of Computer Science, Midwestern State University, Wichita Falls, TX 76308.

Karen Works, Southeastern Representative (2027), keworks@pc.fsu.edu, Department of Computer Science, Florida State University Panama City, Panama City, FL 32405.

Michael Shindler, Southwestern Representative (2026), mikes@uci.edu, Computer Science Department, UC Irvine, Irvine, CA 92697.

Serving the CCSC: These members are serving in positions as indicated:
Bin "Crystal" Peng, Associate Editor, bin.peng@park.edu, Department of Computer Science and Information Systems, Park University, Parkville, MO 64152.
Lucas Cordova, Associate Treasurer, lpcordova@willamette.edu, Department of Computer Science, Willamette University, Salem, OR 97301.
George Dimitoglou, Comptroller, dimitoglou@hood.edu, Department of Computer Science, Hood College, Frederick, MD 21701.
Megan Thomas, Membership System Administrator, mthomas@cs.csustan.edu, Department of Computer Science, California State University Stanislaus, Turlock, CA 95382.
Karina Assiter, National Partners Chair, KarinaAssiter@landmark.edu, Landmark College, Putney, VT 05346.
Ed Lindoo, UPE Liaison, elindoo@regis.edu, Anderson College of Business and Computing, Regis University, Denver, CO 80221.
Deborah Hwang, Webmaster, hwangdjh@acm.org.

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.

Gold Level Partner

Rephactor
ACM2Y
Blossoms

Welcome to the 2025 CCSC Rocky Mountain Conference

Welcome to the 2025 CCSC Rocky Mountain Conference. Welcome to the 34th annual conference of the Rocky Mountain (RM) Region of the Consortium for Computing Sciences in Colleges. The CCSC RM region board members are grateful for the authors, presenters, speakers, attendees, and students participating in this year's conference.

This year we received 26 paper submissions on a variety of topics, of which 18 papers were accepted for presentation at the conference. Multiple reviewers, using a double-blind paper review process, reviewed all submitted papers for the conference. The review process resulted in an acceptance rate of 69%. In addition to the paper presentations, there are two peer-reviewed tutorials/-workshops and four posters. We truly appreciate the time and effort put forth into the reviewing process by all the reviewers. Without their dedicated effort, none of this would be possible. A special thank you goes to the Submission chairs Dr. Karina Assiter and Dr. Mohamed Lotfy.

The CCSC RM region board would like to thank our national gold-level partners Rephactor, ACM2Y, BlossomsAI and the Association for Computing Machinery in cooperation with SIGCSE.

We hope you enjoy the conference and take the opportunity to interact with your colleagues and leave both enthused and motivated. As you plan your scholarly work for the coming year, we invite you to submit a paper, poster, workshop, tutorial, or panel for a future CCSC RM region conference, or to serve as a reviewer or on the CCSC RM region board. Please encourage your colleagues and students to participate in future CCSC RM region conferences.

Michael E Leverington, PhD
Northern Arizona University
Conference Chair

2025 CCSC Rocky Mountain Conference Committee

2025 CCSC Rocky Mountain Regional Board

Mohamed Lotfy, Board Representative	Utah Valley University, UT
Ed Lindoo, Treasurer and Registrar	Regis University, CO
Pam Smallwood, Editor	Regis University, CO
Dan McDonald, Webmaster	Utah Valley University, UT

2025 CCSC Rocky Mountain Conference Steering Committee

Mohamed Lotfy, CCSC Region Representative	Utah Valley University, UT
Ed Lindoo, Treasurer	Regis University, CO
Pam Smallwood, Editor	Regis University, CO
Karina Assiter, Submission Co-chair	Landmark College, VT
Mohamed Lotfy, Submission Co-chair	Utah Valley University, UT
Dan McDonald, Webmaster	Utah Valley University, UT
Jenny Nehring, Publicity Chair	Utah Valley University, UT
Ed Lindoo, Registrar	Regis University, CO
Michael Leverington, Conference Chair	Northern Arizona University, AZ
Xi Chen, Site Co-chair	Utah Valley University, UT
Jenny Nehring, Site Co-chair	Utah Valley University, UT
Mohamed Lotfy, Program Co-chair	Utah Valley University, UT
Imtiaz Parvez, Program Co-chair	Utah Valley University, UT
Michael Leverington, Student Posters Co-chair	Northern Arizona University, AZ
Ranjidha Rajan, Student Posters Co-chair	Metropolitan State University, CO
Dave Loper, Student Programming Competition Co-chair	Utah Valley University, UT
Imtiaz Parvez, Student Programming Competition Co-chair	Utah Valley University, UT

Transformations in Computer Science Programs to Integrate Quantum-AI[*]

Conference Tutorial

Bhaskar Sinha, Mohammad Amin, Pradip Peter Dey
School of Technology and Engineering
National University
9388 Lightwave Ave., San Diego, CA 92123
`{bsinha, mamin, pdey}@nu.edu`

1 Background

What will be the next greatest technological peak? It will be the combined effect of integrating Quantum Computing (QC) and Artificial Intelligence (AI) for real-world problem-solving. Quantum-AI represents the intersection of QC and AI. QC is based on the principles of quantum mechanics and utilizes the computing power of a Quantum Processing Unit (QPU) to augment algorithms used in AI-driven applications[2][4], potentially solving problems that are currently difficult for traditional systems. Quantum-AI is already playing a significant role in the business world, and the demand for specialists in this emerging field is growing. The integration of this technology into Computer Science (CS) programs in academia enriches the discipline. It provides students with the knowledge to address complicated, real-life, and practical challenges facing the industry. In addition to the traditional foundational spaces such as algorithms, data structures, and programming languages, CS curricula are now undergoing significant shifts to integrate Quantum-AI, which has the potential to redefine the future of CS. These changes are not just additives, but are profoundly altering teaching priorities, research directions, and required CS competencies for graduates.

Similar to the AI transformations[5], QC, with QPU, brings in new concepts and models of computation; created and based on the physics of quantum me-

---

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

chanics[3]. This promotes and encourages in-depth analysis and comprehension of QC basics like qubits, superposition, and entanglement, which challenge the binary fundamentals of traditional Von Neumann computing architecture[1]. This enables significant speedups in solving certain types of computationally intensive problems. When these QC features are combined with AI constructs, Quantum-AI capabilities enable breakthroughs in areas of data processing, cryptography, optimization, machine learning, etc., thus addressing the areas of security and big data storage with contemporary vector databases. Teaching these CS concepts in academic programs prepares students for the emerging market demands, encouraging interdisciplinary collaborative research, thus leading to augmented problem-solving expertise. Universities are beginning to offer introductory and advanced courses in quantum algorithms, quantum information theory, and quantum programming languages. These offerings require a rethinking of prerequisite knowledge where linear algebra, quantum mechanics, and complex probability feature more prominently in the academic path of CS students. Research opportunities in quantum machine learning, cryptography, and optimization are gaining recognition, and funding from governments and tech giants is accelerating the development of quantum-AI-focused academic initiatives.

By incorporating Quantum-AI algorithms and models, the CS prospectus aligns with rapid technological developments, enabling students to contribute to fields like national security, robotics, criminal justice, healthcare, data science, and many others. This integration also raises the importance of ethical considerations, ensuring that students not only develop technical expertise but also understand the social consequences of these transformative technologies. The inclusion of Quantum-AI also promotes a deeper understanding of ethical considerations in the global collaborative workplace. However, this paradigm shift for creating content for academic CS curriculum faces many challenges. Rapid pace of technological advancements, steep learning curve of QC and AI concepts, and planning for adequate resources are some of the factors that make it difficult for educators to keep course content up to date. In summary, the rise and adoption of Quantum-AI is creating a fundamental shift in academic CS programs. The integration of these technologies into curricula is not only necessary to stay relevant in a rapidly evolving digital world but also essential to prepare the next generation of engineers, scientists, and administrators with the tools to drive improvements sensibly and efficiently. Academic institutions must continue to adapt and lead, ensuring that students are prepared to navigate the computational developments ahead. As industries increasingly rely on AI for decision-making and as QC transitions from theoretical potential to real-world application, academia must continue to evolve to keep pace with technological advances. Beyond curriculum design, these shifts also have implications for faculty recruitment, laboratory infrastructure, and institutional partnerships with industry and government entities. The participants of this panel will discuss the issues mentioned above and other issues raised in the conference to achieve clarity and quality in the undergraduate CS curriculum.

2 Tutorial Description

This panel provides an interactive and effective learning session where panelists will raise some critical questions, concerns, and discuss potential opportunities. They will also provide valuable suggestions on how Quantum-AI can enrich the CS curriculum. The attendees can interact, ask questions, express their ideas, experiences, and provide suggestions. Principles of AI and QPU-based QC are introduced briefly. Discussions will include how these can be integrated into the CS curriculum, where students can get the opportunities to learn about these emerging technologies and equip themselves to face future challenges and contribute appropriately. The panelists will also address the ethical issues involved in using AI in different CS courses and how to maintain academic standards and integrity.

3 Expected Outcomes

Attendees will gain a good understanding of the concepts of Quantum-AI and their applications. They will have the chance for open discussions, constructive criticisms, and suggestions.

4 Target Audience

Interested faculty who desire to teach computer science courses and related topics.

5 Prerequisites

All educators with an interest in teaching and learning CS courses are welcome to this panel discussion, where intuitive explanations with examples are presented about Quantum-AI related pros and cons.

References

[1] J. D. Dumas. *Computer Architecture: Fundamentals and Principles of Computer Design.* 2nd edition. CRC Press, 2016.

[2] C. Monroe and M. D. Lukin. “Quantum computing: Progress and prospects”. In: *Nature* 591.7848 (2021), pp. 43–44. URL: `https://doi.org/10.1038/d41586-021-00671-5`.

[3] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, 2010.

[4] J. Preskill. “Quantum Computing in the NISQ era and beyond”. In: *Quantum* 2 (2018), p. 79. URL: `https://doi.org/10.22331/q-2018-08-06-79`.

[5] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach.* 4th edition. Pearson, 2020.

How to Install and Use T-Pot Honeypot VM in a Defensive Cybersecurity Course*

Conference Tutorial

Mohamed Lotfy Ph.D.
Utah Valley University
Orem, UT 84058
`MohamedL@uvu.edu`

Preparing IT and cybersecurity graduates to meet industry needs, IT and cybersecurity courses must introduce current offensive and defensive tools and practices. To secure computing resources, systems, services, data, and network services. current defensive cybersecurity tools should be introduced and applied in the different hands-on activities thus allowing students to gain the needed knowledge of current cybersecurity best practices [1, 4, 7]. Honeypots, an active defensive mechanism, are used to deceive attackers in the action on objective cyber-attack phase as a decoy to mislead attackers. Honeypots mimic real systems to attract attackers by making them believe they have found a target with exploitable weaknesses, thus enabling deflecting attacks. Honeypots allow the cybersecurity team to monitor and analyze how the attackers are interacting with the honeypots, identify their actions, techniques, used tools, and their overall objectives. Utilizing a hands-on approach allow students to develop defensive cybersecurity competency enabling them to build layered defenses including honeypots that hardens the systems to advanced persistent threats. Demonstrating defensive cybersecurity practices requires an environment where vulnerable honeypots hosts are deployed enabling the students to perform the different defensive actions. Using an encapsulated virtual environment where the attacks on vulnerable honeypots can be conducted, reduces the risk to institutional networks and systems.

---

*Copyright is held by the author/owner.

Tutorial Description

In this tutorial I will provide a hands-on working example of how to install the T-Pot honeypot VM on a testing environment using VMware Workstation Pro on laptops or PCs[2, 8]. The virtual environment will include an offensive security Kali Linux VM, and a Debian 12 VM where the T-Pot containers will be deployed. I will demonstrate how to build and setup the Debian host, how to deploy the honeypots using Docker [5], update the .ymal deployment file, demonstrate how to attack the honeypots, and show how the different attacks are visualized on the T-Pot elastic stack dashboards.

Tutorial program

In the tutorial I will illustrate and demonstrate the following:

1. How to install and setup a Debian VM in VMware Workstation Pro.
2. Describe how to setup and use Docker.
3. How to deploy the T-Pot honeypot containers.
4. How to adapt the .ymal file to manage the deployed containers.
5. How to attack the honeypots from the Kali VM.
6. How to see the attacks on the T-Pot elastic stack dashboards.

Expected outcomes

Attendees will exit the tutorial with a working VMware environment and learn how to use the T-Pot honeypot system to detect and visualize attacks on the different honeypots.

Target audience

Any faculty who desires to incorporate VMware virtual environment and use T-Pot honeypot in a defensive cybersecurity course.

Prerequisites

Attendees should be familiar with Linux, networking, and some containers and Docker knowledge. It is highly recommended that attendees bring their own laptops with VMware Workstation Pro, a Kali Linux VM, and a Debian netinst image installed [3, 6].

References

[1] Pooneh Nikkhah Bahrami, Ali Dehghantanha, Tooska Dargahi, Reza M Parizi, Kim-Kwang Raymond Choo, and Hamid HS Javadi. Cyber kill chain-based taxonomy of advanced persistent threat actors: Analogy of tactics, techniques, and procedures. *Journal of information processing systems*, 15(4):865–889, 2019.

[2] Broadcom. VMware Workstation Pro, 2025. URL: `https://community.broadcom.com/vmware-cloud-foundation/discussion/downloading-vmware-pro`.

[3] Debian. Network install from a minimal USB, CD, 2025. URL: `https://www.debian.org/CD/netinst/`.

[4] Eric M Hutchins, Michael J Cloppert, and Rohan M Amin. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 1(1):80, 2011.

[5] Docker Inc. Docker, 2025. URL: `https://www.docker.com/`.

[6] OffSec Services Limited. KALI pre-built virtual machines, 2025. URL: `https://www.kali.org/get-kali/\#kali-virtual-machines`.

[7] Mohamed Lotfy. The structure of a graduate defensive cybersecurity course. *Journal of Computing Sciences in Colleges*, 39(2):72–84, 2023.

[8] Telekom Security. Honeypots, 2025. URL: `https://github.security.telekom.com/honeypot.html/`.

Curd: A Curriculum Design Tool[*]

Peter Aldous
Computer Science
Utah Valley University
Orem, UT 84058
peter.aldous@uvu.edu

Abstract

Computer science curricula change rapidly and exhibit complex prerequisite relationships. Consequently, maintenance of these curricula requires careful attention and considerable time. This paper presents Curd, a tool that automates common tasks in curriculum design in an attempt to simplify curriculum maintenance. Instead of encoding the relationships between courses directly, Curd models relationships between topics and skills taught in those courses. As a result, it is capable of exploring hypothetical changes to a curriculum such as moving a topic from one course to another, splitting a course into two, or combining two courses together. Curd induces optimal prerequisite relationships between courses from the relationships between their respective topics. It can also automatically generate graduation plans that respect those prerequisite relationships.

1 Introduction

Because computer science is a new and quickly-changing field, computer science curricula at universities are in a constant state of flux. These curricula are especially difficult to manage because of the long chains of prerequisites that are typical in the field. When educators change a course or a program, they are obliged to manually investigate their curriculum to ensure that the changes do

---

[*] 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.

not create complications for students (e.g., extending the time to graduation or forcing students to take too many courses in a particular term). These complications can be manifest in universities like those in the USA, which specify requirements and permit students to choose which courses to take, and universities in many other locations, which specify a course of study for students in each program. Prerequisite relationships on courses naturally form a partially ordered set; however, standard practice is to perform these analyses manually instead of using computers to analyze them. This manual work is time-consuming and, in departments of computer science, ironic.

The curriculum variants presented in this paper are from Utah Valley University (UVU), a large teaching-focused university in the United States.

This paper presents Curd (an abbreviation of "curriculum design"), a tool designed to streamline the development of curricula. Curd provides two major benefits to educators. First, the process of creating a catalog forces curriculum designers to think deliberately about what topics and skills (what Curd calls **requirements**) are taught in each course and how those requirements relate to each other. Second, once a catalog exists, the exploration of potential changes to curricula is much faster and easier and, crucially, gives curriculum designers the certainty that they did not miss details or fail to explore a possibility that is consistent with the curriculum's constraints.

The contributions of this paper are the following:

- The presentation of a tool for analyzing and organizing curricula (see Section 3) and
- A group of case studies that demonstrate its utility (see Section 4).

Curd is available under the GPLv2 at https://github.com/peteraldous/curd

2 Related Work

Few works have been published on the subject of curriculum design tools. This paper presents related work in two sections: general related work (Section 2.1), which presents background research on curriculum design, and a section on Curricular Analytics (Section 2.2), a tool that shows some similarities to Curd.

2.1 General Related Work

Blank and Kumar [1] are perhaps the first to apply the idea of software design patterns to curricula. Their design patterns are high-level concepts, not mathematical properties of prerequisite graphs.

Klingbeil and Bourne [7] present a mathematics course designed to help students who begin collegiate studies without the necessary mathematical background to study engineering. This problem is common among universities like those in the USA, where students are admitted to a university and not to a specific program. This course was extremely effective at making the study of engineering possible for students who otherwise could not have completed the program. Their success was partially mitigated by lower graduation rates; however, the difference in the number of students who enrolled was much larger than the decrease in graduation rates. Their findings reflect on a specific addition to a specific curriculum but similar modifications are common at other universities—and these modifications are among the simplest that can be made.

Dietz-Uhler and Hurn [3] present a summary of learning analytics tools, which ask questions about the efficacy of learning or how much a group of students uses a particular set of resources. These questions are only tangentially related to questions about the structure of a curriculum.

Several works from the University of New Mexico [4, 5, 6, 9, 11, 12, 10] explore the use of graph theory to analyze curricula and identify potential problems, such as courses that tend to delay graduation because of their large numbers of prerequisites (or the large number of courses that depend on them as prerequisites). The results of this line of work are available as a web application (free to use) called Curricular Analytics [2] (see Section 2.2).

Curd depends on the NetworkX [8] Python library and the Z3 Theorem Prover [13] (via Z3's Python bindings).

2.2 Curricular Analytics

Curricular Analytics [2] performs several analyses using graph algorithms. For example, the delay factor of a course can be computed from node eccentricities; each node's delay factor is the maximum eccentricity of the node at the head of each longest path from that head. The blocking factor of a node is its out-degree in the transitive closure (defined briefly in Section 3.1) of the course dependency graph and the centrality factor of a node is its in-degree in the same graph. While Curd is a prototype and cannot claim the polish of Curricular Analytics, it is capable of performing the same analyses (to demonstrate, Curd calculates blocking and centrality factors). Curricular Analytics performs other analyses that can also be performed using NetworkX algorithms (e.g., cruciality). Curd's analyses can employ finer-grained reasoning because of its focus on requirements, which are parts of courses.

Curd's JSON format and Curricular Analytics's CSV format can be converted into each other (this feature is not yet implemented but certainly possible), although converting to Curricular Analytics's CSV would require discarding information about requirements and the Curd JSON would not contain information about specific requirements; each course would simply satisfy a requirement named after the course.

3 Design

3.1 Overview

The core insight in Curd is that although instructors often think of the topics and skills (again, **requirements** in Curd parlance) that are necessary to a program in terms of its courses, it is the requirements themselves that matter; the courses are simply a means to an end. With a definition of a program's requirements (topics and skills, not courses), a mapping from requirements to the courses that satisfy them, and the prerequisite relationships between requirements, Curd is able to induce a minimal prerequisite relationship between courses and generate a graduation plan that satisfies all program requirements.

It is possible to specify course selections (e.g., "I want to take course X to satisfy this core curriculum requirement") and constraints (e.g., "course X and course Y should be taken in different terms" or "course X should be taken in term 5"). Additionally, it is possible to specify a number of terms that have already occurred; courses that are not already scheduled with constraints will only be scheduled in terms that have not already occurred. Because these constraints can be added incrementally, it is possible to iterate on a graduation plan. Section 3.2 contains additional details about Curd's constraints.

Once the catalog has been assembled, much of the processing is done with NetworkX [8]. For example, it is possible to generate minimal dependencies by first computing the transitive reduction of the prerequisite courses' relationships (in other words, an equivalent graph that omits edges made redundant by transitivity). When reasoning about a subset of a catalog's courses, one must first compute the transitive closure (the opposite of a transitive reduction; it adds edges that can be inferred by transitivity) over the whole catalog, then creating a subgraph that contains only the desired nodes, and finally computing the transitive reduction. All of these operations are supported directly in the NetworkX API.

The task of creating a graduation plan resembles a topological ordering over a partially ordered set (a well-understood problem for which efficient algorithms exist), but most universities have students take multiple courses at the same time. In mathematical terms, generating a schedule when multiple courses occur simultaneously is equivalent to creating a **rankable antichain decomposition** of the partially ordered set represented by the graph. An **antichain** is a subset of a partially ordered set where no two elements in the set are ordered with respect to each other; in the context of a curriculum's prerequisite graph, an antichain is a set of courses that can be taken during the same term because none of them depends on any of the others. A **decomposition** of a set is an assignment of each of the set's elements to exactly one subset. **Rankable** means that there must be an ordering of the antichains that is consistent with the original partial order; in other words, it means that there are no cyclic dependencies between the antichains. To the author's knowledge, there is no published algorithm that accomplishes this task. Curd arranges schedules by encoding them as logical constraints in Z3 [13].

Figure 1 shows the constraints generated for a small catalog with two courses (limited to orderings between courses).

```
CS1_term > 0
CS1_term <= 2
CS2_term > 0
CS2_term <= 2
CS1_term < CS2_term
```

Figure 1: Scheduling Constraints

A Curd catalog is stored as a JSON object that contains data about courses, requirements, programs, saved constraints (see Section 3.2), prerequisites, relationships between courses and requirements, and the institution (e.g., the number of terms). A Curd catalog that does not differentiate between courses and requirements (where each course has one unique requirement) could duplicate the analyses performed in Curricular Analytics [9].

3.2 Constraints

Constraints are represented in aggregate as a conjunction of disjunctions (the same format as a satisfiability problem); in other words, constraints are presented in groups where any constraint being satisfied is sufficient for the entire group to be satisfied and each group of constraints must be satisfied. Each constraint in a disjunction in a catalog is a triple that contains a left-hand operand, an operator, and a right-hand operand. The left-hand operand always identifies a course. The right-hand operand may be a course or a term number. When an operand represents a course, the constraint applies to when the course is scheduled. Operands are represented in an enumeration and adding a new operator requires only a few additional lines of code.

4 Case Studies

We have used Curd to investigate several possible changes to the Utah Valley University computer science curriculum. In all but the first case, the changes seemed beneficial but their viability was uncertain without Curd.

In all cases, the curriculum variants reflect the requirements of the computer science program at UVU, which has general education requirements typical of

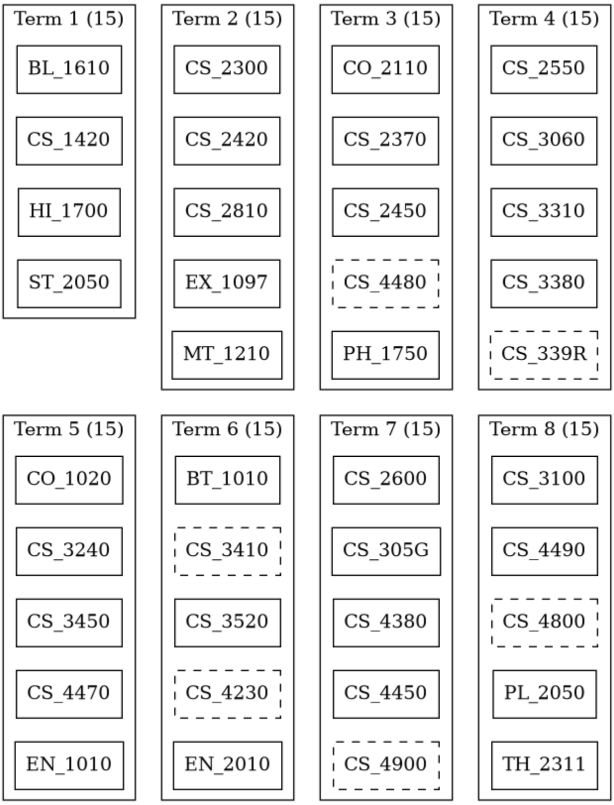


Figure 2: A typical graduation plan

curricula in the United States. A baccalaureate degree typically lasts for eight semesters over two years. A normal course load for a full-time student is fifteen credits per semester. Figure 2 shows a typical graduation plan in this curriculum. In this and other figures, dashed lines indicate electives and each term is annotated with the number of credits included in its schedule.

4.1 Networking courses

The Utah Valley University computer science curriculum historically required two courses on the topic of networking. Networking is a crucial subject, but the depth at which it was presented exceeds contemporary requirements. The two networking courses were split into regions of the OSI model: Networking

I covered layers 1–3 and Networking II covered layers 4–7. More recently, the UVU CS curriculum rearranged the topics in these classes: the first covers the basics of all seven layers of the OSI model and the second covers advanced topics over the same seven layers. Making these changes to the catalog in Curd is straightforward; one simply swaps the courses that satisfy the requirements for basic understanding of OSI layers 4–7 and advanced understanding of OSI layers 1–3. In this case, the benefit of Curd is that it provides the certainty of a mathematical proof to manual reasoning about the curriculum. While this particular case is quite simple, Curd would provide the same certainty for a much more complex change.

4.2 CS1 in a single semester

Historically, the UVU computer science curriculum has presented CS1 as a two-semester sequence. In order to accommodate students who enter the program with a wide variety of background experience, the curriculum now includes a single-semester version of CS1. Adding this single-semester course in Curd simply involves creating a course that satisfies all of the sequence's requirements. Naturally, adding this course frees up three elective credits for students.

4.3 A six-semester schedule

Many students and administrators have recently voiced interest in faster routes to graduation. One possible way to accomplish this would be to take more courses per term. Curd successfully demonstrated that it is not possible to graduate in six semesters instead of eight by taking additional courses in each term; however, taking the accelerated CS1 course makes this possible. It also demonstrates that the resulting workload would be overwhelming. For example, terms 4 and 5 would each have 21 credits (a typical load is 15 credits), all from technical courses (see Figure 3).

4.4 Finishing secondary schooling with an associate degree

Some students in this region avail themselves of options like "concurrent enrollment", where high school students enroll in high school courses that award college credit, to finish their secondary studies with an associate degree. It has long been an open question if high school graduates with an associate degree (which represents two years of study), could save two years in their time to graduation. Curd demonstrated that it is possible to do so, but only with overwhelming course loads similar to the ones discussed in Section 4.3. This analysis showed a redundancy in the associate degree requirements, which required one more science course than the bachelor's degree. Because of Curd,

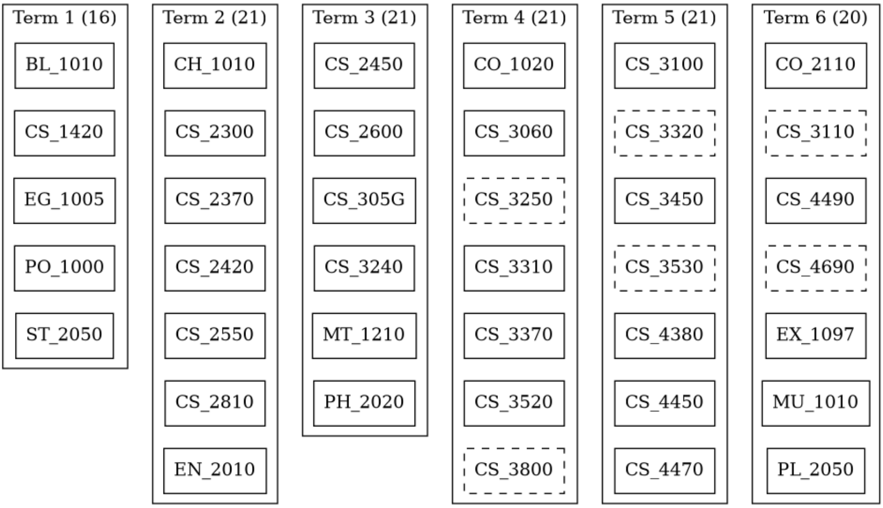


Figure 3: An accelerated graduation plan

the requirements for the associate's degree have been adjusted to remove this redundancy. This analysis also has led to discussions about the possibility of rearranging requirements to allow high school students to do more in high school and have more flexibility later.

4.5 Analysis of Prerequisites

To date, Curd has not led to any specific recommendations for changing course prerequisites with courses structured as they currently are. In one recent iteration, Curd identified two requirement prerequisite relationships made redundant by transitivity; in contrast, 62 course prerequisite relationships were directly reflected in the course prerequisite graph. The proximity of the actual dependencies to a minimized graph suggests that the UVU CS curriculum reflects careful thought about each prerequisite.

5 Limitations

At present, Curd has no GUI (it is open source and collaborators are welcome) and its audience is consequently limited to educators comfortable with CLIs and manually editing JSON files. However, it is designed so that the behaviors a GUI front end would exhibit are available as functions in its API. It can produce visualizations using the Graphviz DOT language, but these visualizations are difficult to read if they include prerequisite relationships. Additional visualizations could also be useful; for example, it would be useful to be able to see the requirements satisfied by each course.

At present, Curd is unable to induce corequisite relationships between courses but they can be specified manually with constraints. It can reason about any one of several courses satisfying a requirement but cannot express requirements that are satisfied by multiple courses (e.g., any three of these five courses). It does something similar to this for electives but this mechanism is not readily generalizable.

The process of building a catalog in Curd is time-consuming, simply because there are many details to record; however, once a catalog is created, it can be maintained with minimal effort. This up-front cost could discourage potential users from adopting Curd.

6 Future Work and Implications

Many of the limitations mentioned in Section 5 can be addressed with additional effort. An algorithm for generating rankable antichain decompositions of partial orders would be a valuable contribution to the field.

Curd is a valuable tool for organizing and reasoning about curricula. It takes time and effort to create a catalog due to the intrinsic complexity of the data; however, the process of creating it is valuable in and of itself to anyone who designs curricula. The ability to generate and iterate on graduation plans can serve students and educators alike.

Although the case studies presented in this paper focus on a single curriculum, Curd could be used for large-scale analyses of curricula. For example, it might examine which year a particular requirement is typically taught across many different departments' curricula.

On a similar note, it may be possible to export data from commonly used tools at universities such as Banner or CourseLeaf to create a simple catalog. It is certainly possible to import data from Curricular Analytics's CSV files into Curd (this feature is not yet implemented). With just the data set from a typical university computer system, it would be possible to make a graph of courses; however, users of Curd who want to perform analyses on requirements would have to enter them manually.

References

[1] Douglas Blank and Deepak Kumar. "Patterns of curriculum design". In: *Informatics Curricula and Teaching Methods: IFIP TC3/WG3. 2 Con-*

ference on Informatics Curricula, Teaching Methods and Best Practice (ICTEM 2002) July 10–12, 2002, Florianópolis, SC, Brazil. Springer. 2003, pages 77–86.

[2] *Curricular Analytics.* `https://curricularanalytics.org/`. Accessed: 2025-01-17. 2023.

[3] Beth Dietz-Uhler and Janet E Hurn. “Using learning analytics to predict (and improve) student success: A faculty perspective”. In: *Journal of interactive online learning* 12.1 (2013), pages 17–26.

[4] Gregory L Heileman et al. “Characterizing the complexity of curricular patterns in engineering programs”. In: *2017 ASEE Annual Conference & Exposition.* 2017.

[5] Gregory L Heileman et al. “Curricular analytics: A framework for quantifying the impact of curricular reforms and pedagogical innovations”. In: *arXiv preprint arXiv:1811.09676* (2018).

[6] Michael S Hickman. “Development of a curriculum analysis and simulation library with applications in curricular analytics”. Master’s thesis. University of New Mexico, 2017.

[7] Nathan W Klingbeil and Anthony Bourne. “The Wright State model for engineering mathematics education: Longitudinal impact on initially underprepared students”. In: *2015 asee annual conference & exposition.* 2015, pages 26–1580.

[8] *NetworkX. Network Analysis in Python.* `https://networkx.org/`. Accessed: 2025-01-17. 2014.

[9] Ahmad Slim. “Curricular analytics in higher education”. PhD thesis. The University of New Mexico, 2016.

[10] Ahmad Slim et al. “Crucial based curriculum balancing: A new model for curriculum balancing”. In: *2015 10th International Conference on Computer Science & Education (ICCSE).* IEEE. 2015, pages 243–248.

[11] Jeffrey Wigdahl. “Assessment of curriculum graphs with respect to student flow and graduation rates”. In: *Albuquerque, New Mexico: The University of New Mexico* (2013).

[12] Jeffrey Wigdahl et al. “Curricular efficiency: What role does it play in student success?” In: *2014 asee annual conference & exposition.* 2014, pages 24–344.

[13] *Z3 Theorem Prover.* `https://github.com/Z3Prover`. Accessed: 2025-01-17. 2015.

A Game Design Approach to CS2:Take Two*

Daniel Cliburn, Jacob Bejarano, Sai Sravan Meduri,
and Srikar Podicheti
Department of Computer Science
University of the Pacific
Stockton, CA 95211
`dcliburn@pacific.edu`
`{j_bejarano1,s_meduri,s_podicheti}@u.pacific.edu`

Abstract

This paper describes an approach to teaching the second course in the computer science major (CS2) through game design. Students develop an original game idea and then create the game through a series of course projects that allow them to achieve course learning objectives. Earlier attempts to teach CS2 though game design were found to be time intensive for the instructor. To help reduce instructor preparation time, upper division and graduate students are involved in the course as student assistants. The role of the student assistants is to implement the game idea that the CS2 students develop using modern game engines. The student assistants then support the CS2 students as they create their own graphical versions of the game for the final project. We were very satisfied with our approach to CS2. We felt it engaged students while providing them with an opportunity to meet course learning objectives and gain appropriate experience using generative AI.

1 Introduction

Previously, we reported an approach to using game design in CS2 (the traditional second course in the computer science major covering data structures)

---

*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.

[4]. We used game design as a way to engage students and provide a context to understand the application of course concepts. The use of games in computing curriculum has been thoroughly explored [16]. Games are often cited as motivating and engaging for students [13]. Educators have used tile games [15], role-playing games [14], text-based adventure games [10], card games [1], and escape room games [7] as ways to motivate and encourage students to think critically about programming assignments. Our approach to CS2 was novel in that it incorporated game design principles into CS2. The students learned aspects of games that make them fun to play, and each student created their own version of an original video game through a series of course projects. The projects were designed to provide students with an opportunity to achieve the learning objectives of CS2, while they created the game.

Many years have passed since our first use of game design principles in CS2, and quite a bit has changed. In particular, generative artificial intelligence (AI) has radically affected how students are learning to code, prompting the need to consider how this impacts the teaching and learning process [3]. An emerging skill necessary for today's computer science students is the ability to prompt an AI for code that correctly solves specified problems [5]. This involves both being able to write natural language prompts to provide to the AI and understanding whether the code generated by the AI correctly solves the specified problem. Computing educators are redefining how they teach students to code, specifically developing assignments that incorporate AI usage while also encouraging students to develop their metacognitive skills [6]. Others allow students to use generative AI to complete assignments, as long as they report how they used AI and write reflections on what they learned [12]. Another approach is to develop "AI-resistant" or "AI-proof" assignments [2, 11]. These are assignments that AI cannot solve easily and that either integrate local context or require students to think critically about how to use AI-generated content when completing the assignments. Because of generative AI, some suggest that courses like CS1 and CS2 may start to focus less on basic coding skills and implementations of data structures, and more on how to design software systems [9].

Another change since our first use of game design principles in CS2 is that game engines, such as Unity, Unreal Engine, and Godot, have become more accessible to novice programmers. It is easier now for first-year programming students to create fully functioning video games than ever before. For these reasons, we felt it was time to revisit and retry our approach to game design in CS2. We hoped that by developing an original game, students would need to think critically about how to integrate AI-generated content into their assignment submissions. We also wanted to incorporate game engines into the course and provide students with an opportunity to create their final version of the game using a game engine. The remainder of this paper describes our revised approach to game design in CS2.

2 Course Changes

Our first change from past offerings of game design in CS2 was to give students a choice between taking our University's traditional Data Structures course and taking the section that would teach Data Structures through game design. Not all students play games, and while rare, some students did not enjoy participating in the development of an original game in the Data Structures course. Therefore, through academic advising and announcements, we made students aware of the time of the game design section of Data Structures. Thus, students that registered for the game design section chose to learn about concepts of game design and to participate in the making of an original game. Students that did not want to learn about concepts of game design were able to register for a traditional section of Data Structures.

Our second change to the course from previous offerings was to allow students to use AI in any way they saw fit, as long as they explained in the comments of their code how they used AI to prepare their submissions. We did this for two reasons. First, it is becoming increasingly challenging to prevent students from using AI to complete programming projects. Second, we hoped that our original game approach would require local context so that AI would not be able to solve it easily. If students chose to use AI, they would need to think critically about how to prompt the AI for content that they could integrate into their assignments.

Our third change was to involve upper division and graduate students in the course as student assistants. One of the issues identified in our previous paper [4] was that incorporating game design into CS2 requires substantial instructor preparation time. Instructors must implement the game designed by the students and divide it up into a reasonable number of assignments for the students to complete during the term. Creating a graphical and event-driven version of the game for students to implement at the end of the term was particularly time consuming. To help with this, three students (two graduate students and one senior undergraduate student) were recruited to implement the game designed by the CS2 students using several game engines (Panda3D, Godot, and Unreal Engine). The goal was to determine which engines would work well for creating the game and to provide the CS2 students examples for developing their own graphical versions of the game. The student assistants earned independent study credit for their help and participation.

Our fourth change was to incorporate usability and play testing into the course. While the course had three midterm exams, there was no final exam.

Instead, during the final two weeks of the term, each student created their own graphical version of the game. During the last class period, students took turns playing each other's games and provided feedback on the usability and enjoyment of the games. Students then had approximately one week to improve their game based on the feedback they received before showing off the final version during the time reserved for the final exam (we called this the Final Project Showcase). Students voted on the best game at the Final Project Showcase, and the student that made the winning game received extra credit added to their final project score.

3 Course Description

In spring 2025, the game design section of CS2 began with a description of the MDA (Mechanics, Dynamics, Aesthetics) framework [8] to provide students with a context for the design of a fun game. Next, we placed the students into groups of two to three around their favorite genres of games. In other words, we paired together students that liked racing games, paired together students that like fantasy role-playing games, and so on, until every student had at least one partner with a common interest. With their group, students brainstormed ideas for an original game of their favorite genre. Once each group formed an idea for a game, they created a playable low-fidelity prototype to represent it. We provided students with paper, scissors, index cards, pencils, tape, small toys, and dice. We gave the groups one class period to make their low-fidelity prototypes. An example low-fidelity prototype is shown in Figure 1.

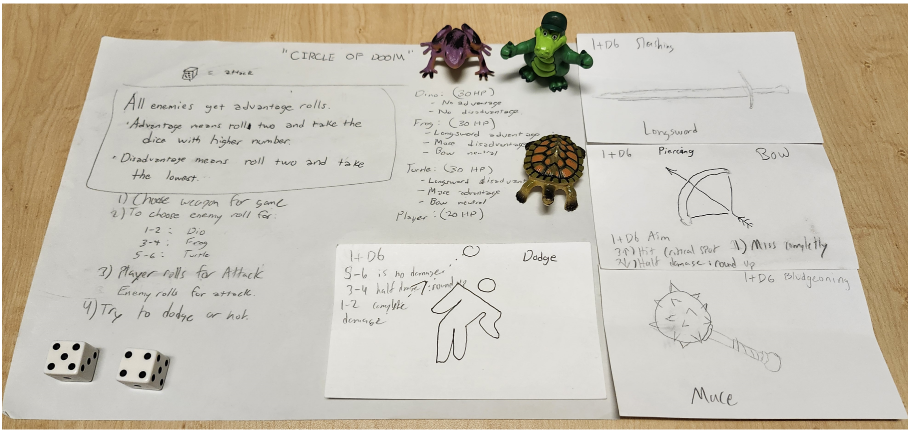


Figure 1: Low-fidelity prototype of a game developed by CS2 students.

During the next class period, the students took turns playing each other's game prototypes. The student assistants also participated in playing the games. At the end of the period, everyone voted for the best game idea, and the winning idea became the basis for the course projects. The winning low-fidelity prototype from spring 2025 (shown in Figure 1) was called "CIRCLE OF DOOM." Each student created their own version of CIRCLE OF DOOM over the course of the term.

During the following class period, the instructor and students discussed major principles of object-oriented programming (encapsulation, inheritance, and polymorphism), and then worked together to design a series of C++ classes that both illustrated the major principles of object-oriented programming and that could be implemented to create the game CIRCLE OF DOOM. After this, the CS2 students began learning the course content and the student assistants began implementing the game in the chosen game engines. The student assistants had two and a half months to complete their work before the CS2 students would begin to create graphical and event driven versions of the game. It should be noted that during this two and a half months, the game design section covered the same material as the traditional section of CS2 (Data Structures), and took the same exams as the traditional section of CS2.

During the term, the CS2 students completed five major course assignments. The objectives for project 1 were for students to learn to create and manipulate objects, and to learn to read and write to text files. For this project, students wrote one of the classes necessary to implement the game. The objectives for project 2 were for students to learn about dynamic memory allocation, inheritance, and polymorphism. For this project, students created a limited text-based version of the game by implementing the remaining classes that were designed earlier in the term. The objective for project 3 was for students to learn to create and manipulate a dynamic data structure, such as a linked list. For this project, students created a fully functioning text-based version of the game, with a linked list used to implement an inventory of weapons that players used during the game. The objective for project 4 was for students to learn to use the C++ Standard Template Library (STL). For this project, each student customized their game to make it more fun to play by choosing two STL classes and incorporating them into their game in some nontrivial way.

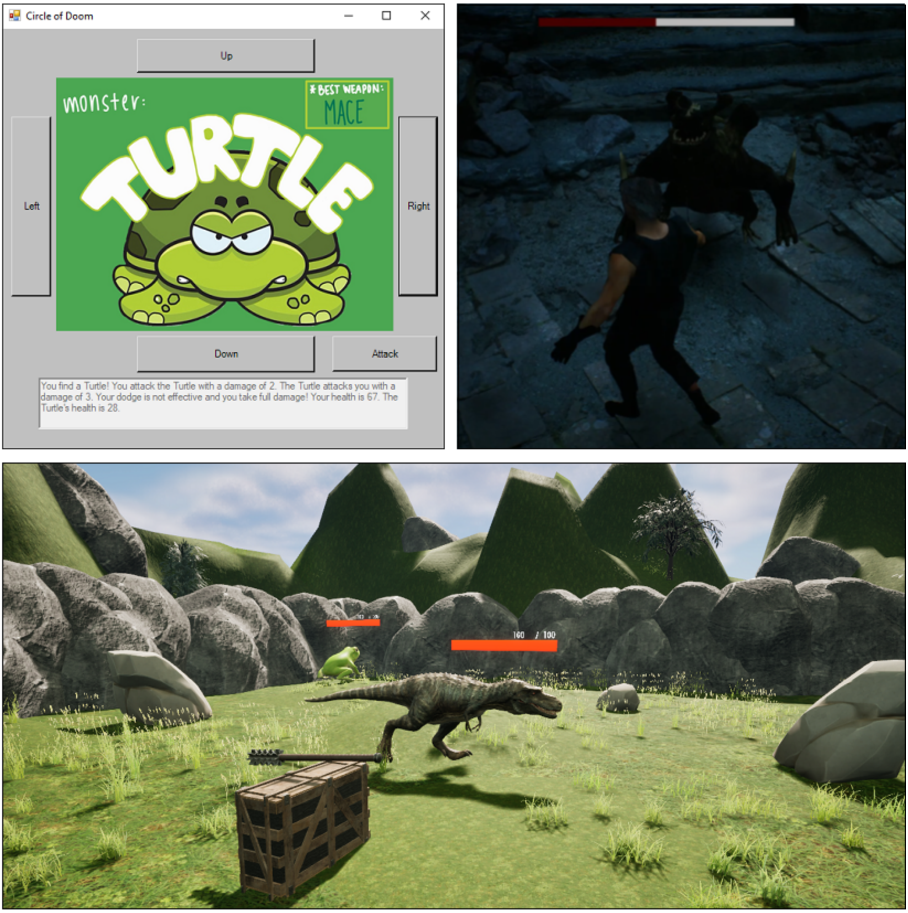
The objective for project 5 (the final project) was for students to learn to create graphical and event driven applications. For this project, students were first shown the implementations of the game developed by the student assistants and the instructor, so that the students could see what was possible with each platform. The CS2 students then created their own graphical versions of the game using the platform that best fit their situation and interests. Students were able to use either their own devices or the University's lab machines to develop their games. In spring 2025, students created versions of CIRCLE OF DOOM using Windows Forms, Unreal Engine, and Godot. One student even created a virtual reality version of the game for the Meta Quest 3 using Unreal Engine. The student final projects were graded during the Final Project Showcase. The grading rubric for the final project focused on the play experience (i.e. Was it clear to users how to play the game? Was it clear to users whether they won or lost the game?). We decided that the grading rubric for the final project should focus on visual and usability aspects of the game, rather than focus on code style and implementation, since we wanted to construct a fair rubric that could be applied to projects created using any of the suggested development environments. Some examples of student final projects are shown in Figure 2.

4 Conclusion

Having upper division and graduate students assist with the course was very helpful to the instructor. While it is difficult to quantify exactly how much time the student assistants saved the instructor, having the student assistants work on creating the graphical version of the game freed the instructor to focus on tasks more traditionally associated with teaching a Data Structures course. This substantially reduced the amount of time that the instructor needed to spend developing the graphical and event-driven versions of the game during the term. We suggest having student assistants attend class sessions at the beginning of the term to play the low-fidelity prototypes, and to participate in classroom discussions on how to design the game. It was also helpful to have the student assistants attend class during the last two weeks of the term, to help the CS2 students develop their final projects.

However, it probably was not necessary to have many options from which the CS2 students could choose to develop their final projects. Having one option that works on multiple platforms would likely be sufficient. What we did notice was that the CS2 students appreciated having video tutorials to follow. Either these videos could be developed by the student assistants, or the student assistants could curate a set of videos they find online that would be helpful for the CS2 students to watch as they are creating their own graphical versions of the game.

CS2 students definitely used AI during the term to assist with the completion of their projects (as we observed in the comments of their code). However, students were not able to use AI to generate an entire solution to any of the course projects. Students needed to test and modify AI generated code in order to develop projects that met assignment objectives. We were very satisfied with our game design approach to CS2. We felt it provided an exciting way to



You find a Turtle! You attack the Turtle with a damage of 2. The Turtle attacks you with a damage of 3. Your dodge is not effective and you take full damage! Your health is 67. The Turtle's health is 28.

Figure 2: Examples of CS2 student final projects.

engage students while allowing them an opportunity to meet course learning objectives and gain appropriate experience with generative AI.

As mentioned earlier in this paper, students were able to choose between the game design section of CS2 and a traditional offering of CS2. The instructor was the same for both sections. We did not attempt to assess student engagement in either section of the course. Future work could attempt to assess student engagement to compare engagement between the game design section and the traditional section of CS2. However, while several class periods covered different material between the sections, and the sections completed different programming projects, students took the same three exams in both sections. We conducted a Repeated Measures ANOVA to compare exam scores between the sections. Exam number was used as the within subjects variable, and course section was used as the between subjects variable. There were no significant main effects of exam number or course section. The interaction between exam number and course section was also not significant. We view this as a positive result, as it suggests students that enjoy learning about game design can choose to enroll in a game design section of CS2 without fear that it may negatively impact their grade. Instructors that enjoy teaching game design concepts can incorporate this into their CS2 courses without fear that it may significantly hinder student learning.

References

[1] Chris Alvin and Lori Alvin. “The Game Guillotine as Inspiration for a Data Structures Course”. In: *Journal of Computing Sciences in Colleges* 39.5 (2023), pp. 32–42.

[2] Wejdan Awadallah Alkouk and Zuheir N Khlaif. “AI-resistant assessments in higher education: Practical insights from faculty training workshops”. In: *Frontiers in education.* Vol. 9. Frontiers Media SA. 2024, p. 1499495.

[3] João Batista, Anabela Mesquita, and Gonçalo Carnaz. “Generative AI and higher education: Trends, challenges, and future directions from a systematic literature review”. In: *Information* 15.11 (2024), p. 676.

[4] Daniel Cliburn. “A game design approach to CS2”. In: *Journal of Computing Sciences in Colleges* 27.5 (2012), pp. 214–221.

[5] Paul Denny et al. “Prompt Problems: A new programming exercise for the generative AI era”. In: *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1.* 2024, pp. 296–302.

[6] Tony Haoran Feng et al. “From Automation to Cognition: Redefining the Roles of Educators and Generative AI in Computing Education”. In: *Proceedings of the 27th Australasian Computing Education Conference.* 2025, pp. 164–171.

[7] Nasser Giacaman and Valerio Terragni. “Empowering Computing Students with Large Language Models by Developing an Escape Room Game”. In: *ACM EngageCSEdu* (2025).

[8] Robin Hunicke, Marc LeBlanc, Robert Zubek, et al. “MDA: A formal approach to game design and game research”. In: *Proceedings of the AAAI Workshop on Challenges in Game AI.* Vol. 4. 1. San Jose, CA. 2004, p. 1722.

[9] Maggie Johnson. “Generative AI and CS education”. In: *Communications of the ACM* 67.4 (2024), pp. 23–24.

[10] Clif Kussmaul. “An extended series of assignments in CS2 involving a text adventure game”. In: *Journal of Computing Sciences in Colleges* 32.6 (2017), pp. 14–23.

[11] Sam Lau and Philip Guo. “From" Ban it till we understand it" to" Resistance is futile": How university programming instructors plan to adapt as more students use AI code generation and explanation tools such as ChatGPT and GitHub Copilot”. In: *Proceedings of the 2023 ACM Conference on International Computing Education Research-Volume 1.* 2023, pp. 106–121.

[12] Eric D Manley et al. “Examining Student Use of AI in CS1 and CS2”. In: *Journal of Computing Sciences in Colleges* 39.6 (2024), pp. 41–51.

[13] Maja Videnovik et al. “Game-based learning in computer science education: a scoping literature review”. In: *International Journal of STEM Education* 10.1 (2023), p. 54.

[14] Mike Wallinga. “Engaging CS2 students via a semester-long in-class game project”. In: *Journal of Computing Sciences in Colleges* 35.2 (2019), pp. 77–88.

[15] Karen E Works. “Three Phase-Adversarial Search-Tile Games”. In: *Journal of Computing Sciences in Colleges* 40.5 (2024), pp. 29–32.

[16] Burkhard Claus Wünsche et al. “Leveling up Learning: Serious Games for Computing Education-Long-Term Opportunities and Risks”. In: *Proceedings of the 27th Australasian Computing Education Conference.* 2025, pp. 134–143.

AI as a Study Aid: Using Artificial Intelligence to Learn Difficult Concepts in Computer Science*

Heather Jackson and Victor Clincy
Department of Computer Science
Kennesaw State University
Marietta, GA 30144
`hjacks66@students.kennesaw.edu`
`vclincy@kennesaw.edu`

Abstract

Computer Science remains one of the most challenging subjects for students to fully understand. With the rapid increase in artificial intelligence tools like ChatGPT, there is growing interest in understanding their potential as a supportive role to student learning. This paper investigates how Computer Science students are currently using AI tools, and evaluates their impact. Through a structured review of recent empirical studies, the research finds that AI tools are most effective for programming tasks, debugging, and reviewing concepts, while being less reliable for theoretical subjects. Critical issues such as incorrect output, excessive reliance on AI tools, and lack of prompt writing skills limit the effectiveness of AI educationally. The findings suggest that incorporating prompt engineering and critical thinking training into the Computer Science curriculum may significantly increase the effectiveness of using AI as a tool. Rather than restricting AI, educators should embrace it as a supplementary aid in preparing students for the use of AI in a future professional environment.

---

*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

Computer science is notorious for being a difficult subject to learn and understand. With the development of Artificial Intelligence, the questions often asked are whether AI can be an effective learning tool to help students progress[8][2], and how students are currently using AI to assist in their academic work[1].

The research questions this paper seeks to answer are; how is AI currently being used by CS students and how beneficial is it in understanding difficult subjects? The purpose of this writing is to explore how AI is currently used in college computer science departments, what benefits may exist, and what challenges may be faced.

2 Proposed Work/Survey

Artificial Intelligence (AI) is a tool that has gained in popularity very recently. Computer science students have increasingly used AI tools like ChatGPT to help complete difficult programming assignments and understand concepts. Although AI tools have been introduced into some curriculums, they have often been offered as supplementary resources and have not been fully integrated into formal instructional practices[3]. AI has demonstrated benefits in programming for heavy subjects, although limitations such as overreliance and unrecognized errors have been observed[7].

Some of the observed benefits of Artificial Intelligence include helping to make computer science content feel more manageable[2]. AI tools like ChatGPT have also been shown to serve as effective tutors to help students complete assignments in programming and database management courses[5]. Furthermore, AI has been shown to encourage faster iteration and problem solving[7].

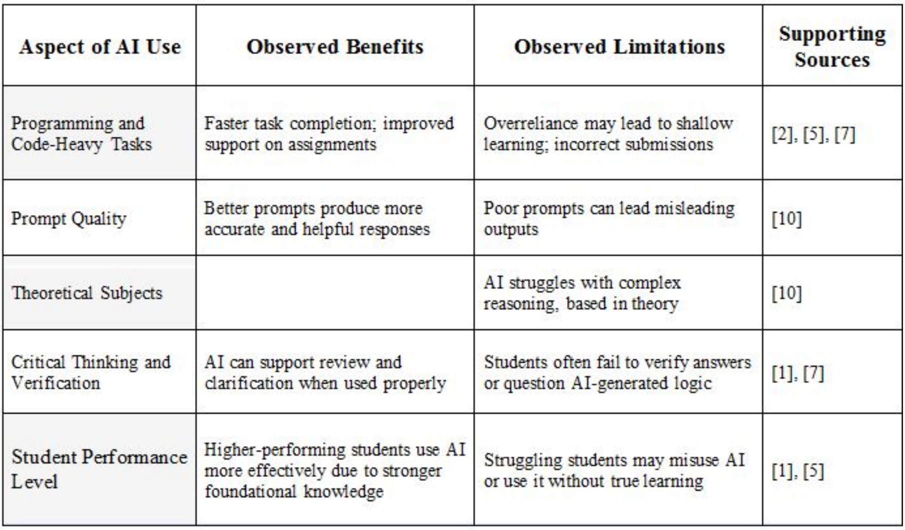
While there have been benefits found in the use of AI in higher education, limitations have also been observed. AI is known to give misleading and/or incorrect answers[7]. Due to the possibility of misleading or incomplete AI responses, students may struggle to decipher accurate versus inaccurate information[10]. Furthermore, if students have not been taught how to effectively interact with AI, they may find the usefulness of this tool to be limited[10].

Those who have been found to benefit most from AI are students with better performance. These students already have a foundational knowledge of understanding[5]. Concerns have been raised that some students may rely too heavily on AI, without being critical enough, thus limiting their own progress[1]. Evidence also suggests students who already have a foundational understanding of the subject matter are the ones seen to gain the greatest benefits from the application of AI[5][9].

3 Experiments/Results

Research has found AI is most effective in helping students with assignments that are heavy in coding, such as programming, database queries, and debugging tasks [7][5]. Similar trends have been observed in K-12 computer science education, where AI tools effectively support programming tasks, but less so for theoretical understanding[4]. AI is also significantly better when used for review and clarification, rather than simple answer regurgitation[5]. Thus, AI is best used as a supplemental tool along traditional forms of instruction, rather than a replacement[10].

While there are benefits to the use of AI in computer science academic settings, there are some short comings as well. It is less effective for theoretical tasks, which some of the notoriously difficult subjects in Computer Science lean heavily. Furthermore, students who lack foundational knowledge when learning new concepts may accept incorrect logic without realizing it. Finally, AI systems lack an understanding in how to effectively instruct students and pedagogical structure[10]. The table in figure 1 summarizes key patterns observed in the use of AI tools in Computer Science. This highlights both the benefits and limitations discussed previously.



Aspect of AI Use	Observed Benefits	Observed Limitations	Supporting Sources
Programming and Code-Heavy Tasks	Faster task completion; improved support on assignments	Overreliance may lead to shallow learning; incorrect submissions	[2], [5], [7]
Prompt Quality	Better prompts produce more accurate and helpful responses	Poor prompts can lead misleading outputs	[10]
Theoretical Subjects		AI struggles with complex reasoning, based in theory	[10]
Critical Thinking and Verification	AI can support review and clarification when used properly	Students often fail to verify answers or question AI-generated logic	[1], [7]
Student Performance Level	Higher-performing students use AI more effectively due to stronger foundational knowledge	Struggling students may misuse AI or use it without true learning	[1], [5]

Figure 1: Patterns Observed in the Use of AI Tools in Computer Science

Common student habits include rarely verifying AI generated answers critically and thus may be misguiding in what they are learning[1][5]. Also, instead of direct prompting, a trial-and-error approach is typically used[1]. Those students who are able to craft better prompts tend to receive more accurate and helpful information, suggesting effective prompting education could further improve the educational usefulness of AI tools[10].

The research suggests instead of banning, or discouraging AI in education, institutions should embrace AI as a supplemental tool. AI can enhance the learning experience, especially in technical fields like programming, database management, and debugging.

My first recommendation would be to teach effective prompt writing skills. The research shows prompt quality greatly affects the accuracy and usefulness of AI responses[10]. Studies in engineering education have also emphasized the need for strategic AI use and student training to maximize the benefits of AI[6]. Even at the K-12 level, researchers highlight the importance of structured education when first introducing AI tools to students[4]. Thus, students should be taught how to write precise and effective prompts. This would help students use AI in a much more strategic manner, rather than to passively accept what information is given to them.

My second recommendation is to encourage critical thinking skills. AI tools are known to generate plausible, but incorrect answers[10]. Therefore, students should be trained to critically assess and verify AI outputs. Teaching these skills could decrease dependency on AI and ensure students are actively engaged in their education, rather than blindly trusting AI and adopting a "copy and paste" mentality.

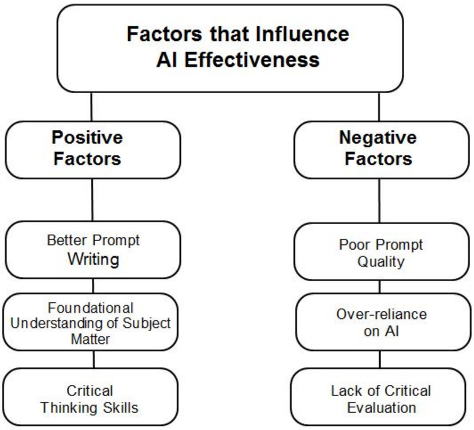


Figure 2: Factors influencing AI effectiveness in Computer Science Education

The overall goal is to give students the necessary skills needed to use AI responsibly and effectively. This approach has the potential to, not only improve learning outcomes, but better prepare students for professional settings, where AI tools are becoming the standard. Figure 2 summarizes the key factors that influence the effective use of AI in computer science education.

4 Conclusion

The use of AI in Computer Science education presents both opportunities and challenges. Research shows that AI, when used effectively, can support student learning, especially in the programing heavy courses, by improving efficacy, enabling faster iteration, and in the clarification of difficult subjects [4][2][6]. However, these benefits are not guaranteed. The successful application of AI by students as an educational aid greatly depends on their foundational understanding of the subject matter, prompt quality and the application of critical thinking[1][10].

