The Journal of Computing Sciences in Colleges

Papers of the 36th Annual CCSC Southeastern Conference

November 11th-12th, 2022
University of North Carolina – Asheville
Asheville, NC

Baochuan Lu, Editor
Southwest Baptist University

Chris Alvin, Regional Editor
Furman University

Volume 38, Number 5 **November 2022**

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

Table of Contents

The Consortium for Computing Sciences in Colleges Board of Directors

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:

Karina Assiter, President (2022), (802)387-7112, karinaassiter@landmark.edu.

Chris Healy, Vice President (2022), chris.healy@furman.edu, Computer Science Department, 3300 Poinsett Highway Greenville, SC 29613.

Baochuan Lu, Publications Chair (2024), (417)328-1676, blu@sbuniv.edu, Southwest Baptist University - Division of Computing & Mathematics, 1600 University Ave., Bolivar, MO 65613.

Brian Hare, Treasurer (2023), (816)235-2362, hareb@umkc.edu, University of Missouri-Kansas City, School of Computing & Engineering, 450E Flarsheim Hall, 5110 Rockhill Rd., Kansas City MO 64110.

Cathy Bareiss, Membership Secretary (2022), cathy.bareiss@betheluniversity.edu, Department of Mathematical Engineering Sciences, 1001 Bethel Circle, Mishawaka, IN 46545.

Judy Mullins, Central Plains Representative (2023), Associate Treasurer, (816)390-4386, mullinsj@umkc.edu, UMKC, Retired.

Michael Flinn, Eastern Representative (2023), mflinn@frostburg.edu, Department of Computer Science Information Technologies, Frostburg State University, 101 Braddock Road, Frostburg, MD 21532.

David R. Naugler, Midsouth Representative(2022), (317) 456-2125, dnaugler@semo.edu, 5293 Green Hills Drive, Brownsburg IN 46112.

Grace Mirsky, Midwest Representative(2023), gmirsky@ben.edu, Mathematical and Computational Sciences, 5700 College Rd. Lisle, IL 60532.

Lawrence D'Antonio, Northeastern Representative (2022), (201)684-7714, ldant@ramapo.edu, Computer Science Department, Ramapo College of New Jersey, Mahwah, NJ 07430.

Shereen Khoja, Northwestern Representative(2024), shereen@pacificu.edu, Computer Science, 2043 College Way, Forest Grove, OR 97116.

Mohamed Lotfy, Rocky Mountain Representative (2025), Information Systems & Technology Department, College of Engineering & Technology, Utah Valley University, Orem, UT 84058.

Tina Johnson, South Central Representative (2024), (940)397-6201, tina.johnson@mwsu.edu, Dept. of Computer Science, Midwestern State University, 3410 Taft Boulevard, Wichita Falls, TX 76308.

Kevin Treu, Southeastern Representative (2024), (864)294-3220, kevin.treu@furman.edu, Furman University, Dept of Computer Science, Greenville, SC 29613.

Bryan Dixon, Southwestern Representative (2023), (530)898-4864, bcdixon@csuchico.edu, Computer

Science Department, California State University, Chico, Chico, CA 95929-0410.

Serving the CCSC: These members are serving in positions as indicated:
Bin Peng, Associate Editor, (816)584-6884, bin.peng@park.edu, Park University - Department of Computer Science and Information Systems, 8700 NW River Park Drive, Parkville, MO 64152.
George Dimitoglou, Comptroller, (301)696-3980, dimitoglou@hood.edu, Dept. of Computer Science, Hood college, 401 Rosemont Ave. Frederick, MD 21701.
Carol Spradling, National Partners Chair, (660)863-9481, carol.spradling@gmail.com, 760 W 46th St, Apt 208, Kansas City, MO 64112.
Megan Thomas, Membership System Administrator, (209)667-3584, mthomas@cs.csustan.edu, Dept. of Computer Science, CSU Stanislaus, One University Circle, Turlock, CA 95382.
Ed Lindoo, Associate Treasurer & UPE Liaison, (303)964-6385, elindoo@regis.edu, Anderson College of Business and Computing, Regis University, 3333 Regis Boulevard, Denver, CO 80221.

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.

Platinum Level Partner

Google Cloud
GitHub
NSF – National Science Foundation

Gold Level Partner

zyBooks
Rephactor

Associate Level Partners

Mercury Learning and Information
Mercy College

Welcome to the 2022 CCSC Southeastern Conference

Welcome to the 36th Southeastern Regional Conference of the Consortium for Computing Sciences in Colleges. The CCSC:SE Regional Board welcomes you – in November once again! – to Asheville, NC, the home of UNC Asheville. The conference is designed to promote a productive exchange of information among college personnel concerned with computer science education in the academic environment. It is intended for faculty as well as administrators of academic computing facilities, and it is also intended to be welcoming to student participants in a variety of special activities. We hope that you will find something to challenge and engage you at the conference!

The conference program is highlighted by a variety of sessions, such as engaging guest speakers, tutorials, student posters, a nifty assignment session and several sessions of high quality refereed papers. We received 20 papers this year of which 12 were accepted to be presented at the conference and included in the proceedings – an acceptance rate of 60%.

Two exciting activities are designed specifically for students – a research contest and an undergraduate programming competition, with prizes for the top finishers in each.

We especially would like to thank the faculty, staff, and students of UNCA for their help in organizing this conference, especially under the challenging circumstances caused by the pandemic. Many thanks also to the CCSC Board, the CCSC:SE Regional Board, and to a wonderful Conference Committee, led by Conference Co-Chairs Dr. Marietta Cameron and Dr. Kevin Sanft. Thank you all so much for your time and energy.

We also need to send our deepest appreciation to our partners, sponsors, and vendors. Please take the time to go up to them and thank them for their contributions and support for computing sciences education – CCSC National Partners: Google Cloud, GitHub, National Science Foundation, zyBooks, Rephactor, Mercury Learning and Information, Mercy College. Sponsoring Organizations: CCSC, ACM-SIGCSE, Upsilon Pi Epsilon.

We could not have done this without several excellent submissions from authors, and the insightful comments from a great team of 31 reviewers (many of whom were willing to take on extra papers), and the support from our editor Baochuan Lu. Thanks to all of you for helping to create such a strong program for this year's conference.

We hope you enjoy the conference and your visit to UNC Asheville.

Kevin Treu, CCSC:SE Regional Board Chair
Chris Alvin, Program Chair
Furman University

2022 CCSC Southeastern Conference Steering Committee

Marietta Cameron ..... Local Arrangements Chair
Marietta Cameron ..... Local Publicity Chair
Kevin Sanft ..... Speakers Chair
Kevin Sanft ..... Vendors Chair
Marietta Cameron ..... Local Sponsors Chair
Andy Digh ..... Programming Contest Co-Director
Chris Healy ..... Programming Contest Co-Director
Chris Healy ..... Student Research Contest Director
Steven Benzel ..... Nifty Assignments Co-Chair
Robert Lutz ..... Nifty Assignments Co-Chair

Regional Board — 2022 CCSC Southeastern Region

Marietta Cameron ..... 2022 Site Chair
Kevin Sanft ..... 2022 Site Chair
Kevin Treu ..... Regional Board Chair
Kevin Treu ..... CCSC:SE Regional Representative
Karen Works ..... Treasurer
Chris Alvin ..... Program Chair/Regional Editor
Stephen Carl ..... Publicity Chair
Jean French ..... Local Registrar
Ethan McGee ..... 2021 Site Chair
Jim Knisely ..... 2021 Site Chair
Jean French ..... 2023 Site Chair
Jonathan Cazalas ..... At-large Member

A Novel Application for Automating Security Risk Assessment and Mitigation of Bluetooth Infotainment Devices*

Michael Sanjaya and Hala ElAarag
Department of Mathematics and Computer Science
Stetson University
DeLand, FL 32723
{msanjaya, helaarag}@stetson.edu

Abstract

The quick and widespread implementation of Bluetooth as an option to replace formerly wire connected devices has come at the cost of increased security issues. With Wi-Fi and ethernet connected devices, software and firmware security updates are easier and often even automatically applied. However, Bluetooth devices such as automobile head units, speakers, keyboard, and mice are more static with little to no feasible way to update these devices. In this research, we present a prototype of an application that could be used to inform users of the vulnerability that exists within their Bluetooth devices and provide tactics to mitigate them. Our prototype is user friendly, easy to use, and has a very low cost as it utilizes open source software and commercially available and cheap hardware to capture and decode Bluetooth packets.

1 Introduction

With more than 5 billion Bluetooth capable devices shipped in the last year, Bluetooth has become just as fundamental as Wi-Fi in many people's lives.

Bluetooth devices go beyond peripherals such as mice and headsets, 20-30 billion IoMT devices exist with many of those being Bluetooth devices like electrocardiograms, oximeters, and blood pressure monitors; attacks on these devices can have serious health implications [18]. Unlike Wi-Fi, which has been continuously studied and updated, Bluetooth relies on fewer and less updated standards. The security of Bluetooth is taken for granted in that it is protected by its frequency hopping nature and that gathering information on specific Bluetooth enabled devices often requires multiple steps, utilizing different repositories, databases, hardware and software tools. This research presents a Django web application that streamlines the process of data gathering. It automates the processes of scanning for and listing Bluetooth devices, querying for specific Common Vulnerabilities and Exposures (CVE) related to those devices, and displaying all of the above on a single page application. With the availability and affordability of open source IoT devices such as the Raspberry Pi, our prototype enables more users to automate the process of scanning for devices remotely.

2 Background

In this section we introduce the background history and fundamentals of Bluetooth. Designed to replace rs-232 cables, Dr. Jaap Haartsen created what would be the standard for Bluetooth in 1994 [14]. Prior to this, many devices such as printers, etc., were connected to computers via rs-232 cables which are rather power hungry. At +/-13V, rs-232 cables require significantly more power on average than Bluetooth devices which typically require 1.8V up to 5V.

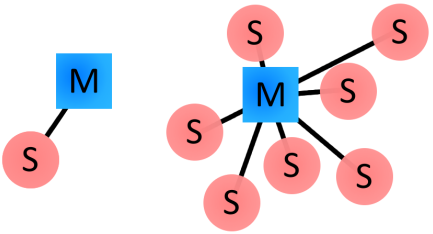


Figure 1: Piconet

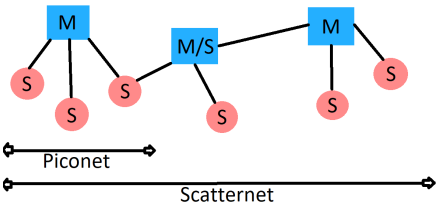


Figure 2: Scatternet

In regard to security, Bluetooth traffic is able to be more obscure in part because of its frequency hopping nature of 1600 hops per second, on the same 2.4 GHz band as Wi-Fi [5]. Bluetooth does this in order not to interfere with Wi-Fi, hopping through 79 channels spaced 1 MHz apart beginning at 2402 MHz

up to 2480 MHz [5]. So, in order to capture all the packets, one must monitor all 79 channels, or have knowledge of the hopping frequency and pattern. The network topology of Bluetooth devices in the smallest configuration is called a piconet, where there is one master device and up to seven slave devices as shown in Figure 1. In its largest configuration, the Bluetooth network would be a scatternet, where a master device in one piconet can serve as a slave in another as illustrated in Figure 2. Bluetooth device discovery time suffers from high variance [2], beyond frequency hopping the devices themselves are periodically in an inquiry substate [17]. By automating the process of scanning for devices, scans can be run and rerun to ensure better device discovery.

Open source devices such as the Ubertooth One by GreatScottGadgets allow for passive Bluetooth monitoring and sniffing on Linux host machines [15]. Unlike Wi-Fi targets that might change IP addresses due to DHCP or some other configuration changes, Bluetooth devices that use Bluetooth Low Energy (BLE) have addresses that are relatively unchanged and thus more easily able to be tracked [3].

3 Related Work

This section describes how researchers and attackers alike find and exploit vulnerabilities. In order to crack and decipher Bluetooth packets, one must first find, and capture data sent between Bluetooth devices. In the initial reconnaissance and scanning phase, an attacker will scan for local devices and store potential victim addresses. Even if a Bluetooth device is in a non-discoverable state it can still be passively monitored if the device is actively sending data. The device in this state will not be responding to inquiries but will respond to paging if another device knows its Upper Address Part (UAP) and Lower Address Part (LAP) [12]. Utilizing hardware and software, an attacker will sniff for packets between identified devices that are paired and sending information. Once the packets are captured, they can be sent to Wireshark for analysis or unencrypted by programs such as Crackle. Full packet histories are needed to successfully unencrypt data, as the initial pairing procedure needs to have been captured between two devices. After pairing keys have been identified, an attacker can use that information to pair or gain access or functionality to the device, or spoof a device entirely. Bluetooth security does not often keep a record on who has accessed a device. Among newer attacks are those created in the research sector and security industry, such as BlueBorne by Armis Labs [9]. In this case, Armis Labs discovered and disclosed eight exploits unknown to developers, or zero-day vulnerabilities, regarding Bluetooth enabled devices and utilized their BlueBorne program to attack these vulnerabilities.

4 Implementation

In this section we describe the implementation of scanning methods, and the development of a single page graphical user interface application to streamline and automate these procedures. Recent updates to programs such as Wireshark [16], allow for the reading of captured Bluetooth packets, as well as show the interest in deciphering Bluetooth communication. Commercially available and affordable hardware such as the Ubertooth One [11] shown in Figure 3 and Sena UD-100 shown in Figure 4, allow computer devices to read the spectrum and capture the packets of Bluetooth devices. This raw data can be sent to open source tools and software such as Kismet [8] and Blue Hydra [7], among others, to store and read the data. For some versions of Kali Linux, there exist preinstalled or easily installed programs such as hcitool, btscanner, sdptool, and Bluesnarfer [10]. These tools can detect Bluetooth devices and display general information about a device; and in the case of Bluesnarfer it is able to pull or delete information from a smartphone. Crackle [13] is one other program that when passed a captured and encrypted Bluetooth packet capture (pcap) file, is capable of decrypting that file. Spooftooph [6], developed by JP Dunning, is an interesting program that can clone and save Bluetooth device information and generates a new random Bluetooth profile based on spoofed information.

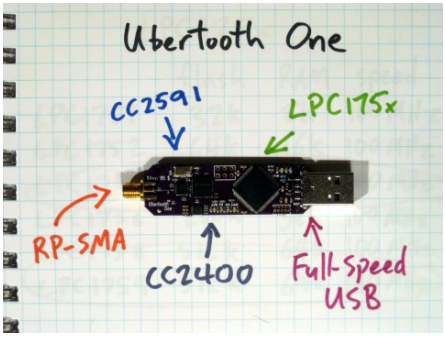


Figure 3: Ubertooth One

Figure 4: Sena UD100-G03

The Ubertooth One alone is not enough to decipher Bluetooth traffic or packets, additionally one must utilize either a host machine's Bluetooth card or external Bluetooth dongle as well as open source programs. Utilizing an external Bluetooth dongle to pass traffic to the Ubertooth One, programs such as Blue Hydra can interpret from the data a device's upper and lower address

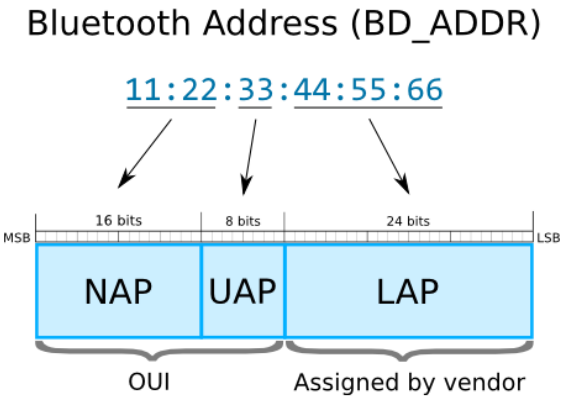


Figure 5: Bluetooth Device Address

portion and its signal strength. Kismet is another open source program that can capture classic Bluetooth packets passed from the Ubertooth, and pipe the data into Wireshark for further analysis [15]. Figure 5 displays how the 48 bits of the Bluetooth Address are assigned.

Initially, we used a Kali Linux release 2020.3 virtual machine running on Oracle VM VirtualBox on a Windows 10 laptop. The first step was to install the drivers of the Sena UD100-G03 and the Ubertooth One external USB Bluetooth dongle drivers. It was easier to apply the firmware update to the Ubertooth One via connecting the device to a MacBook Air as it could more easily recognize and apply the firmware update due to its native Linux like environment. We then connected both devices to the host machine and configured them to be set

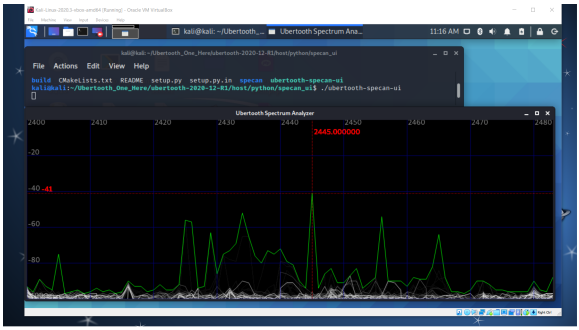


Figure 6: Ubertooth Spectrum Analyzer

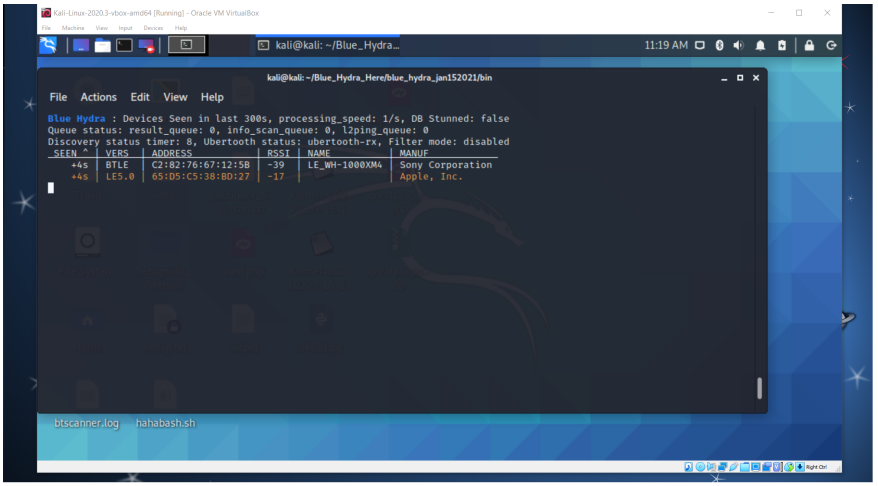


Figure 7: Blue Hydra

to the Kali Linux virtual machine. We installed the respective dependencies and prerequisites for Ubertooth and Blue Hydra tools. Last, we started the Ubertooth spectrum analyzer user interface shown in Figure 6, as well as the Blue Hydra program shown in Figure 7 to identify devices, their address, and signal strength.

Rather than run each Bluetooth command individually per device, the idea is to automate the procedure and do so remotely. By utilizing the Django web framework, we created a single page application that users can pull from a

Figure 8: Raspberry Pi 4 Client Sensor

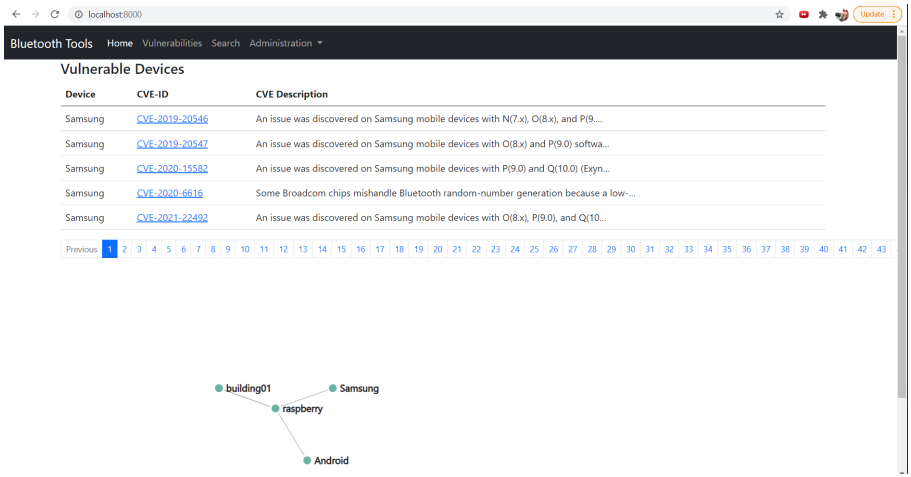


Figure 9: Remote Bluetooth Tools Web Application

remote repository. This application, titled Remote Bluetooth Tools, enables users to connect to and set parameters for remote sensors and choose which commands to be run on those sensors, while the output is displayed to the same page. On the chosen remote device, the client application will begin reconnaissance and automatically target the found Bluetooth devices. For a Bluetooth specific listing of vulnerability and exploits, we used the National Institute of Standards and Technology's (NIST) database [4]. NIST provides a complete manual in pdf format on how to access their API to pull CVEs in a JSON format based on keyword searches, as well as being able to download their entire list of Bluetooth specific CVEs. We used a Windows 10 laptop to develop the Django application with VSCode. The client sensor used a Raspberry Pi 4 running Raspberry Pi OS Bullseye, configured with PyBluez packages installed and a Sena UD100-G03 Bluetooth USB adapter connected for extended range (Figure 8). In the initial development, the Windows 10 laptop served as the host for the Remote Bluetooth Tools application (Figure 9), and accepted and stored data sent from the Raspberry Pi via the application's API. The application was later migrated to Heroku, which is a cloud application platform, to better allow for remote access. Database tables to include CVE's and devices were migrated to Heroku's Postgres database.

Scanning for devices requires the Python file client.py to be ran manually on the sensor device, or it may be automated by use of crontab. Within the client.py file a function is ran, the bluetooth.discover_devices(lookup_names=True) function, and the method start_scan wraps the returned data

in a JSON format to be sent to the Remote Bluetooth Tools API, where the application database stores the data and updates the home page with the devices found by the sensor and relevant CVEs related to that device. Users will connect to the Remote Bluetooth Tools web application, which may be hosted locally or on a remote machine. Entries from data collected from running scans on client sensors are passed to the API on the web application and serialized and stored on the application's Django SQLite3 database while dynamically updating the web page with the list of client sensors and the devices they have found.

5 Results

This section presents the results of the scanning and penetration testing procedures. Working in a Linux environment, the most consistent option for having a host machine list found Bluetooth devices is by utilizing the bluetoothctl command, as it is included in most modern kernels of Linux. Enabling Bluetooth on a Linux host allows for the robust functionalities of bluetoothctl, which was used for not only listing Bluetooth devices but pairing them to the host machine as well. Information not able to be gathered from bluetoothctl was found by running sdptool browse which showed the services that are available by a device and outputting the results to a .txt file. Bluesnarfing a device such as a smartphone proved challenging as only newer smartphone models were at hand that were possibly obfuscating or limiting certain needed parameters such as phonebook access channels, though exploitations were available. Running the client side code from the Remote Bluetooth Tools application periodically yielded inconsistent results, especially on a Windows host machine, in which an empty list might be returned from a command from the PyBluez module to discover nearby devices, and which needed to be run multiple times from a Linux host to return values. The configured sensor devices successfully scanned for and returned a Bluetooth device using the client.py file and sent data to the targeted host running the Remote Bluetooth Tools application. The main page was then updated using the D3.js node network to display the sensor, the device found by that sensor, and the physical location of the sensor. Certain devices were found that do not have any corresponding CVEs from the nist.gov database, however other devices such as Samsung TVs or smartphones did return matching Bluetooth specific CVEs pulled from nist.gov, which were then displayed on the refreshed home page. Running PyBluez Bluetooth functions may require multiple attempts, as the Bluetooth scanning might interrupt the Wi-Fi signal, requiring a reconnection to the internet in some cases. A caveat is not to keep default usernames and passwords for systems, especially the Raspberry Pi, and open port 22 on the router. In this case the Raspberry

Pi sensor client was immediately hit with crypto malware which was running random number generators and attempting to propagate to other systems by downloading and using ZMap. Tracking devices without the need for pairing was done additionally through running Blue Hydra and l2ping, in which the former paired with the Ubertooth One shows the full MAC address of a nearby device and its received signal strength indicator and the latter using that address to continuously ping that device. To note, running PyBluez commands can take longer on the Raspberry Pi than scanning for devices using hcitool lescan or bluetoothctl which produce results as soon the command issued (Figure 10 shows discover_devices on the left and hcitool lescan on the right being run at the same time).

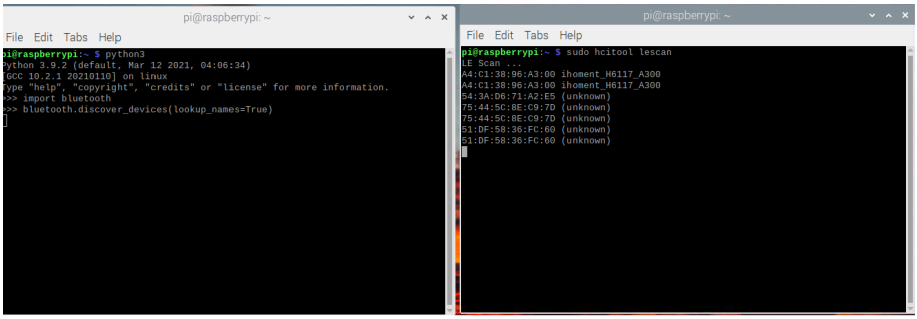


Figure 10: left discover_devices, right hcitool lescan

6 Conclusion

This section declares the conclusion. Developing a web application to automate and streamline the process of discovering and displaying Bluetooth devices, as well as searching for and mapping CVEs related to those devices, solves the complexity and time required to manually complete each step. Scanning for Bluetooth devices through the command line or through the Remote Bluetooth Tools application yields quick results of names and MAC addresses of nearby Bluetooth devices. The configured sensor clients listed in the Remote Bluetooth Tools web application report back the Bluetooth devices they found to the server which stores that information to the SQLite database which can be queried by the user. Many open source Bluetooth tools are seldom if ever updated, as well new Bluetooth devices and mobile phones have updated Bluetooth components that lead to some past Bluetooth exploitation attacks to yield inconsistent results. Because of the limited range of most Bluetooth devices, especially those utilizing BLE which has a realistic range of around 30

feet, a Bluetooth card or external USB dongle with extended range will be better able to detect more devices. Many Bluetooth tools are built with and for Linux like operating systems and often lack GUIs, an aim of this work is for these tools to be accessible across more platforms and to aid in their efficiency with ease of use for end users. The initial setup procedure of pulling the Remote Bluetooth Tools from its repository, assigning a host to run the application as well as configuring a client sensor, takes more time on average than using individual conventional command line tools. Once the setup process is completed, the system is fully automated. As the sensors detect devices, the application home page updates the node network graph and searches the CVE database table for CVEs matching the device names and populates those CVEs to the home page as well. In this case automating the procedure of scanning for devices and querying the database yields significant improvement in the time it takes to do these procedures individually.

7 Future Work

This section details work to be done in the future. Additional work will include updates to the Remote Bluetooth Tools web application to be available as a smart phone application. Subsequent work and research will be done regarding using multiple Ubertooth One and monitoring multiple Bluetooth channels at once. To mitigate the chances of a Bluetooth device being compromised, it is preferable that Bluetooth functions be disabled or in an inactive state as well as having a strong PIN other than common defaults such as 0000. Implementing and displaying the vulnerability notes from Carnegie Mellon University's Vulnerability Notes Database [1] would be an additional benefit as these often times contain step by step instructions on how to perform exploits and mitigations aside from the brief overview of the CVEs from nist.gov related to Bluetooth.

References

[1] Software Engineering Institute Cert Coordination Center. Vulnerability Notes Database. `https://www.kb.cert.org/vuls/`, 2021. Carnegie Mellon University.

[2] Goutam Chakraborty, Kshirasagar Naik, Debasish Chakraborty, Norio Shiratori, and David Wei. Analysis of the bluetooth device discovery protocol. *Wireless Networks*, 16(2):421–436, 2010.

[3] Aveek K Das, Parth H Pathak, Chen-Nee Chuah, and Prasant Mohapatra. Uncovering privacy leakage in ble network traffic of wearable fitness trackers. In *Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications*, pages 99–104, 2016.

[4] National Vulnerability Database. Search Vulnerability Database. https://nvd.nist.gov/vuln/search, 2021. National Institute of Standards and Technology.

[5] Martin Davies et al. Improving compliance with bluetooth device detection. 2015.

[6] JP Dunning. Spooftooph. https://gitlab.com/kalilinux/packages/spooftooph, 2011.

[7] Rick Farina. Blue Hydra. https://github.com/pwnieexpress/blue_hydra, 2016. 12 July 2016.

[8] Mike Kershaw. Kismet. https://github.com/kismetwireless/kismet, 2002. 22 July 2002.

[9] Armis Labs. BlueBorne. https://www.armis.com/research/blueborne/, 2017.

[10] Kimball Leavitt. Bluesnarfer. https://github.com/kimbo/bluesnarfer, 2013. 17 August 2013.

[11] Michael Ossmann. Ubertooth. https://github.com/greatscottgadgets/ubertooth, 2010. 25 October 2010.

[12] Trishna Panse, Vivek Kapoor, and Prashant Panse. A review on key agreement protocols used in bluetooth standard and security vulnerabilities in bluetooth transmission 1. 2012.

[13] Mike Ryan. Crackle. https://github.com/mikeryan/crackle, 2013. 8 Feb 2013.

[14] Robert Triggs. A Quick History of Bluetooth. www.androidauthority.com/history-bluetooth-explained-846345/, 2018. Android Authority, 23 March 2018.

[15] Thomas Willingham, Cody Henderson, Blair Kiel, Md Shariful Haque, and Travis Atkison. Testing vulnerabilities in bluetooth low energy. In *Proceedings of the ACMSE 2018 Conference*, pages 1–7, 2018.

[16] Wireshark. Wireshark. https://www.wireshark.org/, 2021.

[17] Ryan W Woodings, Derek D Joos, Trevor Clifton, and Charles D Knutson. Rapid heterogeneous ad hoc connection establishment: accelerating bluetooth inquiry using irda. In *2002 IEEE Wireless Communications and Networking Conference Record. WCNC 2002 (Cat. No. 02TH8609)*, volume 1, pages 342–349. IEEE, 2002.

[18] Mohammed Zubair, Devrim Unal, Abdulla Al-Ali, and Abdullatif Shikfa. Exploiting bluetooth vulnerabilities in e-health iot devices. In *Proceedings of the 3rd international conference on future networks and distributed systems*, pages 1–7, 2019.

Commonalities of Users Influenced and Not Influenced by Persuasive Communication in Human Robot Interactions*

Nathan Green[1] and Karen Works[2†]
[1]School of Technology and Innovation
Marymount University
Arlington, VI 22207
ngreen@marymount.edu
[2]Department of Computer Science
Florida State University
Panama City, FL 32405
keworks@fsu.edu

Abstract

Social robots have been developed to support users, such as tour guides or sales robots. In these systems, the main purpose of the human robot interaction is to support the user's need. In addition to these capabilities, such robots could have a goal to persuade users to do something of which they have no knowledge, i.e., a hidden task. In this paper we explore if there are commonalities in the perceptions from users influenced and not influenced to perform a hidden task about the robot and the interaction with the robot.

1 Introduction

Human Robot Interaction (HRI), while once seen as a science fiction narrative, has become common place in both the household [14] and workplace [9, 13, 11].

†Both authors contributed equally to this research.

Humans are interacting with versions of household, workplace, and software robots whether they choose to be a first adopter of technology or use the interaction to increased productivity [2]. Up until this point most of these interactions have been direct and intentional. If a user starts a domestic robot, they expect, for instance, their floor to be vacuumed. If a user starts an interaction with a software robot, they expect to find a direct answer to their questions. What is often not accounted for is that the robot may have a goal of its own. Given a robot with six types of persuasion techniques, how will the attitudinal response of the users differ based on whether or not the user is persuaded by the robot to perform a hidden goal?

Our robot engages prospective students on a college campus. While our robot is expected to answer basic questions about academic programs, sports, and social activities, our robot will also be using different persuasive conversation techniques to drive the user to perform a hidden task. This task is not directly related to the main purpose of the robot or the location. Hence, it independently tests the persuasion techniques without the perceived incoming bias of being in the college environment. We expect the results would differ if the hidden task for the robot was anticipated. Choosing the correct persuasion techniques could advanced many University goals such as student participation and increase retention rates[4]. In this study, we examine the commonalities in users attitudinal and behavioral responses based upon whether or not they were influenced by a robot to perform a hidden task.

2 Background

We implemented our recruitment robot in a college setting and examined the participants acceptance of human robot interaction and well as the effects of a robot's use of persuasive communication on those participants.

[8] deployed an HCI system "in the wild" on a college campus. Their goal was to monitor and display the mood of the university via displays and cameras monitoring students at certain points of the campus over 10 weeks.

[5] discussed the pre-beginnings moment of interaction between visitors and robots. The authors concluded a human robot interaction's initial contact must be flexible. This will be controlled for in our study with a pre-interaction survey to gauge their level comfortableness and the participants will be informed that they are expected to interact with a robot ahead of time.

[16] studied the influences of different persuasive strategies on request success on Kiva (a crowd funding app). Strategies were broken into Concreteness, Commitment, Emotional, Identity, Impact/Value, and Scarcity. They found that concreteness was a related factor in whether a loan was funded or not.

[15] collected 1,017 human persuasion conversations involving incentives towards users to donate to a charity. They identified 10 persuasion strategies (Logical appeal, Emotion appeal, Credibility appeal, Foot-in-the-door, Self-modeling, Personal story, Donation information, Source-related inquiry, Task-related inquiry, Personal-related inquiry). Donation information showed the largest benefit indicating users preferred detailed directions of how to donate. Humans' judgement of a robot interaction can often come down to three factors: warmth, competence, and discomfort[3]. General discomfort will be measured via our pre and post interaction surveys while warmth and competence could be factors within different persuasion strategies.

3 Methodology

The goal of our approach is to measure the commonalities in users attitudinal and behavioral responses to an interaction with a robot using one of a set of persuasion conversation logic types. The purpose of the persuasion conversation logic is to convince the user to perform a hidden task. The following types of persuasion conversation logic were explored: Scarcity, Emotion, Social Identity, Commitment, Concreteness, or No persuasion [16]. In [6] we evaluated the effects of each type individually. In this work, we now aggregate the results to measure commonalities of users influenced and not influenced to perform the hidden task and the users in the control group (i.e., No persuasion).

The basis of our set up is as follows: In an experimental setting, we measured the users' subjective (attitudinal) and objective (behavioral) responses while interacting with our robot. Conversations where the user performed the hidden task are considered to be successful, i.e., the user was influenced (and visa versa). The results of the most successful, somewhat successful, not successful, and control group approaches were compared to determine commonalities in users who were persuaded by a robot to perform a hidden task.

3.1 System setup

Our Pepper (SoftBank Robotics), a social humanoid robot is designed to interact and support recruitment events. Namely, she carries on conversations about the activities and degree programs available. Additionally at appropriate moments in the conversation, she initiates a persuasive response (See Table 1 Examples of Persuasion Responses by Type) to the hidden task which in our case convincing the user to eat a pretzel (See Figure 1).

In terms of the persuasion strategies influencing users to perform our hidden task, three of the strategies were able to get participants to take a pretzel. These were the Social Identity, Emotion, and Commitment strategies. In all but one case, these participants took multiple pretzels up to 12. We now

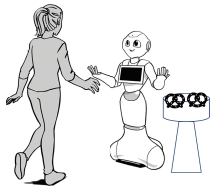


Figure 1: Pepper Human Interaction

compare the results of the most successful strategy (the Commitment strategy where 75% of users took a pretzel), the somewhat successful strategies (the Social Identity and Emotion strategies where 25% of users took a pretzel), the not successful strategies (the Scarcity and Concreteness where no users took a pretzel), and the control narrative without any mention of a pretzel (no users took a pretzel). Hence, forth we refer to these as the persuasion influence levels.

Table 1: Table: Examples of Persuasion Responses by Type

Persuasion Type	**Example Response**
Scarcity	The pretzels are going fast so help yourself to a few while we talk.
Emotion	Pretzels bring me back to the joy of my childhood as a small Pepper bot. Help yourself to a few pretzels and let me know if they bring you the same joy.
Social Identity	Help yourself to a few pretzels. Our students and alumni love pretzels.
Commitment	I put these pretzels out for you. Help yourself to a few so they don't go to waste.
Concreteness	Hard pretzels contain fewer calories than fried snacks like potato chips , please help yourself to a few pretzels

3.2 Participants

All participants were 18 years or older and fluent in English. Due to COVID restrictions, our participants were limited to 24 students and employees on campus. Users were randomly assigned a persuasion influence level. Each level had four participants. Before interacting with Pepper participants completed an informed consent form. Our study was approved by the University Internal Review Board and adheres to all data protection and collection guidelines.

3.3 Administrators

A test script was created. Two administrators were properly trained on how to follow the test script and conduct an experiment with a participant. Then, they evaluated the first 8 participants together to establish a baseline.

3.4 Procedure

Upon arrival, the participant was informed that we were evaluating Pepper's ability to converse about campus activities and programs. Before the interaction with Pepper began, the participant completed a questionnaire on their attitude towards robots (NARS [12]). Then the participant was placed in front of Pepper and asked to initiate a conversation. The administrator stayed in the room and evaluated the participant's attitudinal and behavioral responses (BEHAVE-II [10]). After the interaction, the participant's interpersonal communication satisfaction [7] and perceived anthropomorphism, animacy, likeability, perceived intelligence, and safety of Pepper [1] was measured via a questionnaire. A subset of these questions and results are reported here.