To unlock AI's full educational potential, educational institutions must go beyond simply allowing it's use. Educators should embrace AI as a supplemental tool, while also providing students with instruction on how to effectively write prompts and how to critically think and question the results. By doing this, students will not only learn how to use AI tools effectively, but also understand when to question them. This will ensure AI enhances, rather than eliminates, authentic learning.

References

[1] Rebecca W Black and Bill Tomlinson. "University students describe how they adopt AI for writing and research in a general education course". In: *Scientific reports* 15.1 (2025), p. 8799. DOI: `https://doi.org/10.1038/s41598-025-92937-2`.

[2] Ruiqi Deng et al. "Does ChatGPT enhance student learning? A systematic review and meta-analysis of experimental studies". In: *Computers & Education* 227 (2025), p. 105224. DOI: `https://doi.org/10.1016/j.compedu.2024.105224`.

[3] Youngki Hong. *The impact of an AI-assisted learning tool on student outcomes.* 2025. URL: `https://www.wgulabs.org/posts/the-impact-of-an-ai-assisted-learning-tool-on-student-outcomes-research-brief`.

[4] Zifeng Liu et al. "How ai assisted k-12 computer science education: A systematic review". In: *2024 ASEE Annual Conference & Exposition*. 2024. URL: `https://peer.asee.org/how-ai-assisted-k-12-computer-science-education-a-systematic-review.pdf`.

[5] Daniel López-Fernández and Ricardo Vergaz. "Adoption and impact of ChatGPT in computer science education: a case study on a database administration course". In: *arXiv preprint arXiv:2407.12145* (2024). DOI: `https://doi.org/10.3390/ai5040114`.

[6] Thanh Pham et al. "Digital transformation in engineering education: Exploring the potential of AI-assisted learning". In: *Australasian Journal of Educational Technology* 39.5 (2023), pp. 1–19. DOI: `https://doi.org/10.14742/ajet.8825`.

[7] Basit Qureshi. "Exploring the use of ChatGPT as a tool for learning and assessment in undergraduate computer science curriculum: Opportunities and challenges". In: *arXiv preprint arXiv:2304.11214* (2023). DOI: `https://arxiv.org/abs/2304.11214`.

[8] Ahmed Tlili. "Can artificial intelligence (AI) help in computer science education? A meta-analysis approach". In: *Revista Española de Pedagogía* 82.289 (2024), p. 10. DOI: `https://doi.org/10.22550/2174-0909.4172`.

[9] Chi-Jane Wang et al. "Research on the impacts of cognitive style and computational thinking on college students in a visual artificial intelligence course". In: *Frontiers in Psychology* 13 (2022), p. 864416. DOI: `https://www.frontiersin.org/articles/10.3389/fpsyg.2022.864416/fu`.

[10] Tianjia Wang et al. "Exploring the role of ai assistants in computer science education: Methods, implications, and instructor perspectives". In: *2023 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE. 2023, pp. 92–102. DOI: `https://doi.org/10.1109/VL-HCC57772.2023.00018`.

Teaching AI Ethics through a Reading Project: A Structured and Reflective Approach[*]

Jingsai Liang[1] and Xi Chen[2]
[1]Computer Science Department
Westminster University
Salt Lake City, UT, 84105
`jliang@westminsteru.edu`
[2]Computer Science Department
Utah Valley University
Orem, UT, 84058
`xi.chen@uvu.edu`

Abstract

Artificial intelligence (AI) is rapidly reshaping many aspects of modern life, raising significant ethical questions that computer science (CS) students must be equipped to navigate. However, ethics can be a difficult subject to teach, particularly in technical courses. This paper presents a practical approach to teaching AI ethics through a structured reading and reflection project implemented in an undergraduate AI course. Students chose one of six contemporary AI ethics topics and responded to nine guiding questions, culminating in a written report and oral presentation. This format encouraged research, critical thinking, and personal reflection. With this project, the course provided a close-knit environment for exploring these complex issues. Results suggest that students developed a more nuanced understanding of AI ethics and greater awareness of their responsibilities as future developers. We share our project structure, implementation details, student insights, and lessons learned for educators aiming to incorporate ethics into technical courses.

---

[*] 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

Artificial intelligence (AI) is increasingly influencing decisions in high-stakes domains such as healthcare, criminal justice, finance, and national security, raising urgent ethical questions about accountability, fairness, privacy, and long-term societal impact. Authors in [10, 7] provided a systematic and comprehensive overview of AI ethics, including introductions, guidance, proposed solutions, evaluation frameworks, and goals. For instance, as also highlighted in [18], many real-world AI systems—especially those driven by machine learning—inherit biases from training data, leading to discriminatory outcomes that reinforce social inequities. While in [3], authors study the ethical implications of intelligent machines, with a particular focus on ensuring that their behavior remains aligned with human values and intentions as they become more intelligent than humans. As for solutions, authors in [14] proposed FAST Track principles—Fairness, Accountability, Sustainability, and Transparency—serve as foundational ethical pillars designed to guide responsible AI innovations throughout the AI project life cycle. Together, these works underscore that ethical considerations are not peripheral to AI design, but foundational to its responsible development and use.

Given these risks, paper [11] identified 84 documents containing ethical principles or guidelines for AI across various countries. Concurrently, AI ethics education has been implemented by embedding ethics modules within computer science courses [13, 19, 4] like Harvard's Embedded EthiCS [8].

Teaching ethics directly within AI courses is especially important—students will soon build systems that affect real lives [5]. Without ethical grounding, purely technical instruction risks producing graduates who fail to recognize harmful societal impacts. Existing AI ethics education has also been empirically demonstrated through case studies [9], interactive activities [21], and structured lesson plans [15]. However, many approaches depend on extensive redesign or are difficult to scale in single-instructor settings.

To address this, we designed a project-based module focused on reading, research, and structured reflection. The core goal was to foster student engagement with AI ethics through guided inquiry. Students selected one of six AI ethics topics and produced both a written report and a short presentation in response to nine carefully designed questions. This assignment emphasized real-world connections, stakeholder perspectives, current solutions, and critical reflection.

Our reading-and-reflection project offers a novel, structured approach: students independently research one of six central AI ethics topics, guided by nine reflection prompts, culminating in a written report and short presentation. This design promotes autonomy, critical thinking, and personalized insight into ethical issues—without additional staffing or course overhaul.

The remainder of this paper is organized as follows. Section 2 details the Project Design and nine-question framework. Section 3 outlines Project Implementation, including timeline and instructional support. Section 4 presents Project Reflections, analyzing student understanding and thematic outcomes. Finally, Section 5 concludes with implications and next steps.

2 Project Design

As the culminating project of the AI course, each student conducted independent research on a specific topic related to AI ethics. The goal of this project is to deepen students' understanding of the ethical challenges and responsibilities associated with AI through critical reading, analysis, and reflection.

Students chose one of the topics listed below. Each topic is accompanied by 3–5 suggested resources, such as research articles, conference papers, blogs, technical reports, videos, online courses, or news. Students are encouraged to explore additional sources beyond those provided.

1. Bias and Fairness [6]

   Bias and fairness in AI refer to how systems make decisions that may impact different groups unequally, often due to flawed data or biased design. Ensuring fairness means identifying and mitigating discrimination.

2. Privacy and Surveillance [16]

   This topic focuses on how AI technologies collect, store, and analyze personal data, raising concerns about individual privacy, consent, and government or corporate surveillance.

3. Autonomous Weapons and Military AI [1]

   This topic explores the ethics of using AI in weapons systems capable of making lethal decisions without human input. Discussions often center on accountability, war ethics, and international law.

4. Value Alignment and Superintelligence [22, 12]

   AI systems must be aligned with human values to ensure their goals remain beneficial. This topic considers the risk of misaligned AI systems, particularly as capabilities approach or exceed human intelligence.

5. Accountability and Explainability [20, 2]

   Accountability refers to who is responsible when AI systems cause harm. Explainability deals with how understandable a system's decisions are to humans, especially in high-stakes domains.

6. AI and the Future of Work [17]

   This topic investigates how AI impacts employment, including job displacement, workforce transformation, and the creation of new roles, as well as the ethical obligations to workers.

Students are required to prepare a slide presentation for class and write a report of at least two pages responding to the following questions. All references must be properly cited in both the written report and presentation slides.

1. What is the core ethical issue related to your topic?
2. What are the key challenges AI faces in this area today?
3. What real-world examples demonstrate the impact of this issue?
4. Who are the stakeholders affected, and how?
5. What are the consequences if this issue is not addressed?
6. What current solutions or strategies exist to tackle this issue?
7. What limitations or risks do those solutions present?
8. What should be done next in future by AI developers, policymakers, or society?
9. How has your understanding of this issue changed before and after your research? (Additional requirement on this question: this question is the reflection of your understanding. Students need to write at least 200 words to answer this question.)

3 Project Implementation

The project was implemented in an upper-level AI course and introduced during the final month of the semester. Students were given 2–3 weeks to explore their chosen topic and complete both a written report and presentation slides. At the start of the project, each topic was briefly introduced to help students make informed decisions. Example readings and resources were provided; however, students were encouraged to explore additional sources independently.

In each subsequent class, the instructor offered informal check-ins and personalized support. Class time was also used to discuss research strategies, citation practices, and how to evaluate sources. While students worked independently, they were invited to consult with the instructor for feedback on early drafts and presentation outlines.

The final week was dedicated to student presentations. Each student delivered an 8–10-minute presentation with slides, summarizing their key findings and reflections. Both the report and the slides were required to include proper citations. Presentation slides were expected to include the following sections:

- Introduction
- Past and current status of the topic
- Summary of key findings from sources
- Critical personal reflection and future outlook
- References

The written report was assessed based on clarity, completeness, use of evidence, and quality of reflection. Presentations were graded on organization, clarity, delivery, and slide quality.

4 Project Reflections

In their reports and presentations, students engaged in a thorough and meaningful learning process. They summarized their findings in response to Questions 1–8 and offered personal reflective comparisons—before and after the project—on their chosen topics in Question 9. Overall, for each assigned topic, students explored the past, present, and future challenges, complexities, and impacts. They gained insight into the consequences of failing to address AI ethics properly, as well as the limitations of both current and potential future solutions.

To illustrate the depth of student engagement with external materials, Table 1 presents the average number of references cited in each topic area. These numbers highlight that students actively explored a wide range of scholarly and real-world sources. This independent research effort reflects not only their strong interest in the ethical dimensions of AI but also the development of valuable academic skills, including literature review, source evaluation, and critical synthesis.

Topics ID	1	2	3	4	5	6
Average Number of References	9	4	6.5	6	9.5	4

Table 1: Average Number of References in Each Topic

Student responses to the final reflection question 9 revealed substantial growth in their understanding of AI ethics. While most students began the project with limited or generalized knowledge of their chosen topic, their post-project reflections demonstrated increased depth, critical awareness, and recognition of the complexity surrounding ethical challenges in AI.

For the Bias and Fairness topic, students initially understood bias in AI as a technical issue—primarily related to training data. Through their research, they discovered how systemic inequalities are encoded into algorithms, reinforcing real-world disparities in areas such as hiring and law enforcement. They were particularly struck by the challenges of operationalizing fairness and the difficulties of addressing intersectional concerns in algorithmic systems.

Students who explored Privacy and Surveillance shifted from viewing surveillance as limited to browser tracking or public cameras to recognizing it as a widespread and deeply embedded practice involving mobile devices, social media platforms, and cross-system data aggregation. They were surprised to learn how little control individuals have over their personal data and how vague or outdated many privacy regulations are in the face of advancing AI capabilities.

Those studying Autonomous Weapons and Military AI gained awareness of the distinctions between unmanned, semi-autonomous, and fully autonomous systems. They expressed concern about the speed of development in military AI and the lag in ethical and legal frameworks. Reflections emphasized the dangers of black-box decision-making in life-or-death scenarios and the erosion of human accountability in warfare.

In the Value Alignment and Superintelligence category, students moved beyond general concern about AI safety to grapple with the philosophical and practical challenges of aligning AI with human values. Some questioned whether current alignment strategies are truly viable, given the inconsistency and sometimes harmful nature of human history and behavior. They considered alternative approaches to training AI systems that avoid simply mirroring flawed human norms.

Students investigating Accountability and Explainability began with a focus on technical transparency but came to appreciate the broader socio-technical dynamics at play. They noted that explainability must be tailored to different audiences—technical, regulatory, and public—and recognized that policy enforcement, institutional incentives, and regulatory oversight are as critical as algorithmic clarity. Real-world examples helped them see how poor design or rushed deployment can obscure accountability and exacerbate harm.

Finally, students examining AI and the Future of Work reflected on widespread uncertainty about AI's impact on employment. While they acknowledged both threats and opportunities, they emphasized that adaptability and AI literacy will be crucial in the evolving job market. They also noted the emotional and professional challenges of keeping pace with rapidly advancing technologies, particularly as AI becomes more capable of performing traditionally human tasks.

In summary, the reflection component of the project encouraged students to move beyond surface-level understanding and engage critically with their topics. Their responses showed evidence of personal transformation, deeper ethical reasoning, and awareness of the broader societal implications of AI. These insights underscore the effectiveness of combining structured inquiry with open-ended reflection in ethics education.

5 Conclusion and Future Work

This reading-based ethics project provided a meaningful and engaging experience for students in the AI course. Through guided, independent inquiry, students were able to explore complex and often ambiguous ethical issues in depth. Their reflections revealed significant growth in ethical awareness, critical thinking, and the ability to connect technical concepts to real-world consequences. Many students shifted from superficial understandings to nuanced, well-reasoned perspectives on topics such as algorithmic bias, surveillance, and the societal implications of autonomous systems.

One of the primary contributions of this approach is its adaptability for instructors seeking to integrate ethics into a technical AI course without requiring a co-instructor from philosophy or major curriculum overhauls. By using structured questions and student-chosen topics, the project balances rigor with flexibility. Instructors can adopt this model with minimal resources while still achieving high-impact outcomes in ethical reasoning and student engagement.

Compared to traditional approaches—such as lecture-based coverage or standalone ethics modules—this project fosters deeper personal investment and active learning. Students are not merely exposed to ethical theories but are challenged to explore, synthesize, and reflect on them within the context of current AI practices. The inclusion of both writing and oral presentation also supports the development of communication skills crucial for future AI practitioners.

That said, this approach is not without limitations. Its success depends on student motivation and research ability, which may vary significantly. Additionally, in larger classes, providing individualized feedback and support could become challenging without teaching assistants or scaffolding mechanisms in place. The quality of student learning also depends on the availability of well-curated topic materials and clear guidance.

In future iterations, the project could be enhanced by incorporating structured peer review, milestone check-ins, and sample reports to model expectations. Guest speakers, collaborative topic exploration, or cross-disciplinary co-teaching could further enrich the experience. Additionally, exploring more connections to other computer science courses—such as database or algorithms—could help students better integrate ethical inquiry with their technical knowledge.

Looking more broadly, the teaching of AI ethics will likely become an essential component of computer science education [13], not an optional supplement. As AI systems become more powerful and ubiquitous, students must be equipped not only with the tools to build them but with the ethical frameworks to evaluate and regulate them. Approaches like this reading project demonstrate that it is possible to cultivate ethical awareness through inquiry-based, student-centered learning—and that doing so is both impactful and feasible within a standard AI course.

References

[1] Peter Asaro. *Autonomous Weapons and the Ethics of Artificial Intelligence.* Sept. 2020, pp. 212–236.

[2] Nagadivya Balasubramaniam et al. "Transparency and explainability of AI systems: From ethical guidelines to requirements". In: *Information and Software Technology* 159 (2023).

[3] Nick Bostrom and Eliezer Yudkowsky. "The ethics of artificial intelligence". In: *Artificial intelligence safety and security.* Chapman and Hall/CRC, 2018, pp. 57–69.

[4] Noelle Brown et al. "Teaching Ethics in Computing: A Systematic Literature Review of ACM Computer Science Education Publications". In: *ACM Trans. Comput. Educ.* 24.1 (Jan. 2024).

[5] Emanuelle Burton et al. "Ethical Considerations in Artificial Intelligence Courses". In: *AI Magazine* 38.2 (2017), pp. 22–34.

[6] Zhisheng Chen. "Ethics and discrimination in artificial intelligence-enabled recruitment practices". In: *Humanities and Social Sciences Communications* 10.1 (2023).

[7] Luciano Floridi. *The ethics of artificial intelligence: Principles, challenges, and opportunities.* Oxford University Press, 2023.

[8] Barbara J. Grosz et al. "Embedded EthiCS: integrating ethics across CS education". In: *Commun. ACM* 62.8 (July 2019), pp. 54–61.

[9] Reiko Hishiyama and Tehgfei Shao. *Educational Effects of the Case Method in Teaching AI Ethics.* May 2022, pp. 226–236.

[10] Changwu Huang et al. "An overview of artificial intelligence ethics". In: *IEEE Transactions on Artificial Intelligence* 4.4 (2022), pp. 799–819.

[11] Anna Jobin, Marcello Ienca, and Effy Vayena. “The global landscape of AI ethics guidelines”. In: *Nature Machine Intelligence* 1.9 (2019), pp. 389–399.

[12] HyunJin Kim et al. *The Road to Artificial SuperIntelligence: A Comprehensive Survey of Superalignment.* 2024. arXiv: `2412.16468`. URL: `https://arxiv.org/abs/2412.16468`.

[13] Matthew Kopec et al. “The effectiveness of embedded values analysis modules in Computer Science education: An empirical study”. In: *Big Data & Society* 10.1 (2023).

[14] David Leslie. *Understanding artificial intelligence ethics and safety: A guide for the responsible design and implementation of AI systems in the public sector.* 2019. URL: `https://zenodo.org/record/3240529`.

[15] Phoebe Lin and Jessica Van Brummelen. “Engaging Teachers to Co-Design Integrated AI Curriculum for K-12 Classrooms”. In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems.* CHI ’21. Yokohama, Japan: Association for Computing Machinery, 2021.

[16] Msbah J. Mosa et al. “Ai and Ethics in Surveillance: Balancing Security and Privacy in a Digital World”. In: *International Journal of Academic Engineering Research (IJAER)* 8.10 (2024), pp. 8–15.

[17] Filippo Santoni de Sio. “Artificial Intelligence and the Future of Work: Mapping the Ethical Issues”. In: *The Journal of Ethics* 28.3 (2024), pp. 407–427.

[18] Jonathan Shaw. “Artificial intelligence and ethics”. In: *Harvard Magazine* 30 (2019), pp. 1–11.

[19] Maya Usher and Miri Barak. “Unpacking the role of AI ethics online education for science and engineering students”. In: *International Journal of STEM Education* 11.1 (2024).

[20] Heidi Vainio-Pekka et al. “The Role of Explainable AI in the Research Field of AI Ethics”. In: *ACM Trans. Interact. Intell. Syst.* 13.4 (Dec. 2023).

[21] Randi Williams et al. “AI + Ethics Curricula for Middle School Youth: Lessons Learned from Three Project-Based Curricula”. In: *International Journal of Artificial Intelligence in Education* 33.2 (2023), pp. 325–383.

[22] Yi Zeng et al. *Redefining Superalignment: From Weak-to-Strong Alignment to Human-AI Co-Alignment to Sustainable Symbiotic Society.* 2025. arXiv: `2504.17404`. URL: `https://arxiv.org/abs/2504.17404`.

A Comparative Study of Architectural Patterns in Flutter: A Case Study and Teaching Framework for Undergraduate Mobile Development*

Madhav Neupane and Liqiang Zhang
Computer and Information Sciences
Indiana University South Bend
South Bend, IN 46615
`{mneupane, liqzhang}@iu.edu`

Abstract

Teaching architectural patterns in undergraduate mobile development courses is challenging due to their abstract nature and the need for engaging, practical examples. This paper presents a comparative study of four architectural patterns—Model-View-Controller (MVC), Model-View-ViewModel (MVVM), Model-View-Presenter (MVP), and View-Interactor-Presenter-Entity-Router (VIPER)—implemented in a Flutter-based Virtual Visiting Card application. The app, featuring Firebase authentication, OCR-based business card scanning, drag-and-drop field mapping, and Firestore cloud storage with SQLite caching, serves as a robust case study. We evaluate metrics such as test coverage, Firestore latency, onboarding time, and maintainability, highlighting trade-offs. To support educators, we provide a GitHub repository with the Virtual Visiting Card app and a list of additional example apps, all implemented across the four patterns. We propose a detailed lab assignment, a sample lesson plan, and a rubric, making this study a valuable resource for teaching mobile development using Flutter. Implementation challenges

---

*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.

and lessons learned further enrich the framework, guiding instructors in delivering these concepts effectively.

1 Introduction

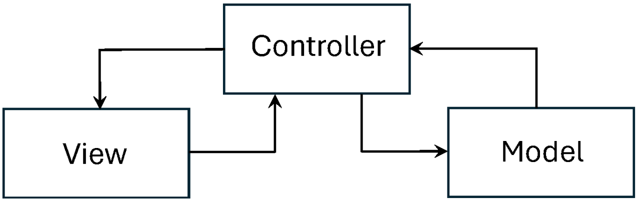
Mobile application development is increasingly popular in undergraduate computer science curricula, driven by the demand for cross-platform solutions like Flutter [2]. However, teaching architectural patterns such as MVC, MVVM, MVP, and VIPER poses challenges: lectures can become dry, examples are often too simplistic, and time constraints limit in-depth exploration [12]. This paper addresses these issues through a comparative study of these patterns, primarily using a Flutter-based Virtual Visiting Card application as a case study. The app, with features like Firebase authentication, Google ML Kit OCR, drag-and-drop field mapping, Firestore storage and synchronization, and SQLite caching, provides a real-world context for students to explore architectural trade-offs. To enhance pedagogical flexibility, we supplement the case study with a GitHub repository containing additional example apps, each implemented in MVC, MVVM, MVP, and VIPER [6]. These apps, ranging from simple to moderately complex, enable educators to tailor assignments to diverse student levels. Our study evaluates each pattern across architectural quality (e.g., separation of concerns), maintainability, testability, and development costs, using metrics like test coverage and onboarding time. Drawing from a graduate project [5], we adapt the findings into a teaching framework, including a lab assignment, lesson plan, and rubric. By providing concrete examples and educational tools, this paper equips educators to make architectural patterns engaging and accessible, aligning with the CCSC's focus on practical pedagogy.

2 Background

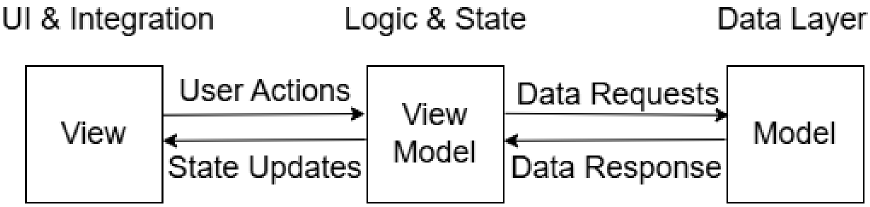
Software architectural patterns define the organization of code components to achieve the separation of concerns [7], [10]. Below, we briefly describe the four patterns under study, illustrated in Figure 1.

2.1 Model View Controller (MVC)

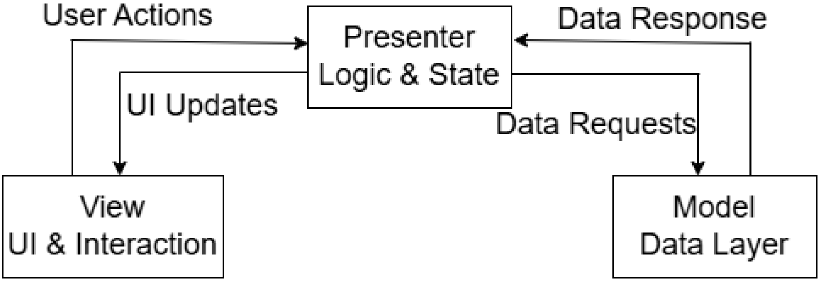
MVC divides an application into three components: the Model (data and business logic), the View (user interface), and the Controller (input handling and coordination). The Controller updates the Model based on user input and refreshes the View, which observes the Model [3].



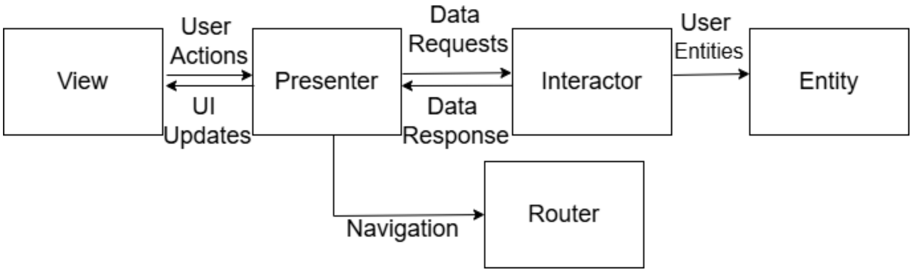
(a) MVC Architecture



(b) MVVM Architecture



(c) MVP Architecture



(d) VIPER Architecture

Figure 1: Architectural Patterns: MVC, MVVM, MVP & VIPER

2.2 Model View ViewModel (MVVM)

MVVM replaces the Controller with a ViewModel, which exposes data and commands to the View via data bindings. The ViewModel abstracts the Model, enabling reactive UI updates. MVVM is well-suited for data-binding-enabled frameworks [3].

2.3 Model View Presenter (MVP)

In MVP, the Presenter acts as an intermediary between the Model and a passive View. The Presenter handles all logic, retrieving data from the Model and updating the View via explicit calls. This separation of concerns enhances testability by decoupling business logic [3].

2.4 View Interactor Presenter Entity Router (VIPER)

VIPER is a modular pattern dividing responsibilities into View, Interactor (business logic), Presenter (view coordination), Entity (data objects), and Router (navigation). VIPER's granularity supports large-scale applications but increases setup complexity [3].

3 Methodology

The Virtual Visiting Card application, developed using Flutter (v3.22.2) and Dart (v3.4.3), serves as the testbed for this comparative study of four architectural patterns: MVC, MVVM, MVP, and VIPER. The app enables users to digitize business cards by scanning them with OCR technology (via google_mlkit_text_recognition), mapping extracted text to contact fields (e.g., name, mobile, email, location, address, company, designation, website) through a drag-and-drop interface, and managing contacts using Firebase Firestore for cloud storage with SQLite (via sqflite) for offline caching. Additional features include favorites management and external integrations for calling, messaging, emailing, map navigation, and website access.

Each implementation adheres to the same core tech stack and dependencies:

- go_router (for navigation)
- path_provider (for file access)
- image_picker (for camera input)
- get_it (for dependency injection)
- google_mlkit_text_recognition (for OCR)
- sqflite (for SQLite-based offline caching)
- url_launcher (for external integrations: calling, messaging, emailing, maps, web)

By refactoring the same codebase from MVC → MVVM → MVP → VIPER, we decouple architectural implications from feature changes. We began with an MVC implementation and successively refactored to MVVM, then to MVP, and ultimately VIPER, maintaining features, UI, and theming (deep-purple Material Design) identical across versions.

To evaluate each pattern, we captured both quantitative and qualitative data across four axes [5]:

- **Code Understandability**: Measured using Average Cyclomatic Complexity (lower values indicate simpler code) and Average Maintainability Index (higher values indicate easier maintenance), computed via dart_code_metrics.
- **Unit Test Coverage**: Quantified as the percentage of code lines covered by unit tests, measured using ***flutter test --coverage*** and ***genhtml*** for report generation.
- **Codebase Complexity**: Assessed by counting Dart files, classes, total imports, and average imports per class in the ***lib/*** directory, reflecting structural overhead.
- **Performance Overhead**: Evaluated by Firestore fetch latency (***ms***) for retrieving all contacts, averaged over 10 runs per pattern, tested on a physical iOS device.

This methodology provides a structured and comprehensive framework for analyzing the strengths and weaknesses of each architectural pattern in the context of a real-world Flutter application, offering valuable insights for educators and developers alike.

4 Implementations

This section details the implementation of the four architectural patterns in the Virtual Visiting Card application. To streamline the discussion and emphasize the most contrasting patterns, we only highlight the implementation of two patterns: MVC and VIPER; please refer to our project report [5] for a discussion on the implementation of MVP and MVVM. Each pattern is applied to the same application, ensuring identical features and user interfaces across all implementations to facilitate fair comparisons. The focus is on code structure, key components, and data flow, highlighting the practical differences and trade-offs between patterns, as detailed in the report [5].

4.1 MVC Implementation

As shown in Figure 1, the MVC implementation organizes the application into three primary components: Model, View, and Controller. The directory structure is as follows:

```
lib/
  controllers/: Contains logic for authentication, contact CRUD,
        and list updates (e.g., auth_controller.dart,
        contact_controller.dart, contact_list_controller.dart).
  models/: Defines data structures (e.g., contact_model.dart,
           user_model.dart).
  pages/: Houses UI components (e.g., auth_wrapper.dart,
          camera_page.dart, contact_details_page.dart).
  database/: Manages SQLite interactions (db_helper.dart).
  utils/: Stores reusable constants and functions.
```

The total file count is 20.

Key Components:

- **Model**: ContactModel encapsulates contact data (e.g., name, mobile, email) with serialization logic for SQLite storage. UserModel maps Firebase user data (e.g., UID, email, display name).
- **View**: Implemented as Flutter StatefulWidgets (e.g., HomePage, CameraPage) that render the UI and handle user input. Views use StreamBuilder for reactive updates based on state changes.
- **Controller**: AuthController manages Firebase authentication (e.g., sign-up, sign-in). ContactController handles CRUD operations for contacts in SQLite. ContactListController manages contact list updates using StreamController, ensuring user-specific operations and real-time updates.

4.2 VIPER Implementation

As shown in Figure 1, the VIPER implementation provides the most modular and testable structure, dividing responsibilities into View, Interactor, Presenter, Entity, and Router. The directory structure is:

```
lib/
  database/: SQLite storage (db_helper.dart).
  entities/: Data models (contact_entity.dart).
  interactors/: Business logic (contact_interactor.dart,
                auth_interactor.dart).
  presenters/: UI coordination (home_presenter.dart).
  routers/: Navigation logic (app_router.dart).
  utils/: Shared constants and functions.
  views/: UI screens with interfaces (home_page.dart,
          camera_page.dart).
  main.dart: App initialization.
```

The total file count is 34.

Key Components:

- **Entity**: ContactEntity defines the data model for contacts, focused on SQLite storage.
- **View**: Implements interfaces (e.g., HomeView) as StatefulWidgets (e.g., HomePage).
- **Interactor**: Handles business logic and data access (e.g., ContactInteractor) using streams for updates.
- **Presenter**: Coordinates between Interactors and Views (e.g., HomePresenter), updating UI via interface methods.
- **Router**: AppRouter manages navigation using go_router, centralizing route definitions.

5 Issues Encountered During Implementation

This section outlines challenges faced during the implementation of the MVC, MVVM, MVP, and VIPER architectural patterns in the Virtual Visiting Card application, highlighting practical trade-offs for educators and students. Implementing the four patterns introduced varying complexities. MVC's tight coupling between Controllers and Views caused code duplication, particularly in error handling for Firebase authentication and SQLite operations, complicating maintenance. iOS-specific issues with url_launcher (e.g., sandbox restrictions for external apps) required platform-specific configurations. MVVM's refactoring to ViewModels and a Repository layer was effort-intensive, with challenges in ensuring Firestore-SQLite synchronization for offline caching. MVP's explicit View-Presenter contracts added boilerplate code, increasing complexity. VIPER's modular structure, with View, Interactor, Presenter, Entity, and Router components, posed a steep learning curve, particularly for centralized navigation with go_router. Despite these challenges, VIPER's modularity minimized failure points, while MVC, MVVM, and MVP required careful design to maintain data integrity and testability.

6 Comparative Analysis

This section delves into a detailed comparative analysis of the four architectural patterns. The evaluation meticulously examines four critical dimensions: Code Understandability, Unit Test Coverage, Codebase Complexity, and Performance Overhead. These dimensions are assessed using robust quantitative metrics, derived from tools such as dart_code_metrics for code quality, ***flutter test --coverage*** paired with ***genhtml*** for test coverage, counts of files, classes, and imports for structural complexity, and Firestore fetch latency for runtime performance.

6.1 Code Understandability

Code Understandability reflects how intuitively developers can grasp and maintain the codebase, measured through two insightful metrics: Average Cyclomatic Complexity, where lower values signify simpler, less convoluted code paths, and Average Maintainability Index, where higher scores indicate code that is easier to modify and sustain. These metrics are computed using the dart_code_metrics tool, offering a window into the clarity of each architectural pattern.

MVC weaves a straightforward yet somewhat tangled tapestry, with an Average Cyclomatic Complexity of 2.68 and an Average Maintainability Index of 68.03. Its simplicity aids quick comprehension, but the blending of concerns in Controllers can obscure clarity in larger projects. MVVM presents a slightly denser landscape, with an Average Cyclomatic Complexity of 2.72 and an Average Maintainability Index of 67.78. The introduction of ViewModels enhances separation but adds a layer of abstraction that may challenge initial understanding. MVP strikes a harmonious balance, boasting an Average Cyclomatic Complexity of 2.60 and a robust Average Maintainability Index of 74.79. Its contract-based structure clarifies responsibilities, making the codebase more approachable and maintainable. VIPER shines as the epitome of clarity, with an impressively low Average Cyclomatic Complexity of 2.24 and the highest Average Maintainability Index of 75.42. Its granular division into distinct components: View, Interactor, Presenter, Entity, and Router creates a crystal-clear structure, ideal for long-term maintenance.

VIPER emerges as the most lucid, followed by MVP, MVC, and MVVM, painting a picture where modularity triumphs in fostering understandable code.

6.2 Unit Test Coverage

Unit test coverage measures the percentage of codebase lines executed by automated tests, calculated using ***flutter test --coverage*** and visualized with ***genhtml***. This metric highlights the reliability and testability of architectural patterns, focusing on pure-Dart layers (data models, database/helper classes, and utility functions) that can be tested in isolation without external dependencies like Firebase SDKs, platform channels, or Flutter's widget framework. Controllers, remote-service classes, and UI widgets, which require extensive mocking or widget-test setups, were excluded to keep the test suite fast and simple.

MVC stands tall with an exceptional 90.1% coverage, testing 82 out of 91 lines across 5 files. Its compact codebase allows for thorough testing, though tight coupling may limit the depth of isolated unit tests [1], [4], [9], [11]. MVVM falters in this arena, achieving a modest 45.8% coverage, with only 163 of 356 lines tested across 6 files. The complexity of ViewModel logic and data-binding dependencies appears to hinder comprehensive testing. MVP delivers a commendable 85.5% coverage, testing 100 of 117 lines across 4 files. Its contract-based design facilitates mocking, ensuring robust testability with minimal gaps. VIPER follows closely with 82.3% coverage, testing 65 of 79 lines across 4 files. Its modular architecture supports isolated testing, though its slightly lower coverage compared to MVP may reflect the intricacy of Interactor and Router components [1], [4], [9], [11].

MVC leads as the most thoroughly tested, trailed by MVP, VIPER, and a distant MVVM, highlighting a surprising strength in MVC's testability despite its structural limitations.

6.3 Codebase Complexity

Codebase Complexity captures the structural intricacy of each pattern, quantified by the number of Dart files, classes, total imports, and average imports per class within the lib/ directory. These metrics reveal the architectural overhead required to support the application's functionality.

MVC offers the simplest structure, with 20 Dart files, 29 classes, 108 total imports, and an average of 3.72 imports per class. Its lean design minimizes overhead, ideal for rapid development but potentially at the cost of scalability[11]. MVVM steps up in complexity, featuring 25 Dart files, 31 classes, 126 total imports, and an average of 4.06 imports per class. The addition of ViewModels and Repositories enriches separation but demands greater organizational effort[11]. MVP further escalates complexity, with 31 Dart files, 31 classes, 118 total imports, and an average of 3.80 imports per class. Its contract-based approach introduces additional components, balancing testability with structural weight. VIPER embodies the pinnacle of complexity, comprising 34 Dart files, 35 classes, 131 total imports, and an average of 3.74 imports per class. Its highly modular design, while robust, requires significant setup and maintenance, suited for expansive applications[11].

MVC reigns as the least complex, followed by MVVM, MVP, and VIPER, illustrating a spectrum from simplicity to sophisticated modularity.

6.4 Performance Overhead

Performance Overhead is gauged by Firestore fetch latency, measured in milliseconds for retrieving all contacts, averaged over 10 runs on a physical iOS device. This metric highlights the runtime efficiency of each architectural pattern.

MVC excels with a swift 303.9 ms latency, its streamlined data flow minimizing processing overhead and delivering optimal performance for simple operations [1], [8], [9]. MVVM follows with a latency of 328.8 ms, reflecting minor additional processing required by ViewModels and the repository layer, yet maintaining high efficiency[1], [8], [9]. MVP records a higher latency of 520.4 ms, its Presenter layer and view contracts introducing noticeable delays in coordinating data retrieval [1], [8], [9]. VIPER trails with the highest latency of 572.2 ms, its intricate data flow through multiple components: Interactors, Presenters, and Routers adding significant overhead, a trade-off for its modularity [1], [8], [9].

MVC leads in performance, followed by MVVM, MVP, and VIPER, underscoring the cost of layered architectures in runtime efficiency.

6.5 Summary of Comparative Analysis

The following table summarizes the performance of each pattern across the evaluation dimensions:

Table 1: Comparative Performance of Architectural Patterns

Metric	**MVC**	**MVVM**	**MVP**	**VIPER**
Cyclomatic Complexity	2.68	2.72	2.60	2.24
Maintainability Index	68.03	67.78	74.79	75.42
Test Coverage (%)	90.1	45.8	85.5	82.3
Files Count	20	25	31	34
Classes Count	29	31	31	35
Total Imports	108	126	118	131
Avg Imports/Class	3.72	4.06	3.80	3.74
Fetch Latency (ms)	303.9	328.8	520.4	572.2

Each pattern offers distinct advantages:

- **MVC**: Simplest for rapid prototyping but lacks scalability due to high coupling.
- **MVVM**: Moderately modular with improved separation via repositories, but complex ViewModel interactions hinder testing and scalability.
- **MVP**: Excels in testability, suitable for test-driven development.
- **VIPER**: Most modular, with granular layers ensuring clear separation and high maintainability, ideal for large-scale systems despite development complexity.

This comparative analysis underscores the importance of selecting an architectural pattern based on project scale, team expertise, and priorities such as speed, maintainability, or testability. Educators can use these findings to guide students in understanding the real-world implications of architectural choices in Flutter development.

7 Educational Applications

To integrate this comparative study of MVC, MVVM, MVP, and VIPER into undergraduate computer science curricula, we propose a teaching framework detailed in the online and publicly accessible project report[5], comprising a lab assignment, a lesson plan, and a rubric. The framework leverages a GitHub repository containing the Virtual Visiting Card application and three additional example apps, each implemented in all four patterns. The lab assignment guides students to analyze and refactor any features from MVC to MVVM, comparing metrics like Cyclomatic Complexity and Maintainability Index. The lesson plan introduces patterns via code snippets and demos, followed by an interactive quiz. The rubric assesses code functionality, quality, and analysis. This approach detailed in the section 9 of the project report [5], fosters hands-on learning, addressing the challenge of abstract lectures by engaging students with practical Flutter development tasks.

8 Conclusion

This study of MVC, MVVM, MVP, and VIPER in a Flutter-based Virtual Visiting Card app, supplemented by a repository of additional example apps [6], provides a robust case study and teaching framework for undergraduate mobile development. By offering detailed implementations, metrics, a lab assignment, a lesson plan, and Github repositories, we empower educators to teach architectural patterns in an effective and impactful way.

References

[1] D. Dobrean and L. Dioşan, “A comparative study of software architectures in mobile applications,” *Studia Univ. Babeș-Bolyai, Informatica*, vol. LXIV, no. 2, pp. 49–64, 2019.

[2] Flutter Team, *Flutter: Build apps for any screen*, `https://flutter.dev/`, 2025.

[3] A. Koval, “Comparative analysis of modern ios architectures in different development stages,” Bachelor’s Thesis, Ukrainian Catholic University, 2021.

[4] H. Magics-Verkman, R.-Ș. Győrödi, D. R. Zmaranda, and C. A. Győrödi, "A comparison of architectural patterns for testability and performance quality for ios mobile applications development," in *Proceedings of the 17th International Conference on Engineering of Modern Electric Systems (EMES 2023)*, IEEE, Jul. 2023, pp. 49–64.

[5] M. Neupane, "A comparative study of mvc, mvvm, mvp, and viper architectural patterns in flutter: Implementation analysis in a virtual visiting card application," Indiana University South Bend, Tech. Rep., 2025. [Online]. Available: `https://www.cs.iusb.edu/~liqzhang/mapr.pdf`.

[6] M. Neupane, *Github profile: Virtual visiting card and additional example apps*, `https://github.com/Madhav1022/`, 2025.

[7] R. Nunkesser, "Using hexagonal architecture for mobile applications," in *Proceedings of the 17th International Conference on Software Technologies (ICSOFT 2022)*, 2022, pp. 113–120.

[8] C. J. Sampayo-Rodríguez, R. González-Ambriz, B. A. González-Martínez, and J. Aldana-Herrera, "Processor and memory performance with design patterns in a native android application," *Journal Applied Computing*, vol. 6, no. 18, pp. 53–61, 2022.

[9] F. Sholichin, M. A. bin Isa, S. A. Halim, and M. F. B. Harun, "Review of ios architectural pattern for testability, modifiability, and performance quality," *Journal of Theoretical and Applied Information Technology*, vol. 97, no. 15, pp. 4021–4033, 2019.

[10] R. Verdecchia, I. Malavolta, and P. Lago, "Guidelines for architecting android apps: A mixed-method empirical study," in *2019 IEEE International Conference on Software Architecture (ICSA 2019)*, IEEE, Mar. 2019, pp. 141–150.

[11] A. Wilson, F. Wedyan, and S. Omari, "An empirical evaluation and comparison of the impact of mvvm and mvc gui driven application architectures on maintainability and testability," in *2022 International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*, IEEE, Jun. 2022, pp. 101–108.

[12] L. Zhang, "Teaching cross-platform mobile development and cultivating self-directed learners – a six-week summer online course experience," *Journal of Computing Sciences in Colleges*, vol. 39, no. 7, pp. 41–51, 2024.

Engaging with the Wiphala: A Code-Generation Hardened CS1 Final*

Calvin Deutschbein and Shouvik Ahmed Antu
Willamette University
Salem, OR 97301
`{cdeutschbein, sahmed}@willamette.edu`

Abstract

As co-instructors in introductory computing classes at a small college, in Fall 2024 we needed an approach to code generation tools that at once equipped new students to be effective in a world in which those tools existed, but also still benefited from the computational thinking and literacy skills often covered in the first semester of a computing course. Many introductory computing assignments, including some subset of SIGCSE Nifty assignments, remain perhaps too challenging for many introductory students even when specified to such a level that freely available code generation tools, such as ChatGPT or Gemini, are able to produce perfect solutions. However, there remain many tasks that are quite difficult for code generation tools, which by construction are specialized to text analysis and text generation. In our Fall 2024 introductory courses, we deployed an experimental final for which the prompt was to generate a certain form of image informed by cultural context to communities adversely impacted by the surge in data center demand in the U.S.. Students would necessarily analyze and describe an image, and critically engage with the consequences of over-reliance on code generation, in order to complete the assignment. We achieved our desired grade distributions and believe we differentiated students who were learning material from students who were outsourcing coursework to automated tools.

---

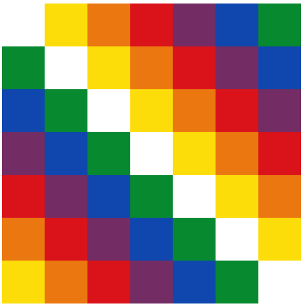
*Copyright ©2025 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear,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

At our university, the entry point for computer science education is “CS 151: Introduction to Programming in Python”, which is a Python translation of an older class in Java based on a Stanford curriculum leveraging SIGCSE Nifty assignments such as Adventure Game [5], Breakout [6], Enigma Machine [7], and Wordle [4]. These are established assignments, with years of success training new computer scientists, and extremely in-depth instructions provided directly to students and available widely online. Coincidentally, making these assignments more accessible for students has also made it easier for students to not engage with these assignments at all due to the availability of ChatGPT, Gemini, and similar frameworks capable of code generation. We perceive a growing consensus that reliance on such tools hampers cognitive development [3] and therefore should be addressed in course design.

In this paper, we will present our novel final on rendering the Wiphala (Fig 1). We believe this final supports student learning and allows students to engage with code generation and its limitations.

Figure 1: The Wiphala is a square emblem commonly used as a flag to represent some native peoples of the Andes.



We are motivated by a particularly troubling phenomenon. The CS1 curriculum leverages a Python port of the Portable Graphics Library “PGL” [8] which often confounded earlier code generators expecting PyGame or similar libraries. However, specifically in the case of the Breakout assignment, both ChatGPT and Gemini generated correct solutions without being provided a description of the Python PGL library while struggling on more general tasks.

We suspect exact solutions to Nifty assignments are present within training data in some capacity. Nifty assignments are unified in having clear (1) learning objectives and (2) assignment formats, which we can mimic for novel assignments intended to defeat code generation.

Our final has the following learning objectives:

1. Fundamental Python
2. Reading Python
3. Strings and Files
4. Interactive Graphics
5. Defining Classes

As a rule, “Strings and Files” maps to the text-based Adventure Game, “Interactive Graphics” maps to graphical Breakout game, and “Defining Classes” maps to the “Enigma Machine” ciphers. “Fundamental Python” and “Reading Python” are expected to be manageable by students with debugging experience. We mapped these objectives to what we expect to be AI-hard subproblems, synthesized subproblems into a coherent assignment, and applied the assignment to a culturally relevant application domain.

We found:

1. Our approach achieved a grade distribution consistent with pre-code generation classes and our personal assessment of student engagement, while allowing use of code generation.
2. Our approach maintained the core learning objectives of an ACM 2013 [2] CS1 course.
3. Our approach equipped students to understand the benefits and limitations of code generation.
4. Our approach raised critical questions about the broader social impacts.

We have made a generalized form of this assignment (replacing all dependencies with PyPI alternatives) available publicly on GitHub [1], with web-hosted instructions[1], available under MIT License for all code artifacts and with fair use images documented with source attribution. We will provide solutions by request to verified instructors.

2 Format

Our introductory course was structured across lectures, problem sets, and projects but the primary learning content is delivered via the projects - the

---

[1] `https://cd-public.github.io/ccscrm/`

four (4) Nifty assignments and a fifth pseudo-Nifty assignment "ImageShop", a rudimentary image editing program.

We regard the first three projects as *scripting* assignments. They are assigned over the course of one week and completed in one file (or module). For these assignments. Students write functions which interact with a provided template. The final two projects - "Enigma Machine" and "Adventure Game" were *programming* assignments spanning multiple weeks, multiple files, and potential multiple team members. In Fall 2024, the average grades dropped sharply when we transitioned from the scripting assignments (averages of 84, 87, 87 percent) to the programming assignments (averages of 79, 69 percent). We believe this drop may be associated with lacking the technical skill to prompt a code generator for more than isolated functions. Therefore, we administered a highly interdependent final exam.

Each assignment was split into "milestones" well suited to daily work. For example, the "Wordle" assignment was released Monday, had six zero-indexed milestones and was due the next Monday, giving students a final day after expected completion to debug any last-minute errors. This milestone structure and its familiarity to students supported a similarly structured final with multiple learning objectives as objectives could be mapped to milestones.

We assigned problem sets and projects via GitHub classroom as a code repository and separately provided instructions in HTML. In response to student feedback, we organized the assignment descriptions into one HTML `<details>` block per milestone.

3 Learning Objectives

We will describe a learning objective, how widespread availability of code generators has impacted assessment, and our solution.

3.1 Strings and Files

Existing finals prompted students to, for example, find the first numeric digit in every line of a file containing lines of ASCII characters. This is trivial for code generation or programmers familiar with library functions.

As an alternative, we ask students to read the names of colors and their hex values from a file. Critically, we introduce what we term *intentional ambiguity*, that is, we do not specify the format in the file or the format which will be most useful for latter milestones. Rather, students must design and then implement a solution.

While we provide a CSV file, and reading CSV files into Python is trivial, we pose a number of complexities:

```
rojo 219 10 19
naranja 236 120 8
amarillo 252 222 2
blanco 255 255 255
verde 1 138 44
azul 6 69 177
violeta 117 40 100
```

We mix alphabetical and numerical values, provide no column labels, provide non-English color names, and space-separate rather than comma-separate. We refer to these as "red-green-blue" color descriptions in the text but otherwise expect students to either know or be able to determine how to work with color data in this format. Students also must independently determine whether, for example, to return multiple lists of strings and integers, or a dictionary of lists, or any other solution. We saw numerous possible solutions within high-scoring final submissions, and are pleased to have perceived space for students to make design decisions.

3.2 Interactive Graphics

Existing final examinations prompted students to, for example, render a pie chart from a user-provided list of percentages. This is obviously trivial to code generators via library functions or rudimentary geometry.

We ask students to generate a Wiphala (Fig. 1). It requires listing colors in the same cycle, but with a different starting point, across rows (or columns). We believe that it is of comparable difficulty to write a loop- or slice-based solution as to instruct a code generator with sufficient precision. Therefore, we feel students necessarily approached the computational thinking problem of multidimensional indexing. Naive generated code often produced a bottom-left-to-top-right diagonal, which we found among low-scoring submissions.

Separately, we take great care to avoid referring to the Wiphala by name in the instructions, as this is often sufficient prompting for a code generator to produce competitive solutions. We aim to resolve this tension in future work, perhaps with a novel application domain.

Lastly, we take care to inform students what the flag represents - the self-determination of the Andean people - with some allusions to the political implications of U.S. tech industry expansions on mineral and labor rights in the Andes. We hope that students carry with them a sense of the true cost of relying on exascale compute clusters for trivial tasks like list/array slicing as they continue their careers.

3.3 Reading Python

Traditionally the final exam began with a code tracing exercise. For example, we may ask a student what the following returns:

```
def conundrum():
    array = []
    for i in range(6):
        array.append(0)
        for j in range(i, 0, -1):
            array[j] += j
    return array
```

We believe we have leveraged the Wiphala to achieve a similar goal. In the prior section we show a specific, regional Wiphala, for the Qulla Suyu (Qulla region). Other recognized regional Wiphalas are differentiated by "long horizontal" color as shown in Fig. 2.

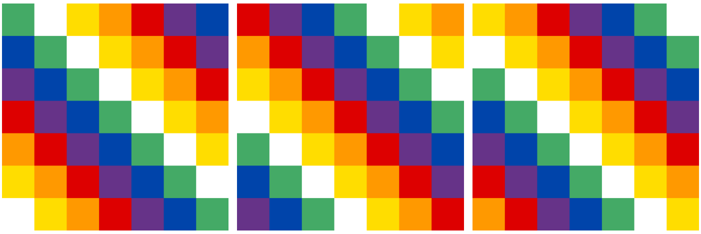


Figure 2: The Wiphala of other Suyu (Anti, Chinchay, and Kunti).

Rather than prompt students as to the outcome of code, we prompt students to write code that generates some outcome specified with imagery, rather than text, which may pose a barrier to code generation. Additionally, we manually render the flags as fixed-pixel HTML elements rather than providing an image that can be pasted into a code prompt.

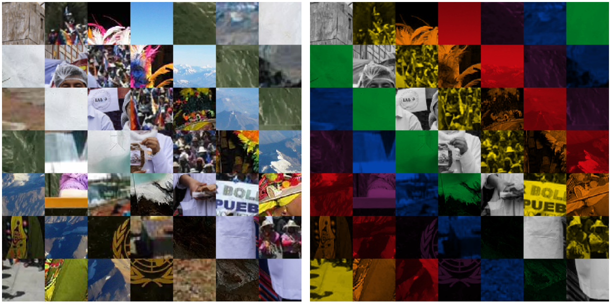
A small number of students entertained manually specifying all 49 tiles in the earlier Wiphala, and this subproblem encouraged use of programming logic.

3.4 Defining Classes

Existing final exams often ended with a larger task involving implementing a `class` that, for example, created numbered labels or recorded the dot distribution of a domino. Given Python iterators and collections, respectively, and expect students will often instead interact with classes implemented within libraries, such as PyTorch tensors, or Flask APIs. While we have a University-specific implementation utilizing classes, we generalize to a NumPy `ndarray`.

We ask students to (1) read file data based on file names provided in the CSV file from an earlier problem, (2) interpret these files as images, (3) crop images to squares, and (4) scale images to be the same size. Finally, rather than creating a Wiphala of colors, they create a Wiphala as a collage over these images as shown in first image of Fig. 3. These are fair use images inspired by each individual color. For example, we selected an image of a Bolivia lithium mine as "el símbolo de las riquezas naturales" (the symbol of the natural resources), represented by the verde color in the Wiphala, and provide a README with this note and source attribution.

Figure 3: True-color and color-scaled collages, both provided to students.



We retain a small amount of *intentional ambiguity*. We could define this operation in pseudo-code: if the pixel at location `(x, y)` would be of color "amarillo", write the "amarillo.png" value at this index to the output pixel array. Rather, we provide example output and a brief recommendation to visually inspect the long diagonal and the constituent files. The distribution of final scores suggests few-if-any students reaching this problem were confused by these instructions.

The collaging effect appears to confound the image processing capabilities of most code generation tools. Gemini inferred it was observing a 5-by-10 grid of 50 individual images with no pattern, even when prompted to use 7 provided images. ChatGPT inferred it was provided with single image file of 7 concatenated tiles and was trying to create a 10-by-10 grid.

3.5 Fundamental Python

In the context of the final exam, the term "Fundamental Python" is used to refer to what is often termed numerical computing, like computing the Racamán sequence. We used NumPy vector operations, the foundation of early numerical computing efforts in Python. For each image, after cropping and scaling, we ask students to convert the true-color images to monochrome. We term this “colorscale” to remind students of grayscale from an earlier assignment (“ImageShop”). To support them, we specify:

1. Example code to compute luminance:

    ```
    r, g, b = color; (4*g + 3*r+b)//8
    ```

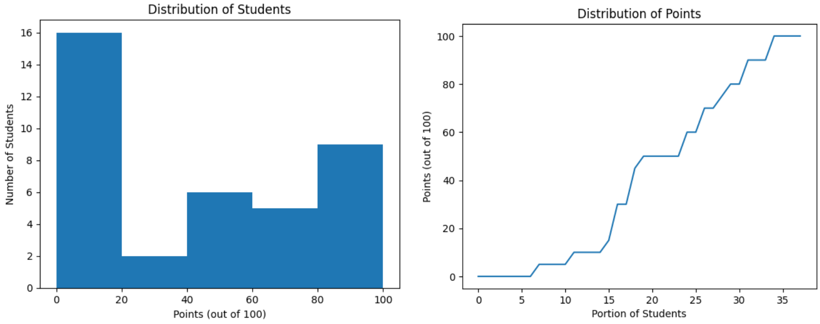
2. That high luminance should become “rojo” monochrome value.
3. That low luminance should become all zeros (black).

Of note, we are not aware of particularly succinct ways to describe this process absent color science terminology outside an introductory computing scope. Separately, we provide a vector-optimized integer luminance calculation to partially confound code generation tools, which seemingly expect to perform these calculations over floating point values. We provide the second image in Fig. 3 as an example to our students with the caveat that their code must additionally be able to produce similar images for the other regions.

4 Results

We achieved our preferred point distribution of approximate uniformity, allowing us to differentiate student achievement. We note we applied a generous curve (not shown).

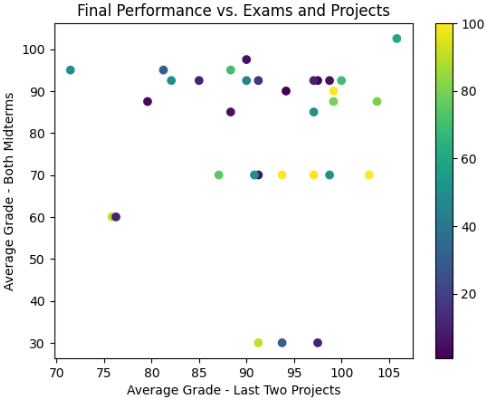
Figure 4: We achieved a quasi-uniform distribution.



An uncomfortably large number of students earned zero or nearly zero points. Anecdotally, from in-class exercises and office hours, we suspected a population of students earned high marks on assignments, but could not write a single line of code, even given hours to do so. We suspect this population gave rise to the large bin of low scores shown in Fig. 4.

We found the final score uncorrelated with project or midterm scores, as shown in Fig. 5. We plotted horizontal homework (code generation allowed), versus vertical midterm (code generation disallowed but suspected), versus colorized final score (code generation unhelpful).

Figure 5: Project (horizontal) and midterm (vertical) performance did not predict final (colorized) performance.



5 Summary

Our impression was that highly motivated students were completing exams and projects manually with minimally but non-zero errors and being scored lower than students who solely used code generation, including in violation of midterm honor policy. Many high scores on the final were students we identified as strong scientists, but lacked interest to complete portions of assignments they did not find algorithmically interesting. Conversely, the final seemed to firmly differentiate an otherwise co-mingled population of relatively high-scoring students between independent and assisted coders. With some relief, we do not seem to be measuring testing anxiety given the low correlation ($r^2 = .0057$) with midterm scores. That said, we want to be more confident we are measuring student learning and not some other factor, and hope to deploy similar midterms and in-class exercises in Fall 2025 for students to be better prepared for the eventual final.

Ultimately, we believe this final rewarded hard-working students. We also hope to have persuaded our class of the usefulness of course content even in an environment with widespread access to industry-grade code generators.

6 Special Thanks

We profusely thank Professor Jed Rembold, our CS 151 co-instructor, for providing thoughtful feedback, showing patience during development, and maintaining all five existing projects, among many other contributions..

References

[1] Calvin Deutschbein. *Final.* https://github.com/cd-public/ccscrm. 2025.

[2] Association for Computing Machinery (ACM) Joint Task Force on Computing Curricula and IEEE Computer Society. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science.* New York, NY, USA: Association for Computing Machinery, 2013. ISBN: 9781450323093.

[3] Nataliya Kosmyna et al. *Your Brain on ChatGPT: Accumulation of Cognitive Debt when Using an AI Assistant for Essay Writing Task.* 2025. arXiv: `2506.08872 [cs.AI]`. URL: `https://arxiv.org/abs/2506.08872`.

[4] Nick Parlante et al. “Nifty Assignments”. In: *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education V. 2.* SIGCSE 2022. Providence, RI, USA: Association for Computing Machinery, 2022, pp. 1067–1068. ISBN: 9781450390712. DOI: `10.1145/3478432.3499268`. URL: `https://doi.org/10.1145/3478432.3499268`.

[5] Nick Parlante et al. “Nifty assignments”. In: *Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education.* SIGCSE ’02. Cincinnati, Kentucky: Association for Computing Machinery, 2002, pp. 319–320. ISBN: 1581134738. DOI: `10.1145/563340.563466`. URL: `https://doi.org/10.1145/563340.563466`.

[6] Nick Parlante et al. “Nifty assignments”. In: *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education.* SIGCSE ’06. Houston, Texas, USA: Association for Computing Machinery, 2006, pp. 562–563. ISBN: 1595932593. DOI: `10.1145/1121341.1121516`. URL: `https://doi.org/10.1145/1121341.1121516`.

[7] Eric Roberts and J. Jedediah Rembold. "Nifty Assignments: Enigma Machine Simulator". In: *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 2.* SIGCSE 2023. Toronto ON, Canada: Association for Computing Machinery, 2023, p. 1276. ISBN: 9781450394338. DOI: `10.1145/3545947.3573220`. URL: `https://doi.org/10.1145/3545947.3573220`.