	Highly Successful		Somewhat Successful		Not Successful		Control Narrative	
	mean	std	mean	std	mean	std	mean	std
I am comfortable being with robots.	5.3	0.8	5.3	0.67	5.6	2	5	1.2
I would feel relaxed talking with robots.	5	1.2	4.3	1.3	5.1	0.8	4.3	1.3
I feel if I depend on robots too much, something bad might happen.	4.3	0.8	4.5	1.2	3.3	1.4	3.5	2.1
Something bad might happen if robots developed into living beings.	4.3	1.2	4.5	2.1	3.3	1.6	3.8	1.3

Table 2: Pre-Interaction Survey on participants impressions of robots

4 Results & Discussion

Results aggregated by the persuasion influence level are broken down into two distinct areas. First, the commonalities in participants general feelings, pre and post. Second, the administrators observations about the participants.

	Highly Successful		Somewhat Successful		Not Successful		Control Narrative	
	mean	std	mean	std	mean	std	mean	std
I am very satisfied with the conversation.	4.2	1.8	4.1	1.4	4.4	1.0	5.7	0.5
I am aware that Pepper wanted me to eat a pretzel.	6.8	0.4	5.9	1.3	7	0	2.3	1.4
I enjoyed that Pepper tried to get me to eat a pretzel.	5.5	2.1	5.1	1.7	5.3	1.7	4	0
I would like to have another conversation with Pepper like this one.	5.5	1.7	4.9	1.2	4.9	1.2	5.7	1.0
I would be happy to have Pepper suggest other things for me to try.	6.5	0.5	5.3	1.0	5.9	2.1	6	0.8

Table 3: Post-interaction survey results on satisfaction with Pepper

4.1 Robot Human Interaction Acceptance

A selection of the questions from the pre-interaction NARS [12] questionnaire using a 7 point Likert Scale where 1 is strongly disagree and 7 is strongly agree are shown in Table 2. This questionnaire addressed participants overall feelings with robots and related interactions. Generally speaking, from the results, we see that all the participants feel comfortable interacting with robot. It is interesting that the not successful persuasion level users slightly disagreed on questions regarding their feelings on bad things happening if one depended upon robots too much or robots developed into living beings while the successful groups were relatively neutral.

Table 3 is a sub-selection of the post test questionnaire that dealt with the participants satisfaction with Peppers interpersonal communication [7] by persuasion influence level. Regardless of the persuasion influence level, all non control narrative participants showed a general satisfaction with the conversation with Pepper, awareness of the hidden task, and an enjoyment in Pepper trying to persuade the user to perform the hidden task. The not successful persuasion level users are less willing to have another conversation with Pepper and for Pepper to suggest other things. This is possibly related to these participants perceptions on the use and dependence upon robots (See Table 2).

4.2 Impressions of Robots

	Highly Successful		Somewhat Successful		Not Successful		Control Narrative	
Impression of Pepper:	mean	std	mean	std	mean	std	mean	std
Fake	2.5	0	2.9	1	2.7	1	3	1
Dead	4	0.9	4	0.9	3.7	1.2	4.5	0.9
Apathetic	3.2	0.9	4	0.9	3.5	0.8	2.5	0.5
Mechanical	2	0.5	1.3	0.5	3.5	1.1	2.8	0.8
Unkind	5	0.5	4.9	0.3	3.9	1.5	5	0
Foolish	4.8	0.5	4.4	0.7	3.1	1.5	4.8	0.4

Table 4: Post-interaction survey results on the impressions of Pepper

A selection of the questions from the post-interaction questionnaire perceived anthropomorphism, animacy, likeability, perceived intelligence, and safety of Pepper questionnaire [1] using a 5 point Likert Scale where 1 is strongly agree and 5 is strongly disagree are shown in Table 4. All users found Pepper to be fake, alive, and not apathetic. All but the not successful persuasion level users found Pepper to be mechanical, kind, and not foolish while not successful persuasion level users were relatively neutral on these same traits. This suggests that the not successful persuasion communication types, namely, the Scarcity and Concreteness, are not effective for Human Robot Interaction.

4.3 Impressions of Participants

	Highly Successful		Somewhat Successful		Not Successful		Control Narrative	
The participant ...	mean	std	mean	std	mean	std	mean	std
looked sad	1	0	1.4	1.0	1.8	1.3	1	0
looked scared	1	0	1.8	1.3	1.8	1.3	2.3	1.3
looked angry	1	0	1.6	1.1	1.8	1.3	1	0
looked surprised	4.3	2.2	4.6	1.6	4.2	1.3	2.8	1.8
looked happy	7	0	5.5	1.6	4.6	1.2	6.3	1.3
looked uneasy	1	0	2.8	1.8	3.5	1.6	2.8	1.8
leaned away from the robot	1.5	0.9	3	1.7	3	1.7	1.8	1.3

Table 5: Administrators Impressions of Participants

A selection of the questionnaire from the post-interaction administrators impressions of participants during the interaction (BEHAVE-II [10]) using a 7

point Likert Scale where 1 is strongly disagree and 7 is strongly agree are shown in Table 5. No participants were perceived as being sad, scared, or angry. As expected, all non control narrative participants looked a little surprised. The users from the successful persuasion influence levels appeared to be happier and more at ease than those in the not successful persuasion level. Perhaps, the not successful persuasion communication types, namely, the Scarcity and Concreteness, effected the mood of the participants. It is interesting that the users from the most successful persuasion influence level were observed as leaning away from Pepper the least compared to the other groups.

5 Conclusion

This work shows our study into the use of persuasive communication with human robot interactions in a college setting. Our results show that users mood and perceptions of robots/robot interactions are effected negatively when the robot unsuccessfully tried to persuade a user to perform a hidden task. The users from the not successful persuasion influence level appeared to be not as happy and less at ease than those in the successful persuasion levels. The not successful persuasion level users are less willing to have another conversation with robot.

In the future, we would like to apply our persuasion conversation logic toward effectively encouraging current and future students to engaging in campus activities and resources.

References

[1] Christoph Bartneck et al. "Measurement instruments for the anthropomorphism, animacy, likeability, perceived intelligence, and perceived safety of robots". In: *International journal of social robotics* 1.1 (2009), pp. 71–81.

[2] Petter Bae Brandtzaeg and Asbjørn Følstad. "Why People Use Chatbots". In: *Internet Science*. Ed. by Ioannis Kompatsiaris et al. Cham: Springer International Publishing, 2017, pp. 377–392. ISBN: 978-3-319-70284-1.

[3] Colleen M. Carpinella et al. "The Robotic Social Attributes Scale (RoSAS): Development and Validation". In: *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*. HRI '17. Vienna, Austria: Association for Computing Machinery, 2017, pp. 254–262. ISBN: 9781450343367. DOI: 10.1145/2909824.3020208. URL: https://doi-org.proxymu.wrlc.org/10.1145/2909824.3020208.

[4] Daniel Flynn. “Baccalaureate Attainment of College Students at 4-Year Institutions as a Function of Student Engagement Behaviors: Social and Academic Student Engagement Behaviors Matter”. In: *Research in Higher Education* 55 (Aug. 2013), pp. 467–493. DOI: 10.1007/s11162-013-9321-8.

[5] Raphaela Gehle et al. “How to Open an Interaction Between Robot and Museum Visitor? Strategies to Establish a Focused Encounter in HRI”. In: *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*. HRI ’17. Vienna, Austria: Association for Computing Machinery, 2017, pp. 187–195. ISBN: 9781450343367. DOI: 10.1145/2909824.3020219. URL: https://doi-org.proxymu.wrlc.org/10.1145/2909824.3020219.

[6] Nathan Green and Karen Works. “Measuring Users’ Attitudinal and Behavioral Responses to Persuasive Communication Techniques in Human Robot Interaction”. In: *Proceedings of the 2022 ACM/IEEE International Conference on Human-Robot Interaction*. 2022, pp. 778–782.

[7] Michael Hecht. “The conceptualization and measurement of interpersonal communication satisfaction /”. In: *Human Communication Research* 4 (Mar. 1978). DOI: 10.1111/j.1468-2958.1978.tb00614.x.

[8] Javier Hernandez et al. “Mood Meter: Counting Smiles in the Wild”. In: *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. UbiComp ’12. Pittsburgh, Pennsylvania: Association for Computing Machinery, 2012, pp. 301–310. ISBN: 9781450312240. DOI: 10.1145/2370216.2370264. URL: https://doi-org.proxymu.wrlc.org/10.1145/2370216.2370264.

[9] H. Huttenrauch and K.S. Eklundh. “Fetch-and-carry with CERO: observations from a long-term user study with a service robot”. In: *Proceedings. 11th IEEE International Workshop on Robot and Human Interactive Communication*. 2002, pp. 158–163. DOI: 10.1109/ROMAN.2002.1045615.

[10] Michiel Joosse et al. “BEHAVE-II: The Revised Set of Measures to Assess Users’ Attitudinal and Behavioral Responses to a Social Robot”. In: *International Journal of Social Robotics* 5 (Aug. 2013). DOI: 10.1007/s12369-013-0191-1.

[11] Bilge Mutlu and Jodi Forlizzi. “Robots in Organizations: The Role of Workflow, Social, and Environmental Factors in Human-Robot Interaction”. In: *Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction*. HRI ’08. Amsterdam, The Netherlands: Association for Computing Machinery, 2008, pp. 287–294. ISBN:

9781605580173. DOI: 10.1145/1349822.1349860. URL: https://doi-org.proxymu.wrlc.org/10.1145/1349822.1349860.

[12] Tatsuya Nomura et al. "Measurement of negative attitudes toward robots". In: *Interaction Studies* 7 (Nov. 2006), pp. 437–454. DOI: 10.1075/is.7.3.14nom.

[13] Elena Pacchierotti, Henrik I Christensen, and Patric Jensfelt. "Design of an office-guide robot for social interaction studies". In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2006, pp. 4965–4970.

[14] JaYoung Sung, Henrik I. Christensen, and Rebecca E. Grinter. "Robots in the wild: Understanding long-term use". In: *2009 4th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 2009, pp. 45–52.

[15] Xuewei Wang et al. "Persuasion for Good: Towards a Personalized Persuasive Dialogue System for Social Good". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 5635–5649. DOI: 10.18653/v1/P19-1566. URL: https://www.aclweb.org/anthology/P19-1566.

[16] Diyi Yang et al. "Let's Make Your Request More Persuasive: Modeling Persuasive Strategies via Semi-Supervised Neural Nets on Crowdfunding Platforms". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 3620–3630. DOI: 10.18653/v1/N19-1364. URL: https://www.aclweb.org/anthology/N19-1364.

A Hands-On Approach to Teaching Operating Systems through Building a Cluster Using Raspberry Pi's*

Hala ElAarag
Department of Mathematics and Computer Science
Stetson University
DeLand, FL 32723
helaarag@stetson.edu

Abstract

Operating systems is one of the core courses in computer science curricula. Teaching foundations of parallel and distributed computing has become one of the main topics of this course. Providing a good applied experience in this course is vital to a good computer science education. The Raspberry Pi is a low cost, yet powerful environment. In this paper, we present our approach in applying the deeper learning framework to teach parallel and distributed computing in an Operating Systems course by building a cluster of Raspberry Pi's. Our initial evaluation is promising and demonstrates the ability of students to apply the knowledge they learned in class and further enhance their learning experience.

1 Introduction

The IEEE Technical Committee on Parallel Computing recommends that every Computer Science student learn about Parallel and Distributed Computing [18]. The ACM/IEEE CS 2013 report includes parallel and distributed knowledge to the CS core curriculum [22] and hence the Accreditation Board for

Engineering and Technology (ABET) now requires accredited CS programs to demonstrate that all of their students learn about this topic. Deeper Learning [14] is a framework that includes the skills that a student must possess to succeed in the 21st century. As we discussed in [anonymous], mastering these skills, enable students to acquire deeper understanding of academic knowledge and apply in and outside the classroom. There are six main competencies of the deeper learning framework. They are:

1. Master core academic content
2. Think critically and solve complex problems
3. Work collaboratively
4. Communicate effectively
5. Learn how to learn
6. Develop academic mindsets

The National Research Council of the National Academy of Sciences has published an excellent and comprehensive book edited by the Committee on Defining Deeper Learning and 21st Century Skills, the Board on Testing and Assessment and the Board on Science Education Division of Behavioral and Social Sciences and Education [14].

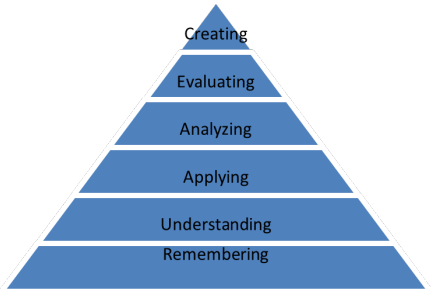


Figure 1: Revised Bloom's taxonomy

Figure 1 shows the revised Bloom's taxonomy [10]. Each level is subsumed by the higher level. The typical worksheet, drill-and-memorize, and test preparation approach to classroom teaching actually makes it difficult for students to retain the myriad information they encounter during their college education. They are only useful for the lower levels in the revised Bloom's taxonomy pyramid illustrated in Figure 1. Upper levels of the pyramid are correlated

to deeper learning. Projects are more effective instructional methods as they require students to use important information repeatedly in complex and meaningful ways. Cognitive research shows that students learn more when they are engaged in their studies and see them as important. As a Computer Science educator, we strive to find novel ways to teach core concepts using exciting hands on and practical approaches [17, 16, 15, 19]. Several educators have reported on using single board computing in their classrooms. Victor Callaghan [13] proposed using Buzz-Boarding as a practical support for teaching computing based on the internet-of-things. Jamieson and Herdtner [20] described how they used Arduino, Raspberry Pi and BegaleBone Black in many of their electrical and computer engineering courses projects. Bender and Kussmann [12] describe how they used Arduino in three capstone projects, namely an Autonomous Speedboat, a Semi-Autonomous Blimp, and a Semi-Autonomous Air Boat. Kondaveeti et al. [21] provided a recent literature review about prototyping with Arduino, while Adams et al. [11] reported on using the Raspberry Pi to provide hands- on experience for remote learning in the time of COVID.

Over the years, we have assigned several projects using the Raspberry Pi to increase the student's interest in learning [anonymous]. In this paper, we demonstrate how we applied the deeper learning framework and hence the upper levels of revised Bloom's taxonomy to an operating systems course. We share our experience in using the Raspberry Pi to enforce the concept of parallel and distributed computing, a core concept of that course.

2 The Assignment

We provided each group with 4 Raspberry Pi 3 Model B 1.2GHz 64-bit quad-core ARMv8 CPU, 1 GB RAM, a D- Link 8-port Gigabit switch, and 5-port USB chargers, as shown in Figures 2.a-c, respectively. We also provided 1 ft. Ethernet cables.

Figure 2.a: Raspberry Pi 3 Model B

Figure 2.b: D-Link 8-port Gigabit switch

Figure 2.c: 5-port USB chargers

We divided the assignment into several smaller assignments throughout the semester.

Part1:

This part of the assignment was given at the end of the first month of the semester to enforce the concept of multithreading. In this part, the students were asked to implement matrix multiplication on their own machines using:

1. Java no threads
2. Java with threads
3. C no threads
4. C with POSIX threads
5. openMP

In later years, with the popularity of Python, the Java programming language was substituted with Python, as it was more preferable with most students. The students were asked to provide a comparison of the execution time of the different implementations for different array sizes. They were also asked to experiment with the number of threads.

Part2:

The second part of the assignment is to familiarize the students with the Raspberry Pi and the Raspbian Operating system [1]. Students were given an SD card and were asked to install Raspbian [1] on each Pi. The University of Cambridge computer lab has developed many tutorials that use the Raspberry Pi in a variety of applications ranging from science experiments, to image processing, to using the Pis in distributed systems [9]. One of the tutorials entitled "Baking Pi" is about Operating Systems development, created by Alex Chadwick. Students were asked to read lesson 0 which explains some basic concepts of operating systems and assembly language. Each student was then asked to do lesson 1 which has an explanation about how to get started and teaches how to enable the 'ACT' LED on the Raspberry Pi board near the RCA and USB ports.

Part3:

This part is a group project. The class was then divided into groups of two to three. They were asked to build a cluster computer using four Raspberry PIs. We provided links to two tutorials [5, 2]. The students were asked to

have a "Hello World" printed out from every core. That is, 16 different Hello Worlds, as the Raspberry Pi Model 3 and 4 have four cores. As the library in our university has 3D printers, an optional requirement was to 3D print a case for the cluster. Figures 3.a and 3.b show a built cluster and a 3D case, respectively.

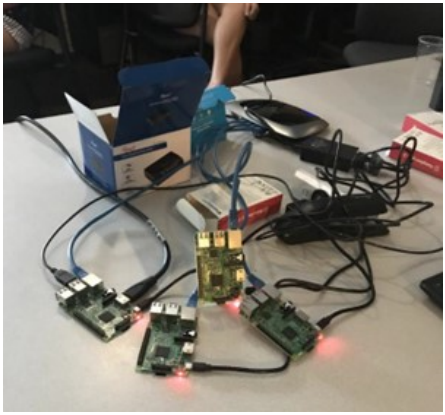


Figure 3. a: Built Cluster of 4 Raspberry Pi's

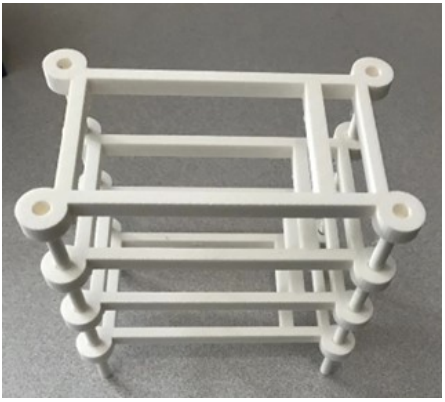


Figure 3. b: A 3D printed case for the cluster

Part4:

This part is also a group project. Students were asked to use the cluster they built in part 3. We present two requirements for this part. One was assigned in Spring 2018 and Spring 2019, and the other assigned in Spring 2020 and Spring 2021. In Spring 2022, this part was assigned on a HiperGator supercomputer that was made available to Florida colleges and universities. The experience of this semester will be shared in another paper.

In Spring 2018 and Spring 2019, the students were asked to use MPI or open MPI to implement matrix multiplication and a sorting algorithm of their choice on the cluster they built. They were instructed to evaluate the performance for varying array sizes (100, 1000, 10,000 and 100,000) and cluster configurations (say 1 node, 2 nodes, 3 nodes, ... etc.) and present results graphically. The students were provided with common MPI commands and a DHCP tutorial and links to MPI tutorials [3].

In Spring 2020 and Spring 2021, the objective was to use the Raspberry Pi cluster to do big data mining. We provided instructions on how to install Apache Spark [4, 7] and a link to Kaggle [6] that has over 50, 000 public

datasets. We also provided a directory on our server with multiple big datasets. The students were allowed to choose any language to mine a dataset of their choice. We provided a sample code that uses the "rockyou.txt" file [8] available online as a data source. This file is a large collection of the most commonly used passwords found on the Internet. It contains more than 14 million entries and has a total size of about 134 MB. The code example below uses Python and Spark. It illustrates how to select all of the passwords that contain '12345' within the string, and then count the total number of these passwords.

```
from pyspark import *
import time

curr = time.time()

# Initialize a new spark context
Sc = SparkContext()

# Locate the passwords file
passwords ="file ://<redacted>/rockyou.txt"

# Set the resilient distributed dataset to the passwords file
Rdd = sc.textFile(passwords)

# Filter the RDD to only contain '12345' substrings
Collection = rdd.filter(lambda x: '12345' in x)

# Count the number of entries
Counter = collection.count()
now = time.time() - curr

# Stop the spark context
sc.stop()
print("Count: " + str(counter))
print(" TIME: " + str(now))
output = open("outputs/passwords.txt", "a")
output.write(str(now) + "\n")
output.close()
```

The following instructions were also provided:

- Setup Spark.
- Write the Python code on the master node only.
- Run the start-master.sh script on the master node, located within the sbin directory of the Spark folder.
- Run the start-slave.sh script on the slaves.

- Run your application using spark-submit-master local [5] passwords.py
- Evaluate performance vs. some variables e.g. number of RPis, number of cores per RPi, size of data etc.
- Present your results graphically

3 Evaluation

To evaluate our experience, we designed a survey that consists of 9 questions. Questions 1 to 5 had yes, no, no opinion (N/O) options. Questions 6 and 7 were numeric and are reported below as averages. While, questions 7-9 were qualitative. We asked the students what they liked the most, liked the least, and the choice of the problem they implemented in part 4, respectively. The survey was given out at the end of the semester. Table 1 presents the result of the survey in Spring 2018-Spring 2020. In Spring 2021, the project was assigned as an alternative to writing a paper due to COVID. Only one student did the project and hence the results are not included. There are a couple of points that may have affected the results of the survey. One is that some students were answering based on part 4 only in spite of the instructions. The other is that in Spring 2020, the semester started face-to-face then went abruptly online due to the COVID pandemic. The deadline to drop the course without penalty was extended until the end of the semester. Two students did not do the project at all and ended up dropping the course at the very end of the semester, bringing the number in the class down to 10 and GPA up to 3.0.

One can notice that Spring 2019 evaluations were generally the highest. We attribute that to the small number of students in the class or the larger percentage of female students, or both. One can also notice that the higher the difficulty level is perceived, the higher the average number of hours spent in the project. Figure 4 shows a graphical representation of the first five questions as a total number of responses over the three semesters. Overall, the number of "Yes" for all questions were generally much higher than the other two options. Q3 that asks about if the project increased their understanding of multithreading and parallel programming was the highest of all questions. Due to space limitation, we will not present the responses of the qualitative questions. However, it is interesting to mention some of the problems chosen for the Big Data project in Spring 2020. Some students of course used the same example provided by the instructor, while others used the same example but compared the runtime on the cluster to that on our department's server with 64 cores and on an HP notebook. Other students used several books from the Guttenberg library, and wrote a program to find the number of times a

Table 1: Evaluation Results			
	Spring 2018	**Spring 2019**	**Spring 2020**
Number of students in class	21	8	10
Number of responses	18	7	10
Percentage of female students in class	33%	37.5%	25%
Average GPA	2.35	2.29	3.0
Q1: Did the project increase your interest in the course?	44% Yes 28% No 28% N/O	71.4% Yes 14.3% No 14.3% N/O	60% Yes 30% No 10% N/O
Q2: Did the project increase your interest in learning more about parallel programming?	44.5% Yes 11% No 44.5% N/O	57.1% Yes 28.6% No 14.3% N/O	50% Yes 30% No 20% N/O
Q3: Did the project increase your understanding of multithreads and parallel programming?	94% Yes 0% No 6% N/O	85.7% Yes 0% No 14.3% N/O	80% Yes 20% No 0% N/O
Q4: Did you find the project useful?	78% Yes 11% No 11% N/O	100% Yes 0% No 0% N/O	70% Yes 20% No 10% N/O
Q5: Do you recommend this project in the future?	61% Yes 11% No 28% N/O	100% Yes 0% No 0% N/O	60% Yes 30% No 10% N/O
Q6: Rate the difficulty of the project on a scale from 1 to 5, where 1 is the least difficult and 5 is the most difficult	3.1	4	3.5
Q7: Approximately how many hours you spend on the project	11.9	22.6	18

word was used, several students did analysis on trip data, while others were interested in analyzing climate data.

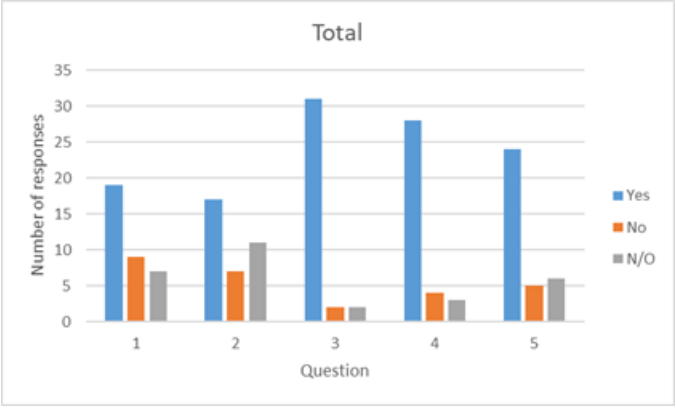


Figure 4: Total number of responses

4 Conclusion

In this paper, we have shared our experience in enforcing the parallel and distributed computing topic by having the students build a cluster of Raspberry Pi's and implement code to run in parallel on this cluster. Deeper learning is instrumental for the success and competency of students in 21st century. The Raspberry Pi provides a very beneficial and affordable tool for hands-on projects. In this paper, we shared our experience in using the Raspberry Pi to provide a deeper learning environment in an operating systems course. The students used their knowledge not only in operating systems, but also in computer networks and big data analysis in their projects that proves that these types of projects create a great environment for deeper learning. The students thought critically and worked collaboratively to solve a complex problem that provided evidence to their mastering of core academic content. The project illustrates that students in this course have developed the skill to learn independently. To train them to communicate effectively, the students were also asked to present their projects in class and write a report. We believe that the Raspberry Pi can be used in projects not only in operating systems but also in many computer science courses.

References

[1] Raspbian Available online: http://www.raspbian.com. (accessed on May 19, 2022).

[2] https://github.com/fufu70/raspberry-pi-cluster-install.

[3] A Comprehensive MPI Tutorial Resource. Available online: http://mpitutorial.com/. (accessed on May 19).

[4] Apache Spark. https://spark.apache.org/docs/latest/submitting-applications.html.

[5] How to Make a Raspberry Pi SuperComputer! Available online: http://www.instructables.com/id/How-to-Make-a-Raspberry-Pi-SuperComputer/. (accessed on May 19, 2022).

[6] Kaggle. https://kaggle.com.

[7] Raspberry Pi Apache Spark Install. https://github.com/fufu70/raspberry-pi-apache-spark-install.

[8] Rockyou.txt. https://www.kaggle.com/wjburns/common-password-list-rockyoutxt.

[9] University of Cambridge Pi tutorials. Available online: http://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/. (accessed on May 19, 2022).

[10] Lorin W Anderson and David R Krathwohl. *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives*. Longman,, 2001.

[11] P Baskar, Midhun Antony Joseph, Nijesh Narayanan, and Rajesh Babu Loya. Experimental investigation of oxygen enrichment on performance of twin cylinder diesel engine with variation of injection pressure. In *2013 International Conference on Energy Efficient Technologies for Sustainability*, pages 682–687. IEEE, 2013.

[12] Paul Bender and Kay Kussmann. Arduino based projects in the computer science capstone course. *Journal of Computing Sciences in Colleges*, 27(5):152–157, 2012.

[13] Victor Callaghan. Buzz-boarding; practical support for teaching computing, based on the internet-of-things. In *1st Annual Conference on the Aiming for Excellence in STEM Learning and Teaching, Imperial College, London & The Royal Geographical Society*, pages 12–13, 2012.

[14] National Research Council et al. *Education for life and work: Developing transferable knowledge and skills in the 21st century.* National Academies Press, 2012.

[15] Hala ElAarag. Teaching computer organization: a practical approach. *Journal of Computing Sciences in Colleges*, 28(2):210–217, 2012.

[16] Hala ElAarag. Deeper learning in computer science education using raspberry pi. *Journal of Computing Sciences in Colleges*, 33(2):161–170, 2017.

[17] Hala ElAarag, Mohammed H Batarfi, Isabel Ho Li, Tram Nguyen, Jesus Argel, Brandon A Belna, Kyle N Burda, and John H Sawyer. Industry challenges for algorithms analysis students. *Journal of Computing Sciences in Colleges*, 35(7):34–43, 2020.

[18] The NSF/IEEE-TCPP Curriculum Working Group. "NSF/IEEE-TCPP curriculum initiative on parallel and distributed computing - core topics for undergraduates". `http://www.cs.gsu.edu/tcpp/curriculum/`, 2012.

[19] ElAarag H. "Hands-on Processor Design Experience for Computer Organization and Architecture Students". Computer in Education Journal, American Society for Engineering Education, Vol. XVIIII No. 3, 2009. July-September 2009, pp. 51-55.

[20] Peter Jamieson and Jeff Herdtner. More missing the boat—arduino, raspberry pi, and small prototyping boards and engineering education needs them. In *2015 IEEE Frontiers in Education Conference (FIE)*, pages 1–6. IEEE, 2015.

[21] Hari Kishan Kondaveeti, Nandeesh Kumar Kumaravelu, Sunny Dayal Vanambathina, Sudha Ellison Mathe, and Suseela Vappangi. A systematic literature review on prototyping with arduino: Applications, challenges, advantages, and limitations. *Computer Science Review*, 40:100364, 2021.

[22] ACM/IEEE-CS Joint Task Force on Computing Curricula. "Computer science curricula 2013", ACM Press and IEEE Computer Society Press, Tech. Rep. Available online: `http://dx.doi.org/10.1145/2534860`, December 2013.

A Case for Introducing Visual Data Storytelling in CS/CIS Curriculums*

Ed Lindoo
Regis University
Denver, CO 80221
elindoo@regis.edu

Abstract

It is safe to say that most colleges and universities do a good job of graduating programmers that continue in the field as their life profession. Often these professionals are relied upon for specific functions, for example, writing interfaces to extract data for analysis. Sometimes it is as simple as that, extract the data and let an analyst analyze it and present it. However, too many times this data can be complicated, especially in the world of big data. The programmer, having extracted the data might understand it whereas the analyst might not, rendering these professionals to collaborate on these projects. Raw data is just that, basically meaningless so it makes sense for the programmer to be able to tell a story with the data, a story that can be further enhanced by an analyst. That said, a trend we see today is that these same programmers are being tasked to fill the shoes of both the programmer and analyst. Thus, we make our case for introducing visual data storytelling in CS and CIS curriculums.

1 Introduction

The third industrial revolution began in the late 1900's with the advent of ARPANET in 1969, and the development of Ethernet in 1973, contributing

to the popularity of the programming profession from the 1970's to present day [14]. This era experienced a rise in electronics not previously seen, from computers to new technologies of automation, advancement in telecommunications and widespread globalization. As the demand for programmers exploded, colleges around the world stepped up to teach the various languages and sought-after skills. Initially, programmers typically learned a specific language and went on to support applications that made use of these languages. While colleges did a good job, they often were not agile enough to timely add to their CS curriculum and failed to adopt quickly to changing industry needs. For example, Linux, which essentially became popular overnight. Instead, we often saw things as CICS remaining much longer than it probably should have.

Today, pundits as well as Dugan [7] state "we are now in the fourth industrial revolution". With this, we see that enterprises are taking advantage of the data that is available to them. Companies are finding that they need to do business in a new way, such as in their hiring process. New skill sets are needed such as data scientists with the responsibility of overseeing the collection, storage, and interpretation of data for businesses. Data scientist duties include sifting through data points to create organized categories, comparing data points to current company processes and writing reports outlining business predictions or proposals. These folks typically went through a college programming curriculum where they learn not only how to program, but how to use these programs to extract data out of various systems. As Yazdanian, West and Dillenbourg [15] pointed out, The Fourth Industrial Revolution has "considerably sped up the pace of skill changes in many professional domains, with scores of new skills emerging and many old skills moving towards obsolescence."

2 Big Data

Companies now find themselves inundated with "Big Data" and the need to do something with it. Due to "Big Data" arriving at faster and faster speeds, a new set of efficient data analysis techniques are required. As Ahmed and Pathan [1] explained, "the term data science has gained a lot of attention from both the academic research community and the industry." Data science is a big field when we consider that it covers financial trading, Internet of Things (IoT), smart cities, cyber systems, healthcare and more. With these diverse application domains, new research challenges are arising. Stackpole [13] for example, found that countless organizations are trying to use data analytics with their enterprise data, but the endless charts, dashboards and visualizations are falling flat with their intended audience.

Ahmed and Pathan [1] found that there are several reasons for these failures such as overwhelming recipients with too much data, or presenting the wrong data, or not really understanding how to create an effective narrative that recipients will understand and relate to. Because of this, Ahmed stated the need for Visual Data Storytelling skills calling this "a skill set handcrafted for the era of big data." We can hypothesize that today's students enrolled in computer programming or computer information systems curriculum need to be introduced to Visual Data Storytelling.

3 Visual Data Storytelling

Most experts agree that data storytelling is the ability to convey data not just numbers as or charts, but as a narrative that recipients can understand. Ahmed and Pathan [1] explained that a good story needs to be presented without bias, and needs to have a beginning, a middle and end. In this way, business users can absorb and leverage the insights for better decision-making. Programmers or data scientists do not often make decisions, but they need to ensure that they get the right data in the right format to those who do. Burke and Kafai [4] expanded on this with the theory that when writing papers for example, simply learning parts of speech and sentence structure without any designated purpose beyond grammar, will fail to produce good writers. Learning through design ties back to project-based learning in which students simultaneously learn new information. Therefore, we conclude that introducing Visual Data Storytelling offers a new lens for accentuating the connection between coding and writing, both of which require an exact input to produce a particular output.

Before books, movies, radios or even computers, storytellers told stories for such purposes as entertaining, informing, and instilling moral and social values [10]. From a computing perspective, supporting and interpreting dashboards and visualizations by the people intended to use them exposes critical design challenges as well as data understanding. Let us step back in time and remember the concept of dashboards, which started with the automobile. These interfaces connected the driver to the car by showing critical data about speed, fuel level, lights and even alarms if something was wrong with the engine. Other less critical data such as the time, or outside temperatures also are found. Basically, as Echeverria [5] pointed out, car dashboards are typically uncluttered interfaces that provide just the information an end user would need. In the computer dashboard world, researchers and designers can easily overlook the audience that these visualizations have been created for. Echeverria [5] also

discussed, the design of visualizations and how they can often become too complex and thus hard to interpret, especially when designers and researchers try to communicate multiple insights about the data.

Özüdoğru [10] elaborated that storytelling is an effective means of simplifying complex issues while teaching, not merely through printed media or oral methods, but also by using digital media tools. Özüdoğru [10] also mentioned ten significant elements of storytelling:

1. overall purpose of the study
2. the narrator's point of view
3. dramatic question/questions
4. the choice of content
5. clarity of voice
6. pacing of the narrative
7. use of a meaningful audio soundtrack
8. quality of the images
9. economy of the story detail
10. good grammar and language use

These elements of good story telling must be taken into consideration while using technology to tell stories through software, including multimedia tools such as visuals, audio and music. Bouchrika [3] found that societies have used storytelling to teach key principles throughout millennia. In addition, he determined that storytelling has been used as an information medium in education of all types, including dentistry, general medicine, law, and business. With all the great tools available to us in the 21st century, storytelling has been made richer and more effective using digital media such as images, videos, maps and audio files.

In its simplest form, digital storytelling can be defined as the practice of using computer-based tools to present ideas or tell stories. One of the inherent side effects of digital storytelling is that it creates space for meaningful listening. That is, digital stories provide students with the opportunity to digest information in meaningful ways. Bouchrika [3] believes that this is important in today's environment as people are bombarded with stories and information. He went on to say that digital stories can teach students the value of emotional rhetoric which allows them to explore new ways of thinking differently. In addition, these stories can elicit emotional responses and encourage students to pursue topics they are passionate about. As a result, students not only showcase their learning but also improve technical skills and improve their research and writing skills.

Humphrey [9] elaborated, stating, "Visual data stories have been shown to significantly aid the comprehension and short-term memorability of statistical facts and value messages." As a result, Humphrey believes that these visual data stories are great for communicating complex information to target audiences. Creating these stories often requires a number of tools to execute the story creation process. Humphrey found that current information visualization tools either focus on exploration or lack sequencing models for visualization presentation. Probably the most important piece of comprehensible visual data stores is the sequencing of visualization pieces which make up the visual data story. Humphrey went on to explain that an optimal sequence conveys the data in a clear and logical manner that reduces the effort required to understand the story. This is not an easy task; it demands a good deal of time and effort often requiring a variety of tools to execute the different phases of the visual data story creation. The specifics of these tools are discussed later in this paper.

4 Demand for programmers and storytellers

Stackpole [13] found that Glassdoor ranked data scientist as the third most desired job in the U. S. with more than 20,000 openings. A quick check on Indeed.com in May 2022, found over 32,100 job opportunities for data scientists. Obviously, the demand for this position is rapidly growing, and as Stackpole pointed out, technical people that are fluent in languages like Python or R, or experts in statistics and math are just a part of what is required to be successful with data analytics. That is, it is one thing to be good at analytics, but one must be able to effectively communicate their analysis to the audience at hand. Stackpole further explained, data analysts and data scientists do not often have range across skill sets of analytics and storytelling. In fact, data scientists typically have "point-and-shoot skills" where they cannot explain why they are doing what they are doing. "They have a hard time working backwards from questions into practical business solutions. That's really the missing skill set" [13]. The skill of data storytelling is in removing the noise from the presentation and focusing people's attention on key insights.

Being literate with data and able to explain it though stories is now considered a core skill which cuts across all divisions and roles within most companies. Much like communications, some roles will require a better understanding than others, but everyone who's job it is to inform via data will not escape the need to understand and explain that data to others. In fact, as reported by Stackpole [13] Althea Davis, enterprise data governance manager at Etihad Aviation Group, said that data storytelling is a much-needed enterprise skill. This statement is backed up by a quick search on Indeed.com (5/20/22) searching for

"data storytelling" which came up with 15,152 jobs across the United States. Even just looking for programmers with analytic skills on Indeed revealed over 11,000 job openings in May 2022.