[8] Eric Roberts and Keith Schwarz. "A portable graphics library for introductory CS". In: *Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education.* ITiCSE '13. Canterbury, England, UK: Association for Computing Machinery, 2013, pp. 153–158. ISBN: 9781450320788. DOI: `10.1145/2462476.2465590`. URL: `https://doi.org/10.1145/2462476.2465590`.

Using LLMs for the Browse Task: An Exploratory Document Clustering*

Daniel McDonald and John E. Anderson
Information Systems & Technology Department
Utah Valley University
Orem, UT 84058
`{daniel.mcdonald,janderson}@uvu.edu`

Abstract

A browsing task is characterized by less planning and less knowledge of what is a relevant document. Support for browsing tasks in information retrieval has largely focused on adding document context to provide topic perspective on the document landscape. Unsupervised clustering has been a popular algorithm to capture a document's topics and group them with related topics from other documents. With the use of Large Language Models (LLMs), clustering output can now add not only document context, but a deeper semantic context for the topics in a collection. Three popular LLMs are tested in their ability to cluster a collection's topics and then label those topics. The LLM output is then compared to the output of clusters created by two information systems faculty. Preliminary results show the LLMs created meaningful clusters that tended to group documents over multiple dimensions, while the human experts clustered over single dimensions.

1 Introduction

Historically, information seeking behavior has often been divided into searching versus browsing tasks [6]. In search tasks, users have a better knowledge of

---

*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.

what they seek and more easily recognize relevant documents. In search, summarized documents often focus on the user's query terms [13]. In browsing, more document context is sought to better understand the document landscape. In the past, tools and visualizations for browse support have largely utilized a bag-of-words approach to document representation [5, 6]. With the rise of Large Language Models (LLMs), textual summaries for browse support offer not only document context, but also knowledge or semantic context of what is being discussed in the document.

The need for browse support has only grown in recent years as more research is published. Reviewing the literature surrounding a topic has become increasingly difficult due to the ever growing volume of literature to understand and synthesize. This problem is often referred to as the information overload problem [7]. Large Language Models offer hope for producing output to better support browsing.

Conducting a preliminary study on how well LLMs can produce summaries of document collections that would be useful to a browse task is the object of this research. In addition, identifying strengths and weaknesses of the LLMs ability to create clusters will be compared to human-generated clusters. Three popular LLMs were used to cluster 105 documents and their output was compared to the results of clusters created by two information systems faculty.

2 Literature Review

While not as common as search, the browse task has been well-researched. More specifically, the use of document clustering to help support browsing has been tested in various settings.

2.1 Browse Task

Browsing has been defined as "an exploratory, information seeking strategy that depends upon serendipity" that is "especially appropriate for ill-defined problems and for exploring new task domains (page 71)." [12] When users are browsing, there is an absence of "planning" [11]. When browsing, documents that are relevant are harder to recognize as users seek background information. Document context has been shown to be important to users [4, 10, 12]. Seeing how documents relate to one another helps users refine their criteria for what is relevant.

2.2 Document Clustering

Clustering is one of the most popular unsupervised techniques in machine learning. It creates groups of similar objects. In text analysis, documents that are similar to each other are grouped in the same group. The document clustering process has the goal of discovering natural groupings and presenting an overview of the topics in a document collection [2].

The k-means clustering method [9, 10] is the most widely used clustering algorithm. k-means clustering starts by selecting k initial documents as clusters. Documents are then iteratively assigned to these clusters, with centroids being updated in a "top-down" manner until there is no longer centroid movement. Because text data is directional, document vectors are typically normalized, and cosine similarity is used instead of Euclidean distance. Despite its popularity due to ease of implementation, k-means has several drawbacks: it relies on random initialization, it can converge to suboptimal local minima, and it is susceptible to outliers and noise [2].

The accuracy of automatic text summarization has been improved using methods such as k-means clustering [8]. Alami et al. pointed out two text summarization problems: 1) most methods do not consider the document context, and 2) the traditional bag-of-words text representations (frequency counts) involving high dimensional and sparse data make it difficult to determine relevant information [1]. They chose to solve these problems by using a combination of clustering, topic modeling, and unsupervised neural networks. The first step in their process was to use document clustering.

The other “classical” clustering method is hierarchical clustering. Opposite of the k-means “top-down” method, hierarchical clustering uses a “bottom-up" method where each document is initially its own cluster which the algorithm successively merges until it converges into one root cluster. Hierarchical clustering keeps the cluster history of this process allowing the user to “browse” this history to see what clusters were formed at each iteration [2].

As clustering has been common in research for topic summarization, having the LLMs produce clusters of topics to support browsing is a reasonable first step in determining the utility of LLMs for this purpose.

3 Research Questions

In this research, we explore the use of LLMs to create content clusters to support browsing. We are focusing on three primary research questions:

1. How does the LLM output of clusters compare against each other?
2. How does the LLM output of clusters compare against human-created clusters?
3. Could LLMs help with the browse task?

4 Methodology

The articles for the clustering were identified via a search for "artificial intelligence" on the ACM and IEEE digital library websites. The resulting titles and abstracts were saved into a CSV file. The CSV file was uploaded to three LLMs: ChatGPT 4o, Copilot, and Gemini 2.5 Flash. Two information systems faculty also read through the CSV file and manually grouped the abstracts into meaningful clusters. The behavior of LLMs is highly dependent on the prompts used. The same prompts were used for the three LLMs as listed below:

1. Create topic clusters using the csv file.
2. Why did you decide on X clusters?

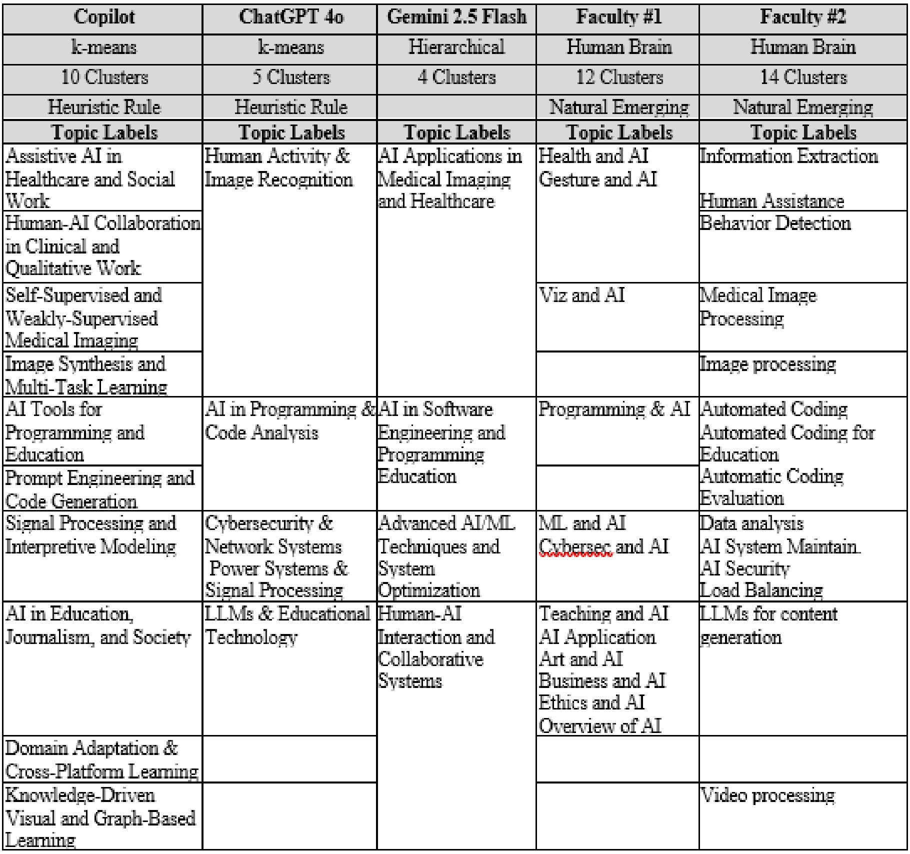
The first prompt uses the term "cluster" purposefully to encourage the LLM to use a clustering technique. At the same time, the first prompt doesn't specify the number of clusters or algorithms to use. The second prompt seeks to understand the LLM's choice of number of clusters.

5 Results

The output of the three LLMs had some similarities, but also several differences. Both Copilot and ChatGPT used the k-means algorithm, while Gemini used a hierarchical clustering algorithm. Even using the same algorithm (k-means), however, Copilot created ten clusters and ChatGPT created five clusters. Gemini created four top-level hierarchical clusters. The LLMs used heuristic rules to determine how many clusters to use, cluster numbers they regarded as best practice. ChatGPT did not immediately produce cluster labels but did after being prompted to do so.

The two faculty created 12 clusters and 14 clusters respectively; both numbers are higher than any of the cluster-counts from the LLM output. Processing speed also varied among the clustering approaches. Copilot and ChatGPT returned results within minutes. Gemini, using deep research, took 10 minutes. The humans took three hours and five hours respectively. The human readers used a bottom-up approach to group similar documents together arriving at a cluster count they felt nicely grouped the 105 abstracts.

Below in Figure 1 you can see a comparison of the cluster labels created by each of the five approaches. Copilot on the far left produced ten clusters. The labels that matched best to the Copilot clusters were placed in similar rows in the table.



Copilot	ChatGPT 4o	Gemini 2.5 Flash	Faculty #1	Faculty #2
k-means	k-means	Hierarchical	Human Brain	Human Brain
10 Clusters	5 Clusters	4 Clusters	12 Clusters	14 Clusters
Heuristic Rule	Heuristic Rule		Natural Emerging	Natural Emerging
Topic Labels	**Topic Labels**	**Topic Labels**	**Topic Labels**	**Topic Labels**
Assistive AI in Healthcare and Social Work	Human Activity & Image Recognition	AI Applications in Medical Imaging and Healthcare	Health and AI Gesture and AI	Information Extraction Human Assistance
Human-AI Collaboration in Clinical and Qualitative Work				Behavior Detection
Self-Supervised and Weakly-Supervised Medical Imaging			Viz and AI	Medical Image Processing
Image Synthesis and Multi-Task Learning				Image processing
AI Tools for Programming and Education	AI in Programming & Code Analysis	AI in Software Engineering and Programming Education	Programming & AI	Automated Coding Automated Coding for Education Automatic Coding Evaluation
Prompt Engineering and Code Generation				
Signal Processing and Interpretive Modeling	Cybersecurity & Network Systems Power Systems & Signal Processing	Advanced AI/ML Techniques and System Optimization	ML and AI Cybersec and AI	Data analysis AI System Maintain. AI Security Load Balancing
AI in Education, Journalism, and Society	LLMs & Educational Technology	Human-AI Interaction and Collaborative Systems	Teaching and AI AI Application Art and AI Business and AI Ethics and AI Overview of AI	LLMs for content generation
Domain Adaptation & Cross-Platform Learning				
Knowledge-Driven Visual and Graph-Based Learning				Video processing

Figure 1: Comparison of Clusters made by the LLMs and Humans

6 Discussion

It was interesting to see that because the word "cluster" was used in the prompt, the LLMs interpreted this literally and used the most common clustering techniques (Copilot and ChatGPT used k-means and Gemini used Hierarchical) to do the analysis.

Another interesting consideration is how the LLMs arrived at their number of clusters (K). The value of K is often the most important variable as to the scope of the clusters. More clusters yield more granular groups while fewer clusters give larger summary groups. Copilot came up with ten topic clusters, ChatGPT chose five and Gemini chose a four-cluster solution. When asked why they chose the number of clusters they did, both Copilot and ChatGPT said they used a heuristic rule, that ten or five were good "rules of thumb" according to the literature. We were not able to find out why Gemini chose a four-cluster solution, when we asked, it replied that it could not do a hierarchical cluster analysis because it could not load the nltk Python library.

When we ran the analysis a second time with the same prompt, Copilot again performed a k-means analysis but optimized using an elbow plot, ChatGPT also again did a k-means analysis and stayed with the heuristic solution, and Gemini chose to not do a hierarchical clustering analysis but changed to a qualitative methodology and chose an eight-topic cluster solution. It is interesting to note that the LLMs are constantly changing, and it was not possible to produce the same clustering result a second time.

6.1 Rater Agreement

The output from each LLM was slightly different. Copilot gave cluster labels, top keywords, and representative article titles. ChatGPT 4o initially did not give cluster labels but added them when asked. ChatGPT gave top keywords, and representative titles. Gemini 2.5 Flash gave the most detailed output. Gemini gave cluster labels, a brief description of the cluster, key AI/ML techniques included in the cluster, and representative applications. Gemini also listed the sub-themes within each cluster and listed overarching themes that transcend individual topic clusters.

There are several topical dimensions over which the research documents could be clustered. We focus on four dimensions in our discussion here. The dimensions are summarized in Figure 2. First, a document could focus on the chosen or developed algorithm. In this research, the focus would be on fitting an algorithm to a problem, i.e. neural network, Bayesian, support vector machine, etc. Second, research could focus on how the algorithm was trained and finetuned. Some examples include whether training sets are single or multi-domain, how the algorithm deals with noise, and whether human feedback is utilized. Third, research could focus on the output of the algorithm. Topics in this area include image segmentation, voice recognition, gesture detection, and information extraction. Finally, research could focus on the use or application of the output. For example, research includes healthcare, cancer detection in medical images, content creation for journalism, and using LLMs for cybersecurity.

The human clustering typically grouped on only one or two dimensions. For example, one faculty grouped abstracts based only on the application, i.e. cybersecurity and healthcare. The other faculty clustered abstracts based mostly on the output of the algorithm, i.e. image segmentation, information extraction, and video processing. In contrast, the LLMs created clusters that

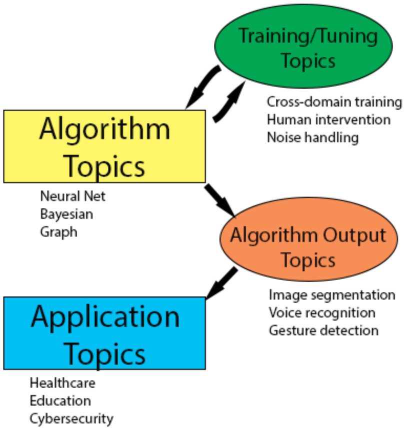


Figure 2: Clustering dimensions

crossed several dimensions even on their labels, i.e. "Self-Supervised and Weak-Supervised Image Processing". This cluster group name includes the training dimension and the algorithm output. Another example would be "Image Synthesis and Multi-Task Learning." This group includes a type of training and the output of the algorithm. Including multiple dimensions in a cluster would seem to make the cluster less inclusive. This is surprising because LLMs used fewer clusters than the human workers.

There have been studies showing that human graders and LLMs have not agreed. For reviewing patient care reports for quality assurance, a non-standardized task, Brant-Zawadzki et al. found that ChatGPT-4 achieved some inter-rater agreement with the human raters (76.2% agreement, kappa coefficient 0.401) in EKG documentation and aspirin administration, but not for other metrics [3]. Gemini performed poorly and was discontinued.

6.2 Usefulness for the Browse Task

The clusters produced by the LLMs do look like they could offer browse support. Especially helpful were the cluster labels produced by the LLMs. The labels did not come from the abstracts or titles themselves, but rather offered a summary based on the larger context of the abstract topic. The keywords, which came from the abstracts, listed by the LLMs were the least helpful. Gemini's hierarchical clustering offered the most detailed analysis. In addition to clusters, Gemini output a list of overarching themes that were repeated across all the clusters. This feature along with the sub-clusters listed within each cluster could be very useful in understanding the content overview versus the detail of each document. The labels of all three LLMs gave a good summary of the document landscape, which could be very helpful to an information seeker.

LLMs were much faster than humans. Copilot and ChatGPT ran in under 60 seconds. Gemini took longer (ten minutes) because it chose to do hierarchical clustering. It took Faculty #1 about 3 hours to categorize the documents and Faculty #2 about 5 hours.

The LLMs having more detailed pattern recognition created cluster areas not noticed by human experts. The LLMs recognized similarities that were not explicitly written in the abstracts. For example, Copilot had a cluster with the label "Self-Supervised and Weakly-Supervised Medical Imaging." In that group of documents, there was an abstract with the title "High-Frequency Space Diffusion Model for Accelerated MRI". In that and other abstracts the words training, supervised, weakly, and medical were never mentioned. The article talked about MRIs and diffusion models using differential equations. LLMs have greater prior knowledge enabling them to use that knowledge to categorize content in ways unfamiliar to human workers.

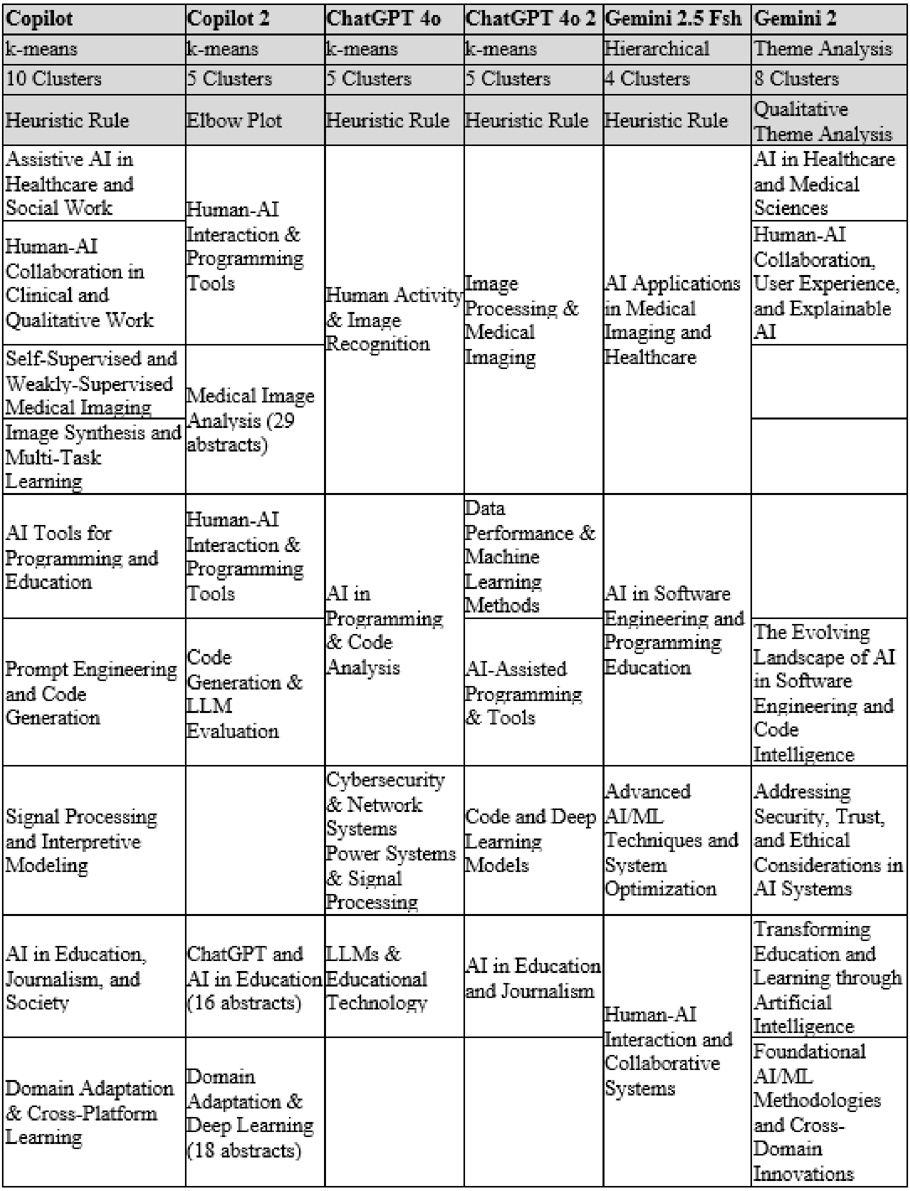
In the process of having the LLMs perform the clustering, one logged out and so we had it redo the analysis. Somewhat surprisingly it did the analysis differently. So, we had all three LLMs do the analysis again in a zero-shot direct without any other prompting method (not trying to edit the prior analyses) and all three new analyses were quite different. Figure 3 2 summarizes the change in output from the first to the second clustering output.

Copilot changed from ten clusters (using a heuristic rule) to five clusters (using an elbow plot). Why did it do this? Using an elbow plot or silhouette scores would be the logical next step in most k-means analyses. Perhaps Copilot was just taking the next step. The first run was much more granular (at ten clusters) compared to the second run (at five clusters).

ChatGPT was more consistent staying with five clusters using a heuristic rule. There is much overlap with the first run topic clusters but there was one new cluster formed that was not in the first. The cluster was combined with another cluster in the first run.

Gemini was perhaps the least consistent changing from four clusters to eight clusters. Also, on the first run Gemini used hierarchical clustering, while on the second run it used a qualitative thematic analysis because it could not load the appropriate Python library.

This highlights the problem of intra-rater reliability or internal consistency of LLMs. Pawitan and Holmes found that there is a wide variability in the tendency of LLMs to change initial answers if asked again, and that the overall accuracy of the second answer is often worse than the first answer [15]. For more structured tasks such as sentiment classification of financial news, Megahed



Copilot	**Copilot 2**	**ChatGPT 4o**	**ChatGPT 4o 2**	**Gemini 2.5 Fsh**	**Gemini 2**
k-means	k-means	k-means	k-means	Hierarchical	Theme Analysis
10 Clusters	5 Clusters	5 Clusters	5 Clusters	4 Clusters	8 Clusters
Heuristic Rule	Elbow Plot	Heuristic Rule	Heuristic Rule	Heuristic Rule	Qualitative Theme Analysis
Assistive AI in Healthcare and Social Work	Human-AI Interaction & Programming Tools	Human Activity & Image Recognition	Image Processing & Medical Imaging	AI Applications in Medical Imaging and Healthcare	AI in Healthcare and Medical Sciences
Human-AI Collaboration in Clinical and Qualitative Work					Human-AI Collaboration, User Experience, and Explainable AI
Self-Supervised and Weakly-Supervised Medical Imaging	Medical Image Analysis (29 abstracts)				
Image Synthesis and Multi-Task Learning					
AI Tools for Programming and Education	Human-AI Interaction & Programming Tools	AI in Programming & Code Analysis	Data Performance & Machine Learning Methods	AI in Software Engineering and Programming Education	
Prompt Engineering and Code Generation	Code Generation & LLM Evaluation		AI-Assisted Programming & Tools		The Evolving Landscape of AI in Software Engineering and Code Intelligence
Signal Processing and Interpretive Modeling		Cybersecurity & Network Systems Power Systems & Signal Processing	Code and Deep Learning Models	Advanced AI/ML Techniques and System Optimization	Addressing Security, Trust, and Ethical Considerations in AI Systems
AI in Education, Journalism, and Society	ChatGPT and AI in Education (16 abstracts)	LLMs & Educational Technology	AI in Education and Journalism	Human-AI Interaction and Collaborative Systems	Transforming Education and Learning through Artificial Intelligence
Domain Adaptation & Cross-Platform Learning	Domain Adaptation & Deep Learning (18 abstracts)				Foundational AI/ML Methodologies and Cross-Domain Innovations

Figure 3: Limited Consistency of the LLMs to Produce the Same Output

found high intra-rater consistency of 90-98% of LLMs [14].

7 Conclusions

All three LLMs produced meaningful cluster labels that gave a good overview of the document topics. The LLMs preferred the k-means clustering algorithm but also used hierarchical clustering. The LLMs all used a different number of clusters. Compared to human-generated clusters, the LLM clusters grouped documents across more content dimensions. Faculty tended to stick with a single way of grouping documents. The LLMs also finished in minutes compared to the faculty taking hours. The clusters produced by the LLMs offer real promise for being useful in a browse task. Their broad and deep background knowledge produced clusters that our human experts did not see.

8 Future Research

In future research, we want to test the utility of the LLM output in a browse task with subjects. In the browse task, we want to use the output from the different LLMs in addition to a Self-Organizing Map (SOM) for comparison.

References

[1] N. Alami et al. "Unsupervised neural networks for automatic Arabic text summarization using document clustering and topic modeling". In: *Expert Systems with Applications* 172 (2021). DOI: `https://doi.org/10.1016/j.eswa.2021.114652`.

[2] N. Andrews and E. Fox. *Recent Developments in Document Clustering.* Report. Virginia Tech, 2007.

[3] G. Brant-Zawadzki et al. "The Performance of ChatGPT-4 and Gemini Ultra 1.0 for Quality Assurance Review in Emergency Medical Services Chest Pain Calls". In: *rehospital Emergency Care* 29.3 (2025), pp. 210–217. DOI: `https://doi-org.uvu.idm.oclc.org/10.1080/10903127.2024.2376757`.

[4] E. Carmel, S. Crawford, and H. Chen. "Browsing in Hypertext: A Cognitive Study". In: *Systems, Man and Cybernetics, IEEE Transactions on* 22.5 (1992). TY - JOUR, pp. 865–884.

[5] H. Chen, C. Schufels, and R. Orwig. "Internet Categorization and Search: A Self-Organizing Approach". In: *Journal of Visual Communication and Image Representation* 7.1 (1996), pp. 88–102.

[6] Hsinchun Chen et al. “Internet Browsing and Searching: User Evaluations of Category Map and Concept Space Techniques”. In: *Journal of the American Society for Information Science* 49.7 (1998), pp. 582–603.

[7] Weiguo Fan et al. “Tapping the Power of Text Mining”. In: *Communications of the ACM* 49.9 (2006), pp. 77–82.

[8] W. Liu et al. “Automatic Text Summarization Method Based on Improved TextRank Algorithm and K-Means Clustering”. In: *Knowledge-Based Systems* 287.C (2024). DOI: `https://doi.org/10.1016/j.knosys.2024.111447`.

[9] S. Lloyd. “Least squares quantization in PCM”. In: *IEEE Transactions on Information Theory* 28.2 (1982), pp. 129–137. DOI: `doi:10.1109/TIT.1982.1056489`.

[10] J. MacQueen. “Some methods for classification and analysis of multivariate observations”. In: *Fifth Berkeley Symposium on Mathematical Statistics and Probability.* Vol. 5. 1967.

[11] G. Marchionini. “An Invitation to browse: Designing full text systems for novice users.” In: *Canadian Journal of Information Science* 12.3 (1987), pp. 69–79.

[12] G. Marchionini and B. Shneiderman. “Finding Facts vs. Browsing Knowledge in Hypertext Systems”. In: *Computer* 21.1 (1988). TY - JOUR, pp. 70–79.

[13] Daniel McDonald and Hsinchun Chen. “Summary in context: searching versus browsing”. In: *ACM Transactions on Information Systems (TOIS)* (2006).

[14] F. Megahed et al. “Reliable Decision Support with LLMs: A Framework for Evaluating Consistency in Binary Text Classification Applications”. Unpublished Work. unpublished. 2025. URL: `https://arxiv.org/abs/2505.14918`.

[15] Y. Pawitan and C. Holmes. “Confidence in the reasoning of large language models”. In: *Harvard Data Science Review* 7.1 (2025). DOI: `https://doi.org/10.1162/99608f92.b033a087`.

Artificial Intelligence in Cybersecurity: Applications, Challenges, and Future Directions*

Diego Sigaran[1], Jeff Richards[1], Bryce Shepard[1],
Kazi Arafat[2], Iftekhar Hossain[3], Xi Chen[1], and Imtiaz Parvez[1]
[1]Utah Valley University
Orem, Utah
`{11005949, 10912073, 10843451, xi.chen, imtiaz.parvez}@uvu.edu`
[2]Atish Dipankar University of Science & Technology
Dhaka, Bangladesh
`cse2010004@adust.edu.bd`
[3]Dhaka University of Engineering & Technology
Gazipur, Bangladesh
`ifthekhar121@gmail.com`

Abstract

The ability to understand and apply artificial intelligence (AI) is increasingly critical in cybersecurity. Selecting and implementing the appropriate AI tools can be the difference between the failure or success of an organization. This paper examines how effective AI tools are in modern cybersecurity, highlighting the features that can contribute to their success and importance in future security efforts. Drawing from recent academic literature, this paper compiles and analyzes current techniques and challenges in the application of AI to cybersecurity. It also explores how AI enhances cybersecurity in the creation of automated and adaptable defensive systems. These systems allow cybersecurity administrators and practitioners to build robust and responsive defenses that reduce both the quantity and severity of threats. This paper also explores the role of AI in cybersecurity education, emphasizing the need

---

*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.

for students to understand the nuances of AI techniques to ensure their effective and ethical application in professional practice.

1 Introduction

In the modern era of digital connectivity, cybersecurity has emerged as a vital component of information safety and infrastructure protection. As both individuals and organizations increasingly depend on digital technologies for daily operations, communication, and data handling, the protection of digital assets against unauthorized access and cyberattacks has become critically important. Cybersecurity involves a comprehensive set of tools, strategies, and practices aimed at securing systems, networks, and data from disruption, theft, or damage. Beyond being a technical requirement, strong cybersecurity measures are essential for sustaining user confidence, ensuring operational continuity, and preserving the integrity of digital ecosystems.

The cyber threat landscape is rapidly evolving, with adversaries employing more sophisticated, adaptive, and persistent tactics [5]. From ransomware and phishing attacks to state-sponsored espionage and advanced persistent threats, the range and complexity of cyber threats have escalated. Businesses face operational disruptions and financial losses, governments encounter threats to national security, and individuals suffer from identity theft and privacy breaches. The increasing integration of digital systems in critical sectors like finance, healthcare, and energy amplifies the potential impact of cyberattacks, making proactive threat detection and response more vital than ever.

The way we approach cybersecurity is being altered by artificial intelligence (AI)[33]. Security systems can learn from data, identify anomalous activity, and react to potential threats more rapidly thanks to AI, as opposed to only depending on manual monitoring and set rules. Large volumes of data can be analyzed in real time, and it can adjust to new attack techniques as they appear. This makes it particularly helpful in addressing the increasing complexity and quantity of cyberthreats. AI is quickly becoming as a crucial component of contemporary cybersecurity methods due to its ability to assist identify threats early and with more accuracy.

AI also enhances cybersecurity by using ML algorithms that evolve over time[26]. Subtle behavioural changes that can point to insider threats, phishing attempts, or zero-day vulnerabilities can be picked up by these algorithms. Because of this flexibility, security systems are able to keep up with hackers who are always evolving the ways they attack. Threat intelligence using natural language processing (NLP) is another promising area in cybersecurity. AI is capable of scanning and analyzing forums, security reports, and communications from the dark web to find early indicators of coordinated attacks.

By taking this proactive stance, organisations can get ready before an attack occurs.

Integrating AI technologies with current security frameworks, including as firewalls, endpoint protection tools, and Security Information and Event Management (SIEM) systems, is crucial to maximizing AI's potential in cybersecurity. Organizations can improve their threat detection capabilities with this hybrid strategy without having to replace their existing infrastructure. By connecting alarms, lowering false positives, and prioritizing issues for human analysts, AI can be a force multiplier. A more effective and responsive security posture results from the combination of AI and conventional methods, particularly in settings with a shortage of human resources.

AI is not only useful for detecting and responding to threats but also for predicting them. Predictive threat intelligence uses AI models to anticipate potential cyberattacks before they occur by analyzing trends, threat actor behaviors, and vulnerabilities. This forward-looking capability helps organizations shift from a reactive to a proactive cybersecurity stance. By forecasting possible breach scenarios or identifying vulnerable assets, AI enables cybersecurity teams to patch systems and strengthen defenses in advance, minimizing the risk of future attacks.

Even while AI has many benefits, there are serious privacy and ethical issues with its application[17]. Unintentional privacy violations may result from the application of AI in behavioral analysis, data monitoring, and surveillance. Additionally, there is the possibility of algorithmic bias, in which, given biased or unrepresentative training data, AI systems may disproportionately target or ignore particular actions. Establishing transparent AI governance frameworks, encouraging responsible data usage, and making sure AI systems continue to be explainable and auditable are all essential steps in reducing these dangers. To preserve compliance and trust, ethical issues must be incorporated into the development and application of AI systems.

Interdisciplinary cooperation, ongoing innovation, and the creation of increasingly autonomous and context-aware systems are key to the future of AI in cybersecurity. A key research objective is to develop AI models that can classify threats more accurately by understanding the context of network actions in addition to detecting anomalies. Additionally, the proliferation of IoT devices and edge computing necessitates the development of lightweight AI models that can function in decentralized settings. Government, business, and academia working together will be crucial to advancing safe and scalable AI systems that are adapted to changing cyberthreats.

This paper aims to explore the transformative role of AI in strengthening cybersecurity frameworks. It delves into various AI techniques employed in the cybersecurity domain, including machine learning (ML), deep learning

(DL), Reinforcement learning (RL), and natural language processing (NLP). It addresses the challenges of implementing AI-driven security solutions, such as data privacy concerns, adversarial attacks, and the need for explainability. Finally, it outlines emerging trends and future directions, offering insights into how AI will continue to shape the future of cybersecurity.

The paper follows this structure. Section 2 discusses the background and key concepts. Section 3 discusses AI Techniques Applied to Cybersecurity, and Section 4 emphasizes AI in Cybersecurity Education. Section 5 and 6 discusses on Challenges and Limitations of AI in Cybersecurity, and Future Directions in AI for Cybersecurity. Finally, concluding remarks are presented in section 7.

2 Background and Key Concepts

This section provides an overview and relevant concepts on AI, cybersecurity, and the intersection of AI and cybersecurity.

2.1 Artificial Intelligence

Artificial Intelligence (AI) stands as a revolutionary field of computer science that enables machines to replicate human intelligence capabilities. The three core processes included within AI are-learning, which refers to the acquisition of information and rules for using it, reasoning which means using rules to reach approximate or definite conclusions and self-correction which enables systems to improve autonomously. The foundational elements of AI consist of three primary areas which are machine learning (ML), deep learning (DL) and knowledge representation. ML functions as an AI subfield that develops algorithms which acquire knowledge from data to generate predictions. ML models enhance their performance by learning patterns from data rather than relying on hardcoded instructions. DL emerges from ML as a specialized version which employs artificial neural networks with many layers to find complex patterns in extensive datasets.

Practical ML operations consist of supervised learning together with unsupervised learning and reinforcement learning paradigms [13]. Supervised learning requires labeled datasets for training algorithms to achieve accurate data classification and outcome prediction. Unsupervised learning reveals natural patterns within input data to solve problems such as clustering and association analysis. The reinforcement learning technique trains models to develop decision sequences through positive outcome-based rewards that now serve robots and cybersecurity systems.

Deep learning models excel at automatic feature extraction because of their fundamental design[32]. The identification of prediction-relevant features in traditional ML requires both significant time and expert expertise. Deep learning models especially autoencoders enable automatic learning of essential representations for classification and detection tasks. Large datasets and significant computational resources become necessary costs when employing this advantage. The unprecedented speed of AI model integration in healthcare alongside finance and transportation and cybersecurity continues to expand despite existing implementation obstacles. The practical application of AI depends on achieving both interpretability and transparency through explainable AI (XAI).

2.2 Cybersecurity

Cybersecurity refers to the collective methods, technologies, and processes used to protect the confidentiality, integrity, and availability of data and computing systems from unauthorized access, cyberattacks, and damage. As digital transformation accelerates across industries, cybersecurity has become a cornerstone of modern information technology infrastructure. It encompasses a broad range of components, including encryption, firewalls, intrusion detection systems (IDS), threat intelligence platforms, and access control mechanisms. These tools are designed to defend systems from increasingly sophisticated threats posed by cybercriminals, nation-state actors, and insider threats.

One of the foundational pillars of cybersecurity is encryption, which transforms data into unreadable ciphertext using mathematical algorithms, ensuring that only authorized parties with the decryption key can access the original information. Encryption protocols such as AES (Advanced Encryption Standard) and RSA (Rivest–Shamir–Adleman) are widely employed for securing communications and data storage [2]. Coupled with encryption are firewalls, which act as gatekeepers between trusted internal networks and untrusted external environments, filtering incoming and outgoing traffic based on predefined security rules. Modern firewalls also incorporate deep packet inspection and behavior analysis to detect suspicious activities.

The IDS acts as a vital component that tracks network traffic to detect any indications of illegal activities. An IDS system functions through two primary methods: signature-based detection matches network activities against recorded threat patterns and anomaly-based detection relies on behavioural analysis to find system behaviour patterns that differ from normal operations. Modern behavioural-based IDS receives better detection results through ML implementations, which decrease incorrect alert frequencies[13]. Threat intelligence means obtaining and analysing and sharing data about active and emerging cyber threats. The analysis of attackers' data through threat intelligence enables organizations to implement proactive defensive adjustments.

Access control systems, including multi-factor authentication, role-based access control and biometric systems, prevent unauthorized access to sensitive systems and data. MFA introduces an essential security layer through its requirement of two or more verification methods that combine something you know (password) with something you have (security token) or something you are (biometric). System and application security flaws receive ongoing evaluation through vulnerability management which leads to identification and treatment, and reporting of security flaws. System protection depends heavily on the implementation of regular patches and updates which most organizations neglect because these practices eliminate recognized malware and ransomware vulnerabilities.

Organizations can use the NIST Cybersecurity Framework (CSF) to receive structured guidance for managing cybersecurity risks and their reduction. The core functions of cybersecurity consist of five elements that include identify, protect, detect, respond and recover. A layered security approach known as "defense in depth" uses different defensive measures spread across multiple IT system layers to match the described functions. The same level of importance applies to organizational cybersecurity functions which include governance as well as policy enforcement and user education. The human element represents the primary weakness for social engineering attacks that use phishing and pretexting because this field requires continuous awareness training.

2.3 Intersection of AI and Cybersecurity

The meeting point of AI with cybersecurity creates new methods for organisations to identify digital threats while they respond and defend against them. ML, along with deep learning capabilities of AI provide organisations with enhanced cybersecurity measures by performing automated threat detection and improving pattern recognition and predictive analytics. The symbiotic bond between these systems becomes essential because modern cyberattacks rise in complexity while becoming more frequent than traditional rule-based security measures can handle. AI-powered cybersecurity solutions generate adaptable real-time responses to emerging threats by closing the weaknesses that traditional signature-based detection systems create [13].

AI makes its most vital contribution to cybersecurity by detecting irregularities in large networks of data. ML models undergo training to identify typical user patterns which helps them detect security breaches through unusual behavioural patterns. Unsupervised learning algorithms such as k-means clustering and self-organizing maps use k-means clustering and self-organizing maps to detect suspicious login activities and irregular file access and data transfers which indicate potential security risks[32]. The AI models eliminate rule-based dependencies to detect previously unknown attack vectors that cybersecurity professionals call zero-day exploits. In particular, meta-learning techniques display potential for better unknown attack detection through generalization across datasets and improved learning from sparse data[45].

Cybersecurity systems require AI integration because it represents an essential requirement for our modern digital world that faces increasing cyber threats and enhanced digital connectivity. Security teams obtain the power to switch from reactive threat defense to proactive threat hunting and automated incident response through AI. The evolving union of AI and cybersecurity research will probably result in the creation of autonomous defense systems that adapt instantly to threats to safeguard essential infrastructure and protected data systems [37].

3 AI Techniques Applied to Cybersecurity

The defense mechanisms of cybersecurity need continuous advancement to successfully combat present threats and upcoming security risks. Systems that maintain rigid operations through manual updates and outdated threat signatures face dual vulnerabilities to new attack methods and elevated operational expenses from increased personnel needs and potential financial losses. The importance of adaptive systems became evident in 2021 when advertisers lost \$42 billion to fraudulent clicks[5] . An effective Intrusion Detection System (IDS) requires the implementation of machine learning methods, which include supervised and unsupervised learning, together with deep learning and reinforcement learning, and NLP [45, 13, 19]. The system achieves enhanced detection capabilities through these techniques, which enable pattern recognition and real-time response, and automated defense operations, leading to faster and more precise detection. We can categorize the AI techniques applied to cybersecurity into the following five major categories.

(1) **Supervised Learning**: The machine learning method of supervised learning trains models through labeled datasets that link each input to its corresponding output. Through this method, the model develops an understanding of hidden patterns, which allows it to predict new data accurately. High-quality training data enables supervised learning to produce strong predictive results. The technique proves most effective in cybersecurity threat detection because it can identify threats immediately after new threat signatures enter the training data[1]. The requirement for labeled data in supervised learning makes it difficult to detect unknown threats which creates a problem for adapting to changing threat environments. The supervised learning techniques include support vector machines (SVM), Naive Bayes, random forests, artificial neural networks (ANN) and decision trees[35].

Supervised learning techniques have recently evolved into more advanced methods that address cyber threats effectively. The intrusion detection performance of Extreme Gradient Boosting (XGBoost) excels through its ability to analyze network traffic efficiently while detecting malicious patterns with high accuracy[2]. The gradient boosting algorithm CatBoost demonstrates successful phishing website detection through its ability to identify minor patterns in URLs and content features [4]. LightGBM demonstrates successful malware detection in large-scale real-time analysis environments because it provides fast training speed and low memory usage[6].

(2) **Unsupervised Learning**: The analysis of unlabeled data through unsupervised learning methods reveals hidden patterns without needing any prior knowledge or predefined labels. The method excels at detecting unknown threats including zero-day attacks and new malware which the system has not seen before. The techniques k-means clustering and deep belief networks (DBNs)[35] are among the most common methods used. The generalization process of these methods produces two main categories which include threats and non-threats. Unsupervised models excel at detecting anomalies but they face challenges when it comes to accurate interpretation of those anomalies [46]. Unsupervised learning provides an advantage because it operates without needing historical or labeled training data which makes it suitable for threat landscapes that change dynamically[1].

Liu et al. showed that autoencoders excel at network traffic anomaly detection by achieving high detection accuracy with low false positive rates[25]. The hybrid unsupervised approach proposed by Zhou and Wang uses self-organizing maps (SOM) to detect botnet activities which achieved high detection accuracy on benchmark datasets[43]. In [12], the authors introduced an innovative Graph Deviation Network methodology based on GNN, which trains a network to model the relationships among sensors and identify deviations from these patterns. The combination of unsupervised methods with supervised learning techniques enables them to overcome their individual weaknesses while enhancing their individual capabilities to produce more effective cybersecurity solutions.

(3) **Deep learning**: The application of deep learning techniques extends to both supervised and unsupervised data processing. The automatic learning of data features and patterns through deep learning eliminates the requirement for manual feature engineering. The application of this capability brings significant value to intrusion detection systems. Deep learning models identify different types of anomalies which include abnormal traffic patterns and volumes, as well as unidentified threats and fraudulent activities[30]. Neural networks serve as the foundation of deep learning through their interconnected nodes which process data. Diverse Neural networks show superior performance in various cybersecurity operations. Convolutional Neural Networks (CNNs) demonstrate exceptional performance when analyzing network traffic data and

system logs[34]. The time-series analysis capabilities of Recurrent Neural Networks (RNNs) and their Long Short-Term Memory (LSTM) variants make them suitable for inspecting network traffic patterns throughout time[40].

The combination of CNNs with LSTMs in hybrid deep learning architectures produces real-time intrusion detection systems with high detection accuracy[18]. The training of deep learning models on poor-quality or imbalanced datasets makes them more vulnerable to security issues. Models develop incorrect or overly specific patterns from these issues which leads to their inability to detect actual security threats. Attackers can take advantage of these weaknesses through adversarial data injection to make the model misidentify malicious behavior.

(4) **Reinforcement Learning**: Reinforcement learning (RL) enables agents to develop adaptive strategies, making it well-suited for simulated defense training and real-time threat response. When combined with deep learning, it enhances the detection of complex and previously unknown fraud and attack behaviors. The recent developments in RL have led to the development of adaptive and autonomous defense mechanisms in cybersecurity. Deep Q-Networks (DQNs) are used for automated intrusion detection and response where agents learn optimal defense policies against network attacks through simulated environments[36]. Proximal Policy Optimization (PPO) has been effectively applied to malware detection and mitigation, demonstrating improved generalization to novel attack patterns compared to static models[24]. Another innovation leverages multi-agent reinforcement learning (MARL) to coordinate multiple defense agents across a distributed system, enhancing resilience to coordinated attacks[42]. Adversarial RL is also gaining traction, where both attackers and defenders are modeled as competing agents, enabling the development of robust strategies against evolving threats[16]. Finally, hierarchical RL has been utilized for deception-based defense systems, where agents learn high-level policies to deploy honeypots and decoys dynamically, increasing attacker's uncertainty and system protection[8]. These emerging RL approaches represent a shift toward intelligent, proactive, and scalable cybersecurity solutions.

(5) **Natural Language Processing**: The analysis of unstructured textual data through Natural Language Processing (NLP) enables cybersecurity professionals to detect potential vulnerabilities and malicious actors. The system uses its text classification abilities to identify threats before they occur which helps protect both system security and public safety initiatives.The recent developments in NLP have improved multiple cybersecurity applications through better analysis of unstructured text information. BERT and RoBERTa transformer models achieve better phishing attack detection through their analysis of email content and linguistic patterns than traditional classifiers[3]. The com-

bination of NER and dependency parsing techniques helps extract Indicators of Compromise (IOCs) from threat intelligence reports which enhances security operations[39]. The cybersecurity-specific corpora training of SecBERT models with CVE descriptions leads to better vulnerability classification and triage results[23]. The development of NLP-based extractive and abstractive methods for threat report summarization helps analysts work more efficiently and make better decisions[14]. Security analysts can access essential information through natural queries when they use NLP-based question-answering systems to interact with technical documentation and reports[22]. Sentiment and intent analysis models that analyze underground forums and dark web discussions help predict upcoming threats and actor activities[9]. The implementation of NLP-driven methods in cybersecurity operations leads to better automation and enhanced contextual understanding and threat prediction capabilities.

4 AI in Cybersecurity Education

In the following subsections, the importance of teaching AI in cybersecurity and its ethical concerns are discussed in detail.

4.1 Teaching AI in Cybersecurity

The integration of AI into cybersecurity education remains essential because the threat environment continues to evolve while automated defense systems become more prevalent[15]. The increasing complexity and frequency of cyberattacks make AI an essential tool for anomaly detection and predictive analytics and real-time threat response. Students who learn about AI tools and techniques become qualified to create and operate and maintain intelligent defense systems. The inclusion of AI in educational programs enables future cybersecurity experts to defend digital infrastructure through both established methods and innovative technological solutions.

Educators must use practical and dynamic teaching methods to effectively instruct AI concepts within cybersecurity education[44]. The teaching of AI requires students to participate in practical labs and simulations while working together across computer science and data science, and ethical frameworks. Students learn about AI tool applications and constraints in cybersecurity through project-based learning and real-world case studies. The use of open-source platforms and AI-powered security software in educational programs enables students to work with industry-standard technologies while developing technical expertise and critical thinking abilities.

4.2 Ethics of AI in Cybersecurity Education

The increasing role of AI in cybersecurity creates substantial ethical issues that need to be handled in educational environments[17]. Students need to learn about the consequences of AI choices, including algorithmic bias in threat detection and the potential for surveillance abuse and data privacy breaches. The education of cybersecurity professionals should develop their sense of responsibility through teaching them about the social effects of their work and the ethical requirements for AI system development and deployment.

Educators encounter practical obstacles in addition to ethical concerns when they try to implement AI into their cybersecurity teaching programs[17]. The integration of AI into cybersecurity education faces three main obstacles which include maintaining technological updates and securing sufficient computational resources and developing interdisciplinary courses that combine theoretical and practical aspects. The advanced nature of AI presents difficulties for students who lack experience in mathematics or programming. The challenges need ongoing curriculum development together with educator training investments and industry partnerships to deliver updated content and practical learning experiences.

5 Challenges and Limitations of AI in Cybersecurity

The following section highlights key open issues and challenges that require further exploration and attention.

- Data Privacy Concerns: The successful deployment of AI in cybersecurity requires the system to gain access to gather sensitive information it needs to defend. The collection of large datasets by the system raises the probability of security breaches and unauthorized access that could reveal confidential data. The ability of AI systems to exploit attacks that remain unidentified or difficult to detect poses a security threat [11].

- Adversarial Attacks on AI Systems: The combination of strong system protection from AI requires less human monitoring, yet remains vulnerable to attacks[27]. Data poisoning techniques used to manipulate data allow attackers to bypass AI security measures, resulting in incorrect negative results. The development of future AI security systems requires strong input data sanitation and robustness measures to defend against potential attacks.

- Lack of Quality Data: AI cybersecurity applications require extensive datasets but poor quality data and biased training sets along with insufficient real-world attack scenarios will severely limit an AI's protection against new sophisticated threats[11]. Attackers will encounter growing challenges when they transition to non-traditional methods of attacking systems. The continuous evolution of attack methodologies will make it harder for AI to combat these approaches unless organizations develop successful methods to obtain high-quality data.

- Scalability and Performance: The proper operation of AI-driven cybersecurity systems requires advanced, high-end hardware infrastructure. A limited architecture makes it difficult for small businesses to use AI systems. Adding AI features to existing security systems creates additional operational complexity for businesses regardless of their size. Such a major change demands business operations to stop either partially or completely for its implementation. A business which lacks proper knowledge about scaling technologies correctly along with utilizing them optimally may suffer short-term damage while trying to maintain their technological capabilities[10].
- Trust and Interoperability of AI: The security process requires complete oversight of every action to prevent the creation of new vulnerabilities[10]. The "black box" nature of AI makes it challenging to trust with decisions of such importance. The advantages of complex AI systems that perform independent checks are matched by the cybersecurity professionals' reluctance to trust them. We aim to verify that AI uses the most effective explanation methods through explainable AI (XAI) patterns.

6 Future Directions in AI for Cybersecurity

The current proposals for AI applications in cybersecurity face multiple open issues (as discussed above) that require future research attention. The following section identifies several open issues and challenges that researchers from academia and industry need to investigate.

- Autonomous Cyber Defense Systems: AI has the potential to develop autonomous cybersecurity systems that can actively protect against cyber threats in real time[31]. Researchers can work on more adaptive and Resilient AI Models that adapt to concept drift—i.e., evolving cyber threats and tactics. Continuous learning from real-world attack data to stay effective against zero-day attacks.
- Emerging AI Technologies: The research needs to examine how new AI technologies, including XAI enhance cybersecurity results[21]. AI-Enhanced Zero Trust Architectures require continuous verification of user and device identity and behavior and context for integration purposes.

Behavioral biometrics and anomaly detection enable zero-trust principles to be enforced dynamically.

- AI and Quantum Computing in Cybersecurity: The potential of quantum computing in conjunction with AI to break traditional encryption methods and develop new forms of cybersecurity defenses[38]. There is a lot of scope in these areas.
- Integration with Blockchain for Data Integrity: The investigation of AI with blockchain technology can offer a powerful approach to enhancing data integrity and security in cybersecurity systems[20]. By leveraging blockchain's decentralized and tamper-proof ledger, AI-generated logs, decisions, and threat intelligence can be securely recorded and shared without risk of unauthorized modification. This ensures full auditability and transparency, allowing organizations to trace the origins and rationale behind AI-driven actions.
- AI-Powered Security for IoT and Edge Devices: Lightweight AI models operate on restricted edge devices to perform local anomaly detection[41]. The technology serves as a vital security measure for smart homes and autonomous vehicles, and industrial IoT systems. A lot of work can be done for finding robust and lightweight AI models.
- Generative AI for Adversarial Simulation: Generative models such as GANs and large language models (LLMs) are increasingly being used to simulate realistic cyber threats, including phishing campaigns, malware variants, and other attack vectors[29]. By generating diverse and adaptive threat scenarios, these models significantly enhance red teaming and penetration testing efforts. This enables security teams to proactively evaluate defenses against sophisticated and evolving attack patterns in a controlled environment, ultimately improving organizational resilience.
- Collaboration Between Humans and AI: The research on human-AI collaboration in cybersecurity aims to create systems that use AI as an intelligent aid instead of a substitute[7]. AI analyzes extensive data sets to detect minor irregularities which enables it to deliver prompt recommendations that guide human experts toward better choices. The partnership between humans and AI systems improves threat detection capabilities and accelerates incident response while decreasing cognitive overload while maintaining human oversight for ethical decisions and context-dependent choices.
- AI vs AI: The upcoming cyber battlefield will feature defensive AI systems battling against offensive AI systems[28]. The defense of AI models

from deception or poisoning requires investigation of strong adversarial machine learning techniques.

7 Conclusion

The evolution of cyber threats surpasses the capabilities of conventional cybersecurity tools. Modern security challenges require AI as a flexible solution to address them. The security domains of intrusion detection and malware analysis and user behavior analytics benefit from machine learning, deep learning, reinforcement learning, and natural language processing techniques. Deep learning systems excel at processing and classifying large volumes of data in real time, and reinforcement learning systems adapt to evolving threats. Natural language processing systems analyze threat reports to detect phishing attempts through language pattern analysis.

The implementation of AI systems enhances detection precision and shortens response periods while decreasing operational expenses. The implementation of AI in cybersecurity does not provide absolute protection against threats. The development of advanced malware through AI usage by attackers has created a new competitive environment. Security professionals need to understand both the capabilities and limitations of AI systems due to data and algorithm biases that exist. The proper implementation of AI in cybersecurity depends on educational programs and ethical awareness to prevent new security risks.

References

[1] Imatitikua D Aiyanyo, Hamman Samuel, and Heuiseok Lim. A systematic review of defensive and offensive cybersecurity with machine learning. *Applied Sciences*, 10(17):5811, 2020.

[2] S. A. Almarri and H. D. Patil. Intrusion detection using extreme gradient boosting: A comparative study on nsl-kdd dataset. *Procedia Computer Science*, 199:1570–1577, 2022.

[3] Mohammed Alsharnouby and Arsalan Khan. Bertphish: Transformer-based phishing detection using email content. *Computers & Security*, 128:103152, 2023.

[4] A. M. Althubiti and A. A. Gumaei. Phishing website detection using catboost with feature engineering techniques. *Sensors*, 23(5):2457, 2023.

[5] Reem A Alzahrani and Malak Aljabri. Ai-based techniques for ad click fraud detection and prevention: Review and research directions. *Journal of Sensor and Actuator Networks*, 12(1):4, 2022.

[6] M. Asif and W. Liu. Efficient malware detection using lightgbm based feature fusion approach. *IEEE Access*, 9:144489–144498, 2021.

[7] K. C. Chan, R. Gururajan, and F. Carmignani. A human–ai collaborative framework for cybersecurity consulting in small businesses. *Journal of Cybersecurity and Privacy*, 5(2):21, 2025.

[8] Rong Chen and Liang Wu. Hierarchical reinforcement learning for deception-based cybersecurity defenses. *IEEE Transactions on Information Forensics and Security*, 18:98–110, 2023.

[9] Nafis Chowdhury and Farhan Reza. Analyzing dark web forums using nlp: A study on threat actor intentions and trends. *Journal of Cybersecurity Research*, 5(2):200–215, 2023.

[10] CIO Influence. Assessing the scalability of ai-powered security solutions in large enterprises. *CIO Influence*, May 2025.

[11] CTO Magazine. Ai data quality and quantity: Striking the balance. *CTO Magazine*, Apr 2025.

[12] Ailin Deng and Bryan Hooi. Graph neural network-based anomaly detection in multivariate time series. *Proceedings of the AAAI conference on artificial intelligence*, 35(5):4027–4035, 2021.

[13] Roger R dos Santos, Eduardo K Viegas, Altair O Santin, and Vinicius V Cogo. Reinforcement learning for intrusion detection: More model longness and fewer updates. *IEEE Transactions on Network and Service Management*, 20(2):2040–2055, 2022.

[14] Rohan Goyal and Zoya Malik. Threatsum: Automatic summarization of threat intelligence reports using nlp. In *Annual Computer Security Applications Conference (ACSAC)*, pages 122–130, 2023.

[15] Shuchi Grover, Brian Broll, and Derek Babb. Cybersecurity education in the age of ai: integrating ai learning into cybersecurity high school curricula. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, pages 980–986, 2023.

[16] Tao He and Yun Chen. Adversarial reinforcement learning for simulating evolving cyber threats. *ACM Transactions on Privacy and Security*, 25(3):1–25, 2022.

[17] Diane Jackson, Sorin Adam Matei, and Elisa Bertino. Artificial intelligence ethics education in cybersecurity: challenges and opportunities: a focus group report. *arXiv preprint arXiv:2311.00903*, 2023.

[18] M. A. Khan and N. Javaid. A hybrid cnn-lstm deep learning model for real-time intrusion detection in smart environments. *Computers & Security*, 130:103144, 2023.

[19] Muhammad Ismaeel Khan, Aftab Arif, and Ali Raza A Khan. The most recent advances and uses of ai in cybersecurity. *BULLET: Jurnal Multidisiplin Ilmu*, 3(4):566–578, 2024.

[20] R. Kumar and S. Thompson. Ai and blockchain: Powering next-gen data integrity in cybersecurity. *DigitalDefynd*, Mar 2025.

[21] J. Lee and M. Patel. Explainable ai in anomaly detection for zero trust security. *ResearchGate*, May 2025.

[22] Sangwoo Lee and Minseok Kim. Qa4sec: A question answering framework for cybersecurity documents. In *Proceedings of the 2023 Workshop on NLP for Security*, pages 1–9, 2023.

[23] Xiaoyu Lin and Yu Huang. Secbert: A domain-specific bert model for software vulnerability assessment. *IEEE Access*, 11:67850–67862, 2023.

[24] Xiaoming Liu and Fang Zhou. Using ppo-based reinforcement learning for dynamic malware mitigation. In *Proceedings of the 2023 ACM Workshop on AI in Security*, page 35–44, 2023.

[25] Y. Liu, X. Zhang, and J. Chen. Unsupervised network anomaly detection using autoencoders. *IEEE Transactions on Network and Service Management*, 19(2):1457–1467, 2022.

[26] Masike Malatji and Alaa Tolah. Artificial intelligence (ai) cybersecurity dimensions: a comprehensive framework for understanding adversarial and offensive ai. *AI and Ethics*, pages 1–28, 2024.

[27] MindGard AI. 6 key adversarial attacks and their consequences. *MindGard AI Security Blog*, Feb 2025.

[28] V. Muley. The new cyber battleground: Defending against adversarial ai. *LinkedIn Pulse*, Jun 2025.

[29] M. Novelli et al. Generative ai simulation in cybersecurity: Enhancing red teaming. *Artificial Intelligence Review*, 2025.

[30] Merve Ozkan-Okay, Erdal Akin, Ömer Aslan, Selahattin Kosunalp, Teodor Iliev, Ivaylo Stoyanov, and Ivan Beloev. A comprehensive survey: Evaluating the efficiency of artificial intelligence and machine learning techniques on cyber security solutions. *IEEe Access*, 12:12229–12256, 2024.

[31] S. Santos and L. Nguyen. Defensive ai in cybersecurity: Building autonomous systems to counter evolving threats. *ResearchGate*, Apr 2025.

[32] Md Nasif Sarwar, Md Shohel Arman, Touhid Bhuiyan, and Fatama Binta Rafiq. Optimizing intrusion detection with hybrid deep learning models and data balancing techniques. In *2025 IEEE 4th International Conference on AI in Cybersecurity (ICAIC)*, pages 1–6. IEEE, 2025.

[33] Atia Shahana, Rakibul Hasan, Sayeda Farjana Farabi, Jahanara Akter, MAA Mahmud, Fatema Tuz Johora, and Gurkan Suzer. Ai-driven cybersecurity: Balancing advancements and safeguards. *Journal of Computer Science and Technology Studies*, 6(2):76–85, 2024.

[34] R. Sharma and S. Tripathi. Cnn-based deep learning framework for intrusion detection in iot networks. *Journal of Network and Computer Applications*, 213:103535, 2023.