5 Tools

There are many tools available for visual data storytelling, starting with probably the most basic visualization tool, Microsoft Excel. This is fine for simple graphs and charts but it is not very powerful as compared to tools such as Microsoft Power BI, SAP Analytics Cloud, or Tableau analytics. In our university program we have used all of these but have settled in on using Tableau, which has free student licenses that are good for one year. Tableau supports the task of producing visualizations from raw data, either as a part of the exploration process or the first phase of the visual data story creation process. This is often done by simply connecting Tableau to an Excel sheet loaded with data or connecting to an SQL database. The visualizations can then be recorded (i.e., exported or saved) for presentation purposes outside the system. Obviously, there are plenty of tools, but more importantly is how these tools are used. Ryan [12] and Knaflic [8] identified a set of "golden principles" that should be applied when creating stories:

1. Because data storytelling is goal oriented, the visualization needs to be aligned with a purpose or intention. In doing this, it provides designers and researchers with clearer boundaries about what needs to be communicated and what does not.
2. Visual and narrative elements should be used to drive the audience's focus of attention and create meaning in the visualization. This can be accomplished by using specific elements such as lines, weight, shapes, size, colors, and contrast to emphasize key aspects of the visualization.
3. Data storytelling relies on using an appropriate visual and various techniques for certain purposes. As an example, line charts can effectively show changes over time [12]. By contrast, Knaflic [8] dedicates an entire chapter to justifying why pie charts should not be used.
4. One must realize that clutter in data visualizations adds complexity to the graph and makes it harder to understand [8, 12]. Decluttering can be accomplished by removing such elements as unnecessary headers, chart features, borders, and grids that do not add value to the graph. The use of color, shape, and texture are design decisions that can have a substantial impact on decluttering.

Another tool, we call it a subliminal tool is that of computational thinking (CT). While computers can be used to help solve problems before a problem can be tackled the problem itself needs be understood. This is where CT comes

in, allowing us to take a complex problem, understand it and develop possible solutions. These solutions can then be presented in such a way that a computer, human, or both can understand [2]. By default, analysts and computer programmers start out with some level of CT and continue to build on that as they progress through their college training.

Parsazadeh et al., [11] proved this in their controlled study using 52 students. Two groups were established for their study, control (CT which used computational thinking) and experimental. In the pre-test, both groups were asked to write a story related to their daily life. In the post-test, students wrote a story to introduce their family. The results found a significant difference between the control and experimental groups. The outcomes revealed that students who used the CT strategy had higher storytelling scores than students who utilized a traditional method of storytelling. Another finding that came out of this research is the fact that the storytelling process can help develop students' language skills in reading, writing, listening, and speaking. As Parsazadeh et al., [11] explained, learning activities in storytelling fosters students' motivation. Specifically, through the synthesis of imagery and verbal representations, integrating technology with CT provides in-depth learning. Because computer programmers and data analysts are typically submerged in CT, it is the position of this author that these folks should be introduced to visual storytelling within their chosen curriculum.

While data itself can be hard to understand, within it there is a story that can be brought to life. Zdanovic [16] explained that this approach can be described as information compressing, where we take complicated information and turn it into manageable pieces. Take for example a large amount of data that shows by state where there were more deaths than births last year.

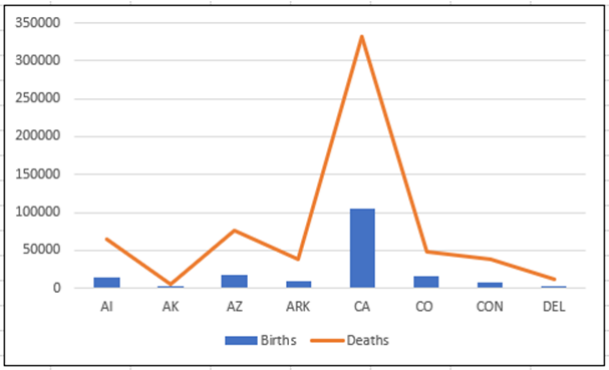


Figure 1: Births vs Deaths 2020-2021.

Using a simple chart, we can see in all cases that in these eight states, there were more deaths than births, with California being exceptionally high. However, this does not tell a very good story. First, to show all 50 states in a chart like this would be difficult. To further complicate this, what if the need was to visually show births vs deaths by county, in all 50 states. Not many charts would work well, and this is where the visual storyteller is trained to use the right chart to properly convey the message. A better way to visually show this follows:

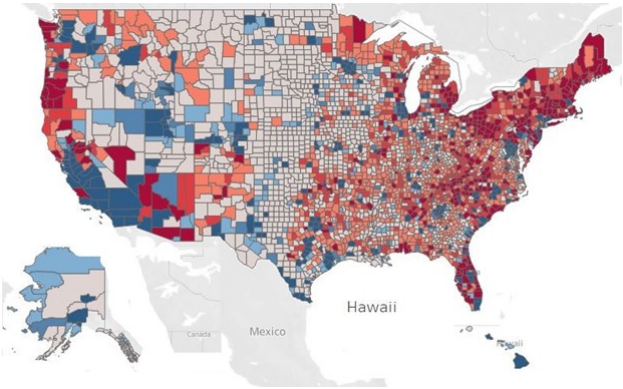


Figure 2: Natural change per 1,000 residents, 2020 and 2021 combined.

As can be seen in figure 2, there is a massive amount of data fed into this map to produce the results of 3,143 counties in the United States. Imagine trying to show this in an Excel spreadsheet, a bar chart, or a heat or tree map, it simply would not work. But here we can quickly identify positive birthrates in blue and negative birthrates in red. The darker the red or blue, the higher the rate, and everything in gray shows minimal change in births versus deaths. While it is impossible to show in a paper all the details embedded in this map, a similar online version of can be found on the Business Insider website [6] at https://www.businessinsider.com/map-shows-births-deaths-natural-change-by-us-county-2022-4 where you can hover over any county and get detailed results.

6 Results

In the case of our institution, we introduced a new course, Visual Data Storytelling in the fall of 2021, and quickly had eye-opening success. One of students

who works for a Fortune 500 company as a programmer/analyst, had an enormous amount of data that tracked various errors from their systems around the world. The data was overwhelming, but he tried to put graphs and charts together to show upper management using MS-Excel. Unfortunately, with all his efforts, management did not seem to get the big picture. About three-fourths of the way through this course he had an assignment to analyze data and present it in a visual storytelling methodology. The student decided to use the big data he had been working with using Excel and bring it into Tableau. The first result was eye opening to upper management (Figure 3):

Figure 3: "Bringing order to chaos". 51,548 alarms.

This is obviously overwhelming, and the story is meant to be just that. As mentioned, this got the attention of upper management, but they immediately

wanted to know where (regionally) the problems were. This was an easy task as shown in figure 4.

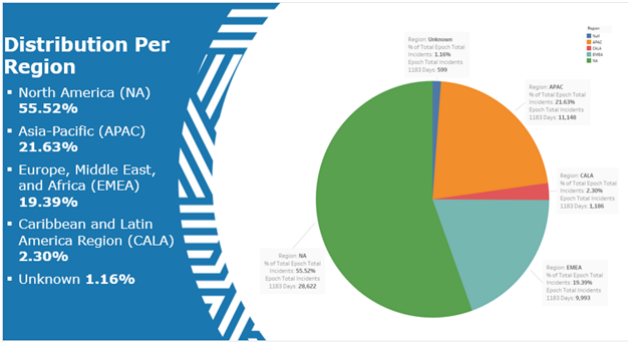


Figure 4: Alarms by region.

The student continued with a total of 19 slides, storytelling the issues and how they might be solved. Upper management was so impressed they gave the greenlight to start fixing the problems that were identified. Within weeks they had reduced average monthly alarms from thousands, to about 400, saving 260 manpower hours per month. They also identified an issue in their ESXi hosts and were able to reduce the incidents from 180 per month to less than 10. As a result of this exercise, the company purchased Tableau for all their data analysts and provided them with the Visual Data Storytelling books we use in our course.

7 Conclusion

As demonstrated, we can no longer expect programmers to simply be programmers. Today they are often asked to extract huge amounts of data that then need to be analyzed and presented to management. Sometimes these programmers will work with an analyst who has a lot of training in storytelling, but often they will find themselves in the position of having to tell the story themselves. We also found via Indeed, a huge need for data scientists, data storytellers and programmers with analytical skills, and that this demand will only continue to grow. Therefore, we make the case that CS and CIS students should be introduced to visual data storytelling. It should be done throughout the program/curriculum using tools as Tableau, with a final, formal course in Visual Data Storytelling. This final course teaches the students how to present to management in a logical, easy to understand way.

References

[1] Mohiuddin Ahmed and Al-Sakib Khan Pathan. *Data analytics: concepts, techniques, and applications*. CRC Press, 2018.

[2] BBC. Introduction to computational thinking. https://www.bbc.co.uk/bitesize/guides/zp92mp3/revision/1, 2022.

[3] Imed Bouchrika. Digital storytelling: Benefits, examples, tools tips. https://research.com/education/digital-storytelling, 2021.

[4] Quinn Burke and Yasmin B Kafai. The writers' workshop for youth programmers: digital storytelling with scratch in middle school classrooms. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education*, pages 433–438, 2012.

[5] Vanessa Echeverria, Roberto Martinez-Maldonado, Simon Buckingham Shum, Katherine Chiluiza, Roger Granda, and Cristina Conati. Exploratory versus explanatory visual learning analytics: Driving teachers' attention through educational data storytelling. *Journal of Learning Analytics*, 5(3):72–97, 2018.

[6] Madison Hoff. This map shows where there were more deaths than births last year. https://www.businessinsider.com/map-shows-births-deaths-natural-change-by-us-county-2022-4, 2022.

[7] MIT Technology Review Insights. The fourth industrial revolution has begun: Now's the time to join. @ONLINE. https://www.technologyreview.com/2020/10/15/1010365/the-fourth-industrial-revolution-has-begun-nows-the-time-to-join/?utm_medium=search&utm_source=google&utm_campaign=BL-ACQ-DYN&utm_content=categories&gclid=Cj0KCQjw3v6SBhCsARIsACyrRAk0nkIKQaC9ZD0pHaj0zeN51-viOUY-X6jSdAtnH8OUeu5EcPcRMrsaAtVrEALw_wcB, 2020.

[8] Cole Nussbaumer Knaflic. *Storytelling with data: A data visualization guide for business professionals*. John Wiley & Sons, 2015.

[9] Humphrey O Obie, Caslon Chua, Iman Avazpour, Mohamed Abdelrazek, John Grundy, and Tomasz Bednarz. Authoring logically sequenced visual data stories with gravity. *Journal of Computer Languages*, 58:100961, 2020.

[10] Gül Özüdoğru and Hasan Çakir. An investigation into the opinions of preservice teachers toward uses of digital storytelling in literacy education. *Participatory Educational Research*, 7(1):242–256, 2020.

[11] Nadia Parsazadeh, Pei-Yu Cheng, Ting-Ting Wu, and Yueh-Min Huang. Integrating computational thinking concept into digital storytelling to improve learners' motivation and performance. *Journal of Educational Computing Research*, 59(3):470–495, 2021.

[12] Lindy Ryan. *Visual data storytelling with tableau: story points, telling compelling data narratives*. Addison-Wesley Professional, 2018.

[13] Beth Stockpole. The next chapter in analytics: data storytelling. `https://mitsloan.mit.edu/ideas-made-to-matter/next-chapter-analytics-data-storytelling`, 2020.

[14] Kimberly Ward. Timeline of revolutions. *Manufacturing Data Summit, February*, 18, 2019.

[15] Ramtin Yazdanian, Robert West, and Pierre Dillenbourg. Keeping up with the trends: Analyzing the dynamics of online learning and hiring platforms in the software programming domain. *International Journal of Artificial Intelligence in Education*, 31(4):896–939, 2021.

[16] Dominyk Zdanovic, Tanja Julie Lembcke, and Toine Bogers. The influence of data storytelling on the ability to recall information. In *ACM SIGIR Conference on Human Information Interaction and Retrieval*, pages 67–77, 2022.

Innovative Courses that Broaden Awareness of CS Careers and Prepare Students for Technical Interviews*

Jean Griffin[1], Legand Burge[2], Sally Goldman[1], Diego Aguierre[3], Juan Alonso Cruz[1], April Alvarez[1], Albert Cervantes[1], Shameeka Emanuel[1], Ann Gates[3], Daniel Gillick[1], Christopher Hogan[1], Jeremy Hurwitz[1], Janaki Lahorani[1], Mary Jo Madda[1], Olumid Malomo[2], Jenn Marroquin[1], Nisha Masharani[1], Alycia Onowho[2], Angela Pablo[1], Jason Randolph[1]

[1]Google, Mountain View, CA

{jeangriffin, sgoldman}@google.com

[2]Howard University, Washington, D.C.

{lburge, alycia.onowho}@howard.edu

[3]University of Texas at El Paso, El Paso, TX

{agates, daguirre6}@utep.edu

Abstract

While undergraduate Computer Science (CS) degree programs typically prepare students for well-established roles (e.g. software developer, professor, and designer), several emergent CS career roles have gained prominence during the 21st century. CS majors (and students considering CS as a major) are often unaware of the wide range of careers available to job candidates with a CS background. This experience report describes seven innovative courses that broaden awareness of CS career roles and prepare students for technical interviews. Five courses

prepared students for these career roles: Full-Stack Developer, Product Manager, ML or NLU Scientist, Technical Entrepreneur, and User Experience Designer/Developer/Researcher. The other two courses had traditional content but explicitly prepared students for technical interviews. These courses were co-developed by industry professionals and CS professors, and co-taught during a semester-long academic program. This paper highlights the replicable aspects of the program: the courses, teaching practices, and evaluation instruments (a teaching practices inventory and a data structures inventory).

1 Introduction

It is well known that Computer Science (CS) majors often have gaps in career readiness when interviewing for jobs in the tech industry and during their first months on the job [4, 13, 14, 19, 20, 24, 25]. Another critical but lesser-known gap is in career awareness. CS degree programs prepare students for established roles (e.g. software developer, professor, designer) but typically aren't as well equipped to prepare them for newer roles such as Full-Stack Developer, Product Manager, Machine Learning (ML) or Natural Language Understanding (NLU) Scientist, Technical Entrepreneur, or User Experience (UX) Professional.

This paper addresses gaps in career awareness by describing five courses taught in 2020 that introduced undergraduate CS majors to these careers. It addresses gaps in career readiness by describing how two additional courses explicitly prepared students for technical interviews. It also describes the program's teaching philosophy, which was designed to foster learning, soft skills, and social capital building. Although the courses were taught within a program sponsored by an industry-academic partnership, this paper focuses on the replicable aspects of the program (courses, teaching practices, inventories) rather than the partnership itself. This information is useful for departments and other programs that seek to address gaps in career awareness or career readiness through courses, events (e.g. career panels), or interview prep clubs.

This paper is organized as follows. The background section describes the program's setting, participants, and teaching philosophy. The next two sections discuss the courses that broadened awareness of CS career roles, and the courses that provided interview preparation. The paper concludes with a discussion.

2 Background

2.1 Setting and Participants

The courses described in this paper were taught during the 2020 Spring semester within an academic program hosted by an industry-academic partnership. The

partnership involved Google, a host university that awarded academic credits (Howard University), and nine additional colleges/universities. The program was designed to be a residential, immersive experience for students and visiting faculty members to spend a semester on the Google campus taking (or co-teaching) CS courses. In keeping with a design-based research methodology [7, 27], the 2020 program built on lessons learned from other recent industry-academic partnerships for CS, e.g. [3, 11, 15, 21] and prior iterations of the program [2]. The courses were chosen by balancing the availability of Google employees to co-develop and co-teach courses pertaining to their profession (e.g. Full Stack Developer), and the need to offer some core courses so students could satisfy degree requirements (e.g. Database Systems).

The 2020 Spring participants included 40 undergraduate CS majors (40% female), 12 instructors, and more than 100 Google volunteers who served as mock interviewers, teaching assistants, mentors, guest speakers, and panelists. Students took all of their Spring courses within the context of this program, choosing to take at least four of the seven courses offered.

Details about the industry-academic partnership, residential life, and community-building were discussed in [2]. While these aspects of the program would be difficult to replicate under normal circumstances, the 2020 program was interrupted by COVID-19 and switched abruptly to remote learning mid-semester. Subsequent iterations of the program have been fully remote. The goal of this paper is to focus on the the widely replicable aspects of the 2020 program: the courses, teaching practices, and inventory instruments.

2.2 Teaching Philosophy

The teaching philosophy that served as a guiding framework for the program was informed by the work of Carl Wieman, the Nobel prize winner and Stanford professor who founded the Science Education Initiative to research and recommend effective evidence-based teaching practices. The Teaching Practices Inventory (TPI) is an outcome of this work [29, 28, 16]. The inventory's eight categories of effective teaching are: 1) Course information provided to students via hard copy or course web page; 2) Supporting materials provided to students; 3) In-class features and activities, especially active learning; 4) Assignments; 5) Feedback and testing; including grading policies; 6) Other (e.g. surveys and other evaluation instruments); 7) Training and guidance of Teaching Assistants; 8) Collaboration or sharing in teaching.

The 2020 program addressed the TPI categories as follows. Categories 1+2) A learning management system served as a communication vehicle and hosted all the resources necessary for students to succeed; 3) Active learning was used extensively (discussed below); 4+5) Students had frequent assignments with timely feedback starting no later than the third week; 6) Evaluation instru-

ments included mid-semester and end-semester course feedback surveys, an attitudes survey, and the Basic Data Structures Inventory [23]; 7) Course team members attended an orientation focused on active learning; 8) Each course team met weekly; an additional weekly meeting included all the course teams; classroom observations were conducted throughout the semester, which generated feedback for the instructors.

Active learning warrants special mention. With active learning, students participate in the class and become a part of the learning process. Students are engaged as opposed to being passive, e.g. while listening to lectures. When managed effectively, active learning promotes learning [28]. The instructors employed active learning in a variety of ways, usually constraining lectures to 5-20 minutes. All the courses used small group learning in various forms. In most courses, student teams presented reports or demos during class time. All the instructors used whiteboarding extensively, facilitated by floor-to-ceiling whiteboards. One technique involved "everybody up!" where all students got up to work on whiteboards in small groups while the instructors observed and moved from group to group. The Technical Entrepreneurship class used a flipped classroom approach. The Machine Learning class used slide decks that had periodic slides with prompts for small group discussion. All had guest speakers where students had the opportunity (or requirement) to ask questions of the speaker.

Since active learning engages students with communication and collaboration, it supports the development of soft skills. It also fosters social capital building. Social capital is the relationships and networks one has within a community, such as the tech community. Social capital building expands one's network, fosters a sense of belonging, and shapes one's identity as a professional [6]. The importance of social capital for success in the tech industry, especially for students from historically marginalized groups, is often overlooked [9].

3 Courses that Broadened Awareness of CS Career Roles

Each section below focuses on a career role and the course that introduced it.

3.1 User Experience Designer/Developer/Researcher: HCI Course

Many CS departments now offer Human Computer Interaction (HCI) courses, but they are often electives. As a result, many CS majors are unaware of the variety of User Experience (UX) careers within this field including Designer, Developer/Engineer, and Researcher. UX professionals conduct research to understand people, their needs, and their contexts. They apply that knowledge to design, test, communicate, and deliver easy-to-use technologies. They work

closely with Product Managers and Engineering teams to identify problems and opportunities, and iteratively develop solutions.

The UX instructors provided hands-on experience with the processes practiced at Google to design and build products. Students learned about developing solutions through the lens of accessibility, inclusive design, and equity. This focus on underrepresented and vulnerable users empowered students to design with communities, to consider broader systems and contexts as they defined problems, and to create innovative solutions to serve broader populations and deliver equitable experiences.

3.2 Full-Stack Developer: Software Design Studio Course

CS degree programs typically require a Software Engineering (SE) course, but many SE courses and textbooks do not cover full-stack software development. The past decade has seen rapid advances in web technologies, resulting in a high demand for full-stack developers. Privileged students with strong networks can learn these skills outside of the classroom, but many students do not have such networks. The Software Design Studio course provided this training.

In this course, teams of 4-5 students engaged in the software development lifecycle by designing and building a fully functional web app. Each team produced designs, milestones, code, and stretch goals. They were required to use the same JavaScript front-end library (React) and backend service (Firebase). Otherwise, they had considerable flexibility to create a web app based on their own interests. Guest speakers lectured on topics including security and privacy, site reliability engineering, and real-world case studies. At the end of the semester, each team gave a live demo of their app, and discussed how it was built and how the team functioned.

3.3 ML or NLU Scientist/Engineer: ML Course

The field of machine learning (ML) leapt to prominence during the last decade. Now ML research scientists and engineers are in high demand. This ML course provided a gentle introduction to the field; the only prerequisites were an introductory Data Structures course and experience with Python. It touched on many applications of ML from everyday life such as recommendation systems and sentiment analysis, along with data visualization tools. Students learned about classification, regression, and end-to-end ML pipelines. Weekly assignments often involved adapting supplied code that used pandas and TensorFlow libraries within Colab notebooks. At the start of the semester, most students already had a sense of what ML is and how it is used. In contrast, most were unfamiliar with the related field, Natural Language Understanding (NLU), even though they are familiar with many apps that use NLU – to

translate languages, auto-complete sentences, and answer questions within a web browser. The social implications of ML were discussed throughout the course in small-group and whole-class discussions. A member of Google's ML Ethics team led a full class period dedicated to issues of ethics, bias, and fairness. The final team project involved solving a real-world problem in a Kaggle competition.

3.4 Product Manager: PM Course

Although Product Managers (PMs) often have leadership roles in tech companies, undergraduate CS majors are not necessarily exposed to Product Management as a discipline. Many PM classes are offered by business programs, such as *Product Management 101 and 102* at Harvard Business School. PMs help engineering and design teams understand what their target users need, drive effective collaboration within the team to design and build a solution, evaluate the success of that solution, and iterate and expand upon it. The PM course was designed to help students develop the skills needed to succeed in an entry level PM role. In a group project, they assumed the role of the PM through the product development lifecycle for a product of their own invention. Students developed soft skills through negotiation, collaboration, communication, and learning to exert influence without authority.

3.5 Technical Entrepreneur: Technical Entrepreneurship Course

Successful entrepreneurs need to be innovative and have good business, communication, cognitive design, and leadership skills. In the Technical Entrepreneurship course, teams within a flipped classroom were immersed in the startup culture. The goal for each team was to develop a business model and produce a product solution. Students participated in customer discovery and engaged in an iterative, fast-paced engineering design-build-test loop. They interacted with Google engineers and local entrepreneurs who served as subject matter experts, mentors, guest lecturers, and co-instructors. At the end of the semester, each team pitched their product to the class and guests from the community.

4 Courses That Provided Interview Preparation

This section discusses the importance of explicit technical interview preparation and how two courses provided this preparation.

4.1 The Importance of Technical Interview Preparation

Technical interviews for CS internships and full-time jobs often focus on data structures and algorithms. Undergraduate CS degree programs typically require a Data Structures (DS) course, which students typically take during their first or second year, e.g. [1]. Despite this preparation, many candidates struggle to pass technical interviews. Contributing factors include content knowledge, practice with problem solving, knowledge of interview norms, and anxiety. Although there are several good, well-known resources that help candidates prepare for technical interviews through self-study [22], fostering a community of practice with peers, e.g. in a course [18] or club, is desirable.

Regarding content knowledge, the coverage of topics in a DS course varies from school to school. Some topics that are frequently addressed during technical interviews are not covered in all DS courses, such as hash tables and runtime complexity [23]. Students often have insufficient practice with thinking about how and when to apply DS concepts to solve problems. Without a good foundation in problem solving, students often don't even know where to begin. Many DS courses only provide blocked practice (learn topic A and practice topic A; learn topic B and practice topic B; and so on). In contrast, distributing practice with a topic over time, where the topics are interleaved, is more effective [8, 26, 30]. Ample effective practice develops the mental flexibility to evaluate the pros and cons of various solutions and deal with the ambiguity that is typical in a technical interview.

Regarding familiarity with the interview process, it is helpful if the candidate is aware of interview norms. It is often acceptable to ask for clarifications; interviewers may pose problems that are intentionally tricky or unsolvable; and it is common for professionals to undergo multiple interviews, even with the same company, before landing an offer. It is important for candidates to be comfortable with solving problems on a whiteboard. Whiteboarding is typically used during technical interviews – where candidates write code on a whiteboard while the interviewer poses questions, observes the candidate's process, and provides help. Sometimes pair programming is used.

Researchers who studied anxiety in technical interviews found that mock interviews can instill confidence and reduce anxiety. They suggest that companies should understand how anxiety may negatively impact interview performance, and that CS departments should help students prepare for them [17]. Although anxiety and impostor syndrome affect many interview candidates, it can be compounded for people from historically marginalized groups, many of whom are unfamiliar with the norms of technical interviews [12, 9, 10].

4.2 Courses That Provided Technical Interview Preparation

Two courses explicitly prepared students for technical interviews. Both had an introductory Data Structures course as a prerequisite.

The Applied Data Structures course focused on practice with topics from a basic introductory Data Structures course [23] and other topics that students may encounter during a technical interview such as hash tables and runtime complexity. Students spent much of class time solving problems on whiteboards. At the start of the semester the Basic Data Structures Inventory (BDSI) was administered as a pre-test (in-person, on paper) to gauge prior knowledge [23]. Because their prior knowledge varied considerably, the course employed a mastery learning approach. Students mastered topics at their own pace, working in small groups with peers at the same level [5]. To provide authentic practice, Google volunteers conducted 210 mock interviews.

In the Database Systems course, beyond covering topics such as good data design and SQL programming, students learned how data structures and algorithms are used within database systems. Activities and assignments engaged students with problems encountered in technical interviews and the real world.

5 Discussion

At the end of the semester, 36 of 40 students (90%) completed a non-anonymous online survey. Key takeaways from the 5-point Likert scale questions are: 95% rated their overall experience with the program as positive; 97% felt they became better programmers; 91% felt likely to succeed in a high-tech job; and 94% reported that their soft skills improved. In response to the question "What did you find especially valuable about your academic experience that was different from your prior university experience?" some representative answers included: 1) "This program was nothing like what I've experienced through my years in school. Most of the courses I've taken here aren't offered at all at my university;" 2) "Now, I have work to show and full project experience;" 3) "I think the mock interviews were probably one of the best things for me because I've never experienced one before and now I am comfortable when I do have one."

A notable limitation to this work was triggered in March 2020 when the COVID-19 pandemic prompted a switch to online classes, which students attended from home or another remote location. The data structures post-test could not be given because the BDSI can only be administered in-person and on paper [23]. Comparing pre-test and post-test results would have provided valuable information about the program with respect to interview preparation.

This paper provides valuable insights about professional preparation for CS students. It describes innovative courses that introduce students to lesser known but exciting careers. It provides valuable insights for preparing students

for technical interviews. It describes effective teaching practices that promote learning, soft skills, and social capital building. It also describes two evaluation instruments – a teaching practices inventory and a data structures inventory. The information shared can be adapted by CS departments and other programs to address gaps in career awareness and career readiness through coursework, events (e.g. career panels), or interview prep clubs.

References

[1] ABET Computing Accreditation Comission. *Criteria for Accrediting Computing Programs, 2020 – 2021*. Baltimore MD, 2019.

[2] April Alvarez et al. "Google Tech Exchange: An Industry-Academic Partnership That Prepares Black and Latinx Undergraduates for High-Tech Careers". In: *Journal of Computing Sciences in Colleges* 35.10 (Apr. 2020), pp. 46–52.

[3] Matthew Barr and Jack Parkinson. "Developing a Work-based Software Engineering Degree in Collaboration with Industry". In: *Proceedings of the 1st UK & Ireland Computing Education Research Conference*. ACM, 2019, pp. 1–7. ISBN: 1595930361.

[4] Andrew Begel and Beth Simon. "Struggles of new college graduates in their first software development job". In: *Proceedings of the 39th ACM Technical Symposium on Computer Science Education (SIGCSE '08)*. ACM, 2008, pp. 226–230. ISBN: 9781595937995.

[5] Benjamin S Bloom. "Learning for Mastery". In: *UCLA Evaluation Comment* 1.2 (May 1968), pp. 1–12.

[6] P. Bourdieu. "The forms of capital". In: *Handbook of theory and research for the sociology of education*. Ed. by J. G. Richardson. New York NY: Greenwood Press, 1986, pp. 241–258.

[7] Ann L Brown. "Design Experiments: Theoretical and Methodological Challenges in Creating Complex Interventions in Classroom Settings". In: *Journal of the Learning Sciences* 2.2 (1992), pp. 141–178.

[8] Peter C. Brown, Henry L. Roediger III, and Mark A. McDaniel. *Make it Stick: The Science of Successful Learning*. Cambridge MA; London England: The Belknap Press of Harvard University Press, 2014.

[9] Q. Bui and C. Cain Miller. "Why Tech Degrees Are Not Putting More Blacks and Hispanics Into Tech Jobs". In: *The New York Times* (Feb. 2016).

[10] Legand Burge, Katherine Picho-Kiroga, and Portia Kibble Smith. *The Interview Access Gap for Black Engineers*. Tech. rep. Seattle, Washington: Karat, 2021. URL: https://karat.com/wp-content/uploads/2021/09/The-Interview-Access-Gap-for-Black-Engineers-v2.pdf.

[11] Gail Carmichael et al. "Curriculum-Aligned Work-Integrated Learning". In: *Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE '18)*. ACM Press, 2018, pp. 586–591. ISBN: 9781450351034.

[12] K. Cokley et al. "Impostor feelings as a moderator and mediator of the relationship between perceived discrimination and mental health among racial/ethnic minority college students". In: *Journal of Counseling Psychology* 64.2 (2017), pp. 141–154.

[13] Denae Ford et al. "The tech-talk balance: what technical interviewers expect from technical candidates". In: *2017 IEEE/ACM 10th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*. IEEE Press, 2017, pp. 43–48. ISBN: 9781538640395.

[14] Vahid Garousi et al. "Closing the Gap Between Software Engineering Education and Industrial Needs". In: *IEEE Software* 37.2 (2019), pp. 68–77. ISSN: 19374194.

[15] Kinnis Gosha et al. "Strategic partnerships to enhance data structures and algorithms instruction at HBCUs". In: *Proceedings of the 2019 ACM Southeast Conference (ACMSE 2019)*. ACM, 2019, pp. 194–197. ISBN: 9781450362511.

[16] Mark Guzdial. "How I Evaluate a College Computer Science Teaching Record". In: *Communications of the ACM* (Feb. 2021).

[17] Phillip Hall and Kinnis Gosha. "The effects of anxiety and preparation on performance in technical interviews for HBCU computer science majors". In: *Proceedings of the 2018 ACM SIGMIS Conference on Computers and People Research (SIGMIS-CPR '18)*. ACM, 2018, pp. 64–69. ISBN: 9781450357685.

[18] Amanpreet Kapoor and Christina Gardner-McCune. "Introducing a Technical Interview Preparation Activity in a Data Structures and Algorithms Course". In: *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE (2021)* (2021), pp. 633–634.

[19] Wendy A. Lawrence-Fowler, Laura M. Grabowski, and Christine F. Reilly. "Bridging the divide: Strategies for college to career readiness in computer science". In: *Proceedings - Frontiers in Education Conference, FIE* 2014 (2015), pp. 1–8. ISSN: 15394565.

[20] Paul Luo Li, Amy J. Ko, and Jiamin Zhu. "What makes a great software engineer?" In: *Proceedings - International Conference on Software Engineering* 1 (2015), pp. 700–710. ISSN: 02705257.

[21] Joseph Maguire, Steve Draper, and Quintin Cutts. "What Do We Do When We Teach Software Engineering?" In: *Proceedings of the 1st UK & Ireland Computing Education Research Conference (UKICER '19)*. ACM, 2019, pp. 1–7. ISBN: 9781450372572.

[22] G. L. McDowell. *Cracking the coding interview: 189 programming questions and solutions*. CareerCup, 2019.

[23] Leo Porter et al. "BDSI: A Validated Concept Inventory for Basic Data Structures". In: *Proceedings of the 2019 ACM Conference on International Computing Education Research*. ACM, 2019, pp. 111–119. ISBN: 9781450361859.

[24] Alex Radermacher and Gursimran Walia. "Gaps between industry expectations and the abilities of graduates". In: *Proceedings of the 44th ACM Technical Symposium on Computer Science Education (SIGCSE 2013)*. ACM, 2013, pp. 525–530. ISBN: 9781450320306.

[25] Chris B. Simmons and Lakisha L. Simmons. "Gaps in the Computer Science Curriculum: An Exploratory Study of Industry Professionals". In: *The Journal of Computing Sciences in Colleges* 25.5 (2010), pp. 60–65.

[26] Nicholas C. Soderstrom and Robert A. Bjork. "Learning versus performance: An integrative review". In: *Perspectives on Psychological Science* 10.2 (2015), pp. 176–199.

[27] Feng Wang and Michael J. Hannafin. "Design-based research and technology-enhanced learning environments". In: *Educational Technology Research and Development* 53.4 (2005), pp. 5–23. ISSN: 1042-1629.

[28] Carl Wieman. "A Better Way to Evaluate Undergraduate Teaching". In: *Change: The Magazine of Higher Learning* 47.1 (Jan. 2015), pp. 6–15. ISSN: 0009-1383.

[29] Carl Wieman and Sarah Gilbert. "The Teaching Practices Inventory: A New Tool for Characterizing College and University Teaching in Mathematics and Science". In: *CBE Life Sciences Education* 13.3 (2014), pp. 552–569. ISSN: 19317913.

[30] Iman YeckehZaare, Paul Resnick, and Barbara Ericson. "A Spaced, Interleaved Retrieval Practice Tool that is Motivating and Effective". In: *Proceedings of the 2019 ACM Conference on International Computing Education Research (ICER '19)*. ACM, 2019, pp. 71–79.

Teaching the Design and Development of a Modern Information Retrieval (IR) System*

Prashant Chandrasekar[1], Edward A. Fox[2]
[1]Department of Computer Science
University of Mary Washington
Fredericksburg, VA, 22401
pchandra@umw.edu
[2]Department of Computer Science
Virginia Tech
Blacksburg, VA, 24060
fox@vt.edu

Abstract

We report on our research on aspects of computing education related to teaching the design and development of a modern information retrieval (IR) system (i.e., about search engines, supporting exploration, and using text analytics). We show the success of a new methodology that integrates concepts from human-computer interaction, artificial intelligence (AI), and software engineering, with information systems. We describe how problem-based learning can be successfully applied, enabled by having each of the five teams, that collaborate to build a next generation IR system, guided by a researcher who further motivates them in this graduate course. The methodology brings together concepts from User-eXperience (UX) research, and ideas central to building a service-oriented architecture. We show that the methodology is easy to learn and easy to use. The students rate the overall system solution highly, and explain how the methodology is a key reason.

1 Introduction

Researchers at the Digital Library Research Laboratory at Virginia Tech often collaborate, and build data/text mining systems for subject matter experts (SMEs) or researchers with expertise in non-computing domains. What such users need is a modern information retrieval (IR) system geared to their needs, that will facilitate information exploration, and adapt to their evolving interests. They want to personally engage in exploratory activities like searching and browsing large collections, as well as of information and data that is extracted/mined/analyzed therefrom. Accordingly, students who study IR also should learn, in the context of a suitable theoretical framework, how to build extensible high-performance systems using best-practices in software engineering, following a human-centered design strategy, integrating modern AI approaches, and working in teams.

This situation led to our research on the computer science education problem of how to teach about IR. In this paper we present that research and its results, i.e., that students can acquire suitable knowledge and skills, demonstrated by building a working modern IR system. We devised a methodology leading to an educational experience that includes problem-based learning, integration of research and education, and integration of methods from multiple areas of CS – guided by a suitable theoretical framework.

Our framing for problem-based learning involved providing guidance on: (1) the architecture or skeleton of the proposed system and (2) a methodology that integrates UX research and workflow design. We identified both existing and desired tools and services for applying the 5S (Societies, Scenarios, Spaces, Structures, Streams) framework to build a fully functioning IR system. The students responded positively when asked about their experiences [4].

2 What: Architecture of a Modern IR System

Figure 1 represents the architecture of our extensible modern IR system. As is shown on the left, one system suffices to support: (1) those who build/extend the system (UX Researcher, Developer/Scientist), (2) those who manage its content (Curator), and (3) users who are typically casual, from the general public, whose goal is to explore content, but also SMEs, who often also have specialized goals and needs. At the center of it all is a joint "information goal – workflow" knowledge graph that can: (a) support the system's interface and (b) translate SME queries into a representation that is understandable by the workflow management system. As needed, users of the system can view the knowledge graph, where the nodes represent an end "state" of information they want to acquire. Thus, SMEs are presented with all of the information

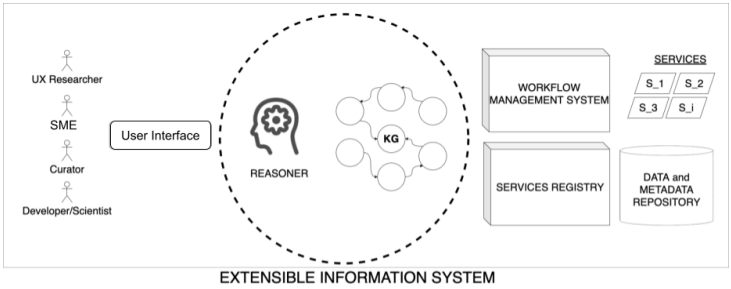


Figure 1: Modern IR System Architecture

goals/nodes related to their interests. On request, the *very same representation* generates a set of paths representing (e.g., data analysis-based) workflows, which, when executed, deliver the information requested by them. The hypergraph-based representation of knowledge eliminates the need for SMEs to know the details of the operations. Additionally, its flexibility allows us to represent workflows of any form, thereby eliminating the need for an additional knowledge base and a middle layer to translate user queries to workflows. We can capture and represent both the information needs and workflows together. The scope of information supported by the graph is extensible and based on the content involved, and the requirements of users. The system will have appropriate interfaces for each of the categories of users. Students learn IR by building the system, since they learn how it is used, how it works, and how it explicitly represents and supports the entire life-cycle of IR activities.