[35] Kamran Shaukat, Suhuai Luo, Vijay Varadharajan, Ibrahim A Hameed, Shan Chen, Dongxi Liu, and Jiaming Li. Performance comparison and current challenges of using machine learning techniques in cybersecurity. *Energies*, 13(10):2509, 2020.

[36] Jie Wang and Qiang Li. An adaptive deep q-learning approach for real-time intrusion response. *IEEE Transactions on Network and Service Management*, 20(2):147–160, 2023.

[37] Zhijia Wang. Research on feature and architecture design of ai firewall. In *2021 5th Annual International Conference on Data Science and Business Analytics (ICDSBA)*, pages 75–78. IEEE, 2021.

[38] A. Williams and R. Smith. Cyber insights 2025: Quantum and the threat to encryption. *SecurityWeek*, Feb 2025.

[39] Lihua Zhang and Renyu Zhou. Automated extraction of iocs from cybersecurity reports using nlp. In *Proceedings of the 2023 IEEE International Conference on Intelligence and Security Informatics*, pages 55–62, 2023.

[40] X. Zhang and Y. Li. An lstm-based approach for network anomaly detection in dynamic traffic environments. *IEEE Access*, 11:50412–50422, 2023.

[41] Y. Zhang and H. Zhao. Securing iot devices with lightweight ai-based intrusion detection systems. *International Journal of Machine Learning Research in Cybersecurity*, 16(01), May 2025.

[42] Yuchen Zhang and Lei Yang. Multi-agent reinforcement learning for distributed cybersecurity defense. *Journal of Cybersecurity*, 8(1):1–15, 2022.

[43] J. Zhou and Y. Wang. Unsupervised botnet detection using self-organizing maps. In *Proceedings of the 2021 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, pages 1–8. IEEE, 2021.

[44] Yujie Zhuang, Ying Wu, Yuhang Guo, and Xuan Luo. Teaching cybersecurity and artificial intelligence via hands-on labs and interdisciplinary projects. In *2020 IEEE Frontiers in Education Conference (FIE)*, pages 1–5. IEEE, 2020.

[45] Tommaso Zoppi, Andrea Ceccarelli, Tommaso Puccetti, and Andrea Bondavalli. Which algorithm can detect unknown attacks? comparison of supervised, unsupervised and meta-learning algorithms for intrusion detection. *Computers & Security*, 127:103107, 2023.

[46] Tommaso Zoppi, Andrea Ceccarelli, Tommaso Puccetti, and Andrea Bondavalli. Which algorithm can detect unknown attacks? comparison of supervised, unsupervised and meta-learning algorithms for intrusion detection. *Computers & Security*, 127:103107, 2023.

A Case for Conveying Design Rationale in Curricular Computing Reports*

Richard Blumenthal
Department of Computer and Information Sciences
Regis University
Denver, CO 80221
`rblument@regis.edu`

Abstract

A case for capturing the rationale used to design a computing curricula report knowledge area is made. To elucidate the argument, several examples underlying the rationale of previous undocumented design decisions for the Algorithmic Foundations Knowledge Area in *Computer Science Curricula 2023* are presented. A model for capturing this curricular design rationale is introduced with a proposal for the computing education community to use such an approach when discussing and developing future computing curricula recommendation reports.

1 Introduction

The design of a computing curricula recommendation report is a laborious multi-year task involving a myriad of undocumented decisions carried out by numerous individuals acting in various roles. In the case of *Computer Science Curricula 2023* (CS2023), ninety-four Task Force members directly contributed to the design over a three year period [17]. As part of this task force, each of the seventeen Steering Committee members formed a working group focused on a subdiscipline Knowledge Area (KA) of computer science, such as Algorithmic Foundations, with the mission of determining the competencies

---

*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.

(knowledge, skills, and dispositions) [9] to include in their KA recommendation. Drawn from experience in participating in the CS2023 Task Force, this article advances an argument for capturing and making transparent the underlying *design rationale* used to decide which competencies to include within a KA. A model for representing this design rationale is proposed along with representative examples of using it to capture the rationale of several undocumented decisions associated with the CS2023 Algorithmic Foundations KA.

Wicked issues are defined as questions that do not have a definitive solution [21]. Consequently, wicked issues are not solved, but are instead resolved. Such resolutions require making judgments that lead to selecting one potential solution from among many conceivable solutions. Arriving at a resolution to a wicked issue is best supported by an argumentative (dialogue) process among interested stakeholders, which requires stakeholders to first identify conceivable solutions and to subsequently select a resolution from among these [18]. From a design perspective, the resolution to a wicked issue constitutes one or more design decisions and the arguments that support or oppose these decisions comprise the underlying design rationale for the resolution. Design rationale models can be proactively used as a tool to arrive at a resolution [12] and as a historical record, which provides the ability to "look back and ask *why did we do it this way?*" [11]. It has been argued that designing a computing curricular report requires addressing numerous *wicked* issues [5].

Previous CS curricular reports, including CS2023, have provided very little explanation, if any, as to why topics were required in the recommendations. As simple examples, the inclusion of Turing Machine or Polynomial (P), Nondeterministic Polynomial (NP), and Nondeterministic Polynomial Complete (NP-C) complexity class topics have appeared as required or elective topics in various CS curricular recommendation reports, with inclusion reversing between subsequent reports [2]. The specific rationale for these reversals is undocumented, leaving the CS computing community to rehash the debates on whether to include such topics in a subsequent curricular revision report, as happened in CS2023. With the availability of previous design rationale, new arguments, and changes to data supporting a previous argument would be easily recognizable, when revisiting decisions to require or remove a competency in a KA.

Capturing a KA's design rationale can be further expected to benefit any *direct* stakeholders interested in contributing to improving the CS curricular recommendations. This rationale would significantly extend the available documented resources, which are limited to existing curricular reports, the scant literature focused on CS curricular content [5], and CS textbooks.[1] As previously suggested, future stakeholders will be able to more easily begin

---

[1] The relation between curicular reports and textbooks as being potentially beneficial or detrimental is explored in [14].

where a previously effort ended, without having to re-debate decisions already made in the preceding report. Potential direct stakeholders include: the "next" Task Force (presumably CS203X), the future community reviewers of CS203X, the Committee on Computing Education in Liberal Arts Colleges [15], the Community College CS Curricular Committee [23], and the recently formed SIGCSE-LACS Computer Science Microkernel working group [7]. Capturing this design rationale can also be expected to benefit *indirect* stakeholders, who are the community members that benefit from an improved set of curricular recommendations and include: computer science students, employers of these students, and society. The later being the ultimate intended beneficiary of the CS education community's curricular work, as suggested by Ethical Principle 1.1 in the *ACM Code of Ethics and Professional Conduct*, which states that "... all people are stakeholders in computing" [20].

In order to begin to better support these stakeholders, the research in this article proposes one approach for capturing curricular design rationale. Specifically, the design rationale underlying several undocumented decisions that shaped the inclusion of topics in the CS2023 Algorithmic Foundations KA (AL-KA) are presented. While limited to this single KA, it is extremely likely this KA will be included in future CS curricula reports. Not only does this design rationale present an historical record of several important CS2023 AL-KA design decisions, it also presents examples of how the next CS curricular Steering Committee in "CS203X" might use such an approach to guide and make transparent their reasoning. Such an approach should also easily extend to other computing curricula efforts (e.g., Cybersecurity [8]). Most importantly, it is hoped that the presentation of this design rationale and approach will spark further discussion in the computing education community focused on the creation and improvement of curricular recommendation reports.

2 Background Literature

2.1 Computer Science Curricula Reports

To date, six computer science curricula reports have been published (CS'68, CS'78, CS'91, CS'01, CS2013, and CS2023) each of which includes required content focused on the nature of algorithmic computation [2]. As suggested in the Introduction section, these reports provide little, if any, rationale for why specific topics were included as required in the recommendations. It has been argued that these "reports are documents that reflect their time. And yet, as written records, they cannot fully capture the context of their time" [14].

2.2 Issue-Based Information Systems

“The concept of [an] Issue-Based Information System (IBIS) rests on a model of problem solving by cooperatives as an argumentative process” [18]. This process is envisioned as a discourse about an unstructured wicked problem area in which *Issues* are disputed using *Arguments* in support of or opposing *Positions* addressing an Issue [18]. IBIS was extended in gIBIS to support design rationale arguments [12]. The gIBIS model was further extended to capture the importance of underlying contextual assumptions that constrain arguments associated with rhetorical dialogue moves [6]. “Indented Text IBIS” (itIBIS) notation [11] provides an approach for capturing design rationale using the following general format:

I: Issue under discussion, which is presented as a question and can be indicated as resolved with a preceding asterisk, ‘*’.
  P : a position responding to an issue, which can be indicated as accepted or rejected using a preceding ‘+’ or ‘-’, respectively.
    Ax = An argument supporting, ‘AS’, or objecting, ‘AO’, to a position.

Specific examples of using itIBIS notation are given in Section 4.

3 CS2023 Contextual Assumptions

The CS2023 AL-KA design is shaped by several *contextual assumptions* made on all KAs in the CS2023 report. While treated as “external” constraints imposed on the design of the AL-KA in this article, the rationale supporting these contextual assumptions could be challenged by the education community. Although such an effort is beyond the scope of this article, the AL-KA design rationale presented in Section 4 provides hints for such challenges.

Several of the CS2023 contextual assumptions are of a definitional nature:

- ***CS Core*** are “topics *every* computer science graduate must know”,
- ***KA Core*** are “topics recommended for more in-depth study”, as found in a dedicated course focused on the KA,
- ***Instructional Hours*** “are hours spent in the classroom imparting knowledge regardless of the pedagogy”, and a
- ***Packaging Course*** contains topics from a KA, which “is assumed to meet for around 40 instructional hours ”[17].[2]

---

[2] Interestingly, U. S. Federal law requires 37.5 clock hours of faculty-supervised instruction for a three credit semester-based course with a clock hour being at least 50 minutes [1]. Equivalent conversions exist for other approaches, such as quarter-based courses.

Additional, contextual assumptions found in the CS2023 Preamble include:

> "The time needed to cover CS Core and KA Core topics is expressed in terms of instructional hours. Students are expected to spend additional time after class practicing related skills and exercising professional dispositions ... CS Core topics at the desired skill levels are estimated to need 270 hours of instructional time. Since every computer science graduate must know CS Core topics, every computer science curriculum is expected to include all 270 hours of instruction" [17].

The Steering Committed also worked under the assumptions that the CS2023 report was to be *evolutionary* and not *revolutionary*, in the sense that it was considered as an update to the preceding CS2013 report [22], was to include a computing competency model guided by CC2020 [9], and the 270 CS Core hours estimate was treated as a target used to minimize the total CS Core hours. The committee self-imposed an assumption to include a Society, Ethics, and Profession (SEP) Knowledge Unit in every KA in order to highlight the importance of SEP competencies in CS.

4 AL-KA Design Rationale

This section presents a few representative AL-KA design rationale examples capturing "*why did we do it this way?*".[11].[3] For each example, determining this "why" is difficult, if not impossible, to ascertain from the CS2023 report or other existing literature. This rationale is presented using the itIBIS notation (Section 2) with an new extension to this model that introduces a counter-argument 'CA' indication. Additionally, the examples are purposively kept terse and summarative in this article for space considerations. In general, entire papers have been written that provide arguments for a single curricular design issue (e.g., [4, 19]). Though, such arguments written as prose are not necessarily the best approach for presenting design rationale (see Section 5).

As deciding which competencies to include in a curricular report is of paramount importance [5], this section begins with a few synoptic examples capturing the rationale supporting such decisions. However, the CS2023 Steering Committee began examination of each KA with three broader questions (issues), which will be presented later in this section. To set the stage, once again, consider the lack of existing design rationale associated with *P, NP, and NP-C Complexity*, whose requirement in the AL-KA has changed between

---

[3] Despite deriving from discussions with the CS2023 Task Force, errors in capturing this historic rationale are the responsibility of the author.

previous curricular reports. In CS2023, this topic is once again required CS Core, as captured in the following rationale synopsis.

- *I: Should *P, NP, and NP-C Complexity* be a CS Core topic?
  - *P: Yes.
    - AS: In the feedback survey, 93% of respondents indicated yes, 7% no.
    - AS: It is the basis for understanding (in)tractability, which had 87% community feedback support for inclusion, with 13% opposing.
    - AS: It is needed to appreciate the unresolved $P \neq NP?$ issue.
    - AS: It is a common topic in Algorithm Analysis textbooks.
    - AS: *Traveling Salesperson* and *Knapsack* are "everyday" NP examples.
  - -P: No
    - AS: It is an advanced topic taught in a theoretical algorithms course.
      - I: Is this really a problem?
        - *P: No
          - AS: 93% of CS programs have a dedicated algorithms course [3].
    - AS: It was elective in the CS2013 and CS'01 curricular reports.
      - CA: It was required in CS78 and CS91.
    - AS: Elimination provides a convenient place to reduce CS Core hours.
      - CA: A dedicated algorithms course has ample instructional hours.

Although the preceding example is terse, the presented rationale highlights the supporting arguments for including this topic as CS core, while also presenting the counter rationale to arguments against its exclusion. The supporting arguments referencing the community feedback survey provide an example of data that could change in the future. For example, if in the future, only 25% of the community supports inclusion of this topic as required CS Core, then perhaps a future curricular committee might reconsider its inclusion. However, they would also know this topic was supported by additional arguments not related to community support.[4] Additionally, if the preceding rationale were presented as hyper-text (or in a graphical form), then it would be possible to link this argument to other design rationale arguments in the KA, or larger CS2023 report. For example, the final Counter Argument (CA) example, namely "ample instructional hours," could be linked to the *CS Core Instructional Hours* Issue, which is presented later in this section.

As a second example in which support of including a topic was more divided, consider the rationale for the decision to exclude *Differential Privacy* as an AL-KA CS Core topic. Unlike the previous example, this issue is not presented

---

[4] An argument in the opposite direction, where community support might increase and thus support inclusion of a topic as CS Core that wasn't required in a previous curricular report is given in the next example issue below, *Differential Privacy.*

as a yes/no question, but as requiring alternative positions, which leads to supporting and opposing arguments that are not necessarily negations of each other. Though, there is still only one supposed resolution to this issue.

*I: How should *Differential Privacy* be handled in the AL-KA?
-P Include it as an *AL-Foundational* CS Core topic.
AO: This is not a foundational algorithm, but a more advanced concept.
CO: a simple *randomized response* technique [13] demonstrates the concept and can be presented to students in a few minutes.
AO: There is no need to give a specific algorithm to demonstrate privacy.
-P Include it as an *AL-Strategies* CS Core topic.
AS: Preserving privacy in algorithmic computation is an increasingly important contemporary topic and an emerging algorithmic strategy.
*AO: In the community feedback survey, 28% support, 33% do not support, and 39% have no opinion as to whether it should be included.
*P Include it as an *AL-SEP* CS Core topic.
AS: *SEP-Privacy* includes it in CS Core, as an example of understanding the ramifications of technology-based privacy protections, hence, include it here as a representative of an algorithm that preserves anonymity.

As captured in this second example, differential privacy was included in AL-SEP CS Core in support of its inclusion as CS Core in the SEP-KA. However, it was indicated as an *example* of an algorithm to preserve privacy.[5] Furthermore, this decision especially reflected the lack of consensus in the feedback from the community for including it as a foundational algorithm, indicated by the '*AO' argument. By making this lack of community support explicit in the decision process, a future curricular committee debating the inclusion of this topic in a proposed curricular recommendation, might decide to include it after realizing community support has changed.

In addition to capturing the rationale for "why" a specific topics are (not) included as CS Core within a KA, there are other important decisions made by a KA working group that affect which topics are included in the KA. Capturing the rationale for these decisions is also important and can be used by future stakeholders. To a large extent these are also contextual assumptions, but ones associated with the KA and not the larger CS2023 report. The next two examples examine two such decisions made at the AL-KA level in CS2023.

---

[5] In CS2023, an early decision was made by the Steering Committee to indicate that while a topical concept, such as Algorithmic Privacy, may be CS Core, a specific example algorithm, such as Differential Privacy, wasn't CS Core itself. Such specific algorithms were indicated with "e.g.". Aside, while this "e.g." rationale is used in the CS2023 report, the rationale for it is not presented. In the AL-KA, it is used precisely in this way.

*I: How should *KA Core* be defined within the AL-KA?

- -P Use the contextual assumption definition of CS2023 (i.e., topics that would be included in a required dedicated algorithms course).
  AS: This keeps intended consistency of KA Core across all the KAs.
  AO: Most CS programs already have a required algorithms course [3].
- *P Associate KA Core topics with an elective *Computation Theory* course.
  AS: *Computation Theory* is a likely algorithmic foundations course [3].
  AO: Only 38% of BSCS programs required a *Computation Theory* course.

This example demonstrates the rationale for treating KA Core differently in the AL-KA. That is, it is assumed that a dedicated algorithms course exists. Hence, KA Core does not consist of competencies that would result from a course dedicated to the KA. This would be unlike a required HCI course since there are not enough HCI CS Core hours in CS2023 to comprise its own course. Though, if a HCI course were required (or offered as an elective), the HCI KA Core specifies what is recommended for inclusion in such a course.

*I: How many CS Core instructional hours should be in the AL-KA?

- -P 48 instructional hours
  AS: Algorithmic content from the AL-Foundational, AL-Strategies, and AL-Complexity KUs are typically covered in CS1, CS2, and a subsequent algorithms related course. A recent study of $N = 500$ Bachelor of Science in Computer Science programs in the United States showed that 100% of these programs required a CS1 and CS2 and 93% require an additional algorithms course for students to graduate [3]. Using the 40 instructional hours per course contextual assumption of CS2023, these three courses allow for 120 CS Core hours. However, the CS2023 SDF and AL KAs only sum to 66 non-overlapping CS Core hours leaving 54 remaining instructional hours in these courses unaccounted for. Consequently, there is considerable more room in actual CS degree programs to present additional AL-KA CS Core topics.
  AS: CS'91 had 47 recommended hours for this KA with the only additional topic being three hours dedicated to Parallel and Distributed Algorithms.
- *P 32 instructional hours
  AS: The need to keep the overall CS Core hours minimal and below 270 hours is a hard constraint.
  AS: 32 instructional hours are more realistic than 40 since most semester courses have additional hours dedicated to reviews, exams, etc.
  - CO: There are still considerable unaccounted for hours in three 32 instructional hour courses (CS1, CS2, and algorithms).

In the preceding example, it is worth noting that discussions related to the number of CS Core hours in CS2023 occurred during every monthly meeting of the Steering Committee. The desire to keep the hours minimal across all KAs significantly affected the design of the AL-KA, which is reflected in this rationale synopsis. This rationale captures the fact that the overall number of CS Core hours in the AL-KA was determined by adherence to keeping the overall CS Core small, as opposed to the hours that appear to be actually available as taught in a significant number of institutions.

In addition to CS Core hour discussions, the CS2023 Steering Committee also spent considerable time focused on competencies. The need to address competencies, as found in the *Computing Curricular 2020* recommendations [9], was treated as a contextual assumption. While a description of CS2023's Combined Knowledge and Competency (CKC) framework and its extension of specifying a task competency as a point in a three dimensional space of components, activities, and constraints can be found in the literature [16, 17], the rationale for why certain dispositions were associated with various KAs isn't. This led to discussions of how to handle competencies, and their associated disposition components, in the AL-KA.

*I: How should Illustrative Outcomes be specified in the AL-KA?
 -P Use the ITiCSE'20 Working Group AL competency statements [10].
   AS: These competency statements are well-thought out and were developed using a proposed approach for generating CS competencies.
   AO: They do not provide complete coverage of the topics in the AL-KA
   AO: The CS2023 CKC framework, especially the requirement of specifying skill levels using only the four verbs explain, apply, evaluate, and develop renders the text for many of these competencies difficult to parse.
 *P Modify and extend the ITiCSE'20 AL competency statements.
   AS: A compromise allowing the essence of the ITiCSE'20 statements.
   AS: Allows adopting the CS2023 CKC framework contextual assumption of specifying the knowledge model and associated skill-level verb forms, separate from the competency model.
   I: How will dispositions be handled?
   *P: They can be included in both the Illustrative Outcome statements and included as separate dispositions (per CKC).
   AS: Provides a compromise.

While the previous rationale is somewhat "in the weeds", since it deals with the grammatical structure of specifying illustrated outcomes as competency statements, a number of revisions were made to the AL-KA as these issues were worked through. Notice, the capturing of this rationale makes clear that the CS2023 committee was attempting to build on the prior work of the ITiCSE'20 Working Group [10], even if some of the AL competency statements from that group are unrecognizable in CS2023. Hence, any discussion focused on revisions of these illustrative outcomes should not focus on whether to use the ITiCSE'20 competency statements, but on how to best capture them. As suggested by the nested Issue in the previous rationale, there were also discussion on dispositions themselves with the ultimate issue being what dispositions to include in the AL-KA dispositions statements.

*I: What dispositions should be included in the AL-KA?
 -P All of the dispositions of the ITiCSE'20 Working Group
  AS: An example and associated argument for demonstrating each of these dispositions in the AL-KA can be made.
  AS: Ultimately, these dispositions should captured in the corresponding Illustrative Outcomes, as competency statements.
  AO: There should be a limit on the number.
 *P Meticulous, Persistent, Inventive
  *AS: These disposition focus on the design of an algorithm (as opposed to, perhaps, the selection of an existing algorithm) to solve a problem.

As presented earlier in this section, the CS2023 Steering Committee began examination of each KA by asking three broader questions (issues). These issues were considered by using CS2013 as a starting context.

*I: Should the CS2013 AL-KA remain in CS2023?
 *P: Yes.
  AS: This KA addressed foundational competencies that are CS Core.
  AS: An AL-KA or course was required in each previous CS report.
  AS: Eighteen AL topics received 100% support as CS Core in the survey.
 -P: No

The first of the three supporting arguments for the resolving "Yes" position captures the fact that every graduate is expected to know the CS Core competencies in this KA and that these competencies are required for further advanced study in CS. The second supporting argument demonstrates the historical importance of this KA, while the third argument supports the currently perceived importance of the competencies within the KA. It is worth noting that this third supporting argument was added after results of the community feedback survey were received (i.e., much later than the decision was made).

Using CS2013 as a starting point, potential additions, deletions, and changes to the AL-KA knowledge units (KU) was subsequently consider. Ultimately, each of these was initiated as a separate issue for each KU. For brevity the rationale for several KUs is combined in the following rationale.

*I: What knowledge units should be removed in the AL-KA?
*P: Advanced Complexity, Advanced Computability, and Advanced Fundamental Data Structures and Algorithms.
AS: These are not separate KUs, but indications of CS vs. KA Core.

An examination of the CS2023 AL-KA reveals that the competencies in the 2013 Basic Analysis and Advanced Computational Complexity KUs were moved to the Complexity Analysis KU in 2023 with advanced topics placed in the KA Core. The above rationale captures the reason for this ontological "clean up".

*I: What knowledge units should be added to the AL-KA?
*P: Society, Ethics, and Profession (SEP)
AS: This addition reflects the Steering Committee's desire to indicate the importance of SEP competencies across the curriculum and not as a separate stand-alone KA or course.

The third initial issue considered for each KA was whether the name of a KA needed to change in CS2023? The folling rationale explains the importance that equating the competencies in the AL-KA played in the resulting name change in CS2023.

*I: What should the AL-KA be named?
-P: Algorithms and Complexity
AS: This is the name used in CS2001 and CS2013 reports.
CA: It was named Algorithms and Data Structures in CS'91.
AO: This name unnecessarily elevates the importance of complexity relative to the other KUs (e.g., AL-Foundations, AL-Models, AL-Strategies).
*P: Algorithmic Foundations
AS: Captures the importance of this KA as CS Core.
AS: This name suggests the KUs are of equal importance.

The examples in this section only provide a synoptic glimpse into the entire rationale underlying the design of the CS2023 AL-KA. Ultimately, the rationale for numerous other discussions and decisions is worthy of capture. A few additional examples are addressed in the next section.

5 Discussion

The fundamental premise in this article is *the computing education community will benefit from capturing and making more transparent the underlying design rationale upon which a computing curricula report depends.* Using itIBIS, a few synoptic examples of the rationale underlying decisions made during the design of the Algorithmic Foundations knowledge area of the *Computer Science Curricula 2023* report were presented. Making such rationale available to the broader community not only increases the understanding of *why the curricular task force did it this way,* but also allows stakeholders interested in resolving associated issues differently to challenge the underlying assumptions for a selected resolution, especially in situations in which the data supporting a decision has subsequently changed. Additionally, by having the design rationale for curricular decisions available, subsequent curricular committees can avoid rehashing previous arguments, especially those leading to "wrong turns". Once again, as a specific example, recall that the rationale as to why the respective previous Steering Committees reversed the decision to require P, NP, and NP-C topics is not available in the existing literature. Consequently, the CS2023 task force was compelled to revisit these decisions without an awareness of the rationale upon which they rested. With a full presentation of the rationale for including this topic and associated competencies in CS2023, as suggested by the synoptic example presented in Section 4, the community and future Steering Committees are better positioned to understand "*why we did it this way*". Perhaps, constructing different supporting and opposing arguments or basing these arguments on newly available data.

As another example, consider the need for the rationale underlying decisions to include (exclude) various topics and whether they are CS versus KA Core. For instance, the underlying rationale for the treatment of Satisfiability (SAT) in CS2023 isn't addressed anywhere. There are subtle arguments in the underlying rationale for this topic. For instance, it isn't technically included as CS Core, but is instead, presented as a representative example (indicated by the use of "e.g." in CS2023) of a problem whose algorithmic solution requires $O(n^2)$ complexity. As there are other such representative algorithms, such as Traveling Salesperson and Knapsack, based only on the CS2023 recommendations, there is no guarantee that a CS graduate would be expected to know any of these important problems (i.e., only that they know one of them). Whether this is a good or bad decision should be debated in the community using some mechanism to capture the rationale for its treatment, one way or the other. There are numerous other subtle issues that could be discussed with the rationale for their resolutions captured, such as the relation in instructional hours between the AL-KA and the Software Development Fundamentals KA. For example, educators sometimes use the implementation of an algorithm as a pedagogical tool to help students better understand the algorithm. However, this approach isn't required since algorithms can be presented, for example, using visual tools. Once again, capturing the rationale for such decisions would benefit the community.

Although there are benefits to capturing (curricular) design rationale and using such an approach to guide decision making when creating curricular recommendations, it comes with a cost. Consider that, while the use of the itIBIS methodology was found to reduce traditional breakdowns in communications among groups, such as clarity in the resolution of issues, when compared to an equivalent representation of the design rationale using prose, there is cognitive overhead associated with using the notation to capture the rationale during meeting discussions and the need to translate the notation when subsequently reviewing the captured rationale [11]. Ultimately, any approach for capturing design rationale must be considered as an integrated part of the process used to find resolutions to issues and not as added afterthought. That is, a curriculum report and its associated rationale must be a active “living document” [11]. Also, capturing design rationale remains in open research area. As a recent example, LLMs have been proposed for generating the design rationale of software architectures [24]. However, it is not clear that such an approach can be used afterwards to create the rationale underlying a curricular report. Additionally, the rationale for including (excluding) competencies in a curricular report aren't always easy to present in a terse form, as exemplified by entire papers making such arguments (e.g., [4, 19]).

In conclusion, the difficulties of capturing the design rationale associated with the creation of a computing curricular report should be viewed as an opportunity for further research in this area. As as starter, the CS education community can adopt a living document approach in which the curricular recommendations and associated design rationale live, for example, on a moderated Web site or Google-Doc like server. Given the potential benefits to all stakeholders resulting from such efforts, it is hoped that this article not only advances the discussion focused on capturing computing curricular design rationale, but also initiates subsequent discussion of this important topic in the computing education community.

References

[1] *38 CFR § 21.4200 - Definitions.* 2025. URL: `https://www.ecfr.gov/current/title-34/subtitle-B/chapter-VI/part-600/subpart-A/section-600.2`.

[2] ACM. *ACM Curricula Recommendations.* 2025. URL: `https://www.acm.org/education/curricula-recommendations`.

[3] Richard Blumenthal. “Alignment Among Normative, Prescriptive, and Descriptive Models of Computer Science Curriculum: The Effect of ABET Accreditation on CS Education”. In: *ACM Transactions on Computing Education* 22.3 (2022), pp. 1–27.

[4] Richard Blumenthal. “Teach More, Not Less Computability Theory”. In: *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education (SIGCSE’22).* 2022, pp. 675–681.

[5] Richard Blumenthal and Johanna Blumenthal. “Moving What’s in the CS Curriculum Forward: A Proposition to Address Ten Wicked Curricular Issues”. In: *Proceedings of the 56th ACM Technical Symposium on Computer Science Education (SIGCSE’25).* 2025, pp. 137–143.

[6] Richard Blumenthal and Johanna Blumenthal. “Using Dialogue Contexts to Support the Tutoring of Ethics”. In: *Proceedings of Artificial Intelligence and Applications (AIA 2010).* 2010, pp. 271–278.

[7] Grant Braught et al. “Towards a Computer Science Curriculum Microkernel”. In: *Proceedings of the 56th ACM Technical Symposium on Computer Science Education (SIGCSE’25).* 2025, p. 1718.

[8] Diana Burley et al. “ACM Joint Task Force on Cybersecurity Education”. In: *Proceedings of the 48th ACM Technical Symposium on Computer Science Education (SIGCSE’17).* 2017, pp. 683–684.

[9] Alison Clear and Allen Parish. *Computing Curricula 2020 (CC2020): Paradigms for Global Computing Education.* 2020. URL: `https://www.acm.org/binaries/content/assets/education/curricula-recommendatio` `cc2020.pdf`.

[10] Alison Clear et al. “Designing Computer Science Competency Statements: A Process and Curriculum Model for the 21st Century”. In: *Proceedings of 25th ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE’18).* 2020, pp. 211–246.

[11] E. Jeffrey Conklin and KC Burgess Yakemovic. “A Process-Oriented Approach to Design Rationale”. In: *Human-Computer Interaction* 6 (1991), pp. 357–391.

[12] Jeff Conklin and Michael L. Begeman. “gIBIS: A Hypertext Tool for Explanatory Policy Discussion”. In: *ACM Transactions on Office Information Systems* 6.4 (1989), pp. 303–331.

[13] Cynthia Dwork and Aaron Roth. “The Algorithmic Foundations of Differential Privacy”. In: *Foundations and Trends in Theoretical Computer Science* 9.3–4 (2014), pp. 211–407.

[14] Sebastian Dziallas and Sally Fincher. "ACM Curriculum Reports: A Pedagogical Perspective". In: *Proceedings of the 11th International Conference on Computing Education Research (ICER'15)*. 2015, pp. 81–89.

[15] Amanda Holland-Minkley et al. "CS2023: Computer Science Curriulum Guidelines: A New Liberal Arts Perspective". In: *ACM Inroads* 16.1 (2025), pp. 40–52.

[16] Amruth N. Kumar et al. "A Combined Knowledge and Competency (CKC) Model for Computer Science Curricula". In: *ACM Inroads* 14.3 (2023), pp. 22–29.

[17] Amruth N. Kumar et al. *Computer Science Curricula 2023*. ACM Press, IEEE Computer Society Press, and AAAI Press, 2024. URL: `https://doi.org/10.1145/3664191`.

[18] Werner Kunz and W. J. Rittel. *Issues as Elements of Information Systems*. Institute of Urban and Regional Develpment, University California, Berkeley, 1970. URL: `http://magrawal.myweb.usf.edu/phd/articles/ibis_wp_70.pdf`.

[19] Marecelo Pias et al. "Should Quantum Processor Design be Considered a Topic in Computer Architecture Education?" In: *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE'22)*. 2022, p. 1184.

[20] ACM Committee on Professional Ethics. *The Code: ACM Code of Ethics and Professional Conduct*. 2018. URL: `https://www.acm.org/code-of-ethics`.

[21] Horst W. T. Rittel and Melvin M. Webber. "Dilemmas in a General Theory of Planning". In: *Policy Sciences* 4.2 (1973), pp. 155–169.

[22] Mehran Sahami et al. *Computer science curriculum 2013*. 2013. URL: `http://dx.doi.org/10.1145/2534860`.

[23] Christian Servin et al. "Community Colleges Perspectives: From Challenges to Considerations in Curricula Development". In: *Proceedings of the 45th ACM Technical Symposium on Computer Science Education (SIGCSE'23)*. 2023, p. 1244.

[24] Xiyu Zhou et al. *Using LLMs in Generating Design Rationale for Software Architecture Decisions*. 2025. URL: `https://arxiv.org/pdf/2504.20781?`.

Breaking Silos: An Agile Approach to Interdisciplinary Software Engineering Pedagogy*

Lynn Roy Thackeray[1] and Susan L Thackeray[2]
[1]Computer Science
[2]Technology Managment
Utah Valley University
Orem, UT 84054
`{Lynn.Thackeray, susan.thackeray}@uvu.edu`

Abstract

The increasing complexity of software development necessitates collaboration among professionals spanning diverse disciplines. Consequently, software engineers must cultivate robust communication and teamwork skills to effectively navigate interdepartmental, organizational, and interdisciplinary environments. While interdepartmental collaborations have been employed in pedagogy, the integration of interdisciplinary concepts within software engineering education remains largely underexplored. This paper examines a novel pedagogical approach that applies an Agile Scrum framework within an interdepartmental, active learning environment to enhance interdisciplinary collaboration and skill development among undergraduate and graduate students. Two case studies are highlighted: the Introduction to Technology Management undergraduate course, which emphasizes project management under constraints, and the graduate-level Software Engineering Leadership course, which focuses on software project deliverables within distributed team settings. Both courses prioritize active learning strategies designed to foster teamwork, communication, and adherence to structured development frameworks involving both technical and non-technical participants.

---

*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.

This study explores the deliberate alignment of course content between disciplines to create an integrated curriculum that prepares students for the multifaceted challenges of the software industry. Through experiential collaboration between students and faculty within the College of Engineering and Technology, insights into interpersonal relationships and project management efficacy are gained. The outcomes of the evidence-based practice demonstrate the significance of interdisciplinary pedagogy in refining engineering student competencies and enhancing project deliverables in a realistic, distributed team environment.

1 Introduction

The inherent complexity of software development has long been recognized as a major challenge for developers, planners, and users alike. As Ray Ozzie famously articulated in a 2005 memo, ”Complexity kills. It sucks the life out of developers; it makes products difficult to plan, build, and test; it introduces security challenges; and it causes user and administrator frustration”[6]. While Ozzie's observation highlighted the difficulties of his era, contemporary software developers contend with an even broader spectrum of complexities. Modern software must accommodate expectations for flexibility in features, connectivity options, performance, and compatibility across multiple platforms, including the Internet of Things (IoT), mobile applications, web-based deployments, and cloud-based systems.

In addition, the current pace of business innovation demands that organizations develop and evolve software systems at unprecedented speeds. To address these demands, software development organizations have widely adopted agile and distributed development methodologies [10]. These approaches prioritize collaboration and communication, necessitating that engineers possess advanced interpersonal and teamwork skills to succeed in increasingly distributed, globalized, and interdisciplinary environments. Cross-functional collaboration, which involves teams from distinct departments, disciplines, or organizations working toward shared objectives, has transitioned from being a desirable attribute to an essential competency. Such collaboration requires an understanding of varied roles and responsibilities to achieve cohesion and efficacy within distributed project teams spanning diverse geographies and time zones.

This paper investigates a pedagogical approach to address the challenge of preparing students for collaborative, distributed work environments. Specifically, it examines an interdepartmental strategy for teaching cross-functional collaboration within software engineering contexts. Through coordinated efforts between college departments, students engage in simulated project-based activities designed to mirror distributed organizational structures. This approach not only encourages a deeper understanding of individual roles and responsibilities but also enhances communication, fosters shared project objectives, and equips students with practical experience in distributed team dynamics.

The findings underscore the transformative potential of interdisciplinary teaching in fostering effective collaboration among engineering students. This study presents a framework for integrating technical and managerial principles in an active learning environment, ultimately preparing students to thrive in the complexity-driven landscape of modern software development.

2 Agile Development Framework

Agile software development serves as an overarching term encompassing frameworks such as Scrum, Extreme Programming, and Feature-Driven Development. Central to Agile is the emphasis on small, multidisciplinary teams composed of individuals with diverse skills and backgrounds who collaboratively design, build, test, and deliver software products. The Agile Manifesto underscores the importance of direct and open communication, prioritizing individuals and interactions over processes and tools, and adaptability to change over adherence to rigid, predetermined plans. A defining characteristic of Agile is its focus on simplifying communication by minimizing the steps required to convey information effectively.

Key principles of Agile development include simplicity, flexibility, and iterative processes[2]. Agile teams evolve their strategies dynamically, making it challenging to predict the specific state of a project or its communication demands at any given point. Consequently, traditional documentation-heavy communication practices are often deemed inefficient and counterproductive in Agile settings. Agile frameworks commonly replace traditional weekly departmental meetings with concise daily stand-up meetings, fostering interactive and transparent communication. However, despite its strengths, Agile's communication-focused methodology has faced challenges with the emergence of distributed software development models, which necessitate alternative approaches [9].

3 Distributed Software Development

Distributed software development refers to the planning, design, construction, testing, and management of software projects executed by decentralized teams operating across multiple physical locations, often spanning different time zones, countries, or even continents [5]. Similar to Agile, distributed development emphasizes team-based collaboration. However, its principles diverge significantly from those of Agile, particularly in communication practices. While Agile primarily relies on informal, synchronous communication, distributed development depends on more formalized mechanisms to address the complexities of decentralized work environments. These mechanisms accommodate asynchronous communication, which is independent of time and location constraints.

Although Agile development frameworks and distributed development models have long been established within the industry and extensively validated [11][12], their implementation requires robust communication and teamwork capabilities. These competencies, once considered supplementary "soft skills," have now become essential prerequisites in the modern software engineering landscape. Employers increasingly regard effective verbal and written communication, along with teamwork, as indispensable qualities. Recent graduates entering the workforce can anticipate interview processes that rigorously evaluate these skills, extending beyond the scope of basic classroom group exercises [7].

4 Curriculum Challenges

Despite the evident demand for interdisciplinary skills, academic institutions often remain confined to traditional, siloed classroom structures. Within these settings, project design and development are typically framed as formal course assignments constrained by semester-based timelines and limited to interactions among students within the same class. Critical processes such as functional specification gathering, operations management, and interdepartmental logistics—encompassing marketing, sales, and customer support—are either excluded or simulated superficially.

These disciplinary silos are perpetuated by institutional structures and performance metrics, such as the reliance on student credit hours, which influence departmental organization and faculty resources. Such structural constraints often result in separate academic programs that limit students' exposure to cross-functional communication. In disciplines like Computer Science and Software Engineering, this focus on specialization prioritizes technical knowledge in designing and developing software products over broader, interdisciplinary competencies.

Faculty members and curriculum designers are aware of these shortcomings, but resource limitations frequently hinder the integration of distributed and cross-functional development into course content. Furthermore, faculty expertise in departments like Computer Science and Software Engineering tends to be concentrated on the technical aspects of software creation and deployment. Consequently, instructors may feel unprepared or reluctant to teach topics that extend beyond their technical domain. Existing teaching materials within these programs are similarly specialized, often lacking an orientation toward integration and interdisciplinary collaboration. As a result, curricula remain narrowly focused, despite increasing demands from industry for graduates who possess strong communication and collaborative skillsets.

5 Literature Review

Recent studies have investigated the efficacy of interdepartmental approaches to teaching cross-functional collaboration in software engineering. For instance, Montero et al.[1] analyzed the impact of such pedagogical strategies on team collaboration and software development performance. The study concluded that interdepartmental collaboration enhanced team dynamics, improved understanding of individual roles and responsibilities, and resulted in higher software quality, as evidenced by fewer bug reports and improved code quality.

Similarly, research by Dutta et al.[8] demonstrated that interdepartmental approaches not only strengthened collaboration and role clarity among team members but also elevated performance metrics, including code quality, task completion rates, and reduced bug frequency. These findings highlight the tangible benefits of interdisciplinary teaching in fostering more effective software engineering practices.

Gudiseva et al.[3] also assessed the impact of interdepartmental strategies on cross-functional collaboration. Their results corroborated previous findings, showing that such approaches significantly enhanced teamwork, role comprehension, and software development outcomes. Higher collaboration levels and improvements in software quality metrics, such as reduced bug reports, further support the value of interdepartmental methodologies in engineering education.

6 Discussion

The findings presented in the reviewed literature indicate that adopting an interdepartmental approach to teaching cross-functional collaboration in software engineering yields significant benefits. Teams engaging in such an approach exhibit enhanced collaboration and a deeper understanding of individual roles and responsibilities, resulting in improved software development performance. These outcomes are reflected in higher code quality and fewer reported bugs. Furthermore, an interdepartmental approach facilitates the development of effective communication skills and fosters a shared understanding of project goals and objectives. Such advancements enable teams to collaborate more efficiently and effectively in distributed and interdisciplinary settings.

7 Motivations

Recognizing the gaps in traditional educational approaches, this study seeks to implement an interdepartmental, cross-functional framework for teaching communication and collaboration in software engineering. The focus of this framework is to equip students with the ability to engage with teams beyond the software development collective, particularly within distributed (non-co-located) organizational structures. While cross-functional collaboration and communication are not new concepts in software engineering education, the approach introduced in this paper uniquely emphasizes distributed team-based learning and communication. Drawing on the team-learning methodology developed by Michaelsen and Sweet[4], this framework facilitates the application of collaborative problem-solving among learners.

The motivation for this course stems from the observation that university students often remain confined within degree-specific silos. As previously noted, programs in Computer Science and Computer Engineering frequently emphasize the technical aspects—the “how”—of software development and management, leaving minimal room for interdisciplinary engagement. Many students lack professional experience and may feel apprehensive about group-based coursework. However, technical projects that require teamwork and foster innovation necessitate a diverse team of subject matter experts extending beyond the scope of a typical Computer Science classroom. A pluralistic system of project development allows technical teams to capitalize on the strengths of diverse expertise for effective collaboration.

This study highlights a successful demonstration of such pluralistic collaboration at Utah Valley University (UVU). The partnership involved two distinct courses: a Computer Science graduate course (Software Engineering Leadership, CS6300) and an undergraduate course (Introduction to Technology Management, TECH 3000). These courses operated independently in different physical locations, with separate class schedules, mirroring the conditions of a typical distributed industry environment. The project required both groups to design, develop, and implement a fully functional product. For successful project completion, the teams were required to integrate and maintain direct, continuous communication throughout the process.

8 The Course

The project brought together two distinct student groups. The graduate course, Software Engineering Leadership (CS6300), offered by UVU's Master of Computer Science program, was paired with the undergraduate course, Introduction to Technology Management (TECH 3000), housed within the Technology Management department. This pairing aimed to create diversity in team composition. Students in the Technology Management program primarily focus on general project management across industries such as manufacturing, food services, and construction, with minimal exposure to technology-focused fields like Software Engineering or Computer Science. In contrast, students in the Software Engineering Leadership course concentrate on managing software development projects, often with a strong technical background in Computer Science.

To integrate the two groups effectively, the project utilized the Scrum framework, an iterative and team-based approach to project management. This framework provided multiple integration points for the teams, facilitating collaboration and communication throughout the development process. Details of these integration points and their role in fostering interdisciplinary teamwork will be discussed further below.

9 The Course Project

The course project was designed to replicate, as closely as possible, a realistic industry scenario. The project introduced students to a simulated corporate environment through a fictitious company, Pegasus Inc., a medium-sized enterprise specializing in the design, construction, and manufacturing of lawn care equipment. Pegasus Inc. operates as a dispersed organization with approximately 5,000 employees distributed across various locations: the corporate headquarters, encompassing the Marketing, Sales, and Project Management departments, is situated in Westport, Connecticut; the manufacturing facilities are located in Athens, Georgia; and the Engineering division, which includes the Software Development department, is based in Orem, Utah.

In this simulation, students enrolled in the undergraduate Technology Management (TECH 3000) course represented the Project Management department located in Westport, Connecticut. Meanwhile, students from the graduate-level Software Engineering Leadership (CS6300) course assumed the role of the Software Development department based in Orem, Utah. To provide oversight and guidance, the instructors of the two courses acted as the project stakeholders.

As part of the scenario, the Software Development department had recently adopted the Agile Scrum framework, while other divisions, such as Engineering and Manufacturing, continued to utilize the Waterfall methodology. Consequently, all projects—whether executed under Scrum or Waterfall—fell under the management of the Project Management department. The Project Managers were tasked with close collaboration across all teams, including the Software Development department. They also served as the primary point of contact for project stakeholders, including the Marketing and Corporate departments. Additionally, Project Managers assumed the critical role of liaison between the stakeholders and the Software Development teams. Within the Agile framework, the Project Manager functioned as the Product Owner, facilitating alignment and communication across all parties involved.

10 Proposed Project

Pegasus Inc., a company with an established and successful product line of both electric and gas-powered lawnmowers, is exploring the potential to expand its offerings with the introduction of an electric robotic lawnmower. Drawing inspiration from the popularity of automated vacuum cleaners, this new lawnmower would be tailored for outdoor use. Unlike its indoor counterparts, the robotic lawnmower would autonomously learn the layout of a lawn, navigate outdoor obstacles, and perform its primary function—cutting grass instead of vacuuming floors.

Prior to committing substantial resources to the design and manufacturing phases of the proposed product, the Marketing department has identified the need for a software simulation. The primary objective of this simulation is to showcase the robotic lawnmower's anticipated functionalities. Marketing and Sales teams would utilize the simulation as a tool to assess market interest and gauge the product's potential acceptance. Additionally, the corporate leadership envisions this simulation serving a dual purpose. Beyond its value as a marketing asset, the simulation would also act as a foundational software prototype for the product's eventual development. This would necessitate the inclusion of a functional prototype and comprehensive supporting documentation.

10.1 Project Requirements

Initial requirements for the simulation system have been outlined collaboratively by the Marketing Research team and corporate planners:

- The System must be able to read in a file that will describe the lawn to be cut, and then track the progress of the lawnmower as it moves around the lawn.

- The lawnmower needs to avoid obstacles, and to stay in-bounds (on the lawn) and cuts only the grass.
- The lawnmower needs to keep track of grass that has been cut so as not to re-do any areas. The lawnmower will need to sense when the entire lawn has been cut and turn itself off.

Additional requirements are currently under review, with several under active discussion. To ensure the simulation meets the broader needs of Pegasus Inc., input from both the Project Management and Software Development departments is essential before finalizing the system's specifications.

10.2 Project Schedule and Deliverables

The project is structured within an 8-week development window, during which Corporate has assigned it high priority. This requires immediate formation of project teams, including the identification of available Project Managers and Development teams at the start of the semester. Within the first week, an official project kickoff will take place. By week five, teams are expected to present a functional design prototype demonstrating as much functionality as possible to Corporate stakeholders. This presentation will act as a critical go/no-go decision point for the project. If approved, the teams will progress toward delivering a fully functional simulator to the Marketing department by week fifteen, marking the conclusion of the semester. During the final week, the Software Development teams will conduct demonstrations and provide training sessions for the Marketing and Sales departments.

These milestones and requirements provide a high-level framework, and Corporate (course instructors) remain open to detailed modifications during project execution. However, the deadline for deliverables is non-negotiable due to business considerations. Following project completion, a post-mortem review (final course evaluation) will be conducted by the stakeholders. A successful project outcome may lead to Phase II development.

11 Project Structure

The project spans the entire semester and mandates the use of Agile Scrum as the management framework. Scrum teams, composed of 3 to 5 members, integrate students from both courses. These cross-functional teams are designed to ensure that all members possess the necessary skills to contribute to value creation during each iterative development cycle, or Sprint. Teams are self-managing, with autonomy to decide on roles, task assignments, and schedules internally.

Project Managers are selected from the Technology Management course (TECH 3000), while software developers are drawn from the Software Engineering Leadership course (CS6300). Within each team, a Scrum Master (SM) is designated and chosen by the team members. Additionally, a lead Project Manager (PM) is identified. To ensure equitable participation, these leadership roles are rotated weekly, allowing all team members to gain experience in both positions. The project schedule accommodates ten one-week Sprints throughout the semester.

11.1 Collaboration & Communication Requirements

For each one-week sprint, students would be required to:

- Plan and hold an initial planning meeting to identify, prioritize and assign development tasks for the upcoming sprint.
- Schedule and hold at least three daily "stand-up" meetings in which all team members would report on their progress and identify any issues.
- Plan and hold an end of sprint review, where new functionality would be demonstrated for the stakeholders and other sprint groups (both classes attending).
- A mid-sprint update report to the stakeholders, to be completed jointly by the SM and PM.
- An end-sprint retrospective report to stakeholders to be created by the Scrum teams and PM department.

The Scrum Master and Project Manager are required to generate weekly reports on the above meetings.

11.2 Project Deliverables

- Working Final project: Approximately 150-200 lines of code per develope
- Functional Specifications document.
- A complete product design document using Unified Modeling Language (UML)
- Technical Documentation (how to set up and run simulation end of project (semester)
- Product demonstration to stakeholders

All documentation and product demonstrations were to be completed jointly as a collaboration between CS6300 and TECH 3000 students.

Because of the distributed nature of the two groups (both courses met in different buildings and at different times), Scrum teams were responsible for formulating and implementing a plan on how and what type of technology they would use to collaborate and communicate.

11.3 Method of Instruction and Evaluation

The scrum requirements listed above necessitated close communication and collaboration between both TECH 3000 and CS6300 students. To support and guide these efforts, an active learning framework combined with a scaffolding teaching strategy was employed. Instructors delivered lessons, guidance, and instructions in distinct segments, providing less and less support as students began to master project and course concepts.

12 Observations and Evaluations

The objective of this paper is to outline the implementation of an interdepartmental approach to teaching cross-functional collaboration. While the purpose was not to produce empirical research findings, key observations were made throughout the course. Students progressively began to recognize their individual subject matter expertise and their contributions to the overall deliverable. This experience fostered an understanding that no single ”correct logic” exists, but rather, multiple valid approaches can coexist within a framework of pluralism and unity, where diverse groups share influence and responsibility. Each participating group played a crucial and interdependent role in delivering the final product.

As previously described, the role of Product Manager (PM), with overarching project management responsibilities, was assigned to undergraduate students enrolled in the Technology Management (TECH 3000) course. Meanwhile, the Software Engineering Leadership (CS6300) graduate students, comprised of software engineers, assumed responsibility for the technical development, encompassing the architecture, design, and construction of the project. This structure deliberately exposed TECH 3000 students to the intricacies of the software development process, offering them a deeper understanding of its inherent complexities. Concurrently, the CS6300 students were required to collaborate closely with a largely non-technical group, which necessitated adapting their communication style to minimize technical jargon and adopt a more professional, business-oriented approach.

Students entered the course with varied levels of familiarity with Agile software development practices, particularly Scrum. For most Technology Management undergraduates, the Agile Scrum framework was novel, while many Software Engineering graduate students were already familiar with Scrum or similar Agile methodologies. Implementing Scrum within an educational setting imposed significant demands on instructors. However, the instructors’ experience in facilitating cross-functional collaboration revealed that fostering a classroom environment where students consistently engage with a clearly defined project is advantageous. Such an environment closely mirrors the real-world application of Agile processes. Additionally, the simulated distributed setting enabled students from distinct academic disciplines to communicate and collaborate effectively.

With clearly defined project deliverables and milestones, students successfully coordinated their efforts. They meticulously prepared and conducted key Scrum ceremonies, including daily stand-ups, sprint reviews, and retrospectives. Teams also established and maintained critical Scrum artifacts, such as burn-down charts, product backlogs, and end-of-sprint reports.

The adoption of Scrum initially introduced complexity to the learning environment. To ensure the effectiveness of Scrum teams, students needed to go beyond superficial adherence to its practices. A foundational understanding of the principles underlying systems and human interaction in Scrum was essential and was incorporated into the course learning objectives.

The project evaluation comprised three sections:

1. Student collaboration and communication.
    - Daily Scrum stand up meetings
    - Team management (Scrum Master)
    - Project progress and reporting (Project Manager)
    - Development progress (Scrum Team)
    - Weekly status presentation to stake holders (Scrum Team, Project Manager)
2. Project Deliverable.
    - Running, workable solutions with required documentation (see Project Deliverables section)
3. Final Presentation.
    - Both classes were required to collaboration in presenting a formal final project demonstration and project recap to state holders (Instructors).

The students did meet the assignment requirements. The project submission was a multi-media solution that included a robust graphical user interface (GUI), robotic lawn mower animation, sound, a user dashboard that includes metrics on run time, gasoline consumption, percentages of lawn cut, etc. Additionally, there was file upload and save option for different lawn scenarios.

The final project presentation was a collaborative effort between both classes. This was a formal scripted presentation where students demonstrated their project to stakeholders (course instructors).

13 Conclusions and Next Steps

13.1 Analysis of the Literature Review

The literature review examined current research on the implementation of an interdepartmental approach to teaching cross-functional collaboration in software engineering. The findings indicate that such an approach enhances team performance, fostering higher levels of collaboration, a clearer understanding of team members' roles and responsibilities, and improved software development outcomes. Furthermore, the research highlights the critical importance of effective communication within interdisciplinary teams. In the authors' view, this process cultivated an environment in which team members actively developed strategies to communicate efficiently and build a shared understanding of project goals and objectives. The observations derived from the course activities align with and reinforce the findings of existing research.

The primary objective of the interdepartmental approach was to simulate a professional environment in which diverse groups, separated not only by time and location but also by distinct professional cultures, could engage with and address common challenges encountered in real-world management and development contexts. It is the authors' assessment that this objective was successfully achieved. By requiring a collaborative project deliverable from two distinctly different groups of students—meeting at separate times and in different locations—the course simulated authentic industry challenges. These challenges necessitated innovative problem-solving to ensure the successful completion of the project.

13.2 Future Directions

The instructors intend to continue incorporating this module into their instructional practices, recognizing its value in bridging disciplinary gaps. Plans are underway to expand the initiative by formalizing research efforts on cross-functional collaboration. This expanded focus will include discipline-specific iterations of the module to collect more comprehensive data on student engagement and outcomes.

References

[1] F Ahmed, LF Capretz, and P Campbell. Evaluating the demand for soft skills in software development. *IT Professional*, 14(1):44–49, 2012.

[2] Lili Cao, Karthik Mohan, Peng Xu, and Balasubramaniam Ramesh. A framework for adapting agile development methodologies. *European Journal of Information Systems*, 18(4):332–343, 2009.

[3] P Dutta, M Hira, and A Kumar. Impact of inter-departmental collaboration on software engineering. *International Journal of Computer Science and Information Security*, 16(2):30–36, 2018.

[4] K Gudiseva, A Goel, A Kumar, S Chaudhary, and V Chawla. An interdepartmental approach for effective cross-functional collaboration in software engineering. *International Journal of Computer Science and Information Security*, 15(3):59(5), 2017.

[5] Mikko Korkala and Pekka Abrahamsson. Communication in distributed agile development: A case study. *In 33rd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROM/CRO 2007)*, pages 203–210, 2007.

[6] Steve Lohr and John Markoff. Windows is so slow, but why? *The New York Times*, March 2006.

[7] A B M Moniruzzaman and D S A Hossain. Comparative study on agile software development methodologies, 2013. DOI: `doi.org/10.48550/arXiv.1307.3356`.

[8] JG Montero, JL Lopez, MI Garcia, and M Garcia. Impact of interdepartmental collaboration on software engineering teams: Evidence from spanish software companies. *Journal of Software Engineering Research and Development*, 7(1):1–17, 2019.

[9] N Rahman, A Wittman, and S Thabet. Managing an engineering project. *International Journal of Information Technology Project Management (IJITPM)*, 7(1):1–17, 2016.

[10] Yvonne Rogers. Ghosts in the network: distributed troubleshooting in a shared working environment. *In Proceedings of the 1992 ACM conference on Computer-supported cooperative work*, pages 346–355, 1992.

[11] Suprateek Sarker, Manish Ahuja, and Sourav Sark.er. Work-life conflict of globally distributed software development personnel: An empirical investigation using border theory. *Information Systems Research*, 29(1):103–126, 2018.

[12] B Sengupta, S Chandra, and V Sinha. A research agenda for distributed software development. In *Proceedings of the 28th international conference on Software engineering*, pages 731–740. Association for Computing Machinery, 2006.

Best Practices for Inclusive Teaching in Computer Science[*]

Thyago Mota
Department of Computer Sciences
Metropolitan State University of Denver
Denver, CO 80204
`tmota@msudenver.edu`

Abstract

This paper outlines practical strategies for inclusive teaching in computer science, with a focus on supporting underrepresented students in higher education. Drawing from experiences at a Hispanic-Serving Institution, it addresses barriers such as bias, lack of representation, and inequitable access to resources, offering actionable steps to promote equity and foster a sense of belonging in the classroom. Inclusion is especially critical in computer science programs, a field historically dominated by certain demographic groups, which has led to a lack of diverse role models and limited representation for many students.

1 Introduction

As faculty at a university that values diversity and respect, we are proud to be part of MSU Denver, a federally designated Hispanic-Serving Institution (HSI) with a long history of welcoming students from various racial and ethnic backgrounds. Recognized as one of the most affordable higher education institutions in Colorado, MSU Denver attracts students from diverse socioeconomic backgrounds. Many of these students are awarded scholarships or receive work-study assistance, enabling them to achieve their dream of completing a four-year degree—often becoming the first in their families to do so.

---

[*] 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.

While we are proud to serve students at a university that embraces diversity, we recognize that much work remains—particularly in our field of computer science, which has traditionally been one of the least diverse [3]. Numerous studies show that Black, Hispanic, and Indigenous students earn undergraduate degrees in STEM fields at rates lower than their representation in the general population [6, 16, 7]. This low graduation rate is reflected in the industry, where these groups remain underrepresented in computer science roles, perpetuating cycles of poverty and inequality.

Diversity is critically important in fields like computer science, where solving complex problems often requires multiple perspectives. Additionally, the issue of underrepresentation extends to gender, as women continue to be underrepresented in science, technology, and engineering fields [12].

Many studies have explored the reasons behind underrepresentation in STEM fields. One significant factor is the higher dropout rate among underrepresented students compared to white students—a gap that is even more pronounced in computer science [14]. The lack of role models for these students further discourages their persistence in these fields. Additionally, our own attitudes and stereotypes—known as implicit biases—can unconsciously influence our actions and decisions in teaching, exacerbating the problem. Cultivating a culture that values inclusion can help mitigate the effects of inequality in both teaching and research activities.

This paper aims to highlight effective and easily implementable inclusive teaching practices, with a particular focus on computer science. It presents five carefully selected strategies, each accompanied by practical examples and resources to support easy and direct application in everyday teaching scenarios. Whether you're leading a classroom or mentoring students in research, we hope that these practices will help you foster a more inclusive and supportive learning environment.

2 Background

Inclusive teaching is generally defined as an approach that ensures all students—regardless of their backgrounds, abilities, or learning styles—have equal access to learning opportunities. It involves designing learning environments that are both equitable, where all students have the opportunity to reach their potential, and welcoming, fostering a sense of belonging.

Inclusive teaching enhances students' perception that their contributions are valued, regardless of gender, race, sexual orientation, ability, religion, socioeconomic status, educational background, health status, or other aspects of identity. It embraces diversity and acknowledges student differences, taking a non-exclusionary approach to creating effective teaching and learning environments.

Being inclusive is important because it helps create the conditions in which all students can fulfill their potential. Reference [1] is a valuable resource for learning more about what it means to be inclusive in teaching and why educators should embrace it.

Although legislation exists to protect students from various forms of discrimination—such as those based on race, creed, gender, sexual orientation, or physical ability—being inclusive in the classroom and in interactions with students requires more than mere compliance with legal standards.

For example, a school may follow anti-discrimination laws by ensuring no student is excluded based on race or ethnicity. However, if the curriculum predominantly reflects the history and culture of only one group, it fails to be inclusive of the diverse backgrounds of all students. Similarly, providing gender-neutral restrooms is not enough if the school culture does not support students, allowing them to face bullying or harassment for using these facilities. True inclusivity involves creating an environment where all students feel respected, supported, and valued.

Successful inclusion in education requires not only institutional support but also the active participation of instructors in their daily teaching practices. This begins with preparing the course curriculum, presenting the course plan to students, planning activities and assessments, conducting classes, and using appropriate teaching tools.

Although there is extensive research in the field of inclusive teaching, this paper aims to highlight a few high-impact practices that have demonstrated effectiveness through empirical studies and to explore how they can be applied in the classroom. For those interested in learning more about equitable and inclusive evidence-based teaching practices, reference [11] is highly recommended.

Being inclusive—particularly in computer science teaching—presents unique challenges. Historically, the field has been dominated by certain demographic groups, resulting in a lack of diverse role models for students. Additionally, students from underserved communities may lack access to up-to-date hardware, software, and reliable internet connections, all of which are crucial for engaging with the CS curriculum.

Other challenges include adapting teaching methods and materials to accommodate different learning styles and abilities, as well as integrating the lived experiences of diverse identity groups into the curriculum. Doing so helps foster a supportive learning environment that values and reflects all students' backgrounds.

3 Inclusive Teaching

There are numerous strategies for creating an inclusive teaching environment. This section highlights a selection of highly effective inclusive teaching practices, supported by research in the field. Each practice is described in detail and accompanied by practical examples and related resources to support implementation.

3.1 Culturally Responsive Teaching Practices (CRTP)

Culturally responsive pedagogy leverages the cultural knowledge, experiences, and strengths of ethnically diverse students to make learning more relevant, effective, and affirming. This approach involves integrating students' cultural elements into the curriculum.

A survey study [2] involving 315 students in grades 6–12 from diverse racial backgrounds across the U.S. explored their perceptions of CRTP in the classroom. The findings revealed positive associations between CRTP and several key outcomes, including academic performance, ethnic-racial identity development, and students' perceptions of inclusiveness and engagement. These results strongly support the idea that CRTP can enhance students' sense of belonging and inclusivity in educational settings.

Instructors can begin embracing culturally responsive teaching practices (CRTP) by sharing an inclusive syllabus statement that acknowledges and values the diverse cultural backgrounds of students. This sets the tone for a classroom environment that respects and supports all students, regardless of their identities.

Below is an example of an inclusive statement that can be added to a syllabus and shared with students:

```
Welcome to [Course Name]! I am committed to creating a classroom
environment that respects and supports all students, regardless
of their identities. This course values the diverse cultural
backgrounds, experiences, and perspectives that each student
brings. We will strive to make learning relevant, effective, and
affirming for everyone.

I encourage open dialogue and active participation, and I am here
to support you in your academic journey. If you have any concerns
or need accommodations, please feel free to reach out to me.
Together, we can create a learning space where everyone feels
valued and included.
```

Course materials should reflect cultural responsiveness by incorporating texts, examples, and case studies that represent a wide range of cultures and lived experiences. Instructors can enhance daily lessons by using analogies and examples that resonate with students' cultural backgrounds.

For example, when teaching sorting algorithms, an instructor might use elements of Indian cuisine—such as organizing spices by name or heat level—to demonstrate how sorting works in a familiar and culturally relevant context. Similarly, the process of assembling a traditional Mexican dish like *tacos al pastor* can serve as a metaphor for algorithm development, illustrating step-by-step logic in an familiar way. The key to inclusive teaching lies in combining creativity with cultural respect, ensuring that examples resonate with students' lived experiences while maintaining academic rigor.

Another culturally responsive analogy could involve comparing communication protocols: TCP (Transmission Control Protocol) can be likened to the formal, structured interactions typical in Japanese culture, while UDP (User Datagram Protocol) resembles the more casual, informal communication style often associated with American culture.

In summary, CRTP can make our teaching practices not only more inclusive but also more engaging, using well-thought-out examples that students are likely to relate to.

A simple yet effective way to demonstrate cultural awareness is by using names from a variety of cultural backgrounds in examples, rather than relying solely on Anglo-Saxon names. Additionally, allowing students to choose topics for projects or assignments that are meaningful to them and their cultural backgrounds can also be a highly effective culturally responsive teaching practice.

Recognizing and celebrating cultural events and holidays within the classroom is an effective way to create an inclusive environment. The Diversity Calendar [4] is a good resource for learning when these events and holidays occur throughout the academic year. Our university's Office of Diversity and Inclusion also maintains a similar events calendar [10].

Ultimately, as educators, we must be able to communicate with our students in the most effective way possible. Given that Spanish is spoken by approximately 20% of the U.S. population—and that one-third of our students identify as Latino—incorporating Spanish into interactions with Hispanic students is a meaningful step toward inclusive teaching and mentoring practices.

At MSU Denver, the Department of World Languages offers a program called DILS (Directed Independent Language Studies) [8], which enables faculty and staff to participate in conversation sessions in languages other than English, including Spanish. By learning the basics of Spanish communication, instructors can take a significant step toward fostering a more inclusive and culturally responsive educational environment.

3.2 Universal Design for Learning (UDL)

Universal Design for Learning (UDL) is a framework aimed at improving and optimizing teaching and learning for all individuals, based on scientific insights into how humans learn. It emphasizes providing multiple means of engagement, representation, and action/expression to accommodate the diverse needs and preferences of learners.

Humans learn in various ways, often categorized into different learning styles and theories. The most common ones include:

- **Visual Learning**: Learning through seeing. Visual learners benefit from diagrams, charts, videos, and written instructions.
- **Auditory Learning**: Learning through listening. Auditory learners excel with lectures, discussions, and audio materials.
- **Kinesthetic Learning**: Learning through doing. Kinesthetic learners prefer hands-on activities, experiments, and physical movement.
- **Reading/Writing Learning**: Learning through reading and writing. These learners thrive on written texts and note-taking.
- **Social Learning**: Learning through interaction with others. Social learners benefit from group work, discussions, and collaborative activities.
- **Solitary Learning**: Learning independently. Solitary learners prefer self-study and working alone.

A core principle of UDL is to create learning opportunities that support the widest possible range of learners. Experiential learning exemplifies this approach by enabling students to gain knowledge and skills through direct experiences and reflective practice. Experiential learning can take many forms—for example:

- Field trips that benefit visual and social learners,
- Group projects that foster collaboration among social learners,
- Hands-on experiments that are especially effective for kinesthetic learners.

Ultimately, the variability in how we teach appears to be key to effectively implementing UDL.

UDL also addresses the needs of visually impaired students by promoting accessible and inclusive learning environments. Its principles aim to reduce barriers and provide multiple means of representation, engagement, and expression, ensuring that all learners—including those with visual impairments—can access and participate in learning activities. Therefore, instructors must make their course materials accessible to visually impaired students by adopting simple strategies, such as:

- Add alternative text descriptions to all visuals—such as images, charts, and graphs—so screen readers can describe the content to visually impaired students.
- Use accessibility checkers available in popular content creation tools like Microsoft Word and PowerPoint.
- Choose high-contrast colors and simple fonts to enhance readability. Avoid shadowed text and ensure color combinations are accessible for students with color blindness.
- Include captions and subtitles for all audio and video content to ensure that all students can access the information.
- Provide digital copies of presentations that are compatible with screen readers and other assistive technologies.

Our university has a special instructional accessibility group [9] that provides faculty training to support the development of more accessible and inclusive instructional content. The group offers 30-minute virtual workshops on instructional accessibility, a semester-long paid workshop designed to encourage faculty to become accessibility liaisons within their departments, and accessibility checks and course review consultations with personalized feedback.

Aligned with UDL principles, instructors should respond promptly and appropriately to student accommodation requests, which may include needs such as a distraction-free testing environment or extended time for assignments. To support this, many universities offer testing services, providing exam administration in specialized rooms and outside regular class hours for students who require a quieter setting or additional time to complete their exams.

As Sanger [15] explains, UDL extends beyond offering multiple means of representation; it also includes multiple forms of engagement and expression. For instance, giving students choices during class activities or assignments can enhance their motivation and engagement in the learning process. Additionally, UDL encourages allowing students to demonstrate their understanding in various ways. For example, a student with an impairment that prevents them from writing or typing on a computer might instead complete an oral examination to showcase their learning.

3.3 Equitable Access

Equitable access means ensuring that all students have fair and meaningful opportunities to access resources, services, and support—regardless of their socioeconomic background or life circumstances. For computer science students, this includes having a computer with sufficient processing power, memory, and storage, as well as access to essential software tools and a stable internet connection. These elements are critical to their academic and professional success.

Since the early 2010s, as personal laptops have become more affordable and powerful, universities—including MSU Denver—have increasingly adopted bring-your-own-device (BYOD) models. However, access to portable computers can still be out of reach for students facing challenging social and economic circumstances. To ensure our program is truly inclusive, we must provide accessible technology alternatives for students who need them.

A growing number of universities offer loaner laptops to registered students for educational use. At MSU Denver specifically, the standard loan period is two weeks (10 business days), though it can be adjusted to range from a single day to an entire semester. These laptops typically come pre-installed with basic software such as an office suite, a PDF viewer, and major web browsers.

Unfortunately, this solution is not ideal for computer science students. Because the laptops are loaned, students often face limitations on what they can install or configure—restrictions that can hinder coursework requiring custom development environments, specialized tools, or administrative access.

Dedicated lab rooms for computer science majors are typically equipped with desktop computers pre-configured with all the software tools students may need. While these machines offer adequate computing capabilities, they are still general-use systems and are therefore subject to similar restrictions as the loaner laptops—such as limited permissions for installing or configuring additional software when needed. Additionally, because these desktops are stationary and intended to be shared, students cannot take them home to complete assignments or projects outside of class, further limiting their usefulness.

A better alternative for computer science students is the Raspberry Pi—a small, affordable, single-board computer developed by the Raspberry Pi Foundation to promote computer science education and digital skills [13]. In particular, the Raspberry Pi 400 model costs approximately \$140 and comes bundled with a mouse, as well as power and other necessary cables. The computer board is embedded directly into the keyboard, making it highly portable. To use it, students simply need to plug it into a power outlet and connect it to an external monitor—a resource that computer science departments can provide free of charge in their classrooms. Offering Raspberry Pi (or similar) portable computers on loan could be an effective way to promote equitable access to technology for students at a very affordable cost.

In addition to providing affordable hardware solutions, the department can further promote equitable access by expanding the availability of specialized software and development environments through cloud-based platforms. Many major cloud providers already offer free tiers targeting educational use:

- **GitHub Codespaces**: Free for students and educators; provides cloud-based development environments with VS Code.
- **Google Cloud Platform (GCP)**: Offers $300 in credits for new users and free-tier services.
- **Microsoft Azure for Students**: Free credits and access to services like VMs, databases, and AI tools—no credit card required.
- **AWS Educate**: Free access to cloud resources, training, and credits for students and educators.

The cloud-based approach eliminates the need for high-end personal hardware while ensuring consistency across student experiences. Moreover, by partnering with software vendors, computer science departments can offer free or discounted licenses for essential tools relevant to their learning, ensuring that all students have access to the professional-grade software needed to succeed in their coursework and projects.

3.4 Role Modeling

One of the main reasons students leave computer science or related fields is the lack of role models who share their gender, race, or cultural background. While our computer science department is fortunate to have a diverse faculty and staff—including Asian, Latino, and strong female representation—departmental diversity alone may not be enough to foster an inclusive community where all students feel seen and supported.

Faculty must actively create opportunities for students to engage with a broader range of role models. One effective approach is to invite guest speakers who not only present course-relevant content and share their personal experiences in the field, but also serve as inspiring examples of successful professionals from underrepresented groups. Establishing ongoing partnerships with alumni or professionals from these groups—who can return regularly to mentor students or speak in classes—is another powerful way to address the lack of visible role models and build lasting connections that support student success.

Another inclusive teaching practice related to role modeling is highlighting diverse contributions within course content. For example, the contributions of Ada Lovelace in the mid-1800s, who worked alongside with Charles Babbage on the design of the Analytical Engine, an early mechanical general-purpose computer, should be highlighted. Ada wrote what is considered the first algorithm intended to be executed by a machine, making her the first to recognize that computers could go beyond arithmetic to manipulate symbols and ideas.

Another influential woman in computer science is Grace Hopper, who invented the first compiler and played a key role in the development of the COBOL programming language. Remembering those key (and other) influential scientists in our field can help students create role models and foster an inclusive teaching culture in the classroom.

Simply sharing your personal journey—your time as a student, your first job in the field, how you discovered your calling to become an educator, the career decisions you faced, and so on—can be a powerful way to connect with your students. More often than we realize, students see parts of their own experiences reflected in our stories. That sense of connection can be the key to becoming the role model they need to envision their own success.

Being a role model for your students takes time and trust. Talk with them outside the classroom, attend department-organized social events that include students, go to their presentations—whether at undergraduate research events or capstone project showcases—and invite them to stop by your office hours just to chat. The more trust you build in your relationships with students, the more likely they are to see you as a mentor and a professional role model to follow.

3.5 Community Engagement

Creating opportunities for students—especially those just beginning their programs—to feel welcomed and develop a sense of belonging within the academic community is essential to any inclusion strategy. Not long ago, senior leaders at our university proposed community engagement as a way to improve first-year student retention and foster a stronger sense of connection. There is supporting evidence that fostering learning communities promotes social and academic integration, which are critical factors in student persistence and success [5].

In alignment with this vision, our computer science department launched a program called Community Hours, which hosts events early in the morning when no classes are typically scheduled. These events include faculty or student research presentations, advising Q&A sessions, discussions on special topics in computer science, and more. Community Hours is a relatively simple yet effective initiative that fosters a stronger sense of connection among students—making it a compelling example of an inclusive strategy.

Building on this commitment to inclusion, the department has also created informal spaces where students can gather outside of class time. We offer a dedicated student lounge that is available for use at any time, providing a comfortable environment for studying, socializing, or relaxing. More recently, the department's conference room has been made accessible to students through a simple reservation process at the front desk, further expanding opportunities for collaboration and community-building.

Another effective community engagement strategy is supporting student clubs. In our department, we have two active organizations: the ACM (Association for Computing Machinery) and Girls Who Code. These clubs provide spaces where students with shared interests, backgrounds, or goals can connect. This sense of community helps students—especially those from underrepresented groups—feel seen, valued, and included in campus life.

Student clubs often serve as safe spaces where members can share experiences, seek advice, and find support during challenging times—reducing isolation and promoting mental well-being. In particular, Girls Who Code plays a vital role in advancing female inclusion in computer science by encouraging girls to explore coding from an early age, helping to close the gender gap in tech. By empowering girls from diverse backgrounds, clubs like Girls Who Code contribute to a more inclusive and representative tech workforce—driving innovation and positive social impact.

Finally, coding competitions, hackathon events, and conference trips are excellent ways to foster peer engagement and strengthen students' sense of community and belonging—factors that have a positive impact on their academic success.

4 Conclusion

This paper presented five easily applicable inclusive teaching practices that research has shown to significantly enhance students' sense of belonging. These practices contribute to key aspects of academic success, including increased first-year retention, improved engagement and academic performance, and reduced dropout rates.

This is a critical topic that deserves prioritization—especially in STEM programs such as computer science—where equitable access to technology plays a vital role in student achievement. The paper demonstrated that even low-cost digital inclusion initiatives can have a meaningful impact on student outcomes.

Ultimately, inclusion initiatives in education aim to ensure that all students—many of whom are the first in their families to enter academia—feel welcomed and free from discrimination. We strive to foster an environment where cultural diversity is valued, different learning styles are respected, essential resources are accessible, students' contributions are recognized, and a genuine sense of belonging is nurtured throughout their academic journey.

References

[1] Tracie Marcella Addy et al. *What Inclusive Instructors Do: Principles and Practices for Excellence in College Teaching.* Sterling, VA: Stylus Publishing, 2021. ISBN: 9781642671921.

[2] Christy M. Byrd. “Does Culturally Relevant Teaching Work? An Examination From Student Perspectives”. In: *SAGE Open* 6.3 (July 2016), pp. 1–10. DOI: `10.1177/2158244016660744`.

[3] Code.org. *Diversity*. Accessed July 28, 2025. URL: `https://code.org/diversity`.

[4] Diversity Resources. *Diversity Calendar 2025*. Accessed: 2025-05-07. 2025. URL: `https://www.diversityresources.com/diversity-calendar-2025`.

[5] Manal Elobaid et al. “The impact of freshman learning communities on students’ academic performance”. In: *Frontiers in Education* 9 (2024), p. 1465809. DOI: `10.3389/feduc.2024.1465809`. URL: `https://www.frontiersin.org/articles/10.3389/feduc.2024.1465809`.

[6] Cary Funk and Kim Parker. *STEM Jobs See Uneven Progress in Increasing Gender, Racial and Ethnic Diversity*. Tech. rep. Accessed July 28, 2025. Washington, D.C.: Pew Research Center, Apr. 2021. URL: `https://www.pewresearch.org/science/wp-content/uploads/sites/16/2021/03/PS_2021.04.01_diversity-in-STEM_REPORT.pdf`.

[7] Neil Hatfield, Nathanial Brown, and Chad M. Topaz. *How Introductory STEM Courses Weed Out Blacks and Other Underrepresented Students*. Accessed July 28, 2025. Oct. 2022. URL: `https://jbhe.com/2022/10/how-introductory-stem-courses-weed-out-blacks-and-other-underrepresented-students/`.

[8] Metropolitan State University of Denver. *Directed Independent Language Studies (DILS) program*. Accessed July 28, 2025. 2025. URL: `https://www.msudenver.edu/languages/directed-independent-language-studies-program`.

[9] Metropolitan State University of Denver. *Instructional accessibility. Center for Teaching, Learning and Design*. Accessed July 28, 2025. 2025. URL: `https://www.msudenver.edu/teaching-learning-design/instructional-accessibility/`.

[10] Metropolitan State University of Denver. *Office of Diversity and Inclusion Events Calendar*. Accessed July 28, 2025. 2025. URL: `https://msudenver.trumba.com/msu-denver-events-calendars-office-diversity-inclusion`.

[11] Emily K. Miller et al. “Strengthening Education Through Equitable and Inclusive Evidence-Based Teaching Practices: A Scoping Review”. In: *Education Sciences* 15.3 (2025), p. 266. DOI: `10.3390/educsci15030266`.

[12] Ryan Noonan. *Women in STEM: 2017 Update.* Tech. rep. Accessed July 28, 2025. Washington, D.C.: U.S. Department of Commerce, Economics and Statistics Administration, Oct. 2017. URL: `https://www.commerce.gov/sites/default/files/migrated/reports/women-in-stem-2017-update.pdf`.

[13] Raspberry Pi Foundation. *Raspberry Pi – Teach, learn, and make with Raspberry Pi.* Accessed July 28, 2025. URL: `https://www.raspberrypi.org`.

[14] Catherine Riegle-Crumb, Barbara King, and Yasmiyn Irizarry. "Does STEM Stand Out? Examining Racial/Ethnic Gaps in Persistence Across Postsecondary Fields". In: *Educational Researcher* 48.3 (2019), pp. 133–144. DOI: `10.3102/0013189X19831006`. URL: `https://doi.org/10.3102/0013189X19831006`.

[15] Catherine Shea Sanger. "Inclusive Pedagogy and Universal Design Approaches for Diverse Learning Environments". In: *Diversity and Inclusion in Global Higher Education.* Ed. by Catherine Shea Sanger and Nancy W. Gleason. Springer, 2020, pp. 31–71. DOI: `10.1007/978-981-15-1628-3_2`. URL: `https://link.springer.com/chapter/10.1007/978-981-15-1628-3_2`.

[16] Maria Temming. "STEM's Racial, Ethnic and Gender Gaps Are Still Strikingly Large". In: *Science News* (Apr. 2021). Accessed July 28, 2025. URL: `https://www.sciencenews.org/article/science-technology-math-race-ethnicity-gender-diversity-gap`.

Reducing Math Anxiety Through MATLAB: A Case Study in Mechatronics Engineering Technology Education*

Rawan Al-Nsour
Technology Managment & Mechatronics Department
Utah Valley University
Orem, UT 84058
`rawan.nsour@uvu.edu`

Abstract

Mathematics has long posed a challenge for students in Engineering Technology (ET) programs, often leading them to reconsider or abandon their academic goals due to difficulty in passing required math courses. At Utah Valley University (UVU), this issue is particularly evident. As an instructor in the Mechatronics Engineering Technology department, I have observed that while many students struggle in traditional math classes, they perform strongly in core technical courses such as Programmable Logic Controllers (PLC) and Computer Numerical Control (CNC). Interestingly, students consistently perform well in MATLAB courses, despite the software requiring substantial engagement with mathematical concepts and problem-solving skills. To investigate this further, this study surveyed Mechatronics students enrolled in a MATLAB course to evaluate their perceptions of math anxiety, comprehension, and the effectiveness of software-based instruction. The findings indicate that students reported significantly lower levels of anxiety and greater clarity when engaging with math through MATLAB compared to their experiences in traditional math courses. These preliminary results suggest that contextual, technology-enhanced learning environments may

---

*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.

offer a more accessible and effective path for teaching mathematics in engineering technology programs. The study provides early evidence supporting the integration of applied tools like MATLAB into math instruction to improve student outcomes and confidence.

1 Introduction

Math anxiety has been widely recognized as a persistent barrier for students pursuing STEM degrees, particularly in engineering technology fields. Defined by Tobias and Weissbrod [7] as "the panic, helplessness, paralysis, and mental disorganisation that arises among some people when they are required to solve a mathematical problem," math anxiety (MA) can manifest in both emotional and physical symptoms. Recent research highlights the growing demand for STEM professionals in the United States and emphasizes that MA continues to impact students' ability to persist in these fields [8]. The same analysis shows that while theoretical mathematics is essential for engineers and natural scientists, students in applied fields such as engineering technology tend to rely more on practical math applications [8]. This distinction may influence their self-efficacy and decision-making when choosing a major. The research underscores a lack of focused studies on MA within engineering and technology disciplines and suggests that the issue is often overlooked due to the overrepresentation of psychology and general education students in MA research. Similarly, Arevee and Vintere [2] investigate cognitive and emotional factors contributing to math learning difficulties in engineering students. Their study identifies math anxiety as a result of inadequate prior knowledge, poor learning strategies, time constraints, and a lack of real-world relevance. They further associate anxiety with low confidence, personality traits, and societal stereotypes about math. Their work highlights the effectiveness of three approaches to reducing MA: the use of mathematical software, real-life application of concepts, and cooperative learning strategies. These methods are supported by survey and interview data showing improvement in student outcomes when anxiety is addressed through targeted strategies [2]. The integration of computational tools such as MATLAB has emerged as a promising solution to improve math comprehension and reduce anxiety. For the current generation of students who prefer interactive, technology-based, and non-traditional learning methods, MATLAB offers an engaging platform that connects abstract mathematical concepts with visual and practical applications. This alignment with their learning preferences not only increases motivation but also enhances their ability to grasp complex math topics, making these tools a valuable resource for succeeding in challenging math courses. Educators advocate for the use of MATLAB in teaching foundational math concepts, including operations with polynomials, factoring, quadratic equations, and linear systems. This technological approach facilitates visualization and helps students engage with material they may otherwise find abstract or inaccessible [9]. Mayer and Moreno's cognitive theory of multimedia learning (Figure 1) illustrates how verbal and visual inputs are processed separately in sensory memory, organized in working memory, and integrated into long-term memory through active processing [5]. Their model supports the idea that tools like MATLAB, which simultaneously engage students through symbolic (code/text) and visual (graphs/outputs) channels, align well with how students learn most effectively in applied settings.A study conducted on first-year university students found that the use of scientific software like MATLAB enhanced their understanding of mathematical content and boosted confidence. The results suggest that students were more willing to engage with challenging material when aided by technology [4]. Further emphasizing MATLAB's benefits, several studies [6, 1] report its use in visualizing geometric and algebraic problems in engineering courses. The findings indicate that students using MATLAB were able to solve a wider range of tasks more effectively and with greater independence. These studies reinforce the tool's value not only for academic success but also for developing career-relevant skills. However, researchers also caution that over-reliance on such tools may lead to a decline in manual and mental math abilities, which remain important in engineering problem-solving. Despite the usefulness of computational tools, emotional resistance to math remains a key concern. One study [3] examined how engineering technology students with high levels of math anxiety tend to avoid symbolic mathematical representations, often preferring experiential rather than rational problem-solving approaches. Their findings align with the broader consensus that MA influences students' academic pathways and may even deter them from continuing in STEM fields. Collectively, these studies reveal a clear pattern: math anxiety significantly

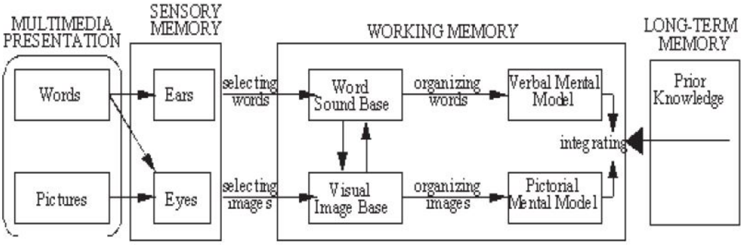


Figure 1: Cognitive theory of multimedia learning, showing dual-channel input and active processing. Adapted from Mayer and Moreno (2003) [5].

affects engineering technology students' academic performance and retention. While technology such as MATLAB offers promising instructional support, it cannot wholly compensate for low math confidence or ineffective learning strategies. Furthermore, the literature reveals a need for deeper, discipline-specific research focused on the intersection of math anxiety, applied learning tools, and student success in engineering technology programs.

2 Methodology

2.1 Participants

The participants in this study were 20 undergraduate students enrolled in MECH 490R MATLAB at Utah Valley University. All students were part of the Mechatronics Engineering Technology program and had previously completed their math prerequisite courses through the university's mathematics department. The sample was purposefully selected to explore math-related experiences and anxiety in the context of applied engineering coursework.

2.2 Survey

Data were collected using a custom-designed survey consisting of six closed-ended questions accompanied by optional comment fields. Each item employed a 5-point Likert scale with the following response options: Definitely Yes , Yes , Neutral, No, and Definitely No. The survey was designed to assess two main domains: first, students' past experiences with traditional math instruction, including anxiety and confidence, It is worth noting that the inclusion of the word "exam" in Question 3 may have influenced students' reported anxiety levels differently than in Question 4, which focused more generally on classroom experience. Second, students' current experiences using MATLAB and their perceptions of its effectiveness in understanding and applying math concepts The optional comment sections allowed participants to elaborate on their responses, providing qualitative context for the quantitative data.

2.3 Procedure

The survey was administered in class to students enrolled in MECH 490R (MATLAB class). Participation was voluntary and anonymous. Completion of the survey constituted informed consent, as outlined in the introductory consent statement. The survey required approximately 5 to 10 minutes to complete. Data collection followed approval from the university's Institutional Review Board (IRB). All responses were compiled and analyzed using descriptive statistics and basic thematic coding for open-ended comments

3 Discussion and Results

The survey results reveal a clear improvement in students' math experience when instruction is supported by MATLAB. Figure 2 illustrates responses to six survey questions, highlighting reduced anxiety and improved comprehension compared to traditional math classes. The results of this case study indicate that MATLAB has a positive effect on engineering technology students' experience with mathematics. While only 25% of students felt confident about passing their previous math courses, just 10% reported feeling anxious about math topics in the MATLAB class. This substantial drop in anxiety suggests that embedding math within a practical, engineering-focused tool like MATLAB reduces the emotional stress typically associated with abstract computation and high-stakes exams. Additionally, 70% of students reported that MATLAB made math concepts easier to understand, and 90% agreed that teaching math prerequisites using MATLAB would be a successful approach. These responses reflect not only reduced anxiety but also improved comprehension. MATLAB appears to shift students' perception of math from a barrier to a functional problem-solving tool. As one student explained, "I liked how MATLAB simplified larger problems and made it easier," highlighting how the tool may reduce cognitive burden by making complex concepts more approachable in applied engineering contexts.

Among all survey responses, Question 6—on students' support for using MATLAB in math instruction—stood out with the highest agreement (90%). This strong result may reflect their positive current experience, though it's also possible that students enrolled in upper-level MATLAB courses tend to be those more comfortable with applied tools.

This observation captures the central outcome of the study: when math is contextualized within students' field of study through applied software like MATLAB, anxiety decreases and understanding improves. As shown in Table 1, students reported notable differences in math confidence and anxiety levels between traditional math instruction and MATLAB-supported learning.

Table 1: Student Responses on Math Anxiety and MATLAB Use (N = 20)

Q#	**Question Summary**	**Agree (%)**	**Mean**	**SD**
Q1	Concerned about passing math classes	25%	2.35	1.13
Q2	Comfortable during math classes	70%	3.70	0.92
Q3	Experienced math exam anxiety	55%	3.35	1.15
Q4	Anxious about math in MATLAB class	10%	2.10	0.85
Q5	MATLAB made math easier to digest	70%	3.75	0.72
Q6	Support for using MATLAB teaching math	90%	3.95	0.60

The bar chart in Figure 2 visualizes the shift in student perceptions across

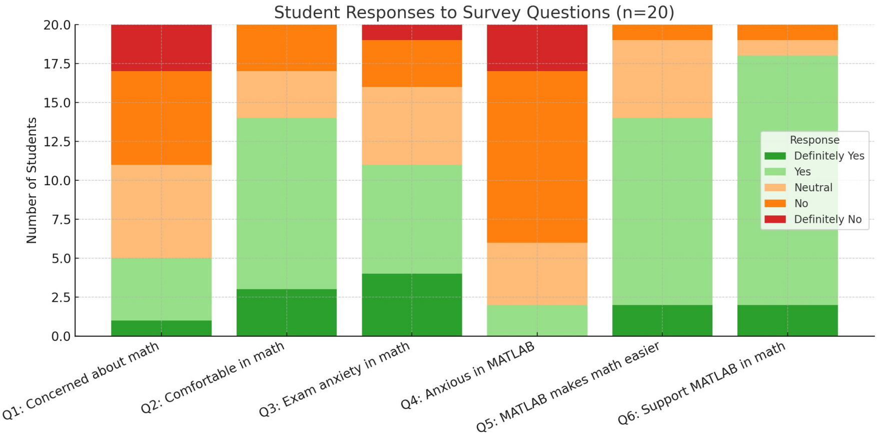


Figure 2: Comparison of student responses to traditional vs. MATLAB-based math learning

six survey questions, contrasting traditional math experiences with MATLAB-supported instruction. For example, while only 25% of students expressed confidence about previous math classes, 90% believed that using MATLAB to teach math would be an effective approach. Similarly, reported anxiety dropped from 55% in traditional classes to just 10% in the MATLAB course. These shifts suggest that context-driven, application-based instruction can significantly reshape students' emotional engagement with mathematics. Beyond reducing anxiety, MATLAB also appears to enhance comprehension. Seventy percent of students agreed that math concepts became easier to understand when delivered through MATLAB, a sentiment supported by qualitative comments such as, "I liked how MATLAB simplified larger problems and made it easier." This reinforces the idea that embedding mathematical learning in hands-on, visual tools not only improves confidence but also deepens understanding and retention. These findings align with research highlighting the benefits of context-driven learning strategies that reduce cognitive barriers in STEM education [8].

4 Recommendations

In light of the findings, it is recommended that engineering technology programs incorporate MATLAB—or similar tools—into the teaching of math prerequisite content. The results suggest that students experience significantly lower anxiety and improved comprehension when mathematical concepts are taught in a software-supported, application-based environment. Embedding math instruction within platforms like MATLAB helps bridge the gap between abstract theory and real-world practice—an obstacle that often discourages STEM learners. To implement this effectively, institutions may consider exploring ways to integrate MATLAB-based instruction earlier in students' academic pathways. This could involve pilot initiatives or interdisciplinary collaborations between math and engineering faculty to develop modules that teach essential math skills through real-world applications. Faculty development remains important; instructors should be supported with training on using MATLAB not just as a technical tool, but as a pedagogical resource that enhances conceptual clarity. These exploratory steps could promote a more applied, confidence-building approach to math education—one that aligns with the learning styles and professional needs of engineering technology students.

5 Limitations and Future Directions

This study offers valuable insights into the benefits of using MATLAB to support math learning in engineering technology. With a sample of 20 students, the research provided focused exploration of student perceptions within a real-world, classroom-based setting. Participation was voluntary, anonymous, and conducted under institutional IRB approval. Despite the limited scope, consistent trends in the data suggest that tools like MATLAB may reduce math-related anxiety and enhance comprehension—highlighting their strong potential in applied STEM education.

Future research should expand the sample size and diversity to include students from multiple courses and institutions. Comparative studies using pre/post-performance data, interviews, or focus groups would provide a deeper understanding of how students engage with applied math tools.

While this study focused on MATLAB, future research could explore comparisons with alternative platforms such as GNU Octave, Mathematica, or Python, particularly given their accessibility and similar computational functionality. As with any instructional tool, there is a risk of over-reliance. While MATLAB supports conceptual understanding, it should be complemented with opportunities for manual and symbolic math practice to ensure well-rounded skill development. Such investigations could help validate and refine technology-enhanced strategies to further improve student outcomes and retention in STEM programs.

6 Conclusion

This study contributes new insights into how technology-enhanced instruction can reshape students' experiences with mathematics in engineering technology programs. Feedback from a MATLAB-integrated course revealed a clear shift in learner attitudes, suggesting that application-driven environments can foster greater confidence and understanding of quantitative material. The visual and practical nature of MATLAB enabled students to engage with mathematical concepts more intuitively, especially those who had previously struggled with abstract instruction. These findings suggest that instructional technologies, when thoughtfully integrated, can serve as both technical platforms and pedagogical tools—enhancing comprehension, confidence, and student retention. As institutions seek to improve STEM engagement and outcomes, adopting such strategies holds strong potential for creating more inclusive and effective learning environments. While these findings are promising, further validation is needed through larger studies and performance-based outcomes.

References

[1] A. Ali and M. Mokhtar. "The effects of MATLAB and simulation in engineering education". In: *Education and Information Technologies* 23.6 (2018), pp. 2805–2818.

[2] I. Arevee and A. Vintere. "Factors Influencing the Development of Mathematics Learning Difficulties in Engineering Students". In: *Mathematics* 11.5 (2023), p. 1089.

[3] Sian L. Beilock and Erin A. Maloney. "More than a feeling: The role of math anxiety in STEM achievement". In: *Current Directions in Psychological Science* 18.6 (2009), pp. 324–328.

[4] Patricia Cretchley et al. "MATLAB in early undergraduate mathematics: An investigation into the effects of scientific software on learning". In: *Mathematics Education Research Journal* 12.3 (2000), pp. 219–233.

[5] Richard E. Mayer and Roxana Moreno. "Nine ways to reduce cognitive load in multimedia learning". In: *Educational Psychologist* 38.1 (2003), pp. 43–52.

[6] M. Simon and A. Simon. "Using MATLAB in visualization of mathematical models in engineering mathematics courses". In: *International Journal of Engineering Pedagogy* 4.4 (2014), pp. 22–28.

[7] Sheila Tobias and Carol Weissbrod. "Anxiety and mathematics: An update". In: *Harvard Educational Review* 50.1 (1980), pp. 63–70.

[8] E. C. Wilkinson. “Holistic Review: Math Anxiety and the STEM Profession”. In: *Journal of Engineering Technology* 40.2 (2023).

[9] Jie Wu and Ronghuai Huang. “Use of MATLAB in teaching algebra”. In: *Proceedings of the International Conference on Information Technology in Education (ITiE 2011)* (2011), pp. 365–368.

Cyber Students' Perceptions of Specifications Grading*

Mathew J. Heath Van Horn and Christopher Warner
College of Business, Security and Intelligence
Embry-Riddle Aeronautical University
Prescott, AZ 863001
`{heathvam, warnerc9}@erau.edu`

Abstract

Specifications grading is increasingly becoming popular in STEM courses. Students are evaluated on a pass/fail basis, and instructors provide detailed specifications (specs) for what constitutes a passing grade. When applied correctly, specs grading upholds high academic standards (B or better) while aligning more closely with student learning outcomes. Students are motivated and experience less stress. Anecdotal evidence suggests that these outcomes are true, but the STEM fields are only now beginning to study student perceptions of this unique grading scheme effectively. Complicating matters is that cyber students are unique in the STEM fields. To measure cyber student satisfaction with specs grading we discovered an instrument developed through use on chemistry students. Therefore, we tested the instrument on a small scale to assess the feasibility of designing a broader study of cyber students.

1 Introduction

The traditional A–F, 100-point grading scales are insufficient in measuring student knowledge and ability [7], [10]. For instance, when a student earns a B in a course with four learning objectives, does this grade indicate the student

---

*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.

partially mastered all four objectives, fully mastered three learning objectives but has no knowledge of the remaining one, or some combination thereof [7]? Cybersecurity employers are increasingly seeking college graduates who can perform tasks, not just pass standardized tests [3], [4]. Adjusting classroom student assessments to reflect employer requirements enhances student preparedness for internships and employment.

Specifications grading is an attempt to restore objective measures of student performance. It serves as an extension of mastery learning and competency-based grading [7]. Specs grading is becoming more popular in STEM courses [10] and consists of a pass/fail evaluation of student performance on every type of assessment [7], [10]. Implementing specs grading in the classroom is beyond the scope of this research. However, when specs grading is implemented in courses, students experience such benefits as a diminished urge to cheat, reduced stress, increased motivation, and less conflict with instructors about grades [7], [10]. The researchers have been using specs grading in their courses for about two years and anecdotally attest to the benefits of the system, but wanted to objectively measure student benefits.

The Perceptions of Grading Schemes (PGS) instrument was developed to objectively measure students' perceptions of the benefits of using specifications grading. The 15-item instrument measures five factors: anxiety, expectations, learning outcomes, feedback, and motivation to learn. It was developed over a period of two years by hundreds of participating chemistry students [10]. We could not find an instance where the instrument was tested on cybersecurity students.

Cybersecurity professionals differ from other STEM professionals in that they have unique personality characteristics. Generally, they need heightened cognitive challenges and will take significant risks—including illegal ones—to achieve their goals. Furthermore, they have few social skills and rarely demonstrate team-building behaviors. Finally, they may be introverted and poor communicators [1]. Considering the unique characteristics of cybersecurity students, we were uncertain whether the PGS instrument would be effective in measuring their perceptions with the implementation of specs grading.

The purpose of our research was to explore the use of the PGS instrument with cybersecurity students using a small sample size. The results would give us direction in designing a larger-scale effort involving hundreds of participants from multiple cybersecurity schools. Ultimately, the results would provide insight for instructors adopting the use of specs grading.

2 Methods

2.1 Participants

The study involved 37 undergraduate cybersecurity students enrolled in *Securing Computer Networks* or *Ethical Hacker – Pen Testing* courses at Embry-Riddle Aeronautical University during the Spring 2025 semester. Participants were voluntarily recruited. Research was carried out under Protocol Number 25-094, which was reviewed and determined to be exempt by the Embry-Riddle Aeronautical University Institutional Review Board.

2.2 Instrument

Specifications grading was implemented following Nilson's framework [7], which employs pass/fail assessments based on explicit criteria to enhance grading transparency and student motivation. Students' perceptions were measured using the Perceptions of Grading Schemes (PGS) instrument, a 15-item survey validated in STEM contexts [10]. The PGS includes 15 positively worded items across five factors, with three items per factor:

- **Reflect Student Learning Outcomes** (e.g., "My grades on assignments represent what I understand," loading = 0.794)
- **Promotes Motivation to Learn** (e.g., "I am motivated to learn as much course material as possible," 0.795)
- **Raises Anxiety** (e.g., "I am anxious about my final letter grade," 0.717)
- **Useful Feedback** (e.g., "The written feedback I receive is helpful," 0.752)
- **Clear Expectations**(e.g., "I understand expectations for success in the course," 0.844)

Items are rated on a 5-point Likert scale (1 = Strongly Disagree, 5 = Strongly Agree), with higher scores indicating positive perceptions. The PGS's five-factor structure was validated with standardized loadings ranging from 0.684 to 0.844 in chemistry courses [10], supporting its adaptation to cybersecurity education.

2.3 Data Analysis

This pilot study employed exploratory analyses to assess the feasibility of the PGS instrument in a cybersecurity context. Internal consistency for each of the five PGS factors was evaluated using Cronbach's alpha. Values of 0.70 or higher were considered acceptable [8], as alpha is widely regarded as the most common measure of scale reliability [2], [5].

Due to the small sample size ($n = 37$), confirmatory factor analysis (CFA) and omega coefficients were not conducted, as larger samples ($n > 100$) are typically required for these procedures [6]. Pearson correlations, along with corresponding p-values, were used to explore interrelationships among the five dimensions.

To examine students' perceptions of grading schemes, factor scores were computed by averaging the three items within each PGS dimension. These scores were then standardized to a scale ranging from -1 to $+1$, where 0 indicates a neutral position. Positive values reflect stronger alignment with specifications grading, while negative values suggest alignment with traditional grading practices.

To help contextualize the results from the cybersecurity sample, a descriptive comparison was made with published PGS findings from two chemistry lab courses reported by Yik et al. [10]. While formal inferential tests were not conducted across groups, mean standardized factor scores from the present study are examined alongside those chemistry findings to explore potential similarities and differences. Results of this comparison are presented in a separate section below. Finally, all analyses were conducted in R [9] using the *psych* and *stats* packages.

3 Results

3.1 Reliability Analysis

Before interpreting students' perceptions of grading schemes, it is important to evaluate the reliability of the instrument used to measure those perceptions. In this study, internal consistency reliability was assessed for each of the five Perceptions of Grading Schemes (PGS) factors: **Reflect Student Learning Outcomes, Promotes Motivation to Learn, Raises Anxiety, Useful Feedback, and Clear Expectations.**

Internal consistency refers to the extent to which items within a given factor measure the same underlying construct. A widely used metric for this purpose is **Cronbach's alpha (α)** which ranges from 0 to 1. Higher values indicate greater consistency among items. While interpretations may vary by field, alpha values of 0.70 or higher are commonly considered acceptable for research use [8]. The statistic was originally introduced by Cronbach to assess the reliability of psychological scales [2].

All five PGS factors demonstrated strong internal consistency, with alpha values ranging from 0.766 to 0.871. These results suggest that students responded consistently to the three items associated with each factor, support-

Factor	Cronbach's Alpha
Reflect Student Learning Outcomes	0.847
Promotes Motivation to Learn	0.766
Raises Anxiety	0.871
Useful Feedback	0.795
Clear Expectations	0.869

Table 1: Cronbach's alpha for each PGS factor

ing the internal coherence of the scale in this new context—an undergraduate cybersecurity course.

In particular, the **Raises Anxiety** and **Clear Expectations** factors showed the highest reliability ($\alpha = 0.871$ and $\alpha = 0.869$, respectively), indicating that these dimensions were especially well captured by the items. The **Promotes Motivation to Learn** factor had the lowest alpha (0.766), but still exceeded the recommended threshold.

Given the exploratory nature of this pilot study and the relatively small sample size, these reliability estimates are encouraging. They suggest that the PGS instrument retains strong psychometric properties even outside the STEM fields in which it was originally developed [10]. Future work should assess test–retest reliability and conduct confirmatory factor analysis (CFA) with larger samples to further establish the instrument's validity and stability.

3.2 Correlations

Pearson correlations among PGS factors, with significance levels, are presented below:

	Outcome	Motivation	Anxiety	Feedback	Expectations
Outcome	1.000	0.484**	-0.452**	0.342*	0.557***
Motivation		1.000	0.029	0.426**	0.587***
Anxiety			1.000	0.094	-0.088
Feedback				1.000	0.672***
Expectations					1.000

Note: *p <.05, **p<.01, ***p<.001 (two-tailed)

Table 2: Pearson correlations among PGS factors

Key to Factor Abbreviations:

- **Outcome:** Reflects Student Learning Outcomes

- **Motivation:** Promotes Motivation to Learn
- **Anxiety:** Raises Anxiety
- **Feedback:** Instructor's Feedback was Useful
- **Expectations:** Course Expectations were Clear

Strong correlations were observed between **Useful Feedback** and **Clear Expectations** (r = 0.672), as well as **Promotes Motivation to Learn** and **Course Expectations were Clear** (r = 0.587). A moderate negative correlation existed between **Raises Anxiety** and **Reflect Student Learning Outcomes** (r = -0.452).

3.3 Perceptions of Grading Schemes

To examine students' perceptions of specifications grading, we calculated factor scores for each of the five PGS dimensions-Reflect Student Learning Outcomes, Promotes Motivation to Learn, Raises Anxiety, Useful Feedback, and Clear Expectations—standardized to range between -1 and +1. These scores reflect how strongly students aligned each factor with either traditional grading (values < 0) or specifications grading (values > 0).

Figure 1 displays the mean score and 95% confidence interval (CI) for each factor. The horizontal axis is anchored at zero, representing a neutral stance—perceiving the factor as equally representative of traditional and specifications grading. Values to the left of zero indicate alignment with traditional grading practices, while values to the right suggest perceptions favoring specifications grading.

Among all five factors, **Clear Expectations** received the highest mean score (M = 0.57, 95% CI [0.42, 0.72]), indicating that students strongly associated this dimension—clarity of expectations, grading transparency, and overall course structure—with specifications grading. **Promotes Motivation to Learn** (M = 0.25, CI [0.10, 0.40]) and **Useful Feedback** (M = 0.20, CI [0.07, 0.34]) also showed statistically significant positive associations with specifications grading, as their confidence intervals do not include zero.

In contrast, **Raises Anxiety** received a slightly negative mean score (M = -0.05, CI [-0.27, 0.18]), though the CI includes zero, suggesting no statistically significant difference in students' perceived anxiety levels between grading schemes. This may indicate mixed experiences, where some students found specifications grading less stressful due to clear expectations and opportunities for revision, while others possibly perceived it as stressful due to the "all-or-nothing" pass criteria.

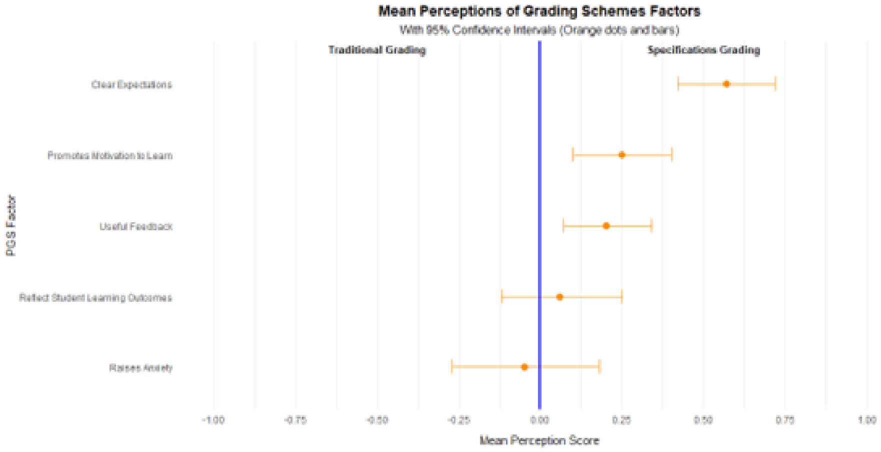


Figure 1: Mean Perceptions of Grading Schemes Factors with 95% Confidence Intervals

The **Reflect Student Learning Outcomes** factor yielded a small positive mean (M = 0.06, CI [-0.12, 0.25]), also with a CI that includes zero, implying neutral perceptions—students viewed both grading schemes as similarly reflective of their actual learning outcomes.

These results suggest that students perceived specifications grading as particularly effective in clarifying expectations and fostering motivation, while still maintaining a neutral to slightly favorable perception regarding its impact on learning outcomes and anxiety. Notably, the relatively wide confidence intervals, especially for **Raises Anxiety** and **Reflect Student Learning Outcomes**, reflect variability in student experiences and reinforce the need for continued refinement of specifications grading practices.

It is worth noting that some confidence intervals, particularly for **Raises Anxiety** and **Reflect Student Learning Outcomes**, are relatively wide. This likely reflects the modest sample size (n = 37), which increases uncertainty in the estimates.

Overall, students appeared to associate specifications grading most strongly with greater **Clear Expectations, Promotes Motivation to Learn,** and **Useful Feedback**. However, neutral or mixed perceptions on **Reflect Student Learning Outcomes** and **Raises Anxiety** suggest that these dimensions may depend on individual preferences or implementation. The variability in scores highlights the importance of ongoing refinement and responsiveness when using specifications grading in practice.

3.4 Comparison with Chemistry Students

To contextualize these findings, perceptions from the current cybersecurity sample (n = 37) were compared with those reported by Yik et al-Figure 5 [10] in two chemistry lab courses. Students in General Chemistry I (GC1Lab) reported near-zero mean standardized scores for Reflect Student Learning Outcomes (0.01) and Useful Feedback (0.00), while those in General Chemistry II (GC2Lab) reported slightly negative values for the same factors (-0.01 and -0.01, respectively). In contrast, cybersecurity students in the present study reported modestly positive means of 0.06 and 0.20 for those factors.

Factor	Cybersecurity Mean (95% CI)	GC1Lab Mean	GC2Lab Mean
Reflect Student Learning Outcomes	0.06 (-0.12, 0.25)	0.01	-0.01
Promotes Motivation to Learn	-0.25 (0.20,0.40)	-0.07	-0.11
Raises Anxiety	-0.05 (-0.27, 0.18)	-0.26	-0.34
Useful Feedback	0.20 (0.07, 0.34)	0.00	-0.01
Clear Expectations	0.57 (0.42, 0.72)	0.18	0.36

Table 3: Comparison with Chemistry Students

Clear Expectations was the most positively rated factor across all three groups, with the highest mean observed among cybersecurity students (M = 0.57, 95% CI [0.42, 0.72]), followed by GC2Lab (0.36) and GC1Lab (0.18). Promotes Motivation to Learn was rated more favorably by the cybersecurity sample (M = 0.25) compared to both chemistry groups (-0.07 in GC1Lab; -0.11 in GC2Lab). Raises Anxiety was consistently negative across all samples, indicating lower stress associated with specifications grading, but less pronounced in the current study (M = -0.05) than in GC1Lab (-0.26) and GC2Lab (-0.34).

These descriptive comparisons, limited by the small cybersecurity sample and unavailable confidence intervals for chemistry data, suggest that the PGS instrument captures similar patterns across STEM disciplines, with Clear Expectations and reduced Raises Anxiety as common strengths. In the cybersecurity sample, only Reflect Student Learning Outcomes and Raises Anxiety had confidence intervals including zero, indicating neutral perceptions, while other factors significantly favored specifications grading. Because PGS items—including the positively worded Raises Anxiety items—were identical across studies [10], differences likely reflect contextual factors, such as cybersecurity's applied focus on practical deliverables (e.g., coding, simulations) versus chemistry's experimental lab context, as noted in the Limitations section. The more favorable ratings among cybersecurity students—particularly for Clear Expectations, Motivation, and Feedback—support the PGS's cross-disciplinary applicability and highlight the need for larger, multi-institution studies to explore contextual influences.

4 Limitations

This pilot study's small sample size (n = 37) severely restricts generalizability to broader cybersecurity or STEM populations and constrains statistical power, leading to wide confidence intervals (e.g., Raises Anxiety CI [-0.54, 0.36]) that limit estimate precision. Confirmatory factor analysis (CFA), which evaluates the PGS's hypothesized five-factor structure, was not feasible due to insufficient sample size, as CFA typically requires n > 100 for robust results [6]. Similarly, McDonald's omega coefficients, which provide a more comprehensive reliability estimate than Cronbach's alpha by accounting for factor loadings, were not computed, as they also require larger samples for stability [6]. The absence of demographic data (e.g., gender, prior cybersecurity experience) precluded subgroup analyses, limiting insights into differential perceptions. The sample, drawn from multiple cybersecurity courses at Embry-Riddle Aeronautical University, a specialized institution, may not represent diverse educational settings.

Despite these limitations, the PGS's strong reliability (Cronbach's alpha = 0.766-0.871) supports its feasibility in cybersecurity education, warranting further validation with larger, multi-institution samples.

5 Conclusion

The findings suggest that the small-scale measure of cybersecurity students aligned with the results of the large-scale chemistry students. Therefore, a larger research design should be undertaken to solidify the early findings and resolve any inconsistencies concerning the cyber students. The sample size is too small for any broad inference about cyber students' perception of the use of specifications grading.

6 Disclosure

The authors used Grok 3 and CHAT GPT, AI tools developed to assist in drafting and revising portions of the manuscript, including refining text for clarity and ensuring style compliance. All research design, data collection, analyses, and interpretations were conducted by the authors, with AI serving solely as a writing and editing aid. The authors take full responsibility for the content and integrity of the final manuscript.

References

[1] Michael Bachmann. "The risk propensity and rationality of computer hackers". In: *International Journal of Cyber Criminology* 4.1/2 (2010), p. 643.

[2] Lee J Cronbach. "Coefficient alpha and the internal structure of tests". In: *psychometrika* 16.3 (1951), pp. 297–334.

[3] Cambrie Eckert. *Just In: U.S. Desperately Needs Cyber Talent, Congress Says*. `https://www.nationaldefensemagazine.org/articles/2023/6/26/us-desperately-needs-cyber-talent-congress-says`.

[4] DoD Cyber Exchange. *Announcements*. `https://public.cyber.mil/wid/dod8140/documents-library/`.

[5] Andy Field. *Discovering statistics using IBM SPSS statistics*. Sage publications limited, 2024.

[6] Joseph F Hair et al. *Multivariate data analysis (Vol. 6)*. Upper Saddle River, NJ: Pearson Prentice Hall, 2006.

[7] Linda B. Nilson. *Specifications grading: Restoring rigor, motivating students, and saving faculty time*. Stylus Publishing, 2015.

[8] Jum C Nunnally. *Psychometric theory. 2*. 1978.

[9] R Development Core Team et al. *The R project for statistical computing*. 2010.

[10] Brandon J Yik et al. "Students' perceptions of specifications grading: Development and evaluation of the perceptions of grading schemes (pgs) instrument". In: *Journal of Chemical Education* 101.9 (2024), pp. 3723–3738.

Prompt-Driven Cybersecurity Education: Exploring LLM-Based Simulation of Security Operations Center Activities*

Basil Hamdan
Department of Information Systems & Technology
Utah Valley University
Orem, UT 84058
basil.hamdan@uvu.edu

Abstract

This paper examines the pedagogical potential of Large Language Models (LLMs), such as ChatGPT, to simulate key workflows within Security Operations Centers (SOCs) in the context of cybersecurity education. We compare two instructional approaches: a traditional lab-based model that employs virtual machines and open-source tools, and a lightweight, LLM-driven framework that generates synthetic telemetry and emulates analyst decision-making. While the traditional model provides high-fidelity interaction with real-world systems, it entails substantial setup and maintenance costs. In contrast, the LLM-based model abstracts away infrastructure complexity, enabling scalable engagement with SOC concepts through a focus on analytical reasoning. To evaluate instructional effectiveness, we present use cases involving malware infection and detection across the stages of initial access, persistence, and intrusion detection. Our findings suggest that LLM-driven simulation offers a compelling complement to hands-on labs, particularly for cultivating conceptual understanding and broadening access to experiential cybersecurity training.

---

[*] 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

The growing demand for skilled cybersecurity professionals has placed increasing pressure on academic institutions to prepare students with practical, job-ready competencies. Traditional approaches to SOC–focused courses often require the deployment of complex virtualized environments that include firewalls, segmented networks, attacker and victim virtual machines, Security Information and Event Management (SIEM) platforms, endpoint detection and response (EDR) tools, log aggregation systems, and simulated adversarial activity. Although these environments offer realistic and immersive learning experiences, they are resource intensive and present scalability challenges for cybersecurity programs, faculty, and students alike. Recent advances in generative artificial intelligence, particularly the emergence of LLMs such as ChatGPT, have introduced new opportunities to simulate cybersecurity operations. This paper explores the extent to which LLMs can serve as instructional surrogates for SOC infrastructure by generating synthetic data, modeling attack scenarios, and emulating analyst workflows, while preserving the pedagogical value of experiential learning.

2 Background

2.1 Established Approaches to Cybersecurity Instruction

Hands-on learning environments have become a hallmark of effective cybersecurity education. Platforms such as NetLab+, the NICE Cyber Range, and institution-specific lab deployments allow students to engage in realistic exercises that involve configuring secure systems, executing controlled attacks, and analyzing digital evidence. These environments foster active learning and expose students to industry-relevant tools and procedures. However, the technical complexity and maintenance demands of these virtual lab infrastructures pose significant challenges, especially as programs scale and the need for flexible, remote access increases. For many institutions, particularly those with limited staffing or IT support, the maintenance of such environments can become prohibitively resource intensive.

2.2 Generative AI in Computing Education: Promise and Limitations in Cybersecurity Contexts

The emergence of generative AI has created new opportunities for enhancing computing education. LLMs such as ChatGPT have demonstrated value in supporting learners through code generation, explanation, debugging assistance, and adaptive instructional feedback [1, 3]. These tools can lower barriers to entry for novice programmers and offer scalable support for self-directed learning.

However, their integration into cybersecurity education is still in its early stages. Preliminary efforts, including [2], [4] and [5], illustrate attempts to embed LLMs into security curricula. These systems commonly use retrieval-augmented generation or structured domain ontologies to deliver quizzes, generate policy templates, or assist with static analysis.

While promising, these tools do not simulate the complex, multi-step workflows found in SOC environments. Critical activities such as log correlation, attack chain reconstruction, and threat hunting require nuanced reasoning that extends beyond simple information retrieval or code synthesis. As such, the pedagogical potential of LLMs in cybersecurity—particularly as substitutes for hands-on SOC simulation—remains underdeveloped and warrants deeper investigation.

2.3 A Comparative Model for LLM-Supported SOC Education

This paper introduces a novel instructional design approach that explores the use of LLMs as simulation engines to emulate SOC analyst workflows. We compare two pedagogical formats: a traditional SOC course built on virtualized lab environments and a parallel version using LLM-driven simulations. The comparison focuses on three representative cybersecurity scenarios—initial access, persistence, and intrusion detection—each grounded in realistic adversary behavior. While exploratory in nature, this work aims to show how LLMs can function not merely as content delivery tools but as scalable alternatives to resource-intensive lab infrastructure. It offers a potential pathway for broadening access to experiential cybersecurity learning, particularly in settings where technical resources are limited and traditional lab deployments are impractical.

3 Methodology

To explore the instructional viability of LLMs in a cybersecurity operations course, we examine two parallel instructional tracks focused on SOC workflows. The first version follows a traditional hands-on lab model, requiring students to work within a VirtualBox-based virtual environment that includes pfSense for network segmentation, Suricata for intrusion detection, NXLog for Windows event forwarding, and Kibana for log visualization. Students engage directly with network and host telemetry, capturing live traffic, configuring log pipelines, and performing threat detection tasks using real tools and data.

In contrast, the second version employs a prompt-driven, LLM-simulated approach. Instead of interacting with physical or virtual infrastructure, students use ChatGPT to emulate SOC analyst activities. While the underlying mechanisms differ, both tracks maintain alignment in learning objectives, assessment criteria, and analytic depth. For example, while a student in the traditional track might investigate an alert using Kibana, their counterpart in the simulated track prompts the LLM for equivalent log artifacts and detection logic, achieving parallel reasoning workflows.