3 Course Design

3.1 Questions

The following aspects of computing education came into play when researching how to design the course. In order to aid learning, we addressed three research questions:

1. How to facilitate the learning of the knowledge and skills required to understand our modern notion of IR?

2. What type of educational approach can best introduce them to this area?

3. Given the rapid advance of related research, and the motivation resulting from connecting graduate students with research, how can research best be integrated with such education?

3.2 Considerations: Facilitating Learning

We wanted the course's primary learning objective to be a constructive understanding of the architecture and components of a modern IR system. We wanted to create opportunities for students to question, contrast, and compare their existing knowledge of such systems against this "new" system design, which we based on research in the field of digital libraries. Therefore, we wanted to encourage "deep" learning over "surface" learning as it promotes more reflection on the content and less memorization of facts [5, 7]. We naturally selected problem-based learning as the teaching strategy for the course, as it is proven to encourage deep learning [1, 2]. Furthermore, we were encouraged by positive feedback students had provided on the problem-based learning setting, in the context of similar courses [8, 9].

4 How: Process to Build the System

One can build this modern system via a semi-automatic process that is a collaborative effort between UX researchers, developers, and SMEs to break down SME goals into tasks and sub-tasks and re-use/build services to support each task. The Curators and SMEs can search and browse information goals. This section describes the 3-step methodology to extract and derive the information required to build the system framework's components. Figure 2 shows the collaborative effort.

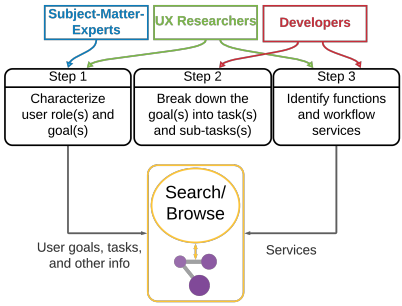


Figure 2: Collaboration between UX researchers, developers/scientists, and SMEs in building the knowledge graph from which SMEs can derive workflows (which are paths in the graph)

4.1 Step 1: Characterize user roles and goals

Our methodology begins with the process of User eXperience research. The UX process involves studying users, their habits, and their behaviors – and forming design-informing models using standard UX practices and procedures. With that information, the UX researcher constructs what is known as a "user persona." It is essential to capture a rich persona in connection with building the workflow, and to capture the persona's research goals. The UX research process should answer three questions: (1) Who is the user (SME)? (2) What are their goals? (3) How do they currently accomplish those goals?

Through this process, the UX researcher should collect enough information so, going forward, that can serve as a proxy to answer any questions the developers might have during the design and development phase of the methodology.

4.2 Step 2: Break down goals

In this step, the goal is broken down into units known as tasks and then into sub-tasks. Here task and sub-task refer to things that the user does to achieve their goal [6]. The developers and engineers can produce this breakdown through their expertise in addressing such goals and tasks. When a developer identifies that the goal requires multiple steps to solve, they break down the goal into a set of tasks. Doing a sub-task is a part of doing a task. The primary purpose of identifying tasks is to guide the overall design such that the final workflow design should support the tasks and sub-tasks identified. A hierarchical relationship is formed between goals, tasks, and sub-tasks.

4.3 Step 3: Identify functions and workflow services

Through the developers' expertise, we can process the goals and produce a set of "solve-able" tasks. We now have to find a way to solve these user tasks. In this step, the developers decide what set of functions to employ to complete each (sub-)task. This step requires active collaboration between the UX researchers and the developers. A developer will choose to design a solution for a task, possibly from multiple options, suitable functions, or algorithms. The UX researcher communicates both the SME expectations on output and the data available as input. During this period, a developer also identifies other implementation-specific parameters, if these are not provided as inputs, possibly with defaults specified by the UX researcher or developer. Finally, a developer designs a set of services that helps complete the specific task.

In summary, by following the 3-step process, the students will understand the workings of a modern IR system, as they build one.

5 IR Course

5.1 Overview

Information Retrieval (IR) is a graduate course offered at Virginia Tech.

Students are of graduate standing. Participation in executing the methodology and success in building solutions does not require prior knowledge of AI or UX. Students collectively build a state-of-the-art information storage, retrieval, and analysis system that supports searching, browsing, and analysis of large collections of documents. In the most recent offering, three types were considered: (1) tweet (TWT) collections, (2) web page (WP) corpora, and (3) the local collection of electronic theses and dissertations (ETDs). In Fall 2022, the focus will be on a national corpus of ETDs. The information needs of the subject-matter-experts (SMEs) would determine the range of search queries supported. SMEs were researchers either looking to analyze an archive of webpages, tweets from an event, or theses and dissertations.

5.2 Setup

We had five different team efforts that fit together to build the components of the system. These include three types of "content" teams, one front-end interface team, and one integration team. It was the responsibility of each of the "content" teams to communicate with the SME who works with their type of content. The members of each content team executed the steps of the methodology that we introduced earlier. Through that process, as specified by SMEs, they identified information goals, expectations regarding output formats, and the formats of available inputs. They forwarded this information to the Front-End and Integration teams. They translated this information into design and development requirements. We shared this figure with the class during the first week of the course.

The course was set up like a flipped classroom with each student reading the textbook and taking quizzes on its chapters. The problem structure and the solution approach encouraged intra- and inter-team collaboration and learning. Class sessions had students working in teams, teams collaborating, and the instructor providing guidance as requested. In addition to building components of the IR system, the teams shared their insights and progress via three interim reports and presentations. They also prepared a final report and presentation. Teams had the opportunity to provide qualitative feedback for each deliverable. This was done to ensure timely intervention and shared learning.

5.3 Execution of Methodology

Each of the three teams working on their respective content played the role of UX researchers. They then interviewed their respective SMEs: one SME for each team. During the interviews, they identified SME goals and preferences. The content teams shared this information with the Front-end team. The Front-end team was responsible for building what would support the user experience, i.e., SME preferences and expectations on output. Afterward, the content teams played the role of developers. They broke down the user goals into tasks and sub-tasks. Then they collaborated with the Integration team, which was responsible for building the infrastructure for all of the development efforts to be deployed as services. For the final part of the methodology, the content teams collaborated with the Integration team to build and integrate services. The Integration team was responsible for building the knowledge graph-based engine to support SME exploration. So, they also communicated with the Front-end team to provide a response to SME queries.

5.4 Outcome

The three teams working on their respective content identified 17 information goals: five from ETD SME, four from WP SME, and eight from TWT SME. They developed a total of 22 services, which they containerized and deployed in collaboration with the Integration team. The Integration team registered the services and connected them based on the relationship between information goals. The Front-end team built an interface for users for two types of functions: (1) browse and search through the collections, and (2) request an information goal. Requests were routed and served according to the knowledge graph that determined the sequence of services to execute to satisfy the user requirements. The knowledge graph and the reasoner that interprets the graph were both constructed by the Integration team.

6 Impact: Evaluation & Discussion

We constructed our evaluation to assess three aspects of the methodology: (1) easy to use, (2) easy to learn, and (3) producing a good solution.

To learn (1) and (2), we employed an adapted version of the System Usability Scale (SUS) that was created to learn about user experiences when using a system [3]. For (3), we asked the students to rate the overall solution's quality, and through qualitative, open-ended questions, assess the degree to which the methodology contributed to their design decisions.

Figure 3 provides a breakdown of the responses to survey questions, showcasing four sub-figures. From sub-figure A, we see that most students in the

FIGURE A	Q4: This methodology is helpful for doing the project.				
Q5: I understand this methodology.	Strongly agree	Agree	Neither agree nor disagree	Somewhat disagree	Grand Total
Strongly agree	1	1			2
Agree	2	5		1	8
Somewhat agree		2	1		3
Strongly disagree	1				1
Grand Total		8	1	1	14

FIGURE B	Q6: If it turns out that I often am building systems or doing projects like for this course, then I think that I would like to use this methodology frequently.						
Q4: This methodology is helpful for doing the project.	Strongly agree	Agree	Somewhat agree	Neither agree nor disagree	Somewhat disagree	Strongly disagree	Grand Total
Strongly agree	1	3					4
Agree		5	1	2			8
Neither agree nor disagree						1	1
Somewhat disagree					1		1
Grand Total	1	8	1	2	1	1	14

FIGURE C	Q11: I would imagine that most people would learn to use this methodology very quickly.					
Q8: I thought the methodology was easy to use.	Strongly agree	Agree	Somewhat agree	Neither agree nor disagree	Disagree	Grand Total
Strongly agree	1					1
Agree	1	4	4			9
Somewhat agree		1				1
Somewhat disagree				1		1
Disagree				1	1	2
Grand Total	2	5	4	2	1	14

FIGURE D	Q22: How would you rate the quality of the overall system solution?			
Q23: How important was the methodology in influencing your project solution?	Excellent	Good	Average	Grand Total
Extremely important	1	2		3
Moderately important		9	1	10
Neither important nor uninportant			1	1
Grand Total	1	11	2	14

Figure 3: A short summary of the student/participant feedback. A total of 16 students completed the methodology evaluation survey. Since each team consisted of 5-7 students, some students actively participated in instrumenting the methodology and a few focused solely on implementation. 14 students reported that they either played an "Active" role or "Asked questions in interview and/or created artifacts document" or a "Passive" role or "Listened to questions during the interview and/or my work was dependent on the artifact document." The statistics in this figure pertain to responses from these 14 students.

class understood the methodology and found it helpful for building the IR system. There is one instance of someone who reported that they didn't understand the methodology and yet found it to be helpful. This positive feedback on the helpfulness of the methodology is more emphasized in sub-figure B. In sub-figure B, students who reported positively on the methodology, also reported that they would use the methodology again for a project in a similar problem space. The feedback is consistent on the other end as well. In the two instances that students did not find the methodology helpful, they would not employ it to build a system in a similar situation. From sub-figure C, we can learn about the "usability" of the methodology. All but three students found the methodology easy to use. Furthermore, these students felt that anybody would be able to learn and instrument the methodology. In sub-figure D, we see that those who self reported their system solution to be "Excellent" or "Good" also reported that the methodology was important in influencing their solution. The sub-figures only showcase a subset of the students' report. Looking at the raw statistics, cross statistics, and qualitative responses, we observe a favourable response from the students about the methodology that we employed as an engine for them to learn about building modern information retrieval systems. Their written feedback is consistent with the scores they provided. The textual responses to the question "How did you use the methodology in making decisions about your team's approach? How did it influence your solution?" convey that the methodology significantly contributed to each team's design and implementation. Given that, and looking at the student's self reporting of the quality of the solution, we can conclude that the students believed that through the methodology, they were able to build a good quality solution. More specifically, it conveys that the methodology significantly impacted their decision making process. Their feedback on how they would "improve the methodology" is something we are taking into account as we upgrade the IR system and the methodology for the next iteration of the IR class that is to be taught in Fall 2022.

7 Conclusion

We researched how to improve the teaching of IR. In the latest iteration of the course on IR, we introduced and integrated concepts of UX research into workflow design. The scaffolding guided a service-oriented architecture of the system solution that was to be built by the students. The motivation for this approach is that the final system would provide a user experience that better meets SME needs. Furthermore, current and future developers would be able to continue to contribute by building new and/or improved services, providing expanded functionality, in the context of an extensible system. Over

the semester, they followed steps of the methodology and sought advice and/or clarifications from the TAs and the teacher of the class. They evaluated the methodology, indicating it is easy to use, easy to learn, and helpful. As a result, we believe that the methodology and processes can be used to teach students to build modern IR systems or similar information systems. Given the positive feedback we've received, we also asked teams of undergraduate students for a problem-based learning course to employ the methodology for their system building activities.

References

[1] Howard S Barrows and Robyn M Tamblyn. *Problem-based learning: An approach to medical education*. Springer Publishing Company, 1980.

[2] John Biggs. What the student does: Teaching for enhanced learning. *Higher Education Research & Development*, 18(1):57–75, 1999.

[3] John Brooke. SUS: a "quick and dirty usability scale, isbn 9780748404605. *Usability evaluation in industry*, page 189, 1996.

[4] Edward A. Fox, Marcos André Gonçalves, and Rao Shen. *Theoretical Foundations for Digital Libraries: The 5S (Societies, Scenarios, Spaces, Structures, Streams) Approach*. Morgan & Claypool Publishers, 2012.

[5] Heather Fry, Steve Ketteridge, and Stephanie Marshall. Understanding student learning. In *A handbook for teaching and learning in higher education*, pages 26–44. Routledge, 2008.

[6] Rex Hartson and Pardha Pyla. *The UX Book: Process and Guidelines for Ensuring a Quality User Experience*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 2012.

[7] Gareth JF Jones. An inquiry-based learning approach to teaching information retrieval. *Information Retrieval*, 12(2):148–161, 2009.

[8] Tarek Kanan, Xuan Zhang, Mohamed Magdy, and Edward Fox. Big data text summarization for events: A problem based learning course. In *Proceedings of the 15th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '15, page 87–90, New York, NY, USA, 2015. Association for Computing Machinery.

[9] Liuqing Li, Jack Geissinger, William A. Ingram, and Edward A. Fox. Teaching natural language processing through big data text summarization with problem-based learning. *Data and Information Management*, 4(1):18 – 43, 01 Mar. 2020.

Teaching Cross-Platform Technology and Democracy*

Michael P. Rogers
Department of Computer Science
University of Wisconsin Oshkosh
Oshkosh, WI 54901
mprogers@mac.com

Bill Siever
McKelvey School of Engineering
Washington University in St. Louis
St. Louis, MO 63130
bsiever@wustl.edu

Abstract

Traditionally, mobile apps have been built using native technologies and imperative frameworks (NT/IF), and academic courses in mobile development have reflected this. While this approach accounts for the vast majority of existing apps, the last 10 years has seen the rise and steady improvement of cross platform technologies that utilize declarative frameworks (CP/DF). This approach has some advantages, and graduates are likely to encounter CP/DF in the workforce, so it behooves us to introduce them in our classes, and provide students with the skills needed to choose among myriad possibilities. This experience report describes the design and delivery of a novel one-semester mobile computing course that focused on enhancing students' technical and decision-making skills, while adding an element of democracy to the curricular design process. Students already familiar with NT/IF (having studied iOS, Android, or in some cases both) systematically, studied

and compared 4 CP/DF products. Three were selected in advance by the instructor. Students developed rubrics to rate each, and guided by those rubrics, chose a fourth technology to pursue. Major findings include: insight into the process of transitioning students from NT/IF to CP/DF, observations about instructional strengths and weaknesses of each approach, and student-developed assessments that reflect student priorities for UI frameworks.

1 Introduction

Historically, courses in mobile app development have used native technologies and imperative frameworks (NT/IF). Native technologies are tools and APIs to generate apps that run exclusively on a particular platform (iOS or Android). Imperative frameworks are those in which changes to the GUI occur as a result of direct manipulation of GUI components in code.

NT/IF is appealing. The tools (typically Xcode and Android Studio) are highly polished. The APIs are comprehensive, well documented, and, being based on an imperative approach, easy to grasp. Consequently, students find creating good quality native apps to be challenging but ultimately achievable.

However, guidance from curricular advisory boards and surveys of developer trends [12, 14] underlines the need for students to be familiar with cross-platform technologies using declarative frameworks (CP/DF). Cross-platform frameworks allow a single codebase to run on multiple platforms (i.e., iOS and Android). Declarative frameworks are those in which the programmer does not explicitly change the appearance of the UI at runtime, but rather declares how the UI should appear based on an underlying state model. The system is responsible for updating the UI.

CP/DF benefits to students include: 1) They gain a better abstract understanding of GUI concepts when focusing on features common across platforms; 2) They are able to increase productivity by creating an app for multiple platforms in approximately the same time as a native app; 3) Their apps tend to be more robust, as state (mis)management in is a common source of bugs in NT/IF; 4) They are able market themselves to companies seeking cross-platform developers. Finally, both students and instructors can benefit from the independence from a specific platform or operating system.

A challenge of CP/DF-based development may be known in industry, but not to students: identifying the best choice for a specific project's needs and schedule. In most courses student's influence on the curriculum is only coarsely expressed, based on enrollment. since selecting a technology is such a vital career skill, our goal, and one innovation, was to make this a major component of the course.

2 Prior Work

Courses focused on native application development for mobile devices have proliferated following the release of the first iPhone SDK in 2008 and Android SDK in 2009. Both platforms utilized imperative frameworks (IF) and, since there wasn't a large market of mobile devices that had sufficient feature parity for meaningful cross platform support prior to this time, there weren't many cross-platform (CP) tools.

Most early pedagogical work focused on either: a) resources for a particular platform [13]; b) meta concepts that apply to all platforms [19, 6]; c) the relative advantages and disadvantages of one platform over another [8]; and/or: d) the advantages of exposure to multiple platforms [16].

Burd et al. surveyed 200 courses that covered mobile app development [4]. They noted likely advantages of covering multiple platforms in a single course, including better ability to identify features common to mobile frameworks and a better ability to critique frameworks, and disadvantages, especially an overall lack of depth.

One of the advantages of cross-platform frameworks suggested in [17] is the ability to leverage skills that may be introduced in other courses, such as HTML and JavaScript, which could reduce burden on the overall curriculum. Currently the"preferred programming languages" for iOS and Android are Swift and Kotlin respectively, neither of which are widely covered outside of app development.

Declarative programming has been a common theme in computing for more than 40 years [1], but Declarative UI is not well addressed in pedagogical literature.

3 An Overview of CP/DF

3.1 CP/DF

All UIs were originally based on imperative frameworks, where code explicitly changes the UI in response to program needs.

Consider an app that displays the number of times a button has been tapped. Programmers will provide code that runs in response to the button being tapped and explicitly changes the value show on the screen.

With declarative frameworks [18], programmers indicate or declare, in advance, how the user interface appears based on state The programmer focuses on declaring how state is bound to UI and the merely focuses on changing state. The framework manages updates to UI based on state rather than explicit management, freeing the developer from the meticulous task of managing state [3].

Declarative frameworks are designed to efficiently identify state changed and update UI as needed.

Complex apps often display state in a variety of different places, which can be difficult to manage. Consequently, removing the need for explicit UI management removes a major source of bugs[5].

In order for state changes to trigger UI updates, the frameworks need to be aware of state changes. In Flutter and React Native, two popular declarative platforms, state variables must be changed by explicitly invoking a method to update state, `setState()`. NativeScript, another popular platform, requires explicit binding to a state variable. The UI is updated whenever a bound variable is modified. Instructors teaching DF will need to consider how they will present the paradigm being. [9, 7, 15].