3.1 Environment Architecture and Infrastructure

Both instructional tracks are based on a common network topology that emulates a small enterprise environment segmented across four zones:

Network Segment	**Purpose**
WAN (10.10.10.0/24)	Simulates untrusted external traffic
LAN (192.168.10.0/24)	Hosts internal workstations and clients
DMZ (192.168.20.0/24)	Provides public-facing services vulnerable to exploitation
CSOC (192.168.30.0/24)	Aggregates logs and supports SOC analyst workflows

Table 1: Network Segments and Their Functional Roles

3.1.1 Traditional Virtual Lab Deployment

In the traditional track, this network is instantiated within a VirtualBox-based environment. Each zone is implemented via virtual machines connected through pfSense, which handles routing, NAT, and firewalling. Students interact directly with these systems to generate, capture, and analyze real telemetry.

Host	**IP Address**	**Functionality**
pfSense	N/A	Routes traffic, enforces segmentation, and applies firewall and NAT rules
Kali Linux (Attacker)	10.10.10.10	Simulates external adversary behavior including exploit delivery and scanning
Windows Workstation	192.168.10.10	Endpoint with Sysmon and NXLog for host-based monitoring
Ubuntu Client	192.168.10.11	Secondary LAN host
Ubuntu Server (DMZ)	192.168.20.10	Hosts Apache and Squid services targeted for web and egress attacks
CSOC Box	192.168.30.10	Aggregates logs and provides SIEM functionality through Splunk and Kibana

Table 2: Simulated Environment: Hosts, IPs, and Assigned Roles

This setup allows for reproducible simulation of multi-stage attacks, enabling students to build detection logic through direct interaction with packet captures, endpoint logs, and SIEM dashboards.

3.1.2 LLM-Simulated Environment

The same environment is abstracted in the LLM track via structured, scenario-based prompts. Students interact with ChatGPT, which assumes the role of an embedded SOC analyst assistant. It responds with synthesized outputs that mirror telemetry generated by tools in the traditional lab.

Prompt Example:

> *You are a cybersecurity analyst assistant embedded within a simulated Security Operations Center (SOC) environment. You have access to synthetic telemetry data, including logs, alerts, and packet captures, representative of real enterprise infrastructure.*
>
> *Your tasks include:*
>
> - *Analyzing logs from tools such as Windows Event Viewer, Sysmon, Suricata, Apache, and DNS*
> - *Simulating attacker behavior and triggering realistic alerts*
> - *Investigating security incidents through iterative, evidence-based analysis*
> - *Explaining detection logic, potential impact, and recommended remediation steps*
>
> *The network includes:*
>
> - *WAN: 10.10.10.0/24 — Kali Linux (10.10.10.10)*
> - *LAN: 192.168.10.0/24 — Windows (192.168.10.10), Ubuntu (192.168.10.11)*
> - *DMZ: 192.168.20.0/24 — Ubuntu with multiple servers (192.168.20.10)*
> - *CSOC: 192.168.30.0/24 — Ubuntu with SIEM Platforms (192.168.30.10)*
>
> *Respond in the tone and format of a professional SOC analyst. Wait for me to provide the specific incident or threat to investigate.*

This LLM-driven approach ensures students receive comparable telemetry artifacts and analyst reasoning paths, but without requiring resource-intensive infrastructure.

3.2 Tools and Telemetry Sources

Students in both tracks engage with a comprehensive suite of cybersecurity tools that support the detection, investigation, and remediation phases of SOC workflows. In the traditional lab, these tools are installed and configured on the virtual machines. In the simulated track, the LLM emulates the behaviors and outputs of these tools through prompt-response interactions.

3.2.1 Virtual Lab (Traditional Track)

The traditional track gives students direct access to a suite of open-source tools commonly used in SOC environments. These tools are configured within the virtual machines and operate on real network and host telemetry:

- **Network Traffic Analysis**: *Wireshark*, *tcpdump* — for packet capture, protocol inspection, and traffic filtering.
- **Endpoint Monitoring**: *Sysmon*, *NXLog*, *Windows Event Viewer* — to monitor process creation, registry access, and network connections.
- **Web and Proxy Services**: *Apache*, *Squid* — to support exploitation, outbound filtering, and egress detection scenarios.
- **Intrusion Detection Systems**: *Suricata*, *Snort* — to detect known exploits, port scans, and abnormal traffic patterns.
- **Log Forwarding and Host Querying**: *Filebeat*, *OSQuery* — to enable structured host queries and log shipping to the CSOC segment.
- **SIEM Platforms**: *Splunk (Free Tier)*, *Kibana* — for log aggregation, dashboarding, and alert correlation.

This toolchain supports realistic SOC workflows and gives students hands-on experience navigating and correlating diverse telemetry sources.

3.2.2 LLM-Simulated Track

In the LLM-simulated track, the same suite of tools is abstracted through structured prompts. Students interact with ChatGPT, which emulates the output of each tool based on the underlying scenario. The LLM simulates log entries, alerts, queries, and dashboard insights, providing a parallel analytical experience without requiring actual infrastructure.

Prompt Example:

> *You are a cybersecurity analyst assistant embedded within a simulated Security Operations Center (SOC) environment. You have access to synthetic telemetry data that mimics the output of real*

> *tools such as Windows Event Viewer, Sysmon, Server logs, Suricata, Kibana, and Splunk.*
>
> *When I provide an incident (e.g., suspicious network scan, PowerShell execution, data exfiltration attempt), return simulated log entries, alerts, or query results consistent with those tools. Use realistic formatting and IP addressing based on the following network segments: WAN (10.10.10.0/24), LAN (192.168.10.0/24), DMZ (192.168.20.0/24), CSOC (192.168.30.0/24).*
>
> *Respond in the tone of a professional SOC analyst. Await further instructions or incident descriptions.*

This approach enables students to develop core SOC analysis skills through guided dialogue and synthetic data, supporting scalability while preserving fidelity to professional workflows.

4 Simulated Malware Execution and Detection

To evaluate the capacity of LLM-based simulations to mirror host and network-level telemetry, we developed a series of use cases derived from common exploitation: (1) Initial Access via trojanized binaries, (2) Persistence via scheduled tasks, and (3) Alerting via Suricata IDS. In each case, students in the simulated track are presented with a prompt describing the incident and are tasked with querying the LLM for logs, reasoning through the sequence of actions, and proposing detection and mitigation strategies.

4.1 Use Case 1: Initial Access via Trojanized Executable

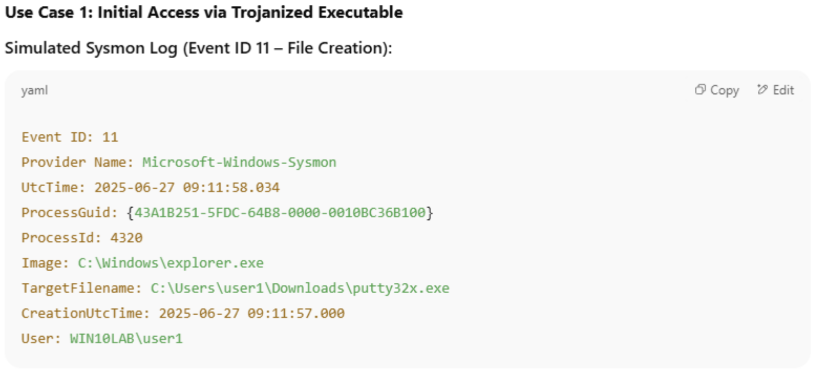
Scenario: An external attacker on the WAN (`10.10.10.10`) uses `msfvenom` to create a back-doored version of PuTTY, named `putty32x.exe`, embedding a `windows/meterpreter/reverse_tcp` payload configured to connect back on port 4444. The malicious file is hosted at `http://10.10.10.10/putty32x.exe`. A user on the Windows 10 workstation (`192.168.10.10`) downloads and executes the file, resulting in initial access.

Prompt A – Sysmon File Creation (Event ID 11):

> Simulate the output of Sysmon logs (Event ID 11) for the creation of `putty32x.exe` in the `Downloads` directory. Include the timestamp of file creation, process ID, image path, parent-child relationships, and command-line arguments where applicable.

Prompt B – Windows Process Creation (Event ID 4688):

Simulate the output of Windows Event Viewer logs (Event ID 4688) for the execution of `putty32x.exe` on a Windows 10 workstation. Provide timestamp, user context, process ID, parent process name, and the full command-line used during execution.

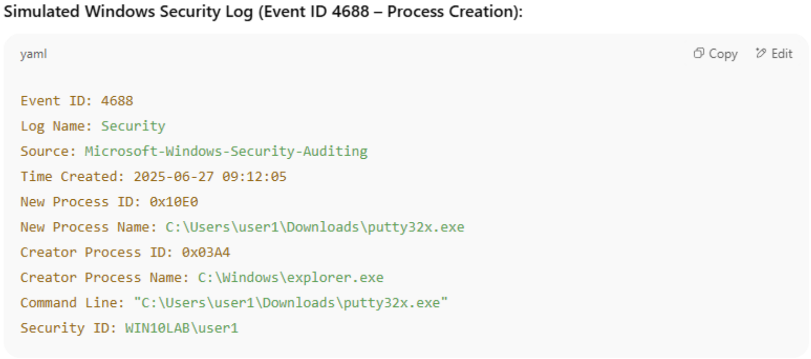


Use Case 1: Initial Access via Trojanized Executable

Simulated Sysmon Log (Event ID 11 – File Creation):

```yaml
Event ID: 11
Provider Name: Microsoft-Windows-Sysmon
UtcTime: 2025-06-27 09:11:58.034
ProcessGuid: {43A1B251-5FDC-64B8-0000-0010BC36B100}
ProcessId: 4320
Image: C:\Windows\explorer.exe
TargetFilename: C:\Users\user1\Downloads\putty32x.exe
CreationUtcTime: 2025-06-27 09:11:57.000
User: WIN10LAB\user1
```

Figure 1: Simulated Sysmon Log for Initial Access (File Creation)



Simulated Windows Security Log (Event ID 4688 – Process Creation):

```yaml
Event ID: 4688
Log Name: Security
Source: Microsoft-Windows-Security-Auditing
Time Created: 2025-06-27 09:12:05
New Process ID: 0x10E0
New Process Name: C:\Users\user1\Downloads\putty32x.exe
Creator Process ID: 0x03A4
Creator Process Name: C:\Windows\explorer.exe
Command Line: "C:\Users\user1\Downloads\putty32x.exe"
Security ID: WIN10LAB\user1
```

Figure 2: Simulated Event Viewer Log for Initial Access (Process Creation)

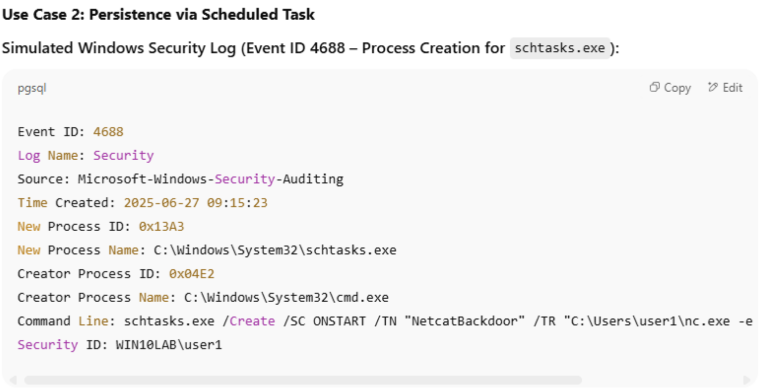
4.2 Use Case 2: Persistence via Scheduled Task

Scenario: Following successful compromise, the attacker uploads `nc.exe` to the Windows 10 workstation and creates a scheduled task to execute Netcat at startup. The task runs a reverse shell command: `nc.exe -e cmd.exe`

`10.10.10.10 3333` to establish persistence.

Prompt:

> Simulate the output of Windows Event Viewer logs (Event ID 4688) for a workstation where `nc.exe` has been scheduled to run on reboot using `schtasks.exe`. Include process creation, command-line arguments, parent-child process relationships, and timestamps.



Use Case 2: Persistence via Scheduled Task

Simulated Windows Security Log (Event ID 4688 – Process Creation for `schtasks.exe`):

```
Event ID: 4688
Log Name: Security
Source: Microsoft-Windows-Security-Auditing
Time Created: 2025-06-27 09:15:23
New Process ID: 0x13A3
New Process Name: C:\Windows\System32\schtasks.exe
Creator Process ID: 0x04E2
Creator Process Name: C:\Windows\System32\cmd.exe
Command Line: schtasks.exe /Create /SC ONSTART /TN "NetcatBackdoor" /TR "C:\Users\user1\nc.exe -e
Security ID: WIN10LAB\user1
```

Figure 3: Simulated Event Viewer Log for Persistence via `nc.exe`

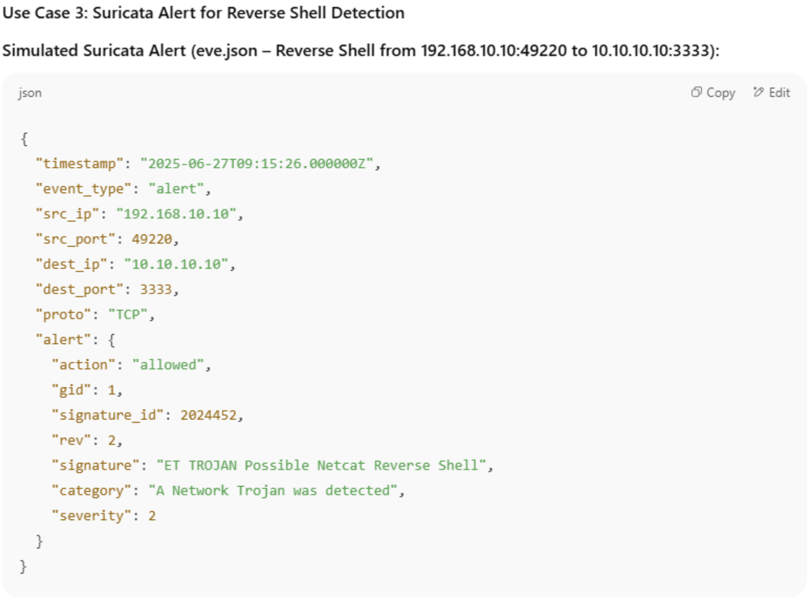
4.3 Use Case 3: Suricata Alert for Reverse Shell Detection

Scenario: Suricata, deployed on pfSense, detects reverse shell traffic from the compromised Windows workstation (`192.168.10.10`) to the attacker (`10.10.10.10`) over port 3333. The alert is generated based on a signature detecting suspicious outbound TCP connections matching reverse shell behavior.

Prompt:

> Simulate a Suricata alert log that detects a Netcat reverse shell from `192.168.10.10:49220` to `10.10.10.10:3333`. Include timestamp, source/destination IP and port, alert signature ID, category, severity, and signature message.

These simulations enable learners to trace attacker behavior across multiple phases of the cyber kill chain using artifacts that mirror real-world telemetry.



Use Case 3: Suricata Alert for Reverse Shell Detection

Simulated Suricata Alert (eve.json – Reverse Shell from 192.168.10.10:49220 to 10.10.10.10:3333):

```json
{
  "timestamp": "2025-06-27T09:15:26.000000Z",
  "event_type": "alert",
  "src_ip": "192.168.10.10",
  "src_port": 49220,
  "dest_ip": "10.10.10.10",
  "dest_port": 3333,
  "proto": "TCP",
  "alert": {
    "action": "allowed",
    "gid": 1,
    "signature_id": 2024452,
    "rev": 2,
    "signature": "ET TROJAN Possible Netcat Reverse Shell",
    "category": "A Network Trojan was detected",
    "severity": 2
  }
}
```

Figure 4: Simulated Suricata Alert for Reverse Shell (Port 3333)

5 Discussion and Conclusion

This paper explores the pedagogical potential of ChatGPT to simulate core workflows within a SOC setting. Specifically, it compares two instructional approaches for SOC education: a traditional lab-based model leveraging virtual machines and a lightweight, LLM-driven simulation framework. The traditional model provides high-fidelity engagement with real-world tools and telemetry. However, its infrastructure requirements and maintenance overhead limit scalability, particularly for institutions with constrained resources.

By contrast, LLM-driven simulation offers a low-barrier, prompt-based alternative for engaging students in SOC analytical tasks. Rather than executing live malware or configuring monitoring infrastructure, learners reason through synthetic telemetry artifacts that emulate real-world system and network behavior. These may include, for instance, process and file creation events, and IDS alerts, representative examples drawn from a much broader set of possible log sources and telemetry formats. By enabling the emulation of events spanning multiple stages of the cyber kill chain, this approach facilitates structured reasoning across diverse detection contexts while removing the logistical overhead associated with traditional lab environments

Although this abstraction omits critical hands-on experience with system configuration and tool operation, it enables scalable access to conceptual SOC training and facilitates rapid iteration through varied threat scenarios. As such, LLM-based simulation serves as a pedagogically valuable complement to traditional lab environments—particularly in the early stages of SOC learning when analytical reasoning and threat comprehension are the focus.

Future work will implement this simulation model within a formal SOC course, enabling empirical evaluation of learning outcomes. Additional development will focus on refining prompt engineering, integrating MITRE ATT&CK mappings, and incorporating adversarial emulation techniques. Ultimately, this work aims to assess whether LLMs can function not only as instructional aides, but also as credible platforms for experiential cybersecurity education.

References

[1] Enkelejda Kasneci et al. "ChatGPT for good? On opportunities and challenges of large language models for education". In: *Learning and Individual Differences* 103 (2023), p. 102274. DOI: `https://doi.org/10.1016/j.lindif.2023.102274`.

[2] Muhammad Mudassar Yamin et al. "Applications of LLMs for Generating Cyber Security Exercise Scenarios". In: *IEEE Access* 12 (2024), pp. 143806–143822. DOI: `10.1109/ACCESS.2024.3468914`.

[3] N. Raihan et al. "Large Language Models in Computer Science Education: A Systematic Literature Review". In: *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1*. ACM, 2025, pp. 938–944.

[4] Yang Xin et al. "Machine Learning and Deep Learning Methods for Cybersecurity". In: *IEEE Access* 6 (2018), pp. 35365–35381. DOI: `10.1109/ACCESS.2018.2836950`.

[5] Chengshuai Zhao et al. "CyberBOT: Towards Reliable Cybersecurity Education via Ontology-Grounded Retrieval Augmented Generation". In: *arXiv preprint arXiv:2504.00389* (2025).

Generative AI and its Use in Defensive Cybersecurity Courses*

Mohamed Lotfy Ph.D.
Utah Valley University
Orem, UT 84058
MohamedL@uvu.edu
Ed Lindoo Ph.D.
Regis University
Denver, CO 80221
elindoo@regis.edu

Abstract

The expanding accessibility of Generative AI (GenAI) chat-bots is fundamentally transforming academic engagement, particularly within specialized technical disciplines like cybersecurity. This investigation critically examines the practical integration and pedagogical implications of GenAI tools in defensive and countermeasures courses, specifically analyzing student utilization for generating network intrusion detection system (NIDS) rules and simulating network traffic. We conduct a comparative evaluation of outputs from leading GenAI platforms, revealing inherent inconsistencies and incompleteness that mandate advanced student expertise for correction. Our findings underscore the vital role of foundational knowledge in discerning AI-generated inaccuracies. This research provides essential insights for educators, guiding the responsible incorporation of GenAI into cybersecurity curricula to foster critical thinking and equip the next generation of professionals for an AI-augmented cyber landscape.

---

*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

The advent of Generative Artificial Intelligence (GenAI), characterized by its sophisticated natural language processing capabilities and rapid content generation across diverse modalities including text, code, and other data forms, is fundamentally reshaping contemporary existence. Concurrently, GenAI poses significant implications for the evolving offensive cyber threat landscape, facilitating the proliferation of misinformation, sophisticated cyber-attacks, advanced social engineering tactics, targeted phishing campaigns, and expedited malware code development [3, 4, 9, 11, 12]. Conversely, GenAI is concurrently being leveraged within defensive cybersecurity paradigms, contributing to enhanced vulnerability investigation, streamlined incident response protocols, advanced digital forensics, the strategic deployment of honeypots, and a general improvement in code security practices [3, 9, 11, 12].

GenAI chat-bots like OpenAI's ChatGPT, Google's Gemini and Microsoft's Copilot are profoundly influencing contemporary student approaches to academic assignments and reports. Before GenAI, students predominantly relied on conventional search engines for information retrieval, including the discovery of examples, solutions, and code fragments pertinent to their coursework. However, the rapid advancement and widespread availability of GenAI chat-bots is reshaping numerous facets of everyday existence, including the methods by which we educate and absorb knowledge. Evidence indicates that students are increasingly using GenAI to summarize articles, generate code and assist in completing writing reports [1, 6]. Consequently, a comprehensive understanding of the responsible integration of GenAI within academic environments, particularly concerning its influence on students' acquisition of specialized knowledge and skills, is imperative and warrants rigorous investigation.

The promising capacity of GenAI to improve the defensive security of complex systems and networks carries significant implications for the future landscape of the IT and cybersecurity professions, especially given the continuous rise in demand for proficient cybersecurity professionals. As the use of GenAI in educational settings becomes more widespread, students, teachers, and educational institutions face the challenge of utilizing these technologies in a responsible manner. GenAI chat-bots such as OpenAI ChatGPT have ignited debates among educators and professionals regarding their capacity to encourage academic dishonesty and disseminate misinformation. At the forefront of educators' concerns regarding GenAI in education is academic dishonesty. Furthermore, an overreliance on GenAI tools may diminish essential human interaction within the learning process and impede the comprehensive mastery of domain-specific content.

In this paper we present findings of an empirical study examining GenAI chat-bots performance on homework assignments designed for defensive and countermeasures courses. The primary objective of this investigation is to assess GenAI ability to help students generate Snort rules for network intrusion detection systems (NIDS) and successful completion of the homework assignments. Recognizing the pedagogical implications and potential for misuse of such tools, the study also examines the patterns of student utilization and challenges encountered during these assignments.

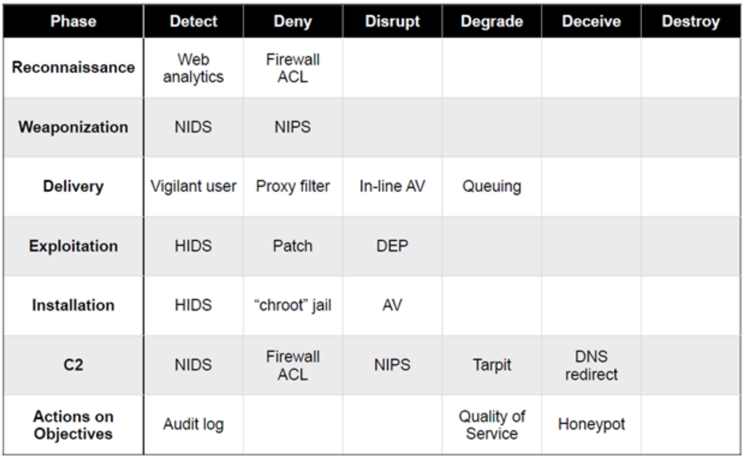
The subsequent sections of this paper introduce the foundational structure of defensive security courses, outline relevant frameworks, and detail the application of NIDS. Further discussion delves into the practical integration of GenAI within cybersecurity defensive curricula and its broader implications for the evolving roles of IT and cybersecurity professionals.

2 Defensive Security

Effective defense against sophisticated cyber-attacks and intrusions necessitates that cybersecurity analysts possess a comprehensive understanding of adversarial patterns, behaviors, tactics, techniques, and procedures (TTPs) [2, 5]. To facilitate this critical understanding, established models such as the Cyber Kill Chain (CKC) and the MITRE ATT&CK framework systematically decompose intrusion attacks into sequential stages. These frameworks empower analysts to discern the intricacies of advanced persistent threats (APTs) and strategically align defensive capabilities to effectively interrupt and neutralize the attack chain [5, 8]. Complementing these analytical frameworks, defensive courses of action are derived from the Department of Defense (DoD) Information Operations (IO) 6D doctrine, encompassing 'detect,' 'deny,' 'disrupt,' 'degrade,' 'deceive,' and 'destroy.' This doctrine provides cybersecurity teams with a structured approach for implementing countermeasures across the various stages of the CKC [5], as illustrated in Figure 1. Furthermore, these 6D courses of action are readily adaptable and can be mapped effectively to the MITRE ATT&CK framework, offering versatile application in defensive strategy formulation.

Network intrusion detection systems (NIDS), play a critical role in identifying and responding to malicious activities as they occur within network infrastructures. These systems operate through strategically placed sensors typically embedded in firewalls, routers, or deployed as standalone or virtual hosts which continuously monitor network traffic for signs of unauthorized access or anomalous behavior. Upon detecting a potential threat, NIDS components can collect forensic data, logging relevant events, issuing real-time alerts, and transmitting this information to a centralized management console for further analysis.

Functionally, NIDS are integral to the early phases of cybersecurity threat detection. Specifically, they align with the initial "Detection" phase of the 6D



Phase	Detect	Deny	Disrupt	Degrade	Deceive	Destroy
Reconnaissance	Web analytics	Firewall ACL				
Weaponization	NIDS	NIPS				
Delivery	Vigilant user	Proxy filter	In-line AV	Queuing		
Exploitation	HIDS	Patch	DEP			
Installation	HIDS	"chroot" jail	AV			
C2	NIDS	Firewall ACL	NIPS	Tarpit	DNS redirect	
Actions on Objectives	Audit log			Quality of Service	Honeypot	

Figure 1: Cyber Kill Chain to 6D course of Action Matrix from [5]

cyber defense model, targeting the identification of weaponization and command and control (C2) activities. Within the context of established threat modeling frameworks such as the Cyber Kill Chain (CKC) and the MITRE ATT&CK framework, NIDS are instrumental in recognizing key stages including initial access, execution, and C2 operations. Their proactive monitoring capabilities enable security teams to detect and potentially neutralize threats before they escalate into more damaging phases of the attack lifecycle.

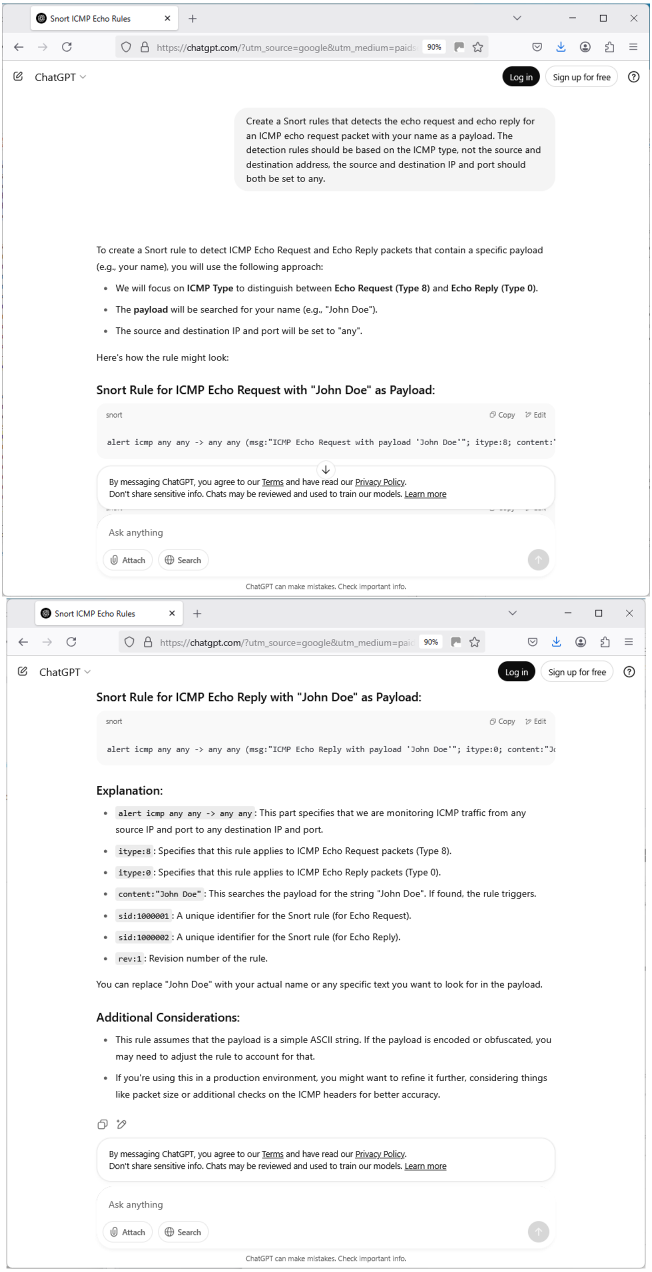
To ensure that graduates are well-equipped for today's rapidly evolving cybersecurity landscape, at Utah Valley University undergraduate and graduate network defense and countermeasures courses introduce current defensive tools and practices to secure computing resources, systems, services, data, and network services. The hands-on assignments in the courses are constructed to allow the student to use current defensive cybersecurity practices and tools. The student uses the CKC model and the MITRE ATT&CK framework knowledge to apply the 6D doctrine within different contexts [7]. The course assignments introduce current defensive cybersecurity practices. For example, the NIDS assignments require the student to install a Security Onion (SO) NIDS VM and write rules to detect ICMP and TCP packets and detect different file types flowing between the hosts on a sub-net. As part of this applied training, each student completes a Network Intrusion Detection System (NIDS) assignment individually, demonstrating the ability to configure detection rules and validate that alerts are triggered appropriately on the Security Onion (SO) manager when corresponding traffic is identified. This exercise reinforces students' understanding of intrusion detection concepts and provides practical experience in monitoring, logging, and analyzing network-based threats in a controlled environment [7].

3 The Use of GenAI to Generate NIDS Rules

To assess how students might leverage ChatGPT, Gemini, and Copilot GenAI chat-bots to generate NIDS Snort rules for detecting ICMP traffic, we posed an identical question taken directly from the NIDS assignment to each chatbot. We didn't create an account or login into the Chat-pot, we just used the website of each chat-bot to ask the following "Create a Snort rules that detects the echo request and echo reply for an ICMP echo request packet with your name as a payload. The detection rules should be based on the ICMP type, not the source and destination address, the source and destination IP and port should both be set to any." Figure 2 shows the OpenAI ChatGPT produced ICMP Snort rules. Figure 3 shows the Google Gemini produced ICMP Snort rules, and Figure 4 shows the Microsoft Copilot produced ICMP Snort rules.

The three GenAI chat-bots provided the two Snort rules with an explanation of each element and options of the rules. All three GenAI generated rules used Snort rule ID (sid) 1000001 for the first rule and sid 1000002 for the second rule. ChatGPT and Copilot rules didn't specify a classification of the rules, thus the alert color of the alerts will be red indicating a high alert. Only Gemini added a "classtype:misc-activity" classification which will display the alerts in orange. ChatGPT was the only GenAI chat-pot that alerted us to change the content string to the actual name. If the student uses the generated rules without changing the sids, the rules will not be added to the set of rules due to sid conflict if there are already rules with the same sids and the rules will never be triggered. In addition, if they don't change the content string to their name, the rules will not be triggered as well. So, if the student doesn't check the GenAI generated Snort rules and changes the options to meet the assignment requirements, the rules might cause the Snort sensors to fail due to sid conflicts or will never get triggered due to the incorrect content string. It is important to note that Gemini and ChatGPT alerted the user that they can make mistakes, so the user need to check (Gemini "double-check") the info.

To probe the robustness and adaptability of the Generative AI (GenAI) chatbots, we presented each of the three platforms with an identical challenge: "Using hping3, craft ICMP packet that contains your name as a data payload." This prompt was designed to assess their ability to generate precise and actionable commands for a specific, non-trivial network task. The one-liner question objective was to see if GenAI will generate an "hping3" command to create an ICMP packet with a name in its payload which would be detected by the ICMP



Create a Snort rules that detects the echo request and echo reply for an ICMP echo request packet with your name as a payload. The detection rules should be based on the ICMP type, not the source and destination address, the source and destination IP and port should both be set to any.

To create a Snort rule to detect ICMP Echo Request and Echo Reply packets that contain a specific payload (e.g., your name), you will use the following approach:

- We will focus on **ICMP Type** to distinguish between **Echo Request (Type 8)** and **Echo Reply (Type 0)**.
- The **payload** will be searched for your name (e.g., "John Doe").
- The source and destination IP and port will be set to "any".

Here's how the rule might look:

Snort Rule for ICMP Echo Request with "John Doe" as Payload:

```
alert icmp any any -> any any (msg:"ICMP Echo Request with payload 'John Doe'"; itype:8; content:'
```

By messaging ChatGPT, you agree to our Terms and have read our Privacy Policy.
Don't share sensitive info. Chats may be reviewed and used to train our models. Learn more

Ask anything
Attach
Search
ChatGPT can make mistakes. Check important info.
Snort ICMP Echo Rules
https://chatgpt.com/?utm_source=google&utm_medium=paic
90%
ChatGPT
Log in
Sign up for free

Snort Rule for ICMP Echo Reply with "John Doe" as Payload:

```
alert icmp any any -> any any (msg:"ICMP Echo Reply with payload 'John Doe'"; itype:0; content:"J
```

Explanation:

- `alert icmp any any -> any any` : This part specifies that we are monitoring ICMP traffic from any source IP and port to any destination IP and port.
- `itype:8` : Specifies that this rule applies to ICMP Echo Request packets (Type 8).
- `itype:0` : Specifies that this rule applies to ICMP Echo Reply packets (Type 0).
- `content:"John Doe"` : This searches the payload for the string "John Doe". If found, the rule triggers.
- `sid:1000001` : A unique identifier for the Snort rule (for Echo Request).
- `sid:1000002` : A unique identifier for the Snort rule (for Echo Reply).
- `rev:1` : Revision number of the rule.

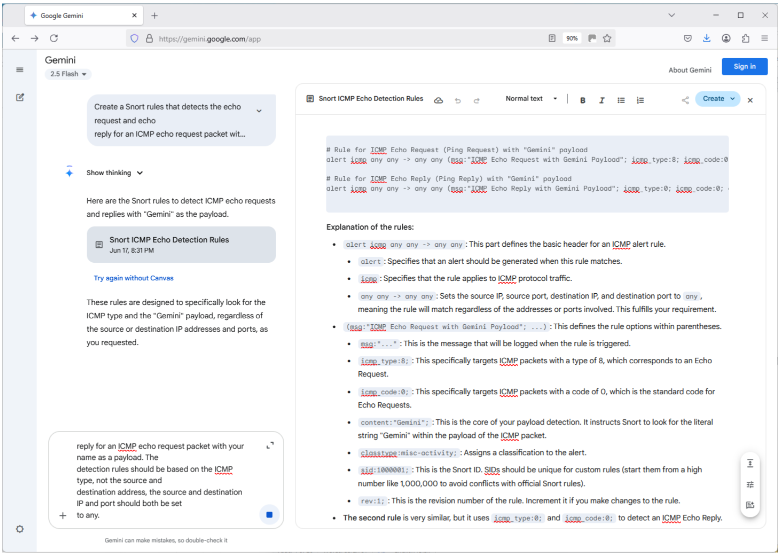
You can replace "John Doe" with your actual name or any specific text you want to look for in the payload.

Additional Considerations:

- This rule assumes that the payload is a simple ASCII string. If the payload is encoded or obfuscated, you may need to adjust the rule to account for that.
- If you're using this in a production environment, you might want to refine it further, considering things like packet size or additional checks on the ICMP headers for better accuracy.

By messaging ChatGPT, you agree to our Terms and have read our Privacy Policy.
Don't share sensitive info. Chats may be reviewed and used to train our models. Learn more

Figure 2: OpenAI ChatGPT ICMP Produced Snort Rule



Create a Snort rules that detects the echo request and echo
reply for an ICMP echo request packet wit...

Show thinking

Here are the Snort rules to detect ICMP echo requests and replies with "Gemini" as the payload.

Snort ICMP Echo Detection Rules
Jun 17, 8:31 PM

These rules are designed to specifically look for the ICMP type and the "Gemini" payload, regardless of the source or destination IP addresses and ports, as you requested.

reply for an ICMP echo request packet with your
name as a payload. The
detection rules should be based on the ICMP
type, not the source and
destination address, the source and destination
IP and port should both be set
to any.

Gemini can make mistakes, so double-check it

Snort ICMP Echo Detection Rules

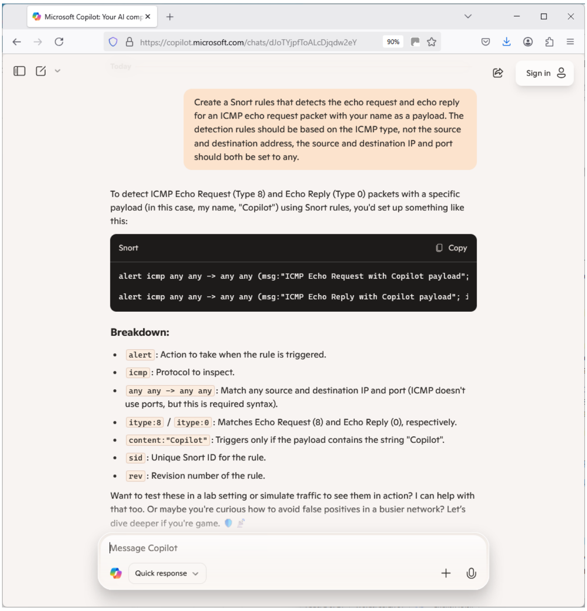
```
# Rule for ICMP Echo Request (Ping Request) with "Gemini" payload
alert icmp any any -> any any (msg:"ICMP Echo Request with Gemini Payload"; icmp_type:8; icmp_code:0

# Rule for ICMP Echo Reply (Ping Reply) with "Gemini" payload
alert icmp any any -> any any (msg:"ICMP Echo Reply with Gemini Payload"; icmp_type:0; icmp_code:0;
```

Explanation of the rules:

- `alert icmp any any -> any any` : This part defines the basic header for an ICMP alert rule.
  - `alert` : Specifies that an alert should be generated when this rule matches.
  - `icmp` : Specifies that the rule applies to ICMP protocol traffic.
  - `any any -> any any` : Sets the source IP, source port, destination IP, and destination port to `any`, meaning the rule will match regardless of the addresses or ports involved. This fulfills your requirement.
- `(msg:"ICMP Echo Request with Gemini Payload"; ...)` : This defines the rule options within parentheses.
  - `msg:"..."` : This is the message that will be logged when the rule is triggered.
  - `icmp_type:8;` : This specifically targets ICMP packets with a type of 8, which corresponds to an Echo Request.
  - `icmp_code:0;` : This specifically targets ICMP packets with a code of 0, which is the standard code for Echo Requests.
  - `content:"Gemini";` : This is the core of your payload detection. It instructs Snort to look for the literal string "Gemini" within the payload of the ICMP packet.
  - `classtype:misc-activity;` : Assigns a classification to the alert.
  - `sid:1000001;` : This is the Snort ID. SIDs should be unique for custom rules (start them from a high number like 1,000,000 to avoid conflicts with official Snort rules).
  - `rev:1;` : This is the revision number of the rule. Increment it if you make changes to the rule.
- **The second rule** is very similar, but it uses `icmp_type:0;` and `icmp_code:0;` to detect an ICMP Echo Reply.

Figure 3: Google Gemini ICMP Produced Snort Rule


Microsoft Copilot: Your AI comp
https://copilot.microsoft.com/chats/dJoTYjpfToALcDjqdw2eY
90%
Today
Sign in
Copy
Message Copilot
Quick response

Create a Snort rules that detects the echo request and echo reply for an ICMP echo request packet with your name as a payload. The detection rules should be based on the ICMP type, not the source and destination address, the source and destination IP and port should both be set to any.

To detect ICMP Echo Request (Type 8) and Echo Reply (Type 0) packets with a specific payload (in this case, my name, "Copilot") using Snort rules, you'd set up something like this:

Snort

```
alert icmp any any -> any any (msg:"ICMP Echo Request with Copilot payload";

alert icmp any any -> any any (msg:"ICMP Echo Reply with Copilot payload"; i
```

Breakdown:

- `alert` : Action to take when the rule is triggered.
- `icmp` : Protocol to inspect.
- `any any -> any any` : Match any source and destination IP and port (ICMP doesn't use ports, but this is required syntax).
- `itype:8` / `itype:0` : Matches Echo Request (8) and Echo Reply (0), respectively.
- `content:"Copilot"` : Triggers only if the payload contains the string "Copilot".
- `sid` : Unique Snort ID for the rule.
- `rev` : Revision number of the rule.

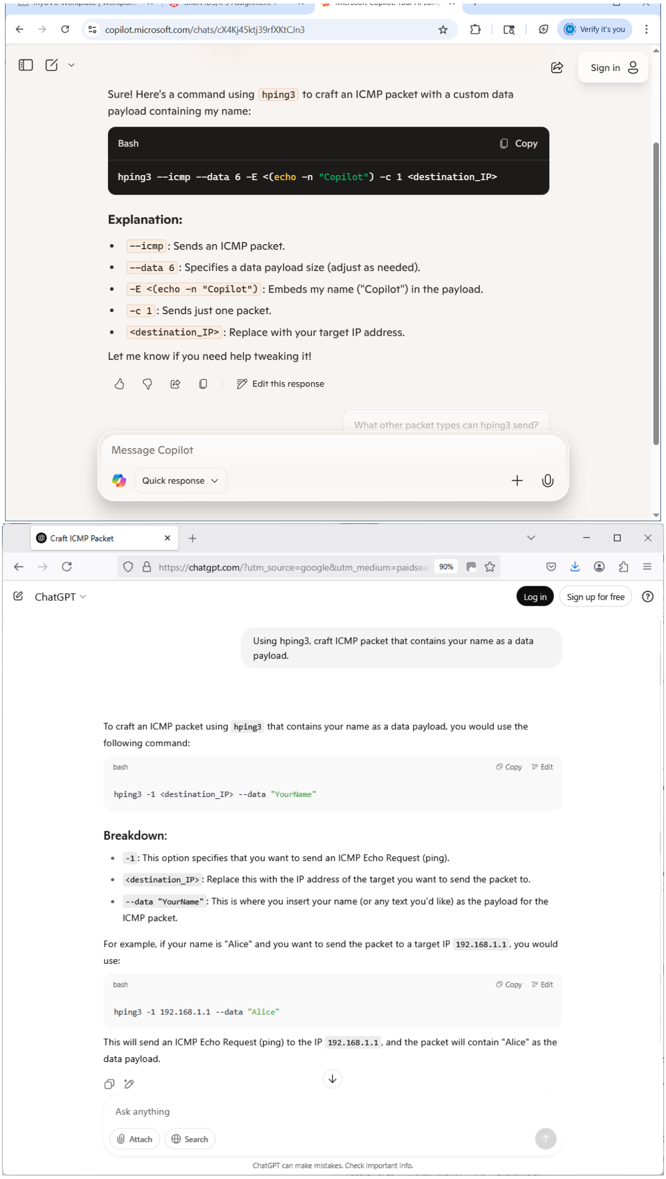
Want to test these in a lab setting or simulate traffic to see them in action? I can help with that too. Or maybe you're curious how to avoid false positives in a busier network? Let's dive deeper if you're game. 🛡️

Figure 4: Microsoft Copilot ICMP Produced Snort Rule

echo request and echo response Snort rules. Analysis of the GenAI chatbot responses revealed notable divergence, with each platform producing a distinct solution to the "hping3" query. Figure 5 illustrates the outputs from ChatGPT and Copilot, while a snippet from Google Gemini's response is also presented. A critical observation was that the "hping3" commands generated by both ChatGPT and Copilot omitted the prerequisite for privileged execution (e.g., sudo). This omission could potentially mislead students, resulting in execution errors within a Linux shell environment. In contrast, Google Gemini's GenAI response correctly included the "sudo" prefix and additionally provided an illustrative example demonstrating file generation for payload inclusion Figure 6.

It became obvious that by using any GenAI chat-bot with just a little effort on the student part, the student could easily write simple Snort rules that allow the NIDS to detect ICMP traffic. To see if GenAI chat-bots can generate advanced ICMP rules, we asked the following: "Create a Snort rules that detects traceroute commands. The detection rules should be based on the ICMP type, not the source and destination address, the source and destination IP and port should both be set to any." ChatGPT and Copilot provided two Snort rules one detecting an ICMP packet with "itype:8" (detecting an echo request) while the other rule detecting an ICMP packet with "itype:11" (detecting a ICMP packet time-to-live not (TTL) exceeded). ChatGPT added a TTL not equal to zero (ttl:!0) to ensure it is a traceroute traffic since traceroute packets start with TTL equals 1 or more. Figure 7 shows the OpenAI ChatGPT produced ICMP traceroute Snort rule.

Regarding the detection of traceroute traffic, the Gemini GenAI chat-bot distinguished itself by providing three distinct Snort rules, in contrast to the two offered by other platforms. The initial two ICMP rules generated by Gemini paralleled those from Copilot and ChatGPT, targeting ICMP packets with "itype:8" (Echo Request) and "itype:11" (Time Exceeded). Crucially, Gemini's third rule was designed to detect ICMP packets with "itype:3" and "icode:3" (Destination Unreachable), which are indicative of a UDP-based traceroute successfully reaching its destination. Furthermore, the Gemini response included a valuable section discussing potential false positives that could arise from detecting "itype:8" (due to standard ping traffic) and "itype:11" packets, acknowledging their broader usage beyond traceroute operations. In addition, Gemini talked about how to make the rules more complete by using the "flowbits" option. The Gemini response reminded the user to place the rules in "local.rules" file. All the sids in the rules from the three GenAI chat-bots started from 1000001, thus if the student forgets to change the sids, the sensors will fail and the rules will not be triggered. Figure 8 shows a snippet from the Google Gemini response.



Sure! Here's a command using `hping3` to craft an ICMP packet with a custom data payload containing my name:

```bash
hping3 --icmp --data 6 -E <(echo -n "Copilot") -c 1 <destination_IP>
```

Explanation:

- `--icmp` : Sends an ICMP packet.
- `--data 6` : Specifies a data payload size (adjust as needed).
- `-E <(echo -n "Copilot")` : Embeds my name ("Copilot") in the payload.
- `-c 1` : Sends just one packet.
- `<destination_IP>` : Replace with your target IP address.

Let me know if you need help tweaking it!

Edit this response
What other packet types can hping3 send?
Message Copilot
Quick response
Craft ICMP Packet
https://chatgpt.com/?utm_source=google&utm_medium=paidsea
90%
ChatGPT
Log in
Sign up for free

Using hping3, craft ICMP packet that contains your name as a data payload.

To craft an ICMP packet using `hping3` that contains your name as a data payload, you would use the following command:

```bash
hping3 -1 <destination_IP> --data "YourName"
```

Breakdown:

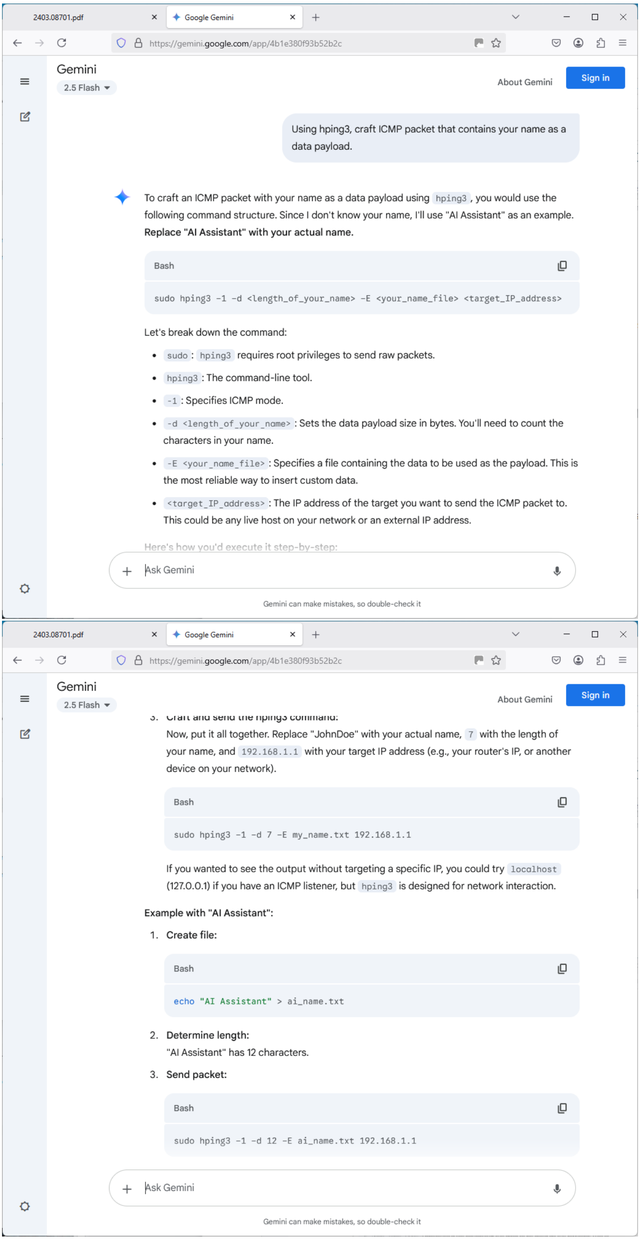
- `-1` : This option specifies that you want to send an ICMP Echo Request (ping).
- `<destination_IP>` : Replace this with the IP address of the target you want to send the packet to.
- `--data "YourName"` : This is where you insert your name (or any text you'd like) as the payload for the ICMP packet.

For example, if your name is "Alice" and you want to send the packet to a target IP `192.168.1.1`, you would use:

```bash
hping3 -1 192.168.1.1 --data "Alice"
```

This will send an ICMP Echo Request (ping) to the IP `192.168.1.1`, and the packet will contain "Alice" as the data payload.

Figure 5: Microsoft Copilot and OpenAI ChatGPT hping3 Commands


2403.08701.pdf
Google Gemini
https://gemini.google.com/app/4b1e380f93b52b2c
Gemini
2.5 Flash
About Gemini
Sign in
Ask Gemini
Gemini can make mistakes, so double-check it

Using hping3, craft ICMP packet that contains your name as a data payload.

To craft an ICMP packet with your name as a data payload using `hping3`, you would use the following command structure. Since I don't know your name, I'll use "AI Assistant" as an example. **Replace "AI Assistant" with your actual name.**

```bash
sudo hping3 -1 -d <length_of_your_name> -E <your_name_file> <target_IP_address>
```

Let's break down the command:

- `sudo`: `hping3` requires root privileges to send raw packets.
- `hping3`: The command-line tool.
- `-1`: Specifies ICMP mode.
- `-d <length_of_your_name>`: Sets the data payload size in bytes. You'll need to count the characters in your name.
- `-E <your_name_file>`: Specifies a file containing the data to be used as the payload. This is the most reliable way to insert custom data.
- `<target_IP_address>`: The IP address of the target you want to send the ICMP packet to. This could be any live host on your network or an external IP address.

Here's how you'd execute it step-by-step:

3. **Craft and send the hping3 command:**
   Now, put it all together. Replace "JohnDoe" with your actual name, `7` with the length of your name, and `192.168.1.1` with your target IP address (e.g., your router's IP, or another device on your network).

```bash
sudo hping3 -1 -d 7 -E my_name.txt 192.168.1.1
```

If you wanted to see the output without targeting a specific IP, you could try `localhost` (127.0.0.1) if you have an ICMP listener, but `hping3` is designed for network interaction.

Example with "AI Assistant":

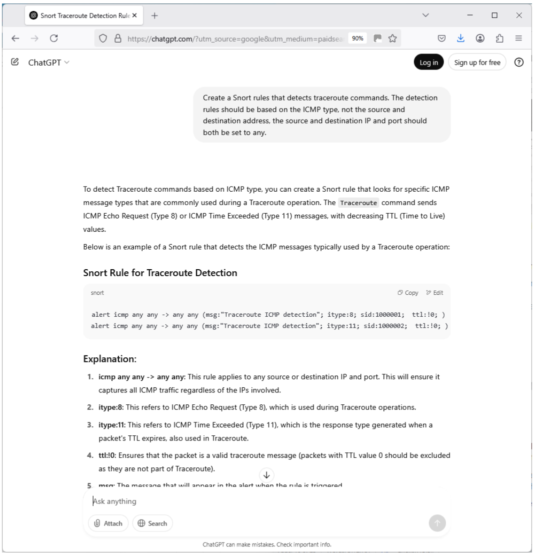
1. **Create file:**

```bash
echo "AI Assistant" > ai_name.txt
```

2. **Determine length:**
   "AI Assistant" has 12 characters.
3. **Send packet:**

```bash
sudo hping3 -1 -d 12 -E ai_name.txt 192.168.1.1
```

Figure 6: Google Gemini hping3 Command


Snort Traceroute Detection Rule
https://chatgpt.com/?utm_source=google&utm_medium=paidsea
90%
ChatGPT
Log in
Sign up for free

Create a Snort rules that detects traceroute commands. The detection rules should be based on the ICMP type, not the source and destination address, the source and destination IP and port should both be set to any.

To detect Traceroute commands based on ICMP type, you can create a Snort rule that looks for specific ICMP message types that are commonly used during a Traceroute operation. The `Traceroute` command sends ICMP Echo Request (Type 8) or ICMP Time Exceeded (Type 11) messages, with decreasing TTL (Time to Live) values.

Below is an example of a Snort rule that detects the ICMP messages typically used by a Traceroute operation:

Snort Rule for Traceroute Detection

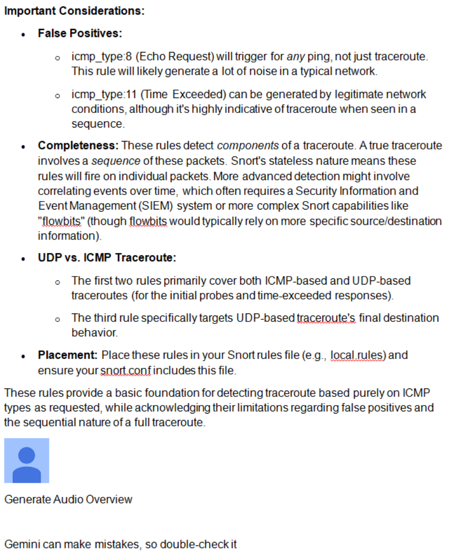
```
alert icmp any any -> any any (msg:"Traceroute ICMP detection"; itype:8; sid:1000001;  ttl:!0; )
alert icmp any any -> any any (msg:"Traceroute ICMP detection"; itype:11; sid:1000002;  ttl:!0; )
```

Explanation:

1. **icmp any any -> any any:** This rule applies to any source or destination IP and port. This will ensure it captures all ICMP traffic regardless of the IPs involved.
2. **itype:8:** This refers to ICMP Echo Request (Type 8), which is used during Traceroute operations.
3. **itype:11:** This refers to ICMP Time Exceeded (Type 11), which is the response type generated when a packet's TTL expires, also used in Traceroute.
4. **ttl:!0:** Ensures that the packet is a valid traceroute message (packets with TTL value 0 should be excluded as they are not part of Traceroute).
5. **msg:** The message that will appear in the alert when the rule is triggered

ChatGPT can make mistakes. Check important info.

Figure 7: OpenAI ChatGpt ICMP Produced traceroute Snort Rule



Important Considerations:

- **False Positives:**
  - icmp_type:8 (Echo Request) will trigger for *any* ping, not just traceroute. This rule will likely generate a lot of noise in a typical network.
  - icmp_type:11 (Time Exceeded) can be generated by legitimate network conditions, although it's highly indicative of traceroute when seen in a sequence.
- **Completeness:** These rules detect *components* of a traceroute. A true traceroute involves a *sequence* of these packets. Snort's stateless nature means these rules will fire on individual packets. More advanced detection might involve correlating events over time, which often requires a Security Information and Event Management (SIEM) system or more complex Snort capabilities like "flowbits" (though flowbits would typically rely on more specific source/destination information).
- **UDP vs. ICMP Traceroute:**
  - The first two rules primarily cover both ICMP-based and UDP-based traceroutes (for the initial probes and time-exceeded responses).
  - The third rule specifically targets UDP-based traceroute's final destination behavior.
- **Placement:** Place these rules in your Snort rules file (e.g., local.rules) and ensure your snort.conf includes this file.

These rules provide a basic foundation for detecting traceroute based purely on ICMP types as requested, while acknowledging their limitations regarding false positives and the sequential nature of a full traceroute.

Gemini can make mistakes, so double-check it

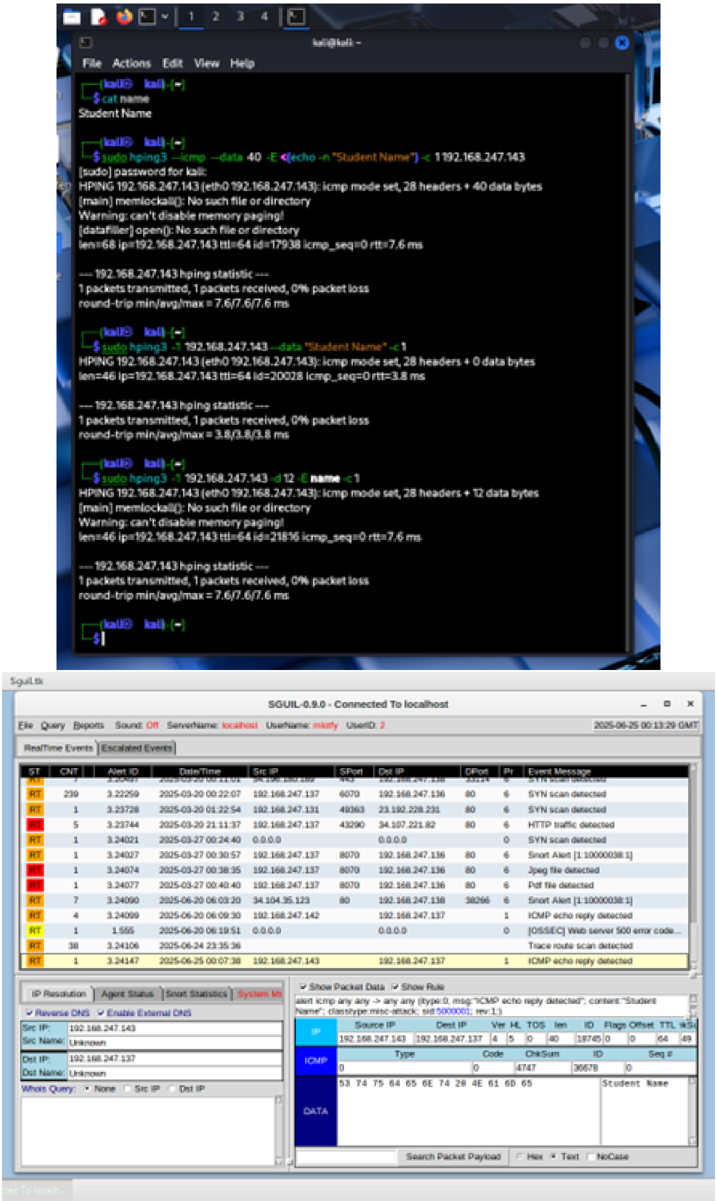
Figure 8: Snippet from Google Gemini ICMP Produced tracrroute Snort Rule

4 Discussion and Results

Given the pedagogical context, where students are instructed on crafting NIDS Snort rules for diverse ICMP traffic detection and utilizing “hping3” commands for traffic generation, it is evident that Generative AI (GenAI) can readily produce initial versions of these rules and commands. However, the efficacy of this direct generation is contingent upon the student’s ability to critically analyze, modify, and adapt the GenAI-provided outputs to meet specific assignment requirements. Despite the relatively straightforward nature of the rules involved, the GenAI-generated outputs consistently necessitated post-generation refinement by the student, indicating that they are not immediately deployable without human intervention and correction. We tested the provided GenAI rules and “hping3” commands using a Kali VM, a Security Onion VM and a Debian 10 VM in VMware. The GenAI ICMP echo request and reply were updated to check for the ”Student Name” in the content, added the classtype, and updated the SID numbers, which make the ChatGPT and Copilot rules similar to the Gemini provided rules. We added the rules to the “local.rules” file and updated the Security Onion rules and restarted the sensors. We ran the three GenAI “hping3”commands on the Kali VM to generate the ICMP traffic to the Debian 10 VM to see if they will be detected by the SO NIDS VM on the subnet. We adjusted each GenAI “hping3” command content text and the data size. Figure 9 shows a Kali term with all the commands and the Squil triggered Snort rules.

Only the Google Gemini GenAI tweaked “hping3” command, simialr to what the faculty provided in the course presentation, generated the correct traffic. The ChatGPT “hping3” command was incorrect since the “--data” option requires the size of data not a string. The Copilot “hping3” command did not generate the required traffic as well, thus not detected on Squil. As seen on Squil, only one packet was detected despite we generated three packets, one from each command. We then tested the rules to detect traceroute traffic. We added a rule to trap for “itype:11”, TTL expires, and included “ttl:!0” option as provided by Google Gemini. Updating the rules caused the Snort sensor to fail. The failure was due to the use of “ttl:!0”. Fixing the Gemini rule by changing the option to “ttl:>0” and adding a second rule omitting the “itype:11” option and just checking that the “ttl:>0” only. Updating the rules went fine and the Snort sensors were up. We generated a “traceroute” command to “Google.com” and the Debian 10 VM (192.168.247.143). The Google Gemini rules didn’t trigger an alert on Squil. Only the rule we added trapping for packets with TTL greater than zero triggered an alert. Figure 10 shows a Kali term with the “traceroute” commands and the Squil triggered Snort rules.

Analyzing student submissions during the Fall 2024 and Spring 2025 under-



```
┌──(kali㉿kali)-[~]
└─$ cat name
Student Name

┌──(kali㉿kali)-[~]
└─$ sudo hping3 --icmp --data 40 -E <(echo -n "Student Name") -c 1 192.168.247.143
[sudo] password for kali:
HPING 192.168.247.143 (eth0 192.168.247.143): icmp mode set, 28 headers + 40 data bytes
[main] memlockall(): No such file or directory
Warning: can't disable memory paging!
[datafiller] open(): No such file or directory
len=68 ip=192.168.247.143 ttl=64 id=17938 icmp_seq=0 rtt=7.6 ms

--- 192.168.247.143 hping statistic ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 7.6/7.6/7.6 ms

┌──(kali㉿kali)-[~]
└─$ sudo hping3 -1 192.168.247.143 --data "Student Name" -c 1
HPING 192.168.247.143 (eth0 192.168.247.143): icmp mode set, 28 headers + 0 data bytes
len=46 ip=192.168.247.143 ttl=64 id=20028 icmp_seq=0 rtt=3.8 ms

--- 192.168.247.143 hping statistic ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 3.8/3.8/3.8 ms

┌──(kali㉿kali)-[~]
└─$ sudo hping3 -1 192.168.247.143 -d 12 -E name -c 1
HPING 192.168.247.143 (eth0 192.168.247.143): icmp mode set, 28 headers + 12 data bytes
[main] memlockall(): No such file or directory
Warning: can't disable memory paging!
len=46 ip=192.168.247.143 ttl=64 id=21816 icmp_seq=0 rtt=7.6 ms

--- 192.168.247.143 hping statistic ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 7.6/7.6/7.6 ms

┌──(kali㉿kali)-[~]
└─$
```

SGUIL-0.9.0 - Connected To localhost

Sound: Off ServerName: localhost UserName: mlotfy UserID: 2 2025-06-25 00:13:29 GMT

ST	CNT	Alert ID	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	239	3.22259	2025-03-20 00:22:07	192.168.247.137	6070	192.168.247.136	80	6	SYN scan detected
RT	1	3.23728	2025-03-20 01:22:54	192.168.247.131	49363	23.192.228.231	80	6	SYN scan detected
RT	5	3.23744	2025-03-20 21:11:37	192.168.247.137	43290	34.107.221.82	80	6	HTTP traffic detected
RT	1	3.24021	2025-03-27 00:24:40	0.0.0.0		0.0.0.0		0	SYN scan detected
RT	1	3.24027	2025-03-27 00:30:57	192.168.247.137	8070	192.168.247.136	80	6	Snort Alert [1:10000038:1]
RT	1	3.24074	2025-03-27 00:38:35	192.168.247.137	8070	192.168.247.136	80	6	Jpeg file detected
RT	1	3.24077	2025-03-27 00:40:40	192.168.247.137	8070	192.168.247.136	80	6	Pdf file detected
RT	7	3.24090	2025-06-20 06:03:20	34.104.35.123	80	192.168.247.138	38266	6	Snort Alert [1:10000038:1]
RT	4	3.24099	2025-06-20 06:09:30	192.168.247.142		192.168.247.137		1	ICMP echo reply detected
RT	1	1.555	2025-06-20 06:19:51	0.0.0.0		0.0.0.0		0	[OSSEC] Web server 500 error code...
RT	38	3.24106	2025-06-24 23:35:36						Trace route scan detected
RT	1	3.24147	2025-06-25 00:07:38	192.168.247.143		192.168.247.137		1	ICMP echo reply detected

☑ Reverse DNS ☑ Enable External DNS

Src IP: 192.168.247.143
Src Name: Unknown
Dst IP: 192.168.247.137
Dst Name: Unknown

Whois Query: ● None ○ Src IP ○ Dst IP

☑ Show Packet Data ☑ Show Rule

alert icmp any any -> any any (itype:0; msg:"ICMP echo reply detected"; content:"Student Name"; classtype:misc-attack; sid:5000001; rev:1;)

IP	Source IP	Dest IP	Ver	HL	TOS	len	ID	Flags	Offset	TTL
	192.168.247.143	192.168.247.137	4	5	0	40	18745	0	0	64

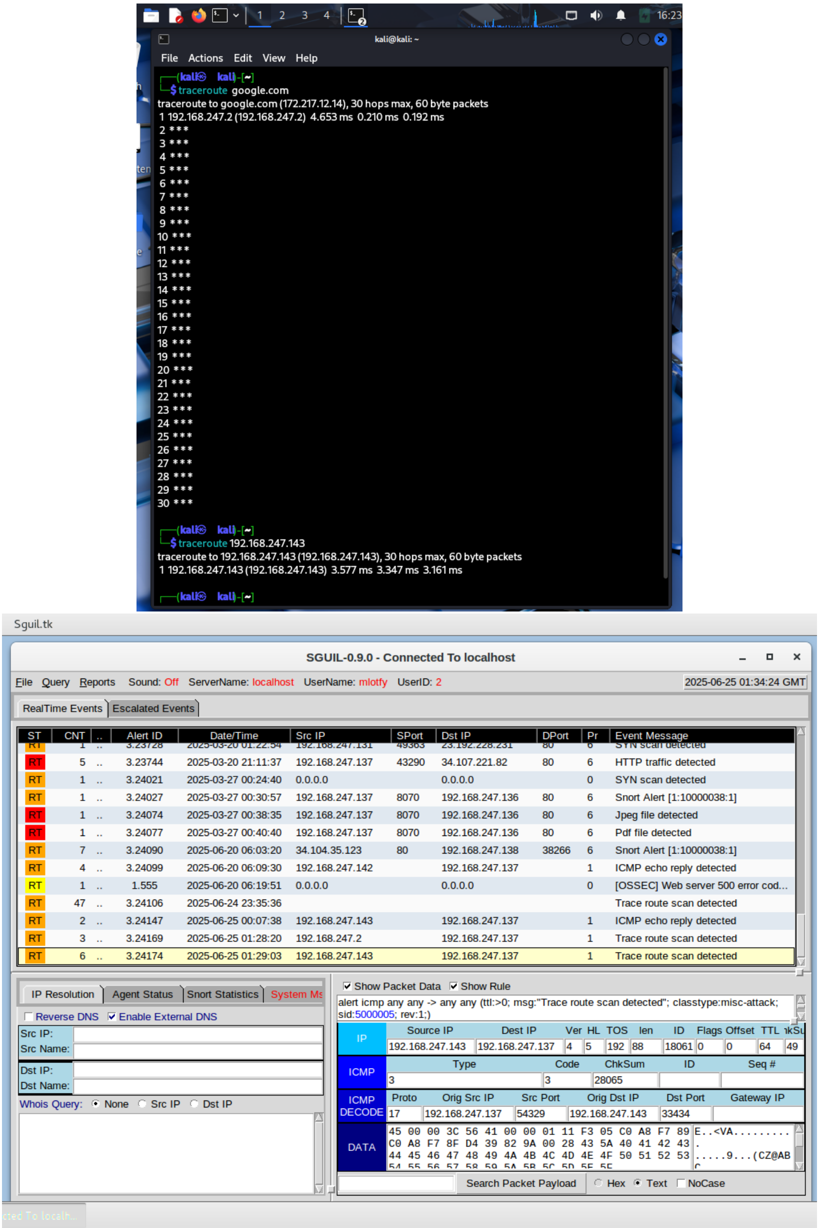
ICMP	Type	Code	ChkSum	ID	Seq #
	0	0	4747	36678	0

DATA

```
53 74 75 64 65 6E 74 20 4E 61 6D 65        Student Name
```

○ Hex ● Text ☐ NoCase

Figure 9: GenAI hping3 commands and the Triggered Snort Rule



```
┌──(kali㉿kali)-[~]
└─$ traceroute google.com
traceroute to google.com (172.217.12.14), 30 hops max, 60 byte packets
 1 192.168.247.2 (192.168.247.2) 4.653 ms 0.210 ms 0.192 ms
 2 * * *
 3 * * *
 4 * * *
 5 * * *
 6 * * *
 7 * * *
 8 * * *
 9 * * *
10 * * *
11 * * *
12 * * *
13 * * *
14 * * *
15 * * *
16 * * *
17 * * *
18 * * *
19 * * *
20 * * *
21 * * *
22 * * *
23 * * *
24 * * *
25 * * *
26 * * *
27 * * *
28 * * *
29 * * *
30 * * *

┌──(kali㉿kali)-[~]
└─$ traceroute 192.168.247.143
traceroute to 192.168.247.143 (192.168.247.143), 30 hops max, 60 byte packets
 1 192.168.247.143 (192.168.247.143) 3.577 ms 3.347 ms 3.161 ms

┌──(kali㉿kali)-[~]
```

SGUIL-0.9.0 - Connected To localhost

Sound: Off ServerName: localhost UserName: mlotfy UserID: 2 2025-06-25 01:34:24 GMT

ST	CNT	..	Alert ID	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	5	..	3.23744	2025-03-20 21:11:37	192.168.247.137	43290	34.107.221.82	80	6	HTTP traffic detected
RT	1	..	3.24021	2025-03-27 00:24:40	0.0.0.0		0.0.0.0		0	SYN scan detected
RT	1	..	3.24027	2025-03-27 00:30:57	192.168.247.137	8070	192.168.247.136	80	6	Snort Alert [1:10000038:1]
RT	1	..	3.24074	2025-03-27 00:38:35	192.168.247.137	8070	192.168.247.136	80	6	Jpeg file detected
RT	1	..	3.24077	2025-03-27 00:40:40	192.168.247.137	8070	192.168.247.136	80	6	Pdf file detected
RT	7	..	3.24090	2025-06-20 06:03:20	34.104.35.123	80	192.168.247.138	38266	6	Snort Alert [1:10000038:1]
RT	4	..	3.24099	2025-06-20 06:09:30	192.168.247.142		192.168.247.137		1	ICMP echo reply detected
RT	1	..	1.555	2025-06-20 06:19:51	0.0.0.0		0.0.0.0		0	[OSSEC] Web server 500 error cod...
RT	47	..	3.24106	2025-06-24 23:35:36						Trace route scan detected
RT	2	..	3.24147	2025-06-25 00:07:38	192.168.247.143		192.168.247.137		1	ICMP echo reply detected
RT	3	..	3.24169	2025-06-25 01:28:20	192.168.247.2		192.168.247.137		1	Trace route scan detected
RT	6	..	3.24174	2025-06-25 01:29:03	192.168.247.143		192.168.247.137		1	Trace route scan detected

☐ Reverse DNS ☑ Enable External DNS

Whois Query: ● None ○ Src IP ○ Dst IP

☑ Show Packet Data ☑ Show Rule

```
alert icmp any any -> any any (ttl:>0; msg:"Trace route scan detected"; classtype:misc-attack;
sid:5000005; rev:1;)
```

IP	Source IP	Dest IP	Ver	HL	TOS	len	ID	Flags	Offset	TTL	ıkSu
	192.168.247.143	192.168.247.137	4	5	192	88	18061	0	0	64	49

ICMP	Type	Code	ChkSum	ID	Seq #
	3	3	28065		

ICMP DECODE	Proto	Orig Src IP	Src Port	Orig Dst IP	Dst Port	Gateway IP
	17	192.168.247.137	54329	192.168.247.143	33434	

DATA

```
45 00 00 3C 56 41 00 00 01 11 F3 05 C0 A8 F7 89 E..<VA.........
C0 A8 F7 8F D4 39 82 9A 00 28 43 5A 40 41 42 43 .
44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 .....9...(CZ@AB
```

Search Packet Payload ○ Hex ● Text ☐ NoCase

Figure 10: “traceroute” commands and the Triggered Snort Rule

graduate and the Spring 2025 graduate network defense and countermeasures courses, revealed instances where students utilized GenAI to generate both the requisite Snort rules and "hping3" commands for assignment completion. Table 1 shows the use of GenAI to complete the ICMP and "traceroute" assignments for the undergraduate Fall 2024, and Spring 2025 classes, and the graduate Spring 2025 class. Despite the use of GenAI to complete the assignments in the undergraduate courses was low (5% - 8%), only one student managed to tweak the rules and make them work. A particularly noteworthy finding emerged from the graduate course, where 21% of students employed GenAI to generate the "traceroute" Snort rules. Surprisingly, even after students attempted modifications, these rules consistently failed to trigger alerts on Squil. This outcome empirically reinforced our preliminary observations that the GenAI-generated Snort rules were either incomplete or necessitated significant manual refinement to achieve operational functionality. Also, the GenAI "hping3" commands needed to be fixed to make them work. If students fail to go over the course presentations or coverage of the content and rely on GenAI, they will not be able to identify the issues in the GenAI generated Snort rules and commands. As Gupta, Akiri, Aryal, Parker, and Praharaj (2023) mentioned, GenAI chatbots have their strengths and weaknesses, "it is crucial for users to be aware of these aspects to make informed decisions about their use" [3]. In addition, system vulnerability might increase due to the "misunderstanding between the IT and cybersecurity professionals and the GenAI tool" [11].

Table 1: Use of GenAI to Complete the Assignments Results

	GenAI Use (%)	Correct	Incorrect
Undergrad Fall 2024 (N= 24, n=2)			
ICMP rules	8.33	1	1
hping3 command	4.66	-	1
Undergrad Spring 2025 (N=19, n=1)			
hping3 command	5.26	-	1
Grad Spring 2025 (N=19, n=4)			
traceroute rule	21.05	-	4

Regardless of whether the use of GenAI tools is permitted in a class or not, instructors must understand how students utilize them. Recognizing the inevitability of students leveraging GenAI tools, even when output accuracy is uncertain, educators must adapt pedagogical strategies. We posit that assignments should be designed as modular building blocks, compelling students to synthesize information and integrate components independently. It is crucial to emphasize that GenAI serves as an assistive tool, providing preliminary content, but often lacks the contextual understanding necessary for autonomous problem-solving. Consequently, mastery of the underlying content and techniques becomes paramount; without this fundamental comprehension, students will be incapable of identifying and rectifying inaccuracies in GenAI-generated responses, thus hindering successful assignment completion.

Furthermore, if GenAI use is permitted, a valuable pedagogical approach involves encouraging students to utilize and compare outputs from various GenAI tools. This fosters critical evaluation skills and enables them to identify the most suitable tool for a given task. Alternatively, GenAI can be employed as a meta-cognitive aid, allowing students to evaluate and deepen their understanding of core concepts [10]. In addition, faculty should include a GenAI usage policy in their courses. The policy should include the learning opportunity, acceptable use, and unacceptable use of GenAI. The policy should stress that the use of GenAI to complete graded assignments or assessments is not allowed as well as writing the assignment reports. Students should acknowledge and agree to adhere to the GenAI policy outlined in the course syllabus [6].

5 Conclusion

As GenAI tools and chat-bots continue to proliferate across technological and educational landscapes, it is imperative for educators to recognize both their growing accessibility and their inevitable use by students. These tools have become increasingly integrated into widely used platforms such as internet search engines and social media applications, thereby lowering the barrier to entry for even novice users. In the field of cybersecurity, the dual-use nature of GenAI poses considerable implications, as it can facilitate both beneficial applications and malicious activities. GenAI has demonstrated the ability to disseminate misinformation, generate code for cyber-attacks, conduct social engineering and phishing campaigns, assist in the development of malware, and contribute to vulnerability analysis, the creation of intrusion detection rules, incident response protocols, and code security improvements.

In light of these capabilities, it is essential that instructors in IT and cybersecurity programs provide structured guidance to students on the strategic and ethical use of GenAI. Educators must emphasize both the advantages and limitations associated with GenAI-generated outputs, including rules, commands, and code. Network intrusion detection systems (NIDS), in particular, rely on the precision and completeness of rule sets. An uncritical dependence on GenAI-generated rules may lead to undetected malicious traffic, compromising the integrity of network defenses. Furthermore, despite the utility of GenAI in automating routine tasks and generating syntactically valid content, such tools are not substitutes for the human creativity, judgment, and analytical reasoning that underpin effective defensive cybersecurity operations.

Future research should explore how the integration of GenAI tools will reshape pedagogical approaches to teaching both offensive and defensive cybersecurity in academic settings. Exploratory studies are needed to investigate how these tools influence instructional design, student engagement, and the practical application of cybersecurity concepts in the classroom. Additionally, empirical research must assess the impact of GenAI-assisted learning on students' acquisition of critical cybersecurity skills and foundational knowledge. Future investigations should also examine the evolving role of GenAI in threat detection, the maintenance and updating of security systems and networks, and its broader implications for workforce readiness and the nature of cybersecurity employment.

References

[1] Moatsum Alawida, Sami Mejri, Abid Mehmood, Belkacem Chikhaoui, and Oludare Isaac Abiodun. A comprehensive study of chatgpt: advancements, limitations, and ethical considerations in natural language processing and cybersecurity. *Information*, 14(8):462, 2023.

[2] Pooneh Nikkhah Bahrami, Ali Dehghantanha, Tooska Dargahi, Reza M Parizi, Kim-Kwang Raymond Choo, and Hamid HS Javadi. Cyber kill chain-based taxonomy of advanced persistent threat actors: Analogy of tactics, techniques, and procedures. *Journal of information processing systems*, 15(4):865–889, 2019.

[3] Maanak Gupta, CharanKumar Akiri, Kshitiz Aryal, Eli Parker, and Lopamudra Praharaj. From chatgpt to threatgpt: Impact of generative ai in cybersecurity and privacy. *IEEE Access*, 11:80218–80245, 2023.

[4] Basil Hamdan. Integrating chatgpt in cybersecurity education: Use cases and implications. *Journal of Computing Sciences in Colleges*, 40(2):105–114, 2024.

[5] Eric M Hutchins, Michael J Cloppert, and Rohan M Amin. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 1(1):80, 2011.

[6] Ed Lindoo and Mohamed Lotfy. Generative ai and its impact on the cs classroom and programmers. *Journal of Computing Sciences in Colleges*, 40(2):35–50, 2024.

[7] Mohamed Lotfy. The structure of a graduate defensive cybersecurity course. *Journal of Computing Sciences in Colleges*, 39(2):72–84, 2023.

[8] Mitre. Att&ck matrix for enterprise, 2025. Visited June 9, 2025. URL: `https://attack.mitre.org`.

[9] Shafi Parvez Mohammed and Gahangir Hossain. Chatgpt in education, healthcare, and cybersecurity: Opportunities and challenges. In *2024 IEEE 14th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0316–0321. IEEE, 2024.

[10] E Mollick, L Mollick, O Acar, and M Weiss. 4 simple ways to integrate ”ai” into your class. *Harvard Business Publishing Education*, pages 4–10, 2024.

[11] Siva Sai, Utkarsh Yashvardhan, Vinay Chamola, and Biplab Sikdar. Generative ai for cyber security: Analyzing the potential of chatgpt, dall-e and other models for enhancing the security space. *IEEE access*, 2024.

[12] Yagmur Yigit, William J Buchanan, Madjid G Tehrani, and Leandros Maglaras. Review of generative ai methods in cybersecurity. *arXiv preprint arXiv:2403.08701*, 2024.

Vast, Low-cost Honeypot Architecture*

David Charles Loper[1], Jenny Valentine Nehring[2],
and Prakash Ranganathan[1]
[1]Center for Cyber Security Research
College of Engineering and Mines
University of North Dakota
Grand Forks, ND 58202
[2]Information Systems and Technology
Smith College of Engineering and Technology
Utah Valley University
Orem, UT 84057
`{david.loper,prakash.ranganathan}@und.edu`, jnehring@uvu.edu

Abstract

This paper presents a low-cost honeypot architecture designed to assist cybersecurity researchers and practitioners in collecting high-quality threat data using legacy equipment and open-source tools. By reusing older capital expense (CAPEX) devices and leveraging underutilized Internet Protocol (IP) and port resources, this framework enables the scaling of honeypot deployments across a wide range of interaction levels without requiring significant new investment. This paper outlines a flexible approach that combines virtualization, containerization, and scripting to build segmented environments capable of capturing both opportunistic and targeted attacks. The design includes the Principle of Least Purchase (PoLP), a practical strategy for converting surplus infrastructure into a valuable defense asset. We also provide a simple model for estimating attacker behavior across an expanded attack surface.

---

*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

Honeypots provide security researchers with critical data about threats to their network while simultaneously frustrating hackers by inducing them to reveal their secrets. The provided intelligence may include simple items, such as the attacker's IP addresses, to complex questions about one's adversary, including their tactics, techniques, and procedures (TTPs). However, with the increasing use of Residential Internet Protocol Address (ResIP) proxies by malicious attackers, traditional IP reputation intelligence is insufficient to stop advanced persistent threats (APTs). This paper introduces methods to reduce honeypot costs, enabling increased honeypot deployment. Specifically, it details leveraging legacy equipment and under-leveraged network resources (addresses and ports) to gather significant data that IT professionals could use in classical systems and local AI analysis to produce contemporary and real-time data for defense.

1.1 Background and Motivation

Our research is closely related to botnet detection and the application of artificial intelligence (AI). The use of cost-effective methods to gather information about emerging threats is a key tool for ongoing research.

In common with local governments [14], academic work in cybersecurity is often constrained in funding but may leverage historically available internet protocol (IP) address space [20] [7]. Because our research is exploratory, the flexibility to investigate various levels of Bianco's Pyramid of Pain (PoP) is needed [2]. This framework enables us to explore various collection domains under the PoP.

We are further motivated to explore solutions to the prevalent lack of cybersecurity training data available for research. When considering AI use against a constantly evolving threat vector, the application of historical data is not valid. Furthermore, it is essential to establish a foundation of experimental repeatability through the production process, which serves as a means to validate claims. The repeatability of the process is crucial, given that threat actors continually update their tools, tactics, and procedures (TTPs). Consequently, a training set that captures a valuable set of training data will typically become useless later, due to obsolescence [13] [1].

1.2 Problem

Most honeypot solutions focus on delivering a single paradigm and do not provide a framework for providing various levels of strength and countermeasures to areas of weakness. These solutions lack a framework-of-frameworks approach and focus instead on specific interaction levels, whereas our proposal provides a space for these excellent, focused tools.

Low-interaction honeypots are lightweight and easy to deploy. They scale well and are well-suited for automated analysis. However, they often lack realism and are easily identifiable (i.e., fingerprinting). Medium interaction honeypots provide better deception and more insight. While they are safer than high-interaction systems, they are still limited and somewhat fingerprintable while incurring additional complexity. High-interaction honeypots provide greater detail because they accurately mimic employable hackable systems, which allows for a more thorough analysis of hacker TTPs. They require greater resources for analysis, and they can incur real-world harm if not well-isolated. Additionally, they can be challenging to scale and often struggle to differentiate between APTs and opportunistic attacks.

1.3 Importance

Like insurance premiums, many organizations view cybersecurity expenses as non-essential, as the perceived value in cybersecurity is often only realized following an incident or event [16]. Because our architecture emphasizes leveraging legacy devices purchased as capital expenditures (CAPEX), arguing for a return on investment for sunk costs is always a consideration. In short, if an organization has legacy (i.e., surplus) resources that are not well leveraged for cybersecurity, it is not utilizing its resources effectively.

1.4 Gaps, Shortcomings, and Tradeoffs

The use of older equipment for cybersecurity has a downside, particularly in terms of electrical expenses and human capital productivity, past a certain point. This paper does not provide guidance for when electrical expenses and employee wages exceed the value of leveraging older CAPEX devices.

This paper also advocates for extensive use of free software and frameworks, which include open source software (OSS) and open standards. This open framework is reliable and is well-known for its cost efficiency when using limited resources [3]. For some institutions, investing in the training and education of existing staff on the use of OSS software and solutions may be necessary.

1.5 Organization of this Paper and Contributions

To create this solution, practitioners will need to build on skill and ability in open standards, OSS, and industry best practices. Analogs of any of these items are encouraged and may be more effective in professional cybersecurity infrastructures. As we cover related and prior work, it will be important for the reader to become familiar with these concepts when attempting their implementation.

This paper introduces some key systems architecture related to the practical nature of implementing botnet detection. First, as a way to make our framework affordable, we introduce the Principle of Least Purchase/Spend (PoLP, PoLS, or PoL$), an intentional misnomer to the cybersecurity principle of least privilege (PoLP). This evolution of equipment constitutes a technological exaptation, a term we borrow from biology. With this concept in place, we discuss a non-linear way to reduce an attack surface by proportionality that is made possible through the vast resources provided by the PoL$. We detail how this reduction can help cybersecurity professionals discriminate between opportunism and intentionality, revealing APTs.

Finally, we provide a robust approach that is viable for our future research purposes and benefits other researchers and cyber-professionals. We illustrate this approach through the example of two networks we created, which provided low-interaction and high-interaction networks for cybersecurity testing and data collection.

2 Prior and Related Work

Our architecture relies heavily on existing open standards. IEEE 802.3AD allows us to increase bandwidth on the network by combining network interfaces into a single, logical device [4]. IEEE 802.1Q provides a method that allows us to create segmentation between our networks, critical to our security and flexibility [6]. IEEE 802.1D provides the ability to place virtual machines on the physical network interface of a virtualization host[5].

Miura et al. demonstrate that the combination of IEEE 802.3AD, IEEE 802.1Q, and IEEE 802.1D provides higher bandwidth and greater flexibility [12]. Winter et al. provide jumbo frames as a means for bundling large packet transfers, increasing bandwidth while reducing CPU overhead [21]. Kanthla demonstrates that the use of jumbo frames in cloud virtualization applications yields greater performance [8]. Since we are creating a functional, private cloud, these standards are vital for optimal performance.

Oleksiuk and Oleksiuk demonstrate that the use of Proxmox VE in an academic cloud environment offers a cost-effective, scalable, and high-productivity solution for virtualization tasks [15]. Sharma et al. [18] and Lekkala [10] demonstrate the utility of automation with Terraform for scalable private-cloud architecture.

3 Fundamental Architecture Principles

We emphasize leveraging virtualization for cost savings, but it is also a key factor in our security model. Virtualization and containerization enable us to segment resources into granular components, which can be assigned to a honeypot duty cycle. Cyber security targets on a network segment should be isolated. Compromised virtual machines can then be quarantined and analyzed in a safe post-mortem environment. Additionally, simply restarting containers can return them to a clean state. Automation and orchestration make the whole system manageable and optimize the investment of human capital by making their efforts applicable across a continuous stream of hardware devices.

Our architecture begins with design principles that differ from typical production deployments, transforming punitive ideas and weaknesses into cost-saving strengths. To maximize cost savings, we advocate for virtualization and containerization, as well as the use of open-source software, penny-pinching, indifference to uptime, power efficiency, and a perception of security apathy, which can lead to increased attack surfaces. These unintuitive means allow for greater data collection from threat actors.

Where possible, repurposed hardware can emulate actual exposed attack services since this hardware was previously used for production services. Without the burden of production-based loads, such hardware can typically bear a longer lifecycle when faced only with the minimal cybersecurity loads imposed on their processes. The emulation, replication, and conversion to honeypot-oriented purposes will attract hackers who specifically target the same class of service that an enterprise runs in their production environments. It is recommended that a faithful replication of the production environment is not reproduced, as this can give a roadmap for attackers to tool their attacks. Low-interaction honeypots limited to port interactions may be sufficient to understand the first phases of the Cyber Kill Chain, while higher interaction honeypots may be needed to understand attackers at later phases [11].

3.1 Principle of Least Purchase

Enterprise systems will often include on-premises CAPEX equipment. This equipment is typically only valuable for the organization while the warranty period is active. Because reliability is critical for production systems, the utility of servers and network switches are significantly diminished after the warranty period, leading to its designation as surplus equipment.

Organizations owe no promise of availability to the threat actor community. This behavior can be unintuitive to trained IT professionals who regularly provide the best experience for customers and partners. Threat actors deserve the bad experience of unstable, false attack surfaces. To that end, devices that

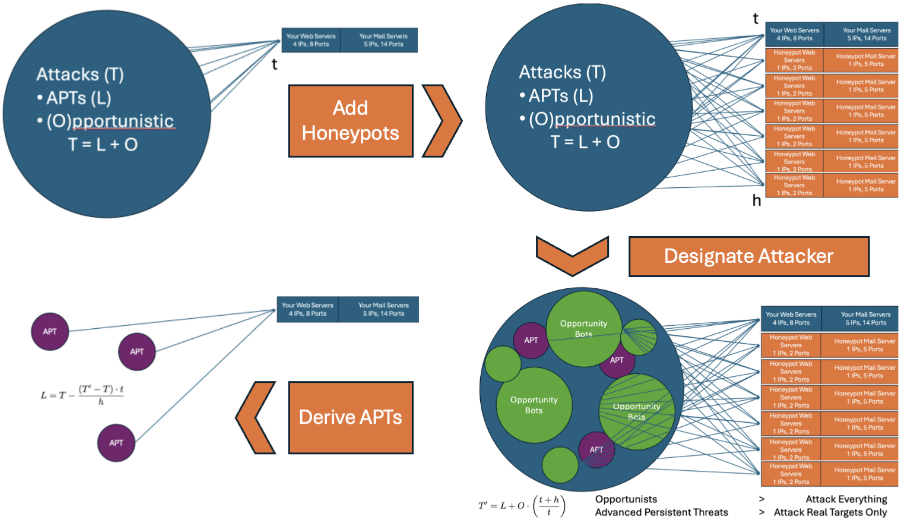

Attacks (T)
• APTs (L)
• (O)pportunistic
T = L + O
Your Web Servers 4 IPs, 8 Ports
Your Mail Servers 5 IPs, 14 Ports
t
Add Honeypots
Honeypot Web Servers 1 IPs, 2 Ports
Honeypot Mail Server 1 IPs, 5 Ports
h
Designate Attacker
Opportunity Bots
APT
Derive APTs
$L = T - \frac{(T' - T) \cdot t}{h}$
$T' = L + O \cdot \left(\frac{t+h}{t}\right)$
Opportunists > Attack Everything
Advanced Persistent Threats > Attack Real Targets Only

Figure 1: Using a Large Attack Surface to Detect APTs

no longer are serviceable under warranty are ideally suited for technological exaptation, becoming honeypot analogs. In this model, nearly every piece of deprecated infrastructure becomes useful for cybersecurity. Found utility in legacy hardware provides full depreciation capability to the business, a steady stream of new equipment for cybersecurity teams, a vast playground for wasting threat actor resources, and other benefits. We do not advocate a zero spend on cybersecurity. However, leaving this equipment unused represents an inefficiency in infrastructure spending.

3.2 Evaluating Opportunistic and APT attacks

It is a common cybersecurity principle to reduce a site's attackable surface [19]. At some point, the attack surface size will reach an irreducible number without compromising features and functions. The only way to reduce attacker focus on these reduced points is to confuse opportunistic attackers with false surfaces. Kocaogullar et al. identify that low-interaction honeypots reduce attacker interest, which can force intentional attackers to blocklist targets [9]. We weaponize this behavior to produce a list of APTs and rich entrapment targets for our high-interaction honeypots. Our architecture is well-suited to this approach by increasing the distinction between automated and non-automated threats. We make an assumption that APTs are conducting directed attacks at specific targets while opportunistic attacks, like port scanning, will broadly probe and attack our entire surface through procedure and automation.

Using the opportunistic attacks as a baseline, we can create honeypots at various levels of the PoP, starting with simple IP Address banning. While this approach is similar to many log-based IP banning methods (such as fail2ban and Fortiguard), it provides feedback mechanisms at additional layers. This framework can collect data at each honeypot node to create a mask based on opportunistic systems and threat actor AI systems, as shown in Fig. 1.

Let:

- T: total number of attacks observed **before** honeypot deployment,
- T': total number of attacks observed **after** honeypot deployment,
- L: number of targeted or APT attacks (e.g., adversaries with intent),
- O: number of opportunistic attacks (e.g., scanners, bots, random probes),
- t: number of production (real) systems,
- h: number of honeypot systems deployed.

We assume:

$$T = L + O \tag{1}$$

$$T' = L + O \cdot \left(\frac{t+h}{t}\right) \tag{2}$$

Opportunistic attacks are distributed proportionally across all visible systems, whereas targeted APT attacks focus solely on specific targets.

Subtracting (1) from (2):

$$T' - T = O \cdot \left(\frac{h}{t}\right) \tag{3}$$

Solving for O and L:

$$O = \frac{(T' - T) \cdot t}{h} \tag{4}$$

$$L = T - \frac{(T' - T) \cdot t}{h} \tag{5}$$

4 Practical Architecture and Our Implementations

If all has gone right so far, an organization will have spent nothing in new CAPEX following this model. We acknowledge that this model favors on-premises deployments in environments with existing, retired hardware and new, retiring hardware on the horizon. For greenfield deployments, purchasing used hardware is a low-cost way to enter the paradigm. For cloud deployments, cybersecurity professionals can leverage lower-tier and low-resource virtual environments.

4.1 Physical Machines

If a physical machine dies, remove it. These machines should be considered low-maintenance priorities. In this paradigm, more hardware is on its way as production machines age out out of support and production life-cycles.

In our first implementation, we used separate VLANs on servers and switches that were over 10 years old. As in Fig. 4, we leverage support for NIC bonding, VLAN support, and bridging to create a flexible network for our virtual machines and containers using the three physical interfaces depicted in Fig. 2. We have used this architecture as a blueprint for three (and growing) implementations.

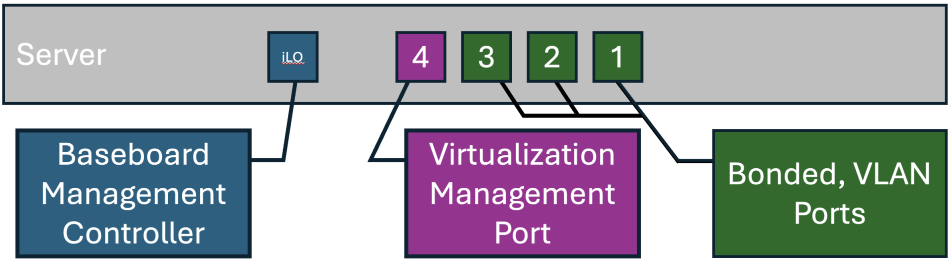

Server
iLO
4
3
2
1
Baseboard Management Controller
Virtualization Management Port
Bonded, VLAN Ports

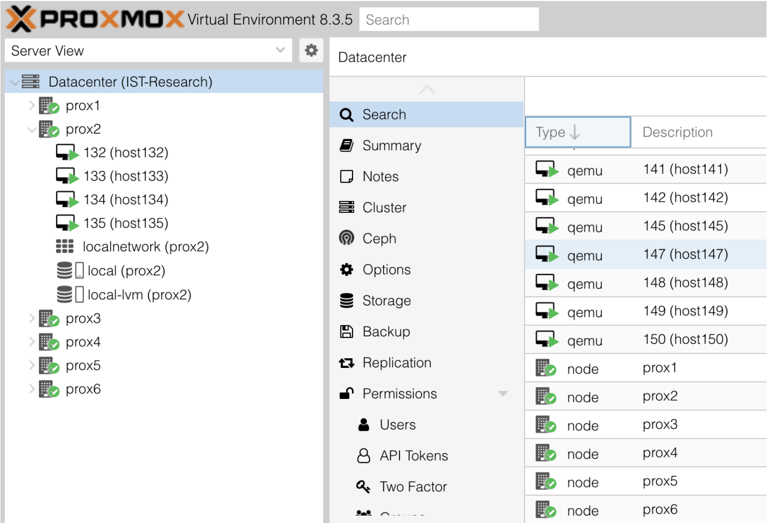
Figure 2: **Server Network Interface Designations**

Our first implementation utilized hardware consisting of six repurposed HP ProLiant DL 360 Gen 9 servers (now HPE) that had 80-96 GB of RAM, single power supplies, 1 TB SAS drives, an iLO baseboard management controller, and four 1 GB NICs. This hardware was obtained for this purpose after its duty cycle and before being sent to surplus. We estimate the total value of all six machines in this cluster to be 600 USD. New machines of this model would have cost more than 4,800 USD each.

4.2 Virtualization and Containerization

Virtualization and containerization offer key benefits in terms of robustness and flexibility, as demonstrated by the successes of cloud infrastructure. Modern platforms support virtualization and containerization, even if they were designed for other purposes. Cloud architecture is known for robustness, and the basic functions of the public cloud are also available in private cloud infrastructures.

The deployment of virtualization and containerization provides increased flexibility, allowing us to deploy low, medium, and high-interaction honeypots of various types and capabilities [17]. We will not speak to the plethora of these solutions, but this architecture can support a vast majority of high and low-interaction honeypots.



Type	Description
qemu	141 (host141)
qemu	142 (host142)
qemu	145 (host145)
qemu	147 (host147)
qemu	148 (host148)
qemu	149 (host149)
qemu	150 (host150)
node	prox1
node	prox2
node	prox3
node	prox4
node	prox5
node	prox6

Figure 3: **Proxmox Cluster Organization and Virtual Machines**

Figure 3 shows our first implementation using Proxmox VE for virtualization and containerization. In some instances where we needed containers other than the LXC containers (native to Proxmox VE), we ran Docker and/or Kubernetes on virtualized hosts. On these systems we used Secure Shell (SSH) services and versions similar to those exposed in adjacent IP addresses. This multiplied the attack surface explorable and served as additional falsified attack points without providing a technical path for any exploitation.

4.3 Scripting, Automation, and Orchestration

When combined with automation and orchestration tools, it is easy to scale hundreds of virtual machines in an organization. Using these tools, we can leverage IP and port resources to scale our public-facing resources, thereby extending our support for our discrimination architecture.

Our implementations started with simple Bourne-Again Shell (BASH) scripts. Later implementations leveraged Terraform and Ansible. In some cases, we used the MAC address of the virtual machine to pass information between the virtual environment and the local running machine, which was pre-programmed with initialization routines that would activate on bits specified within the MAC address designation. Scripts worked well on certain low-interaction honeypots that required secure deployment.

4.4 Attack Surface

We learn from this paradigm that we should leverage all IP and port resources. Using up every IP, over-stacking ports, and forwarding unused ports on production IPs to our honeypots only serves to tighten the resolution of APTs. As much as possible, confuse, distract, and get IP addresses blocklisted by criminals as honeypots. Although this contradicts the advice of Kocaogullar et al. [9], we leverage these predicted outcomes to great success. Moreover, we harvest their patterns every time automated systems attack our expanding infrastructure. This process allows us to translate their attacks into training data for AI defensive structures. Because this data is modern, it is rich with current TTPs.

In our low-interaction implementation, we assigned multiple IP addresses and ports to the same virtual machine. This process is typically discouraged to avoid blocklisting. However, we turned the obvious downside of easy fingerprinting into an advantage when our goal is to achieve blocklisting by the threat actor community. Today, our hosts are primarily targeted by emerging threats and poorly written threat automation systems. We can leverage this information to easily block and remove these attacks from attempts against our production services.

4.5 Server and Network

It is crucial that deployments in this model also utilize legacy networking equipment where feasible. First, older network equipment is often compatible with older server infrastructure. Second, we recommend using older network equipment when feasible with the virtualization hosts. Threat actors will expect it when they are allowed under the hood in high-interaction environments.

We encourage the use of features in newer security devices and firewalls to constrain this entire environment into a proper, yet segmented, layered sandbox. In our first implementation, we used separate VLANs in tagged networks within the virtualization environment. The segmentation and one-way firewall configuration enabled us to keep our networks completely separate

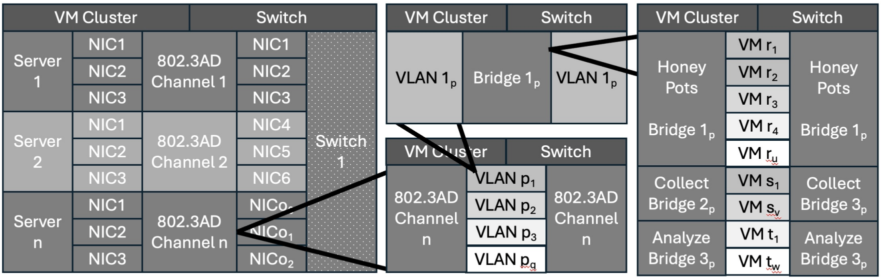


Figure 4: **Physical to Logical Design for Cluster to Network Traffic**

from our production environments and to segment them adequately from each other.

Our prototype architecture used VLANs on a used Cisco switch for network segmentation. Alternatives could include using multiple switches; however, there are significant benefits to using VLANs. These include reduced electricity consumption, the flexibility of bonded interfaces, and ease in transitioning devices between networks through capabilities in our virtualization cluster. Like our server, our switch was retired from production and used instead of being allocated to surplus. The cost of using this legacy equipment amounted to thousands of dollars in savings.

We used one NIC on the server to act as a segmented network interface for management. We bonded the three other NICs using IEEE 802.3ad (See Fig. 2), placed that bond on an IEEE 802.1Q VLAN (See Fig. 4), and assigned bridges (IEEE 802.1D) in our Proxmox cluster using Linux methods. OpenVSwitch would also be a suitable alternative here, offering a low-cost option. We had a choice of network hardware and opted for an older 24-port switch with a fiber optic uplink. We bonded ports using 802.3ad over port channels and layered VLANs on top of the port channels as trunks.

We configured virtual private networks (VPNs) in our first deployment to allow external access to our management networks. Our honeypot method was an implementation of a low-interaction honeypot strategy. We configured our honeypot VLANs to disallow egress traffic.

In our second deployment, access to management aspects of the honeypot network was only available from the LAN environment. Our third deployment employed two sites, separating the public elements and the lab. External road warriors were allowed access to the management side, which we extended to a site-to-site VPN topology. Honeypots were not deployed on our third deployment, but this site heavily implemented automation and orchestration to test that aspect of the model.

It was unimportant for our honeypot hosts to have access to the Internet. We created our honeypot templates in the management network, where we hand Internet access for updates and software installation, and then cloned them to the honeypot network as needed. Our revisions used a process that includes continuous integration and continuous deployment (CI/CD).

4.6 Firewall Considerations

When properly done, honeypot hosts will have no egress ability. The inability to egress to the Internet is vital for low-interaction honeypots. Medium and high-interaction honeypots will have severely QoS-constrained bandwidth abilities and will likely be highly restricted. We suggest burst-sensitive policies, which may appear to offer high bandwidth but are severely constrained when heavy network loads are applied.

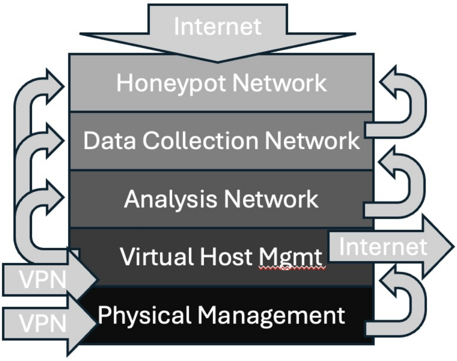


Figure 5: **One-way Data Flows**

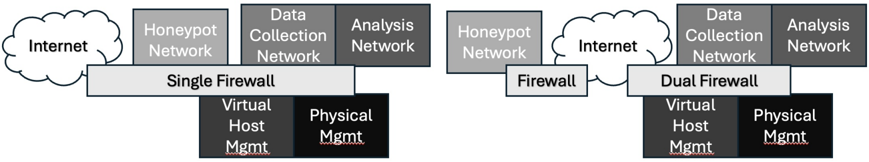


Figure 6: **Single and Dual Firewall Placement Configurations**

As seen in Fig. 5, the firewall is configured to ensure that the honeypots do not have any access to other networks. For low-interaction honeypots, configure hosts to have Internet access so that they can respond to requests from threat actors. However, honeypots should not be allowed (by the firewall) to initiate connectivity to the Internet in case they become compromised. The Data Collection Network has access to the honeypot network but should not have access to any other network. Properly done, this network does not need Internet access. Likewise, the Analysis Network only needs access to the Data Collection Network and does not need Internet Access. These one-way layers ensure abstraction and remove capability should they, too, be compromised. The Virtual Host Management Network has the most significant communication capability. It should have robust, zero-trust protections, including limited access to external Internet assets (such as updates and template provisioning).

We used our modern firewall instead of legacy equipment to segment traffic between our production network and our honeypot structures. This flexibility allows us to forward unused ports on production IPs to the low-interaction honeypots, encouraging blacklisting by threat actors. Additionally, we prohibited our low-interaction honeypots from accessing the Internet, including DNS. This enforces the low-interaction nature of these honeypot hosts.

5 Evaluation and Discussion

Our original site faithfully returns data on opportunistic actors continually under analysis. Our experience demonstrated that old hardware worked well and had resources to spare, indicating that we could add more low and high-interaction honeypots. The scope of our project did not include a collection of production systems, but our information was provided to our cybersecurity teams.

Our observations revealed a 12 percent decrease in attacks over a two-month period. We also observed a 3-5 percent recycling of attacker IPs after two months, indicating that some of the original attackers are still attempting to compromise our low-interaction systems.

6 Conclusion and Future Work

This paper outlined a practical, cost-effective honeypot architecture built on legacy hardware, open-source tools, and flexible, component-based design. By focusing on affordability, scalability, and ease of deployment, we demonstrated how organizations can build useful threat intelligence platforms without significant capital investment. We realized savings in infrastructure costs by a factor ten when using equipment that was out of warranty and was no longer fit for production use. Our architecture helps separate opportunistic attacks from targeted threats by analyzing how attackers behave across a broadened attack surface while supporting AI-based analysis with up-to-date training data. The architecture's novelty lies in its intentional use of surplus infrastructure and its reversal of conventional attack surface thinking, transforming outdated hardware into a cutting-edge resource for adversary analysis and AI training. Future work includes expanding automation capabilities, refining attacker classification using machine learning, and testing this architecture in more diverse network environments to validate and further improve its effectiveness.

References

[1] Alkan, Akin. “Analysis of Failed SSH Attempts for Intrusion Detection”. PhD thesis. University of Nevada, Reno, May 2024. URL: `https://scholarwolf.unr.edu/server/api/core/bitstreams/a85d8c5c-9fe7-4677-a0dd-6e4916faac17/content`.

[2] Bianco, David J. *Enterprise Detection & Response: The Pyramid of Pain.* Mar. 2013. URL: `https://detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html`.

[3] Hoffmann, Manuel, Nagle, Frank, and Zhou, Yanuo. “The Value of Open Source Software”. In: *SSRN Electronic Journal* (2024). ISSN: 1556-5068. DOI: `10.2139/ssrn.4693148`. URL: `https://www.ssrn.com/abstract=4693148`.

[4] IEEE. “IEEE Standard for Information Technology - Local and Metropolitan Area Networks - Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications-Aggregation of Multiple Link Segments”. In: *IEEE Std 802.3ad-2000* (June 2000), pp. 1–184. DOI: `10.1109/IEEESTD.2000.91610`. URL: `https://ieeexplore.ieee.org/document/850426`.

[5] IEEE. “IEEE Standard for Local and metropolitan area networks: Media Access Control (MAC) Bridges”. In: *IEEE Std 802.1D-2004 (Revision of IEEE Std 802.1D-1998)* (June 2004), pp. 1–281. DOI: `10.1109/IEEESTD.2004.94569`. URL: `https://ieeexplore.ieee.org/document/1309630`.

[6] IEEE. “IEEE Standard for Local and Metropolitan Area Network–Bridges and Bridged Networks”. In: *IEEE Std 802.1Q-2018 (Revision of IEEE Std 802.1Q-2014)* (July 2018), pp. 1–1993. DOI: `10.1109/IEEESTD.2018.8403927`. URL: `https://ieeexplore.ieee.org/document/8403927`.

[7] IPClear. *IPClear.* URL: `https://cic.edu/resource/ipclear/`.

[8] Kanthla, Arjun Reddy. *Network Performance Improvement for Cloud Computing using Jumbo Frames.* 2014. URL: `https://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-143806`.

[9] Kocaogullar, Yekta et al. "Hunting High or Low: Evaluating the Effectiveness of High-Interaction and Low-Interaction Honeypots". In: Accepted: 2022-08-30. Copenhagen, Denmark, Aug. 2022, pp. 15–31. URL: `https://kar.kent.ac.uk/102122/`.

[10] Lekkala, Chandrakanth. "Automating Infrastructure Management with Terraform: Strategies and Impact on Business Efficiency". In: *European Journal of Advances in Engineering and Technology* 9.11 (Nov. 2022), pp. 82–88. ISSN: 2394-658X. DOI: `10.5281/zenodo.12737493`. URL: `https://zenodo.org/records/12737493`.

[11] Lockheed Martin. *Cyber Kill Chain.* URL: `https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html` (visited on 07/31/2025).

[12] Miura, S. et al. "Low-cost high-bandwidth tree network for PC clusters based on tagged-VLAN technology". In: *8th International Symposium on Parallel Architectures,Algorithms and Networks (ISPAN'05).* ISSN: 2375-527X. Dec. 2005, 8 pp.–. DOI: `10.1109/ISPAN.2005.57`. URL: `https://ieeexplore.ieee.org/abstract/document/1575810`.

[13] Najafabadi, Maryam M. et al. "Machine Learning for Detecting Brute Force Attacks at the Network Level". In: *2014 IEEE International Conference on Bioinformatics and Bioengineering* (Nov. 2014). Conference Name: 2014 IEEE International Conference on Bioinformatics and Bioengineering (BIBE) ISBN: 9781479975020 Place: Boca Raton, FL, USA Publisher: IEEE, pp. 379–385. DOI: `10.1109/BIBE.2014.73`. URL: `http://ieeexplore.ieee.org/document/7033609/`.

[14] Norris, Donald F. et al. "Cybersecurity at the Grassroots: American Local Governments and the Challenges of Internet Security". In: *Journal of Homeland Security and Emergency Management* 15.3 (Sept. 2018). Publisher: De Gruyter. ISSN: 1547-7355. DOI: `10.1515/jhsem-2017-0048`. URL: `https://doi.org/10.1515/jhsem-2017-0048`.

[15] Oleksiuk, Vasyl P. and Oleksiuk, Olesia R. "The practice of developing the academic cloud using the Proxmox VE platform". In: *Educational Technology Quarterly* 2021.4 (Dec. 2021). Number: 4, pp. 605–616. ISSN: 2831-5332. DOI: `10.55056/etq.36`. URL: `https://acnsci.org/journal/index.php/etq/article/view/36`.

[16] Palmer, Danny. *Bosses are reluctant to spend money on cybersecurity. Then they get hacked.* Dec. 2021. URL: `https://www.zdnet.com/article/too-many-bosses-are-reluctant-to-spend-money-on-cybersecurity-then-they-get-hacked/`.

[17] Priya, V. S. Devi and Chakkaravarthy, S. Sibi. "Containerized cloud-based honeypot deception for tracking attackers". In: *Scientific Reports* 13.1 (Jan. 2023). Publisher: Nature Publishing Group, p. 1437. ISSN: 2045-2322. DOI: `10.1038/s41598-023-28613-0`. URL: `https://www.nature.com/articles/s41598-023-28613-0`.

[18] Sharma, Sachin, Agarwal, Piyush, and Tyagi, Ranu. "High Level Cloud Architecture for Automated Deployment System Using Terraform". In: *2023 Global Conference on Information Technologies and Communications (GCITC)*. Dec. 2023, pp. 1–6. DOI: `10.1109/GCITC60406.2023.10425997`. URL: `https://ieeexplore.ieee.org/abstract/document/10425997`.

[19] Theisen, Christopher et al. "Attack surface definitions: A systematic literature review". In: *Information and Software Technology* 104 (Dec. 2018), pp. 94–103. ISSN: 0950-5849. DOI: `10.1016/j.infsof.2018.07.008`. URL: `https://doi.org/10.1016/j.infsof.2018.07.008`.

[20] Ulven, Joachim Bjørge and Wangen, Gaute. "A Systematic Review of Cybersecurity Risks in Higher Education". In: *Future Internet* 13.2 (Feb. 2021). Number: 2 Publisher: Multidisciplinary Digital Publishing Institute, p. 39. ISSN: 1999-5903. DOI: `10.3390/fi13020039`. URL: `https://doi.org/10.3390/fi13020039`.

[21] Winter, Robert et al. "Ethernet Jumbo Frames". In: *Ethernet Alliance* (2009). URL: `https://ethernetalliance.org/wp-content/uploads/2011/10/EA-Ethernet-Jumbo-Frames-v0-1.pdf`.

From Confusion to Clarity in Introductory Computer Science through Intelligent Discussion Board Systems*

Tirth Shah[1], Saloni Tilekar[1], Kalpit Sharma[1]
Abbas Attarwala, Ph.D.[1], Ed Lindoo, Ph.D.[2]
[1]Computer Science Department
California State University
Chico, CA, 95973
`tshah@csuchico.edu`, `sbtilekar@csuchico.edu`, `ksharma12@csuchico.edu`
`aattarwala@csuchico.edu`
[2]Computer Information Systems
Anderson School of Business
Regis University
Denver, CO, 80221
`elindoo@regis.edu`

Abstract

First and second year computer science students often struggle to connect theoretical concepts with practical applications, a challenge that becomes particularly evident during online discussions. Although these discussions are valuable for peer interaction, confusion can arise when students cannot find a solution on their own and instructor responses are delayed, especially around foundational topics like data structures, algorithms, and core programming concepts. This paper presents an intelligent discussion board system designed specifically for introductory CS courses that addresses this challenge through AI-powered just-in-time

---

*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.

learning support. Our system allows students to engage in peer discussions while providing access to a specialized AI assistant that can be triggered using “Explain” prompts when confusion arises. The AI assistant uses retrieval-augmented generation (RAG) to pull relevant content from instructor-uploaded course materials, including textbook chapters, lecture slides, and supplementary resources, ensuring that responses are grounded in official course content. The system does not provide direct solutions to programming problems. Instead, it acts as a tutor by offering simplified explanations, generating pseudocode scaffolds, and providing step-by-step conceptual breakdowns that guide students toward understanding while maintaining academic integrity. We present the system architecture and highlight key features, such as the content retrieval mechanisms and the design of tutoring-focused AI responses. Our work contributes to the growing field of AI-enhanced educational technology by demonstrating how course-specific content retrieval can support student learning without compromising educational objectives.

1 Introduction

Discussion boards have emerged as fundamental tools for collaborative learning in CS education, enabling students to engage with course material through peer interaction and instructor guidance. Previous research has demonstrated their pedagogical value, showing that students often use online forums such as Piazza not only for logistical questions but also to support problem-solving and reasoning [13]. Platforms like Piazza are now widely used to facilitate structured question-and-answer exchanges between students and instructors. These platforms support active learning by encouraging students to articulate their understanding, seek clarification on complex topics, and learn from the perspectives of their peers’. However, existing discussion board systems face significant limitations in providing immediate contextual support when students encounter confusion, particularly outside of regular class hours or instructor office hours, when instructors may not be available to respond promptly. Research has shown that in the absence of timely instructor feedback, students can experience confusion or propagate misconceptions, underscoring the need for more responsive support mechanisms in asynchronous discussions [16]. Although students frequently use general-purpose AI tools such as ChatGPT when human guidance is unavailable, these tools function independently of course and context-specific information. This disconnect can lead to misleading or inconsistent explanations, especially in domains that require precision and structured conceptual development. AI-generated responses risk introducing misinformation through hallucinations and confusing outputs, potentially leading to serious negative consequences when users internalize and spread inaccurate information [17].

This paper introduces ChatBook, an intelligent discussion board system designed to enhance student engagement through AI-assisted dialogue grounded in course-specific materials. Originally conceived to support textbook-based conversations, ChatBook also accommodates a broad range of instructional content typically shared in PDF format, including lecture slides, supplementary handouts, and publicly available textbooks. Aimed at first and second-year CS courses, the system bridges the gap between asynchronous discussion and timely, contextualized support by integrating RAG to ensure all AI responses are based on instructor-curated resources. When students encounter confusion and invoke the AI using 'Explain' prompts, ChatBook retrieves relevant passages and delivers simplified explanations, or conceptual scaffolding—without directly solving problems—thereby upholding academic integrity while supporting understanding. Another feature in ChatBook is the "I am still confused" feature that analyzes the previous five messages in a discussion thread to provide contextually aware follow-up responses, supporting iterative learning that builds over time and also may generate pseudocode if necessary.

Although students can access the AI assistant at any time, instructors are encouraged to adopt a discussion-first model that invites peer engagement before AI support. This approach reflects established pedagogical principles, particularly social constructivism and instructional scaffolding [14], in which learners co-construct knowledge before receiving individualized assistance. By constraining AI output to vetted course materials, ChatBook promotes alignment with instructor pedagogy and mitigates the cognitive dissonance that can occur with general-purpose AI tools.

For instructors managing large-enrollment CS courses, ChatBook offers practical benefits: it offloads routine explanation tasks, provides consistent and pedagogically aligned support, and generates discussion logs that surface common misconceptions and learning patterns—enabling data-informed teaching adjustments. Although developed with CS courses in mind, the architecture of the system is broadly applicable to any discipline where students benefit from context-based AI support, from literary analysis to historical interpretation. Students benefit from immediate and reliable assistance that complements peer collaboration and instructor guidance. In traditional classroom settings, ChatBook can enable students to revisit concepts, prepare thoughtful questions, and extend the discussion beyond the time constraints of live sessions. In classes taught online, whether synchronously or asynchronously, the system can promote a sense of connection, helping to counteract isolation that can undermine learning and engagement. By integrating scalable, course-aligned AI assistance into collaborative learning environments, ChatBook addresses a key gap in CS education. It ensures that technology supports—not replaces—human interaction, preserving the social and instructional dynamics essential to effective learning.

2 Literature Review

Previous work in digital learning environments has examined a wide range of tools and frameworks designed to support student engagement, collaborative learning, and real-time assistance. These include live chat tools for synchronous support, educational chatbots for project-based learning, curriculum-aligned AI systems, and RAG systems. The work in this paper builds on by offering a pedagogically grounded, course-specific, and discussion-integrated AI system—ChatBook—that supports just-in-time learning assistance without replacing peer collaboration or instructor authority.

Studies have consistently found that immediacy and synchronicity in communication tools can improve students' sense of support and connection—particularly in online learning contexts. For example, [2] found that the live chat features provided online and blended learners with a valuable channel for just-in-time academic support, and students particularly appreciated instant responses during assessment periods and when immediate help was needed. Similarly, [4] demonstrated that the Google Hangouts improved student satisfaction with online teamwork and helped students develop a sense of community in virtual settings, highlighting the value of real-time communication in online learning environments. Although ChatBook shares the goal of timely support, it differs in its asynchronous architecture, offering immediate contextualized AI assistance even when live human interaction is unavailable. Unlike live chat or video conferencing, ChatBook integrates directly into asynchronous discussion threads, offering responsiveness without requiring coordination across time zones or schedules.

Several studies have explored the integration of chatbots and AI-driven conversational agents in educational settings. [6] and [5] developed chatbot systems aimed at supporting collaborative learning, offering scaffolding and task-related feedback. Although these systems demonstrated positive impacts on teamwork and cognitive engagement, they did not consistently improve learning outcomes, emotional engagement, or motivation—partly due to limited contextual awareness and mistimed interventions. Similarly, [9] proposed the CHAT-ACTS framework to integrate chatbots with self-regulated learning and active learning models. Their work provides pedagogical guidance for chatbot integration, but focuses on theoretical principles rather than implementation with specific course materials. ChatBook builds on these efforts by incorporating RAG and aligning all AI output with instructor-provided course materials, such as textbooks and lecture slides. This ensures epistemic consistency and mitigates the risk of conflicting or hallucinated explanations—a limitation in many generative AI tools [17].

[7] introduced EduBot, a curriculum-aligned language learning chatbot fine-tuned. EduBot outperformed ChatGPT in maintaining topical relevance and vocabulary alignment, demonstrating the benefits of tailoring AI systems to specific curricular content. ChatBook extends this principle by embedding RAG-based retrieval across a wide variety of instructional formats (not just language learning), including PDFs of textbooks, lecture slides, and handouts. Unlike EduBot, which is domain-specific, ChatBook is designed as a flexible cross-disciplinary platform for contextualized discussion support.

Several systems have addressed collaborative learning through technical affordances and social interaction modeling. [11] contrasted forums and instant messaging tools in Computer-supported collaborative learning environments, showing that discussion forums promote more knowledge-building interactions while instant messaging facilitates social interaction. [8] incorporated social networking awareness into a mobile assessment platform, finding that peer visibility and social influence enhanced sustained engagement. [12] developed PeerTalk to enable synchronous collaboration and AI support in MOOCs. Although these systems underscore the importance of peer-to-peer interaction, none explicitly embed AI assistance within student-led discussion workflows. ChatBook is designed not only to preserve collaborative learning but to amplify it by offering just-in-time scaffolding within the peer discourse itself, optionally triggered via 'Explain' or 'I am still confused' prompts. This model complements discussion rather than displacing it, aligning with social constructivist pedagogy [14].

The work most technically aligned with ChatBook is ChatEd [15],which integrates large language models with retrieval-based chatbots to support contextual accuracy in student queries. ChatEd focuses on LMS integration and response accuracy, outperforming ChatGPT on course-specific tasks. However, ChatEd functions more as an on-demand Q&A bot, whereas ChatBook is embedded within the ongoing discourse of a discussion board. The ChatBook iterative response mechanism (e.g., the feature "I am still confused") represents a novel contribution to the RAG+AI design space, allowing AI to consider previous context in shaping its reply.

3 System Design

Our system is built as a full-stack web application with a modular architecture that separates content processing, real-time communication, AI assistance, and user interface components. This design ensures scalability and maintainability while supporting the complex workflows required for intelligent educational discussions. The backend foundation relies on Flask with Flask-SocketIO to enable real-time bidirectional communication, supporting both synchronous discussions and immediate AI assistance delivery. Authentication and security are managed through Auth0 for identity management and JWT tokens for session control, to ensure that students can only access their course-specific content.

Content processing begins when instructors upload PDF course materials to the system. PyPDF2 extracts raw text and metadata including page numbers and chapter headings, which are then stored in MongoDB Atlas. We chose MongoDB Atlas for its flexibility in handling both structured course metadata and unstructured textual content as vector embeddings that vary significantly across different educational materials. A critical design decision involves maintaining dual storage approaches to address competing technical requirements. Raw text content is processed and indexed for AI operations, as language models work most effectively with plain text for semantic understanding. However, students require properly formatted, page-specific content for effective learning, so original PDFs are preserved and rendered using PDF.js. This dual approach allows the system to display exact page references with proper formatting, figures, and layout when presenting AI-generated responses to students. The core intelligence emerges through our RAG pipeline, which processes student "Explain" queries by using OpenAI API embeddings for semantic similarity matching between student questions and stored course content. Once relevant content sections are identified, the OpenAI API generates contextually appropriate responses, with specific prompts to reference page numbers from course materials. For the "I am still confused" feature, we use Google Gemini API for analyzing the last five discussion messages to provide increasingly targeted explanations such as generating pseudocode and maintaining conversation context throughout the learning process. The frontend implementation uses HTML5/CSS3 with Bootstrap for responsive design across different devices. Socket.IO manages real-time message delivery to ensure that students see new discussion posts and AI responses immediately, while Marked.js handles markdown rendering and Highlight.js provides syntax highlighting for code snippets. The seamless PDF.js integration allows students to view specific referenced pages without leaving the discussion context, maintaining optimal learning flow from AI explanation to source material review. Security and performance considerations include HTTPS communication, CORS policies, and CSRF protection for secure operations. Performance optimization utilizes Gzip compression and browser caching, while lazy loading efficiently handles large PDF documents and extensive discussion threads that could otherwise impact system responsiveness. The deployment strategy supports both development and production environments through virtual environments and LocalTunnel for testing, while production deployment utilizes Gunicorn with Nginx as a reverse proxy for load balancing and static file serving. This architecture successfully enables multiple concurrent discussions with immediate, contextually relevant AI assistance grounded in instructor-curated content, scaling effectively to support large courses while maintaining educational effectiveness.

4 Features and Capabilities

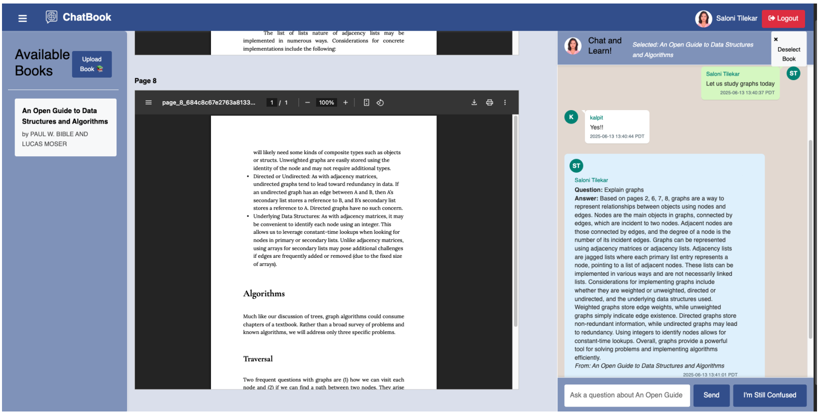
Although formal user testing and deployment in actual courses remain part of our future work—including planned randomized controlled trials comparing our system against Piazza following IRB approval—this section demonstrates the current capabilities and pedagogical value of our application through practical examples of student-AI interactions.

The core functionality of ChatBook centers on its ability to respond to student queries by retrieving relevant course content and generating contextualized, pedagogically aligned explanations. This feature addresses a critical challenge in asynchronous learning environments: providing timely, content-specific help when human assistance may be delayed or unavailable [2, 4]. Immediate feedback on code comprehension can support learning in introductory programming courses, where students frequently struggle with algorithmic abstraction [10].

In Figure 1, we show a typical use case in which two students engage in a discussion about graph theory. When a student invokes the AI with the phrase "Explain graphs," the system retrieves specific page references (2, 6, 7, and 8) from the uploaded textbook and displays them alongside a tailored explanation. This design aligns conversational practice with curriculum content, ensuring students engage with topics and vocabulary from their textbooks [7].

ChatBook supports seamless transitions in discussion dynamics as students join or leave an ongoing conversation. New participants can review the full discussion history to quickly gain context. In the example shown in Figure 2, while the roster of participants changes, an existing student, Saloni, requests help with "weighted graphs." Our system responds by retrieving relevant material and providing a focused explanation. This design provides adaptive scaffolding that responds to students' evolving needs during collaborative learning [5, 1].

A distinctive pedagogical benefit lies in the system's support for page-specific content retrieval. Figure 3 shows how our system can focus entirely on the content of a single page (e.g., page 6) to provide fine-grained and localized explanations. This mirrors the instructional value of worked examples in programming education and facilitates incremental comprehension—a critical skill in early CS learning [10]. By allowing students to query and reflect on a specific page's content, the system turns passive reading into an interactive process.


ChatBook
Saloni Tilekar
Logout
Available Books
Upload Book
An Open Guide to Data Structures and Algorithms
by PAUL W. BIBLE AND LUCAS MOSER
Page 8
page_8_684c8c67e2763a8133...
1 / 1
100%
Chat and Learn!
Selected: An Open Guide to Data Structures and Algorithms
Deselect Book
Ask a question about An Open Guide
Send
I'm Still Confused

The list of lists nature of adjacency lists may be implemented in numerous ways. Considerations for concrete implementations include the following:

will likely need some kinds of composite types such as objects or structs. Unweighted graphs are easily stored using the identity of the node and may not require additional types.

- Directed or Undirected: As with adjacency matrices, undirected graphs tend to lead toward redundancy in data. If an undirected graph has an edge between A and B, then A's secondary list stores a reference to B, and B's secondary list stores a reference to A. Directed graphs have no such concern.
- Underlying Data Structures: As with adjacency matrices, it may be convenient to identify each node using an integer. This allows us to leverage constant-time lookups when looking for nodes in primary or secondary lists. Unlike adjacency matrices, using arrays for secondary lists may pose additional challenges if edges are frequently added or removed (due to the fixed size of arrays).

Algorithms

Much like our discussion of trees, graph algorithms could consume chapters of a textbook. Rather than a broad survey of problems and known algorithms, we will address only three specific problems.

Traversal

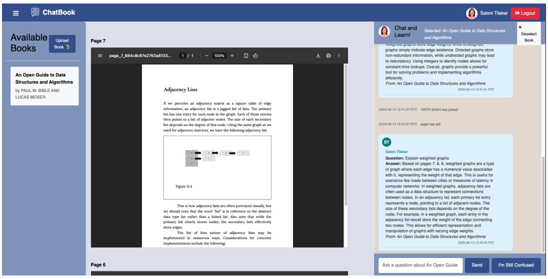
Two frequent questions with graphs are (1) how we can visit each node and (2) if we can find a path between two nodes. They arise

Saloni Tilekar
Let us study graphs today
2025-06-13 13:40:37 PDT

kalpit
Yes!!
2025-06-13 13:40:44 PDT

Saloni Tilekar
Question: Explain graphs
Answer: Based on pages 2, 6, 7, 8, graphs are a way to represent relationships between objects using nodes and edges. Nodes are the main objects in graphs, connected by edges, which are incident to two nodes. Adjacent nodes are those connected by edges, and the degree of a node is the number of its incident edges. Graphs can be represented using adjacency matrices or adjacency lists. Adjacency lists are jagged lists where each primary list entry represents a node, pointing to a list of adjacent nodes. These lists can be implemented in various ways and are not necessarily linked lists. Considerations for implementing graphs include whether they are weighted or unweighted, directed or undirected, and the underlying data structures used. Weighted graphs store edge weights, while unweighted graphs simply indicate edge existence. Directed graphs store non-redundant information, while undirected graphs may lead to redundancy. Using integers to identify nodes allows for constant-time lookups. Overall, graphs provide a powerful tool for solving problems and implementing algorithms efficiently.
From: An Open Guide to Data Structures and Algorithms
2025-06-13 13:41:01 PDT

Figure 1: Students engaging in discussion with AI-triggered explanation of graphs, showing relevant textbook pages and AI-generated summary.


ChatBook
Saloni Tilekar
Logout
Available Books
Upload Book
An Open Guide to Data Structures and Algorithms
by PAUL W. BIBLE AND LUCAS MOSER
Page 7
page_7_684c8c67e2763a8133...
1 / 1
100%
Figure 11.4
Page 6
Chat and Learn!
Selected: An Open Guide to Data Structures and Algorithms
Deselect Book
Ask a question about An Open Guide
Send
I'm Still Confused

Adjacency Lists

If we perceive an adjacency matrix as a square table of edge information, an adjacency list is a jagged list of lists. The primary list has one entry for each node in the graph. Each of those entries then points to a list of adjacent nodes. The size of each secondary list depends on the degree of that node. Using the same graph as we used for adjacency matrices, we have the following adjacency list.

This is how adjacency lists are often portrayed visually, but we should note that the word "list" is in reference to the abstract data type list rather than a linked list. Also note that while the primary list clearly stores nodes, the secondary lists effectively store edges.

The list of lists nature of adjacency lists may be implemented in numerous ways. Considerations for concrete implementations include the following:

graphs simply indicate edge existence. Directed graphs store non-redundant information, while undirected graphs may lead to redundancy. Using integers to identify nodes allows for constant-time lookups. Overall, graphs provide a powerful tool for solving problems and implementing algorithms efficiently.
From: An Open Guide to Data Structures and Algorithms
2025-06-13 13:41:01 PDT

[2025-06-13 13:41:23 PDT] TIRTH SHAH has joined

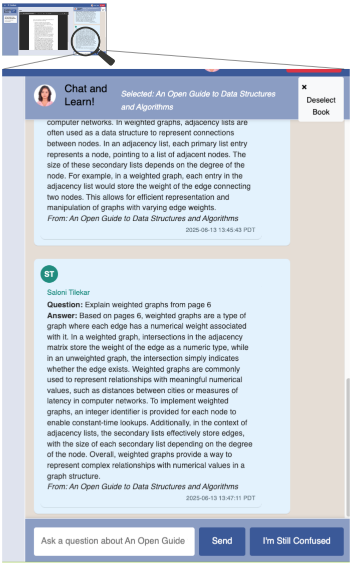
[2025-06-13 13:42:35 PDT] kalpit has left

Saloni Tilekar
Question: Explain weighted graphs
Answer: Based on pages 7, 6, 8, weighted graphs are a type of graph where each edge has a numerical value associated with it, representing the weight of that edge. This is useful for scenarios like roads between cities or measures of latency in computer networks. In weighted graphs, adjacency lists are often used as a data structure to represent connections between nodes. In an adjacency list, each primary list entry represents a node, pointing to a list of adjacent nodes. The size of these secondary lists depends on the degree of the node. For example, in a weighted graph, each entry in the adjacency list would store the weight of the edge connecting two nodes. This allows for efficient representation and manipulation of graphs with varying edge weights.
From: An Open Guide to Data Structures and Algorithms
2025-06-13 13:45:43 PDT

Figure 2: Dynamic discussion showing student participation changes and AI response to weighted graph query with specific page references.

Such targeted engagement exemplifies active learning, which has been shown to enhance motivation and learning outcomes in introductory programming courses [3].

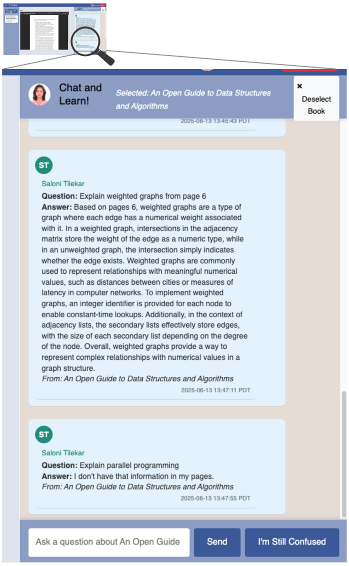
Figure 4 shows the system's content alignment enforcement feature. When a student asks about a topic not covered in the uploaded instructional materials (e.g., parallel programming), our system explicitly states that the topic lies outside its current scope. This maintains alignment with instructor-provided content, promoting epistemic fidelity and reducing the risk of misinformation—a concern noted in previous evaluations of generative AI in education [15].



computer networks. In weighted graphs, adjacency lists are often used as a data structure to represent connections between nodes. In an adjacency list, each primary list entry represents a node, pointing to a list of adjacent nodes. The size of these secondary lists depends on the degree of the node. For example, in a weighted graph, each entry in the adjacency list would store the weight of the edge connecting two nodes. This allows for efficient representation and manipulation of graphs with varying edge weights.
From: An Open Guide to Data Structures and Algorithms
2025-06-13 13:45:43 PDT

ST
Saloni Tilekar
Question: Explain weighted graphs from page 6
Answer: Based on pages 6, weighted graphs are a type of graph where each edge has a numerical weight associated with it. In a weighted graph, intersections in the adjacency matrix store the weight of the edge as a numeric type, while in an unweighted graph, the intersection simply indicates whether the edge exists. Weighted graphs are commonly used to represent relationships with meaningful numerical values, such as distances between cities or measures of latency in computer networks. To implement weighted graphs, an integer identifier is provided for each node to enable constant-time lookups. Additionally, in the context of adjacency lists, the secondary lists effectively store edges, with the size of each secondary list depending on the degree of the node. Overall, weighted graphs provide a way to represent complex relationships with numerical values in a graph structure.
From: An Open Guide to Data Structures and Algorithms
2025-06-13 13:47:11 PDT

Figure 3: Student requesting detailed explanation of weighted graphs from specific page 6, showing AI's ability to focus on particular content sections.



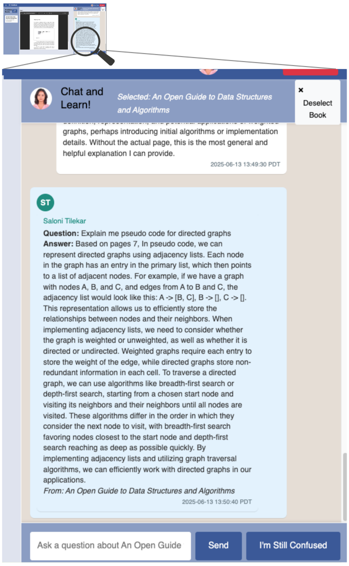
2025-06-13 13:45:43 PDT

ST
Saloni Tilekar
Question: Explain weighted graphs from page 6
Answer: Based on pages 6, weighted graphs are a type of graph where each edge has a numerical weight associated with it. In a weighted graph, intersections in the adjacency matrix store the weight of the edge as a numeric type, while in an unweighted graph, the intersection simply indicates whether the edge exists. Weighted graphs are commonly used to represent relationships with meaningful numerical values, such as distances between cities or measures of latency in computer networks. To implement weighted graphs, an integer identifier is provided for each node to enable constant-time lookups. Additionally, in the context of adjacency lists, the secondary lists effectively store edges, with the size of each secondary list depending on the degree of the node. Overall, weighted graphs provide a way to represent complex relationships with numerical values in a graph structure.
From: An Open Guide to Data Structures and Algorithms
2025-06-13 13:47:11 PDT

ST
Saloni Tilekar
Question: Explain parallel programming
Answer: I don't have that information in my pages.
From: An Open Guide to Data Structures and Algorithms
2025-06-13 13:47:55 PDT

Figure 4: AI appropriately responding that parallel programming information is not available in the course content, maintaining course-specific boundaries.

In Figure 5, our system generates pseudocode in response to a query about directed graphs. Because the relevant pages only describe adjacency lists, the AI adapts its response accordingly—demonstrating a key instructional design goal: help students generalize from partial knowledge without overreaching the curriculum boundaries.


Chat and Learn!
Selected: An Open Guide to Data Structures and Algorithms
×
Deselect Book
ST
Ask a question about An Open Guide
Send
I'm Still Confused

graphs, perhaps introducing initial algorithms or implementation details. Without the actual page, this is the most general and helpful explanation I can provide.

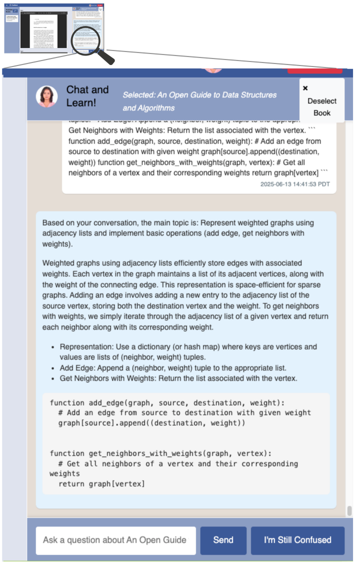
2025-06-13 13:49:30 PDT

Saloni Tilekar

Question: Explain me pseudo code for directed graphs
Answer: Based on pages 7, In pseudo code, we can represent directed graphs using adjacency lists. Each node in the graph has an entry in the primary list, which then points to a list of adjacent nodes. For example, if we have a graph with nodes A, B, and C, and edges from A to B and C, the adjacency list would look like this: A -> [B, C], B -> [], C -> []. This representation allows us to efficiently store the relationships between nodes and their neighbors. When implementing adjacency lists, we need to consider whether the graph is weighted or unweighted, as well as whether it is directed or undirected. Weighted graphs require each entry to store the weight of the edge, while directed graphs store non-redundant information in each cell. To traverse a directed graph, we can use algorithms like breadth-first search or depth-first search, starting from a chosen start node and visiting its neighbors and their neighbors until all nodes are visited. These algorithms differ in the order in which they consider the next node to visit, with breadth-first search favoring nodes closest to the start node and depth-first search reaching as deep as possible quickly. By implementing adjacency lists and utilizing graph traversal algorithms, we can efficiently work with directed graphs in our applications.
From: An Open Guide to Data Structures and Algorithms

2025-06-13 13:50:40 PDT

Figure 5: AI generating pseudocode for directed graphs based on adjacency list representation from page 7 of the textbook.


Chat and Learn!
Selected: An Open Guide to Data Structures and Algorithms
×
Deselect Book
Ask a question about An Open Guide
Send
I'm Still Confused

Get Neighbors with Weights: Return the list associated with the vertex. ``` function add_edge(graph, source, destination, weight): # Add an edge from source to destination with given weight graph[source].append((destination, weight)) function get_neighbors_with_weights(graph, vertex): # Get all neighbors of a vertex and their corresponding weights return graph[vertex] ```

2025-06-13 14:41:53 PDT

Based on your conversation, the main topic is: Represent weighted graphs using adjacency lists and implement basic operations (add edge, get neighbors with weights).

Weighted graphs using adjacency lists efficiently store edges with associated weights. Each vertex in the graph maintains a list of its adjacent vertices, along with the weight of the connecting edge. This representation is space-efficient for sparse graphs. Adding an edge involves adding a new entry to the adjacency list of the source vertex, storing both the destination vertex and the weight. To get neighbors with weights, we simply iterate through the adjacency list of a given vertex and return each neighbor along with its corresponding weight.

- Representation: Use a dictionary (or hash map) where keys are vertices and values are lists of (neighbor, weight) tuples.
- Add Edge: Append a (neighbor, weight) tuple to the appropriate list.
- Get Neighbors with Weights: Return the list associated with the vertex.

```
function add_edge(graph, source, destination, weight):
  # Add an edge from source to destination with given weight
  graph[source].append((destination, weight))


function get_neighbors_with_weights(graph, vertex):
  # Get all neighbors of a vertex and their corresponding
weights
  return graph[vertex]
```

Figure 6: Contextual response when student clicks "I am still confused," incorporating both course content and discussion thread context.

The "I am still confused" feature, shown in Figure 6, allows students to solicit more nuanced help. By analyzing the five previous messages in the discussion, the AI offers context-sensitive elaborations. This strategy embodies the principles of effective question-asking during problem-solving [1] and provides synchronous support that students find valuable for just-in-time help [2]. Together, these capabilities position ChatBook as a scalable learning companion that supports active, self-regulated, and peer-mediated learning. Its design reflects effective pedagogical principles: students benefit the most when instruction is interactive, provides real-time support, and encourages active problem-solving [3, 1]. Rather than replacing peer or instructor feedback, ChatBook complements it by ensuring that students have continuous access to support that reinforces instructor-defined goals and curriculum coverage.

5 Conclusion

This paper presents an intelligent discussion board system designed to address the growing need for AI-enhanced educational tools that maintain pedagogical integrity while providing immediate learning support. By integrating RAG with instructor-curated content, our system bridges the gap between collaborative peer learning and contextual AI assistance in introductory CS education. The key innovation of our system lies in its constraint-based approach to AI integration. Rather than relying on general-purpose AI tools that may provide inconsistent or inappropriate guidance, our system ensures that all AI responses are grounded in official course materials uploaded by instructors.

Acknowledgment

We gratefully acknowledge the use of OpenAI's ChatGPT for proofreading, grammatical checks, and other text editing tasks.

References

[1] Kristy Elizabeth Boyer et al. "Principles of asking effective questions during student problem solving". In: *Proceedings of the 41st ACM technical symposium on Computer science education*. 2010, pp. 460–464.

[2] Jaclyn Broadbent and Jason Lodge. "Use of live chat in higher education to support self-regulated help seeking behaviours: a comparison of online and blended learner perspectives". In: *International journal of educational technology in higher education* 18.1 (2021), p. 17.

[3] Ricardo Caceffo, Guilherme Gama, and Rodolfo Azevedo. "Exploring active learning approaches to computer science classes". In: *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*. 2018, pp. 922–927.

[4] Jinxia He and Xiaoxia Huang. "Collaborative online teamwork: Exploring students' satisfaction and attitudes with google hangouts as a supplementary communication tool". In: *Journal of Research on Technology in Education* 49.3-4 (2017), pp. 149–160.

[5] Wanqing Hu, Jirong Tian, and Yanyan Li. "Enhancing student engagement in online collaborative writing through a generative AI-based conversational agent". In: *The Internet and Higher Education* 65 (2025), p. 100979.

[6] Jeya Amantha Kumar. "Educational chatbots for project-based learning: investigating learning outcomes for a team-based design course". In: *International journal of educational technology in higher education* 18.1 (2021), p. 65.

[7] Yu Li et al. “Curriculum-driven edubot: A framework for developing language learning chatbots through synthesizing conversational data”. In: *arXiv preprint arXiv:2309.16804* (2023).

[8] Jian-Wei Lin, Hsieh-Hong Huang, and Chia-Wen Tsai. “Mobile collaborative learning with social network awareness”. In: *Journal of Computer Assisted Learning* 39.5 (2023), pp. 1690–1700.

[9] Michael Pin-Chuan Lin and Daniel Chang. “CHAT-ACTS: A pedagogical framework for personalized chatbot to enhance active learning and self-regulated learning”. In: *Computers and Education: Artificial Intelligence* 5 (2023), p. 100167.

[10] Leigh Ann Sudol-DeLyser, Mark Stehlik, and Sharon Carver. “Code comprehension problems as learning events”. In: *Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education*. 2012, pp. 81–86.

[11] Zhong Sun et al. “A tale of two communication tools: Discussion-forum and mobile instant-messaging apps in collaborative learning”. In: *British Journal of Educational Technology* 49.2 (2018), pp. 248–261.

[12] Stergios Tegos, Apostolos Mavridis, and Stavros Demetriadis. “Agent-supported peer collaboration in MOOCs”. In: *Frontiers in Artificial Intelligence* 4 (2021), p. 710856.

[13] Mickey Vellukunnel et al. “Deconstructing the Discussion Forum: Student Questions and Computer Science Learning”. In: *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*. SIGCSE ’17. Seattle, Washington, USA: Association for Computing Machinery, 2017, pp. 603–608. ISBN: 9781450346986. DOI: `10.1145/3017680.3017745`. URL: `https://doi.org/10.1145/3017680.3017745`.

[14] Lev S Vygotsky. *Mind in society (M. Cole, V. John-Steiner, S. Scribner, & E. Souberman, Eds.)* 1978.

[15] Kevin Wang, Jason Ramos, and Ramon Lawrence. “ChatEd: a chatbot leveraging ChatGPT for an enhanced learning experience in higher education”. In: *arXiv preprint arXiv:2401.00052* (2023).

[16] TII Washington et al. “Today’s discussion boards: the good, the bad, and the ugly”. In: *The Online Journal of New Horizons in Education* 9.3 (2019), p. 219.

[17] Huiwen Xiong. “Research on confusing responses based on ChatGPT”. In: *Applied and Computational Engineering* 57 (2024), pp. 90–97.

Voice-Driven Conversational Mixed Reality for Linear Algebra in Computer Science Education Using an AI Avatar Tutor*

Sheldon Henriques[1], Abbas Attarwala, Ph.D.[1],
and Ed Lindoo, Ph.D.[2]
[1]Computer Science Department
California State University
Chico, CA, 95973
svhenriques@csuchico.edu, aattarwala@csuchico.edu
[2]Computer Information Systems
Anderson School of Business
Regis University
Denver, CO, 80221
elindoo@regis.edu

Abstract

This paper presents an advanced mixed reality (MR) application that integrates conversational artificial intelligence (AI) with voice-driven mathematical visualization for linear algebra education in computer science (CS) courses. Building upon previous work in MR-based mathematical learning, our system introduces a virtual Albert Einstein avatar that students can interact with through natural language commands to create, manipulate, and explore mathematical concepts in Meta Quest 3s. Students can verbally request vector and plane generation, perform vector operations, and execute step-by-step Reduced Row Echelon Form (RREF) algorithms through conversational interaction with the

---

*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.

AI avatar. Another contribution of this work is the integration of real-world astronomical visualizations, where students can observe how rotational matrices control the Earth's rotation, the Moon's rotation, and the Moon's orbital motion around Earth, providing tangible context for abstract linear transformation concepts in linear algebra. This voice-driven approach addresses different learning modalities compared to traditional hand-tracking interfaces, enabling students to focus on conceptual understanding rather than manual manipulation. The preliminary evaluation among first-year students demonstrates enhanced engagement and comprehension, particularly for auditory learners who benefit from the conversational interaction model. The system represents a significant advancement in multimodal educational technology, combining the immersive benefits of MR with the accessibility of natural language interaction and the pedagogical advantages of AI-assisted tutoring for complex mathematical concepts.

1 Introduction

Linear algebra serves as a foundational course for students of mathematics and engineering, but its abstract nature presents significant pedagogical challenges [9]. Traditional classroom instruction relies heavily on symbolic manipulation and two-dimensional representations, leaving many students struggling to develop an intuitive understanding of concepts that are inherently three-dimensional. Our recent work [4] has demonstrated the potential of MR for mathematical visualization. Our previous research shows that interactive 3D environments can effectively support linear algebra learning through hand-tracking and controller-based manipulation of vectors, planes, and geometric visualization of the RREF process. Although this initial work successfully demonstrated that MR visualization enhances comprehension, particularly for visual and kinesthetic learners, it relied primarily on manual interaction modalities that still required students to navigate complex interfaces and controller mechanisms. Building upon these findings, this paper introduces a fundamentally different approach that integrates conversational AI with voice-driven mathematical visualization. Conversational agents using natural language interactions have been shown to significantly reduce anxiety toward mathematics among students [10]. Natural language interaction mirrors the way humans naturally process and communicate mathematical ideas, allowing students to articulate their thinking, ask questions spontaneously, and receive immediate feedback in a familiar format. When students can verbally request "generate two planes that intersect on a line" or "generate three planes that intersect on a point" or "show me how this rotation matrix affects the Earth's motion," they engage with mathematics as a dialogue rather than a series of mechanical operations. This conversational approach reduces the cognitive load by eliminating the need to navigate complex interfaces, allowing students to focus entirely on conceptual understanding, a limitation identified in our previous work, where students occasionally struggled with controller sensitivity and navigation complexity. The integration of a virtual Albert Einstein avatar represents a strategic pedagogical choice that extends beyond our previous work's focus on geometric visualization. Animated pedagogical agents (PA) that utilize real-time rendering and anime-style esthetics can mitigate the uncanny valley effect, resulting in more natural and comfortable interactions, as demonstrated by significantly higher user preference ratings compared to realistic talking-head approaches [2]. Such systems show potential to increase student engagement and improve learning outcomes through adaptive, on-demand support, and personalized feedback [2]. The avatar serves not only as an interface, but as a virtual mentor who can guide students through their mathematical journey with personality and context, addressing the need for more intuitive interaction identified in our initial pilot study feedback. However, the true innovation lies in the multimodal learning experience that seamlessly combines voice interaction with the proven benefits of physical manipulation demonstrated in our previous work. After students use voice commands to generate mathematical objects through conversation with Einstein, they can transition to manipulating these objects with their hands and controllers—leveraging the successful hand-tracking capabilities established in our initial application. As students physically adjust a plane's orientation or modify a vector's direction, Einstein's avatar provides real-time feedback, verbalizing the new equations, and explaining the mathematical implications of their actions. This creates a continuous loop of conversation, visualization, and tactile manipulation that addresses multiple learning modalities simultaneously, based on our previous finding that the combination of visual and kinesthetic interaction significantly enhances student comprehension. While our previous work [4] focused on the visualization of fundamental concepts such as vector addition, span, and RREF visualization, this new application also extends into advanced linear algebra applications through real-world astronomical visualizations. By engaging multiple senses and interaction modalities simultaneously, students can construct understanding through their preferred learning channels while being exposed to alternative approaches that strengthen their overall comprehension. The result is a learning environment that transforms abstract linear algebra into an interactive, conversational, and deeply engaging experience that addresses the usability concerns identified in our initial pilot study while dramatically expanding the scope of mathematical concepts that can be effectively visualized and manipulated in MR environments.

2 Literature Review

The IllusionX system by [12] proposes integrating conversational AI with MR technologies for academic support, providing a conceptual framework for our AI Avatar Tutor approach. Although its MR components remain untested, their system architecture demonstrates how LLM can be combined with immersive interfaces, and their evaluation of LLM-based educational tasks showed marginal improvements in generating coherent course content when leveraging knowledge embedding. The affective computing capabilities of IllusionX validate the pedagogical potential of MR-based conversational interfaces to increase engagement and personalize learning in mathematical education.

The systematic review by [5] analyzing 66 studies from 2015 to 2023 demonstrates that AR significantly improves mathematical creativity, student motivation, and spatial understanding. Their findings show that AR applications promote active learning by combining physical objects with digital simulations, particularly effective for visualizing abstract mathematical concepts. These insights directly support our Avatar Tutor system, which combines MR visualization with conversational AI to address linear algebra's cognitive challenges through multimodal engagement.

[6] examines the integration of Intelligent Tutoring Systems within AR and VR environments, demonstrating that this combination creates effective immersive learning environments that support personalized education. These systems identify, monitor, and adapt to the characteristics of the students while providing tailored real-time feedback. Students in such environments demonstrate increased engagement, motivation, and improved knowledge retention, directly supporting our Avatar Tutor approach that combines conversational AI with MR visualization.

[8] compared the effectiveness of ChatGPT-generated mathematics hints with human-authored hints in a randomized study with 274 participants in four math topics (Elementary Algebra, Intermediate Algebra, College Algebra, and Statistics). Participants were assigned to one of three conditions: no hints (control), human tutor hints, or ChatGPT hints. The results showed that the ChatGPT hints produced a 17% learning gain that was statistically significant compared to the control group's 1.85% gain. The human tutor hints yielded an 11.62% gain, but this difference from the control was not statistically significant. Importantly, there was no statistically significant difference between the ChatGPT and the human tutor conditions, suggesting that ChatGPT performed similarly to human-authored help. However, quality control revealed that 32% of the initial responses of ChatGPT's contained errors. The researchers applied a self-consistency technique—generating 10 responses per problem and selecting the most common answer—which dramatically reduced error rates to nearly 0% for algebra problems and 13% for statistics problems.

These results demonstrate that LLMs can be as effective as human-generated content for mathematics instruction, supporting our Avatar Tutor's conversational AI approach.

[3] evaluated Rori, an AI-powered math tutor accessible via WhatsApp, involving approximately 500 students in Ghana. The conversational agent uses NLP and LLMs to deliver more than 500 micro-lessons with scaffolded practice questions. The results showed statistically significant higher math growth scores. This validates the effectiveness of chat-based AI tutoring in mathematics education, supporting our Avatar Tutor approach that employs conversational AI for mathematical instruction.

[7] presented an end-to-end evaluation of a Spoken Dialogue System to teach basic mathematics through play-based learning. The system deployed in real-world school settings enabled multimodal interactions including speech-based questions and manipulation of tangible objects. The study focused on improving Natural Language Understanding for interpreting mathematical intentions and provided insights into error propagation between AI components, demonstrating the practical viability of conversational AI systems for mathematics education.

[13] explore human-machine teaming paradigms within collaborative AR environments, investigating how virtual PA as Learning Companions can be integrated with human collaborators to facilitate social learning. Their study conducted three collaborative AR experiments with medical students learning to interview patients with stroke symptoms. The research demonstrates how learners can benefit from observing, interacting with and receiving feedback from both virtual pedagogical agents and human partners in AR settings, achieving significant results in self-efficacy and conceptual and procedural learning despite challenges in natural language processing technology. This work supports our Avatar Tutor approach, which similarly combines virtual pedagogical agents with immersive technology for educational enhancement.

[11] introduce Tutor CoPilot, a Human-AI approach providing real-time guidance to tutors. Their randomized controlled trial involving 900 tutors and 1,800 K-12 students found that students whose tutors had access to Tutor CoPilot were 4 percentage points more likely to master topics, with the greatest benefit for students of lower-rated tutors (9 percentage points improvement) and lower-experienced tutors (7 percentage points improvement). At $20 per tutor annually, this demonstrates how Human-AI systems can effectively scale expertise.

3 System Design

Our conversational MR application represents the integration of multiple components of AI and immersive technology, designed to create seamless voice-driven mathematical learning experiences. The system architecture combines real-time voice processing, language modeling, and immersive visualization within a unified Unity-based framework running on the Meta Quest 3 platform.

The foundation of our conversational interface relies on Meta's Wit.ai SDK, integrated through the Oculus platform, which provides voice recognition capabilities optimized for MR environments. This component handles both partial and full speech transcriptions, enabling real-time processing of student utterances as they interact with the Einstein avatar. The Wit.ai integration is particularly well-suited for our educational context, as it can effectively process mathematical terminology and commands while maintaining low latency essential for natural conversational flow. The system processes voice input through a multi-stage pipeline that begins with audio capture via the Quest 3's built-in microphones, followed by real-time speech-to-text conversion through Wit.ai's cloud-based recognition engine. This approach ensures high accuracy for mathematical vocabulary while providing the responsiveness necessary for interactive learning experiences.

At the core of our system's intelligence lies Google's Gemini API, specifically utilizing the gemini-2.5-flash-preview-04-17 model with function-calling capabilities enabled. This integration allows the Einstein avatar to not only engage in mathematical conversations, but also trigger specific Unity actions based on student requests. The function-calling feature is particularly crucial, as it enables translation between natural language commands and 3D mathematical visualizations. When students make requests such as "generate two intersecting planes" or "show me the RREF steps," the Gemini model processes these utterances and determines the appropriate mathematical actions to execute. We have especially prompted our prompts in Gemini to provide feedback in a very specific structure that makes it easy to not only parse it out, but also call the appropriate functions in our code base for triggering the visualizations. The system also maintains a conversation history to provide coherent and contextual responses that build upon previous interactions.

For audio output, we employ ElevenLabs' multilingual streaming Text-to-Speech (TTS) API, configured to generate responses in a distinctive voice profile designed for the Einstein avatar. The ElevenLabs integration streams audio directly to the Quest 3's spatial audio system, enabling natural conversational timing and supporting the immersive learning environment. The streaming nature of TTS ensures that students receive immediate audio feedback, maintaining the natural flow of conversation essential for effective tutoring interactions.

4 Conversational Learning Through the Einstein Avatar

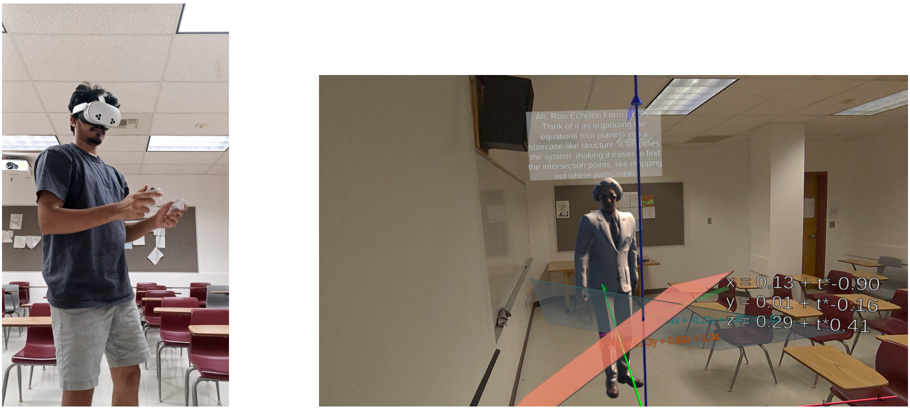
Our application transforms abstract linear algebra instruction into engaging conversational experiences through the Einstein avatar, fundamentally changing how students interact with mathematical concepts. Rather than navigating complex interfaces or memorizing procedural steps, students engage in natural dialogue that mirrors authentic mathematical discourse, creating a more intuitive and pedagogically sound learning environment. The conversational interface enables students to request mathematical visualizations through natural language, with Einstein responding both visually and auditorily to reinforce conceptual understanding. When a student speaks the command *"draw me two planes that intersect on a line,"* Einstein immediately responds with engaging mathematical dialogue while simultaneously rendering the requested geometry.

Einstein's response demonstrates sophisticated pedagogical awareness:*"Ah, a fascinating request! Imagine two pieces of paper cutting through the air; where they meet forms a line, doesn't it? I can certainly generate two such planes for you. I'll choose some values for their equations."* This response exemplifies several key pedagogical principles: it connects abstract mathematical concepts to tangible real-world analogies (pieces of paper), validates the student's mathematical curiosity, and provides transparency about the mathematical process being executed.

Figure 1 illustrates this interaction, showing both geometric visualization and Einstein's conversational engagement. The parametric equation of the resulting line is displayed within the MR environment, allowing students to observe the mathematical relationship between the intersecting planes and their linear intersection.

Similarly, when students request more complex scenarios such as *"Generate three planes that intersect at a single point in 3D space"*, Einstein provides mathematically rich dialogue that contextualizes the visualization within broader mathematical understanding. His response—*"Ah, excellent! You wish to see how three planes can come together to meet at a single, unique point in space. Much like how three different paths can cross at one specific location. I shall generate three such planes for you now"*—demonstrates how conversational AI can scaffold learning by connecting abstract mathematical concepts to spatial intuition.

In Figure 2 a student likes to visualize the addition of vectors geometrically. The student asks *"Draw me two random vectors and then perform vector addition on it"*. Enstien responds back *Aha! Adding vectors, a fundamental*

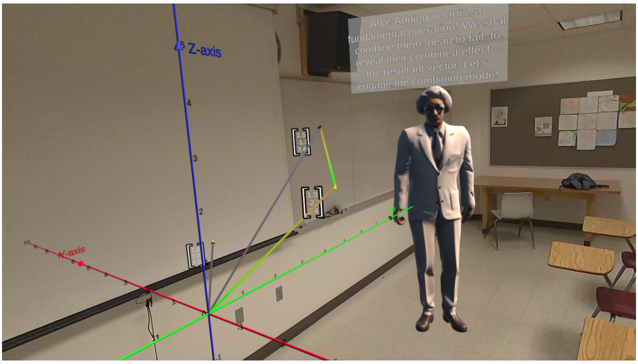


Ah, Row Echelon Form (RREF)! Think of it as organizing the equations (our planes) into a staircase-like structure. It simplifies the system, making it easier to find the intersection points, like mapping out where paths meet.

x = 0.13 + t*-0.90
y = 0.01 + t*-0.16
z = 0.29 + t*0.41

Figure 1: Left: Student with the Meta Quest 3 hardware in the classroom. Right: The Einstein avatar responds to a student's request to display two intersecting planes, providing both a geometric visualization and the parametric equation of their line of intersection. In this figure, the text above Einstein displays a transcript of his spoken response. Additionally, the figure captures the beginning of a step-by-step demonstration of the RREF algorithm, with Einstein's on-screen dialogue reflecting his explanation of the initial steps.

operation! We shall combine them, head to tail, to reveal their combined effect - the resultant vector. Let's engage the combining mode!

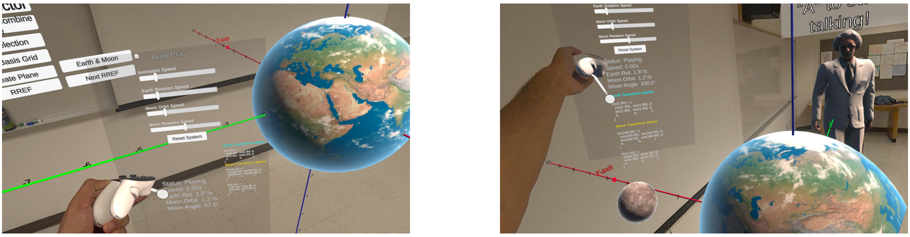


Aha! Adding vectors, a fundamental operation! We shall combine them, head to tail, to reveal their combined effect – the resultant vector. Let's engage the combining mode!

Figure 2: Vector addition where two random vectors are shown as column vectors in 3D space as [1;1;1] and [1.3;-2.3;2.4] and the resultant vector of [2.3;-1.3;3.4]

The application extends beyond basic geometric visualization to demonstrate real-world applications of linear transformations through an immersive Earth-Moon rotational system. This feature transforms abstract rotational matrices into tangible, observable phenomena that students can manipulate and control through voice commands and physical controllers.

Students can observe three distinct rotational transformations simultaneously: Earth's rotation around its own axis, the Moon's rotation around its own axis, and the Moon's orbital motion around Earth. Each rotation is governed by its corresponding rotational matrix, allowing students to witness the direct mathematical relationship between matrix parameters and physical motion. Traditional linear algebra instruction often presents rotational matrices as abstract mathematical objects disconnected from their geometric meaning, creating a significant barrier to student understanding [1]. Our paper addresses that gap by providing concrete, embodied interactions with 3D transformations.



Status: Playing
Speed: 1.00x
Earth Rot: 1.8°/s
Moon Orbit: 1.2°/s
Moon Angle: 57.0°

Status: Playing
Speed: 0.00x
Earth Rot: 1.8°/s
Moon Orbit: 1.2°/s
Moon Angle: 100.0°

Figure 3: Earth-Moon rotational system illustrating linear transformations through astronomical visualization. Left: The three rotational matrices are displayed alongside controller options, with the Earth centered at the origin. Students can manipulate these matrices using voice commands or physical interaction. Right: The Moon orbits the Earth, and the user dynamically adjusts rotational parameters via the controller to observe the real-time effects of transformation.

The conversational interface enables intuitive control over these transformations through natural language commands such as *"increase the speed of the earth rotation"* or *"make the moon go slower around the earth."* These voice interactions allow students to immediately observe how changes in rotational matrix parameters affect the corresponding physical motions, creating a direct connection between mathematical abstraction and observable phenomena.

The conversational approach offers several distinct advantages for applied linear algebra instruction in CS curricula. First, the natural language interface reduces cognitive load by eliminating the need to learn complex software interfaces, allowing students to focus entirely on mathematical concepts. This is particularly valuable in CS education, where students often struggle with the transition from algorithmic thinking to mathematical abstraction.

Second, Einstein's dialogical responses provide immediate conceptual scaffolding that connects mathematical operations to their geometric and physical interpretations. When students request specific visualizations, they receive not only the visual representation, but also the explanatory context that reinforces understanding. This conversational scaffolding is especially effective for CS students who benefit from understanding the practical applications of mathematical concepts in computational contexts.

Third, the real-time nature of the interactions enables iterative exploration and hypothesis testing. Students can quickly generate multiple geometric configurations, observe their properties, and develop an intuitive understanding through experimentation. This exploratory approach aligns with inquiry-based pedagogical methods while leveraging the unique capabilities of conversational AI and immersive visualization.

Finally, astronomical applications demonstrate the relevance of linear algebra concepts to computer graphics, game development, and simulation—core areas of CS where rotational transformations are fundamental. By observing how rotational matrices control planetary motion, students develop a deeper appreciation for the practical importance of linear algebra in computational applications, potentially increasing motivation and engagement in the subject matter. The integration of conversational AI with immersive visualization represents a significant advancement in mathematical pedagogy, offering CS students a more intuitive, engaging, and conceptually rich approach to linear algebra learning that bridges the gap between abstract mathematical theory and practical computational applications.

5 User Feedback and Planned Improvements

To evaluate the initial effectiveness and user experience of our conversational MR application, we conducted a pilot study with eight students who participated in a three-week linear algebra workshop designed for incoming students at California State University, Chico. These participants represented diverse academic backgrounds, including mathematics, chemistry, electrical and computer engineering, and arts, providing valuable insights into the application's appeal across different STEM and non-STEM disciplines. The workshop focused on solving systems of linear equations, visualizing their geometric interpretations, and understanding the geometric RREF process.

The survey results, presented in Table 1, reveal consistently positive responses in all measured dimensions, with mean scores ranging from 3.625 to 4.375 on a 5-point Likert scale. The highest scores were fundamental visualization capabilities (Q1: Mean = 4.375) and engagement with the Einstein avatar (Q3: Mean = 4.375), suggesting that both core technical functionality and the conversational interface effectively support learning objectives. The data indicates strong validation for our primary hypothesis that conversational MR enhances mathematical visualization. Question 1, addressing the core visualization of planes, lines and vectors, achieved the highest mean score with minimal variation (SD = 0.5175), indicating the broad consensus that the application effectively supports geometric understanding. Similarly, Question 5 (Mean = 4.25, SD = 0.7071) confirms that the 3D MR environment successfully builds intuition for spatial relationships fundamental to linear algebra.

In particular, Question 2 regarding linear transformations through the Earth-Moon system received the lowest mean score (3.625), though still indicating a positive reception. This result is particularly meaningful given that participants had not formally studied linear transformations at the time of trying the MR app, suggesting that the astronomical visualization approach successfully makes advanced concepts accessible to novice learners. The moderate standard deviation (0.7440) indicates some variation in how effectively different students connected rotational matrices to planetary motion, highlighting opportunities for improved scaffolding in future iterations.

The voice command system (Q4: Mean = 3.875, SD = 0.9910) showed the highest variability in responses, suggesting that while many students found voice interaction beneficial, some experienced difficulties that warrant technical improvements. This aligns with qualitative feedback indicating that voice recognition accuracy and system responsiveness require refinement for an optimal user experience. Student responses to the question about the interaction of the Einstein avatar reveal several key themes: The most prominent theme is enhanced engagement through personification of the learning experience. Students consistently noted that having a *"character to help explain"* and someone to *"tutor you while studying"* transformed the learning dynamic from passive absorption to active conversation. One student articulated this clearly: *"more interactive hands on activities are more engaging. And engagement leads to interest, and interest leads to learning."* The responses also highlight the motivational impact of the expert persona. One student noted that the avatar *"can motivate the user to want to keep learning about this subject"* because *"he is an expert in that field."* This suggests that the Einstein's avatar successfully leverages aspirational learning, where students feel inspired by interacting with a renowned scientific figure. However, feedback also reveals important usability considerations. One student expressed preference for *"creating equations myself"* instead of relying on avatar generation, indicating that the balance between guided instruction and student agency requires careful calibration. This feedback suggests implementing more user control options while maintaining the conversational benefits.

Although these preliminary results are encouraging, several limitations must be acknowledged. The small sample size (n=8) and non-randomized design limit generalizability, particularly for CS students who represent our primary target population. The diverse academic backgrounds of participants, while providing breadth, may not reflect the specific needs and prior knowledge of CS students taking applied linear algebra. Future evaluation plans include deployment with CS students during the applied linear algebra course, enabling comparison with traditional instruction methods through randomized controlled trials. This will allow measurement of learning outcomes beyond engagement and self-reported understanding, including performance on standardized assessments and retention of conceptual knowledge. Technical improvements based on user feedback will focus on enhancing voice recognition accuracy, particularly for mathematical terminology, and providing more granular user control over mathematical object creation. Additionally, the Earth-Moon visualization will be enhanced with more explicit connections to underlying rotational matrices to improve conceptual clarity for students without linear transformation background. The positive reception across diverse academic disciplines suggests broader applicability beyond CS education, though domain-specific adaptations may be necessary to optimize effectiveness for different student populations and mathematical concepts.

Question#	**Statement**	**Mean**	**Std Dev**	**Max**	**Min**
1	The app helped me better visualize and understand concepts such as planes, lines, and vectors in linear algebra.	4.375	0.5175	5	4
2	Even though I haven't formally studied linear transformations, the rotation of the moon around the Earth made intuitive sense to me.	3.625	0.7440	5	3
3	The ability to talk to Albert Einstein and have him show examples improved my interest in the topic.	4.375	0.7440	5	3
4	The voice command system made it easier to explore linear algebra visually.	3.875	0.9910	5	2
5	The 3D mixed reality environment helped me build an intuition for how planes and lines intersect.	4.25	0.7071	5	3
6	After using the app, I feel more motivated to learn the math behind what I saw.	4.00	0.7559	5	3
7	Having a virtual tutor like Albert Einstein made the learning experience more personal and engaging.	3.75	1.0351	5	2
8	Being able to talk to a virtual tutor directly will improve my ability to learn linear algebra concepts.	4.25	1.0351	5	2

Table 1: Survey results for the MR linear algebra app on Meta Quest 3. Participants rated each statement on a 5-point Likert scale: 1 = Strongly Disagree, 2 = Disagree, 3 = Neutral, 4 = Agree, 5 = Strongly Agree.

6 Conclusion

This work advances math education in computer science by combining conversational AI with MR to support linear algebra learning. Extending our prior research on MR visualizations [4], we introduce a new paradigm where students interact naturally with a virtual Einstein avatar to request visualizations and receive immediate, pedagogically aware feedback. Our system integrates Meta Wit.ai, Google Gemini, and ElevenLabs within Unity, enabling seamless voice-driven interaction and immersive visualization. The Earth-Moon system exemplifies how familiar astronomical motion can ground abstract concepts like linear transformations in physical intuition—particularly relevant for computer science topics like graphics and machine learning. Preliminary feedback from eight users is promising, though larger studies are needed. Future work will expand to other linear algebra topics and improve error detection. By merging natural language interaction with MR, this research lays a foundation for AI-powered educational tools that make advanced math more accessible, engaging, and inclusive.

Acknowledgment

We gratefully acknowledge the use of OpenAI's ChatGPT for proofreading, grammatical checks, and other text editing tasks.

References

[1] Samantha D. Aguilar et al. "Learning 3D Matrix Algebra Using Virtual and Physical Manipulatives: Qualitative Analysis of the Efficacy of the AR-Classroom". In: *Learning and Collaboration Technologies*. Ed. by Panayiotis Zaphiris and Andri Ioannou. Cham: Springer Nature Switzerland, 2024, pp. 3–16. ISBN: 978-3-031-61691-4.

[2] Eason Chen et al. "VTutor: An Animated Pedagogical Agent SDK that Provide Real Time Multi-Model Feedback". In: *arXiv preprint arXiv:2505.06676* (2025).

[3] Owen Henkel et al. "Effective and scalable math support: Evidence on the impact of an AI-tutor on math achievement in Ghana". In: *arXiv preprint arXiv:2402.09809* (2024).

[4] Sheldon Henriques et al. "Enhancing Computer Science Education Through Immersive Geometry-Based Approaches to Linear Algebra". In: *Proceedings of the 2025 Consortium for Computing Sciences in Colleges Midwest Region (CCSC MW).* To appear. Chico, CA, 2025.

[5] Flavia Aurelia Hidajat. "Augmented reality applications for mathematical creativity: a systematic review". In: *Journal of Computers in Education* 11.4 (2024), pp. 991–1040.

[6] Georgios Lampropoulos. “Augmented Reality, Virtual Reality, and Intelligent Tutoring Systems in Education and Training: A Systematic Literature Review.” In: *Applied Sciences (2076-3417)* 15.6 (2025).

[7] Eda Okur et al. “End-to-end evaluation of a spoken dialogue system for learning basic mathematics”. In: *arXiv preprint arXiv:2211.03511* (2022).

[8] Zachary A Pardos and Shreya Bhandari. “ChatGPT-generated help produces learning gains equivalent to human tutor-authored help on mathematics skills”. In: *Plos one* 19.5 (2024), e0304013.

[9] Ragnhild Johanne Rensaa, Ninni Marie Hogstad, and John Monaghan. “Themes within lecturers’ views on the teaching of linear algebra”. In: *International journal of mathematical education in science and technology* 52.1 (2021), pp. 107–123.

[10] Choo Kim Tan et al. “Influence of conversational agent on students’ attitude toward mathematics”. In: *Journal of Research in Innovative Teaching & Learning* (2025).

[11] Rose E Wang et al. “Tutor CoPilot: A human-AI approach for scaling real-time expertise”. In: *arXiv preprint arXiv:2410.03017* (2024).

[12] Ramez Yousri et al. “IllusionX: An LLM-powered mixed reality personal companion”. In: *arXiv preprint arXiv:2402.07924* (2024).

[13] Marjorie A Zielke et al. “Exploring social learning in collaborative augmented reality with pedagogical agents as learning companions”. In: *International Journal of Human–Computer Interaction* 41.4 (2025), pp. 2424–2449.

Reviewers — 2025 CCSC Rocky Mountain Conference

Achee, Bonnie	Southeastern Louisiana University, Hammond, LA
Aldous, Peter	Utah Valley University, Orem, UT
Alkadi, Ghassan	Southeastern Louisiana University, Hammond, LA
Amin, Mohammad	National University, San Diego, CA
Assiter, Karina	Landmark College, Putney, VT
Attarwala, Abbas	California State University, Chico, CA
Blumenthal, Richard	Regis University, Denver, CO
Chen, Xi	Utah Valley University, Orem, UT
Cliburn, Daniel	University of the Pacific, Stockton, CA
Clincy, Victor	Kennesaw State University, Kennesaw, GA
Deutschbein, Calvin	Willamette University, Salem, OR
Fellah, Aziz	Northwest Missouri State University, Maryville, MO
Glass, Michael	Valparaiso University, Valparaiso, IN
Grujic, Olivera	California State University, Stanislaus, Turlock, CA
Hamdan, Basil	Utah Valley University, Orem, UT
Harris, Laurie	Southern Utah University, Cedar City, UT
Heath Van Horn, Mathew	Embry-Riddle Aeronautical University, Prescott, AZ
Hemmes, Jeffrey	United States Air Force Academy, Colorado Springs, CO
Kirkman, Stephen	Regis University, Denver, CO
Leverington, Michael	Northern Arizona University, Flagstaff, AZ
Lindoo, Ed	Regis University, Denver, CO
Liu, Jigang	Metropolitan State University, Denver, CO
Lotfy, Mohamed	Utah Valley University, Orem, UT
McDonald, Dan	Utah Valley University, Orem, UT
Mota, Thyago	Metropolitan State University, Denver, CO
Nehring, Jenny	Utah Valley University, Orem, UT
Parvez, Imtiaz	Utah Valley University, Orem, UT
Rhee, Junghwan	University of Central Oklahoma, Edmond, OK
Sajal, Sayeed	Utah Valley University, Orem, UT
Shabanian, Hanieh	Western New England University, Springfield, MA
Smallwood, Pam	Regis University, Denver, CO
Zeng, Hongbiao	Fort Hays State University, Hays, KS
Zhang, Liqiang	Indiana University South Bend, South Bend, IN
Zuo, Fei	University of Central Oklahoma, Edmond, OK