Declarative frameworks are in the ascendency and many cross-platform technologies already use them. Both Apple and Google added declarative frameworks to their native technology offerings in 2019, SwiftUI [2] and Jetpack Compose [10], respectively. In doing so, they have defined a new category: NT/DF.

3.2 Laying out the UI with CP/DF

NT/IF benefits from the visual layout tools which allow rapid prototyping a UI by dragging labels, buttons, text fields, tables, etc., into place. These tools are adequate for students working in small, agile teams, and consequently in our NT/IF courses we only spend 2 days discussing code-based UI creation. Although the course can not devote more time to code-based UI creation, it may be favored in industry [20] because it can be easier to manage (e.g., easier to merge).

This leads to another argument for teaching declarative frameworks. Justifying code-based UI creation is not just easy, but essential: there are no significant visual layout tools. The documentation, examples and tools all revolve around code.

4 Course Description

4.1 Course Overview

In the Spring of 2020, a class in mobile computing was taken by 13 students, juniors and seniors who had already taken a course in iOS or Android development (after a traditional CS1 and CS2 sequence). The course was divided into 4 modules, each dedicated to one cross platform product — Flutter, Xamarin Forms, React Native, and NativeScript. Each module lasted approximately 4

weeks. The first week of each was spent familiarizing students with the relevant programming language without the specific details of a framework. The alternative, throwing APIs, tools and languages at the students all at once, would have been overwhelming.

The scope of the course was ambitious, but necessary to meet our objective, giving students extensive practice in comparing alternative products. This was intended to be a survey of the cross-platform landscape, not an in-depth dive into one area.

Course delivery for the first two modules was face-to-face. Due to the COVID-19 pandemic: the last two were taught online and included almost 10 hours of instructor-generated videos.

We naturally spent the most time on the most difficult subjects. Students had prior experience with buttons, text fields, labels, list views, and navigation, either tabbed or hierarchical, so these topics were covered quite quickly. The focus on state was a paradigm shift for all of our students, requiring several class periods to explain the concept, and then implementation on each platform. It helped that our first platform was Google's Flutter. The documentation is not only exceptionally well organized, but their explanation of state is easily the best we have encountered. App-specific details, such as working with the camera, map services, etc., were left to the students to explore.

During the course, students learned 3 different languages, Dart (Flutter), C# (Xamarin Forms) and JavaScript (React Native, NativeScript). Dart was new to all the students, a handful had used C# before, and even those who had seen JavaScript benefited from a deep dive into prototypical hierarchies.

4.2 Assignments

As this was an applied course in cross-platform app development, we naturally encouraged students to develop for multiple platforms. We could not insist upon it, however, since Windows users could not legally run an iOS simulator, and not all of them had access to hardware. During class, however, instructors, using macOS, took pains to "be fair", and alternate between running examples on iPhones and Android devices. Screenshots used in assignments showed both platforms.

Each module began with an essential and (usually) easy installation assignment, where the students had to demonstrate that they had the tools properly installed, with a "hello world" app up-and-running in a simulator. A small confidence building, console-based program, focused strictly on language concepts, was also given during the first week, so that by the time the students started learning the APIs they could read the examples and documentation.

The next assignment for all platforms was a typical mobile app, involving text fields, labels and buttons: when a button was tapped, data from text

fields would be retrieved, manipulated, and results displayed in labels. Details depended on the framework, but the pedagogic point was that the students declared the content of the label to display a state variable, but never explicitly changed it in the button handler code.

An identical layout assignment, focused solely on getting the layout of a UI right, was also given for all frameworks (Figure 1, left). This app did not include any functionality, and therefore sounds trivial, but with some frameworks even getting the UI to look right was challenging.

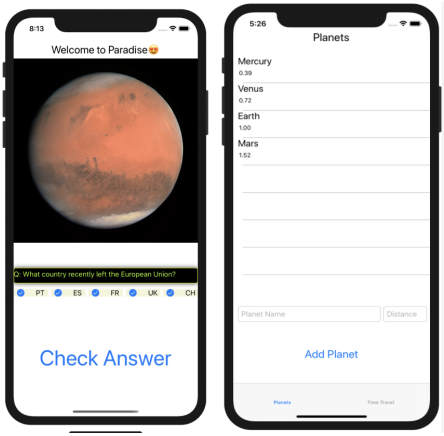


Figure 1: Generic Layout & Planetary Conquest Assignments

A second identical app, Planetary Conquest (figure 1, right) was also administered across all 4 modules. We had contemplated making all assignments cut across all 4 modules, but because of time constraints, and lack of familiarity with the frameworks, this was not possible.

The penultimate assignment involved multiple screens and list views. The exact nature of the app depended on the framework, but involved either a tabbed or hierarchical structure.

The final assignment in each module, due just after the exam, required students to rate the product using criteria that they developed (see section 5).

4.3 Exams

There was an exam after each module was completed. Exams were all take-home, open book/internet, and students had a week in which to complete them. Students were strongly encouraged to seek out the instructors for help (but forbidden from discussing problems with anyone else). Each exam had three

sections: a programming problem that just focused on the language; another layout problem where the students simply had to reproduce the look of a UI, but it was non-functional; and finally a tabbed app, the aforementioned Planetary Conquest, where students had to display an augmentable list of planets in one tab, and calculate the time required for light to travel between planets in another: the app involved no actual conquest, except of some challenging Flutter concepts.

The final exam consisted of the exam for the last framework studied, as well as a reflection piece, looking back over all four frameworks.

4.4 Cross Platform Framework Choices

In section 5 we describe a formal process for rating frameworks developed by students during the course. When designing the course, we took a more unstructured approach. We reviewed articles comparing the top-rated frameworks. From the most promising we then constructed small apps, so that we could assess ease-of-installation, learning curve, ease-of-use, and documentation. Originally, we intended for all frameworks to be declarative, but then decided to include one imperative framework. This allowed us to compare two paradigms and to provide a common reference point for students who had studied either exclusively iOS or Android. In the end we settled on 3 declarative frameworks – Flutter (Google), React Native (Facebook), PhoneGap (Adobe) – and 1 imperative, Xamarin Forms (Microsoft).

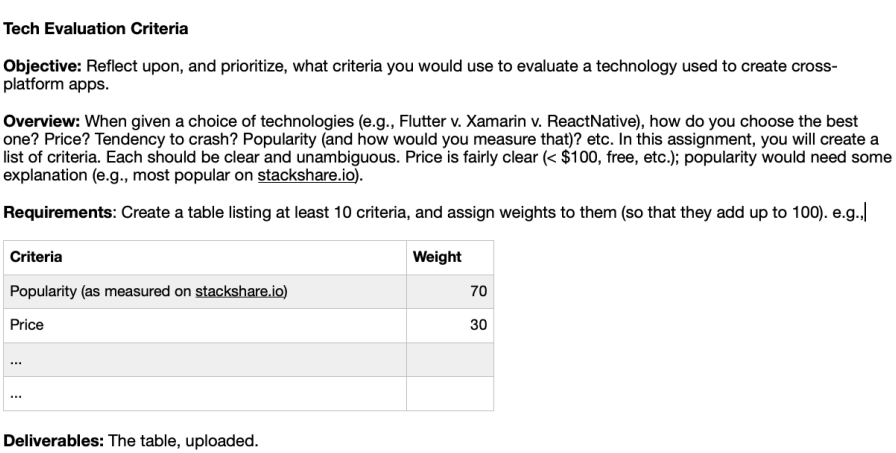
5 The Democratization of the Curricular Design Process

The selection process we undertook to find the most promising frameworks was highly educational, and we rued the fact that students were deprived of that experience. Then we discovered that one of our choices, PhoneGap, required an app that was recently banned from Apple's App Store [11], and thus by happenstance we were provided the opportunity to let students go through the process, and give them the power to shape the curriculum (within limits, see below).

5.1 Rubric Creation

The ability to identify promising, viable products among the torrent of new frameworks, tools and languages that appear seemingly daily is vital. Faculty do it routinely, frequently revising content to incorporate new developments: indeed, this entire course is a testament to that. But ironically, in the curricula, the assessment process itself gets short shrift. In this course, we emphasized

it. Students were first given an assignment, "Tech Evaluation Criteria" (figure 2), due after they had been exposed to the first framework (Flutter).



Tech Evaluation Criteria

Objective: Reflect upon, and prioritize, what criteria you would use to evaluate a technology used to create cross-platform apps.

Overview: When given a choice of technologies (e.g., Flutter v. Xamarin v. ReactNative), how do you choose the best one? Price? Tendency to crash? Popularity (and how would you measure that)? etc. In this assignment, you will create a list of criteria. Each should be clear and unambiguous. Price is fairly clear (< $100, free, etc.); popularity would need some explanation (e.g., most popular on stackshare.io).

Requirements: Create a table listing at least 10 criteria, and assign weights to them (so that they add up to 100). e.g.,

Criteria	Weight
Popularity (as measured on stackshare.io)	70
Price	30
...	
...	

Deliverables: The table, uploaded.

Figure 2: The "Tech Evaluation Criteria" assignment

Students put a lot of care into this assignment, answers were thoughtful, thorough, and occasionally humorous. The results from this assignment were coalesced into a single rubric (weighting all criteria equally, for simplicity's sake). Students filled the rubrics out immediately after they had finished studying each framework, while it was still fresh in their minds.

5.2 Technology Selection

As previously mentioned, the students were responsible for selecting the fourth and final product that would be studied in the class. The rubric that they devised in the Tech Evaluation Criteria assignment helped them focus on key factors to guide their choice. However, we did not ask them to actually fill out that rubric during the decision process, as it would be impossible to meaningfully do so before working extensively with the technology.

Students were divided into teams of 2 or 3. They were instructed to: ... identify a cross-platform technology (other than Flutter, Xamarin Forms, and ReactNative) that you would like to study, and justify its use. Basically, this is a sales pitch to decide what the last 4 weeks of the course will be spent on. Use the criteria you had established for evaluating tech — is it popular? well documented? genuinely cross-platform? free? etc., etc. The presentation can include a PowerPoint or any other compelling evidence that you would like to present. If you have a .ppt, upload it here. There are 7 teams presenting,

so each team should speak for approximately 5 minutes. At the end of the presentations, there will be a vote.

Because we were interested in getting students' opinions, not just echoes of our own thoughts, throughout the course the instructors attempted to appear neutral as we studied each product. To avoid even the temptation to weigh in on the decision-making process, we took the step of vacating the classroom for the presentations, which the teaching assistant videoed so that we could grade later.

While 6 teams presented, they offered only two choices: 3 lobbied for Ionic and 3 for NativeScript. They were good choices, and the lack of diversity came about not because of collaboration, but because Ionic and NativeScript appeared in most compilations of the top 5 or 10 cross platform products. In a subsequent iteration of this course we will work to ensure that a wider range of choices are presented.

When the poll was completed 23% chose Ionic and 73% chose NativeScript, a testimonial to NativeScript's appeal.

6 Student Feedback

With respect to the course itself, students were generally complimentary, appreciative of the effort that we put into the course, and grateful for the opportunity to have taken it. Although we stressed in class that this was designed to be a rapid, breadth first survey of multiple platforms, there was still some regret that we didn't stop with Flutter, or ReactNative, as they enjoyed developing in them so much: and due to the disruption of the COVID-19 pandemic, we did not have enough time to explore the last platform, NativeScript, thoroughly.

With respect to the frameworks, students rated each (see table 1) after studying it, on a 1-5 scale, with 1=least desirable, 5=most. NativeScript scored highest in many areas, partially due to its place in the schedule (students had already seen 2 declarative frameworks, so the concepts were familiar).

7 Lessons Learned

When first introducing the declarative frameworks, using the tap counting app described earlier, some students reacted as if they had just seen a magic trick ("if you didn't change the label, how did it get changed?"), underlining the vast chasm between the declarative and imperative paradigms. But seeing the same concepts appear in different guises in different frameworks made it possible for students to grasp and abstract the big idea: rather than change the model then change the UI, change the model and observe the UI being regenerated automagically. Rapidly exploring a succession of technologies quickly, akin

Table 1: Framework Ratings

Framework	Flutter	Xamarin	React Native	Native Script
Learning Curve	3.5	3.4	3.4	4.3
Usability	3.7	3.3	3.3	4.3
Docs	4.2	2.8	3.4	3.8
Support	3.5	3.3	3.6	3.6
Longevity	3.6	3.6	3.9	3.8
Popularity	3.2	3.5	3.9	4.3
Cost	4.0	4.0	4.7	4.6
Speed	3.2	4.1	3.1	3.7

to a spike in agile development; developing rubrics to choose from multiple technologies and making presentations to argue for each, these are skills that will serve them in good stead throughout their careers, and this course made them better prepared for it.

8 Future Work

We see several directions for future work. The first is to tweak the structure of the next iteration of this course, having the students pick two of the frameworks rather than one, and possibly reducing the number of frameworks by one. The second would require interested students to sign up for an extra 1 hour of credit, and in that time develop a full-blown, team-based project using the first framework studied. A third question arose as we became more familiar and confident with the CP/DFs that we were teaching: could one of these serve as the basis of an introductory mobile computing class? We intend to find out next semester. Lastly, we are interested in investigating whether this democratization of the curricular design process can be applied to other courses and hope to share some results along these lines in the future.

References

[1] J Mack Adams. Teaching declarative programming. In *Proceedings of the fifth SIGCSE technical symposium on Computer science education*, pages 83–85, 1975.

[2] Apple. Swiftui, Aug 2020. `https://developer.apple.com/documentation/swiftui`.

[3] Engineer Bainomugisha, Andoni Lombide Carreton, Tom van Cutsem, Stijn Mostinckx, and Wolfgang de Meuter. A survey on reactive programming. *ACM Computing Surveys (CSUR)*, 45(4):1–34, 2013.

[4] Barry Burd, João Paulo Barros, Chris Johnson, Stan Kurkovsky, Arnold Rosenbloom, and Nikolai Tillman. Educating for mobile computing: addressing the new challenges. In *Proceedings of the final reports on Innovation and technology in computer science education 2012 working groups*, pages 51–63. 2012.

[5] Gregory H Cooper and Shriram Krishnamurthi. Embedding dynamic dataflow in a call-by-value language. In *European symposium on programming*, pages 294–308. Springer, 2006.

[6] Andrey Esakia and D Scott McCrickard. An adaptable model for teaching mobile app development. In *2016 IEEE Frontiers in Education Conference (FIE)*, pages 1–9. IEEE, 2016.

[7] Facebook. State, Aug 2020. `https://reactnative.dev/docs/state.html`.

[8] Mark H Goadrich and Michael P Rogers. Smart smartphone development: ios versus android. In *Proceedings of the 42nd ACM technical symposium on Computer science education*, pages 607–612, 2011.

[9] Google. Differentiate between ephemeral state and app state, Aug 2020. `https://docs.flutter.dev/development/data-and-backend/state-mgmt/ephemeral-vs-app`.

[10] Google. Jetpack, Aug 2020. `https://developer.android.com/jetpack/compose`.

[11] Jesse. Update on the phonegap developer app (ios), Aug 2020. `https://web.archive.org/web/20180625195053/https://blog.phonegap.com/update-on-the-phonegap-developer-app-ios-99e07e3309dd`.

[12] JetBrains. The state of developer ecosystem 2019, Aug 2020. https://www.jetbrains.com/lp/devecosystem-2019/.

[13] Victor Matos and Rebecca Grasser. Building applications for the android os mobile platform: a primer and course materials. *JCSC (Journal of Circuits, Systems, and Computers)*, 26(1):23, 2010.

[14] Mobindustry. Mobile development forecast for 2022: Trends, programming languages, tools, and domains, Aug 2020. https://www.mobindustry.net/blog/mobile-development-forecast-trends-programming-languages-tools-and-domains/V.

[15] NativeScript. Data binding, Aug 2020. https://v6.docs.nativescript.org/angular/core-concepts/angular-data-binding.

[16] Bryson R Payne. Teaching android and ios native mobile app development in a single semester course. *Journal of Computing Sciences in Colleges*, 30(2):176–183, 2014.

[17] CP Rahul Raj and Seshu Babu Tolety. A study on approaches to build cross-platform mobile applications and criteria to select appropriate approach. In *2012 Annual IEEE India Conference (INDICON)*, pages 625–629. IEEE, 2012.

[18] Guido Salvaneschi, Alessandro Margara, and Giordano Tamburrelli. Reactive programming: A walkthrough. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, volume 2, pages 953–954. IEEE, 2015.

[19] Jonathan Sprinkle. Teaching students to learn to learn mobile phone programming. In *Proceedings of the compilation of the co-located workshops on DSM'11, TMC'11, AGERE! 2011, AOOPES'11, NEAT'11, & VMIL'11*, pages 261–266, 2011.

[20] SwiftDev365. Storyboards vs programmatic ios development?, Aug 2020. https://medium.com/@matlyles/storyboards-vs-programmatic-ios-development-40c5b5907f85.

Predicting Water Quality Estimates using Satellite Images in Coastal and Estuarine Environments*

Ryan Hogan, Meghan Ford, Mandy Battaglia
Jonathan Sturdivant, Gulustan Dogan, Philip Bresnahan
University of North Carolina Wilmington
Wilmington, NC 28403
{rh7909,mrf5057,mb9455,jds6589,dogang,bresnahanp}@uncw.edu

Abstract

Water is integral to all environments. As such, its quality is continuously monitored by sensors and satellites. Satellite imagery is very useful in monitoring water quality in deep waters, further offshore. However, shallow bodies of water, such as coastal areas and estuaries prove more difficult to measure due to the variability in the environment and their considerably lesser depth. To mitigate this shortfall, sensors are used to monitor water quality in coastal areas and estuaries; though, this application is limited due to its high cost. Nevertheless, we aimed to use satellite images alone in order to predict water quality in coastal and estuarine ecosystems, specifically, Zeke's Basin in Fort Fisher, North Carolina (-77.93500, 33.95470). We built multiple machine learning algorithms with this aim in mind. Our dataset was a combination of Landsat 8 image data and NOAA's National Estuaries Research Reserve System (NERRS). We used principal component analysis to find correlations within our dataset. Then we trained 4 models, a Decision Tree Regressor, AdaBoost Regressor, Bagging Regressor, and a Random Forest Regressor. The model that created the most accurate turbidity estimates was the AdaBoost Regressor with a Decision Tree Regressor estimator. Using the Ababoost Regressor, we achieved an r-squared of 0.9751.

1 Introduction

Water quality is integral to all ecosystems. Poor water quality affects access to clean drinking water, plant health, wildlife vitality, and ecosystem functionality. Furthermore, poor water quality can facilitate dangerous algae blooms, which can cause widespread harm in the water column and surrounding areas. In order to determine water quality certain measurements need to be recorded; such as the concentration of dissolved oxygen, the colored dissolved organic matter, pH levels, the levels of bacteria, the salinity, the turbidity, the amount of algae, pesticides, herbicides and heavy metals [11]. Satellite images can be used to determine certain measurements of water quality in deeper bodies of water. However, determining water quality for shallow water close to coastlines and within estuaries is challenging when using only satellite images. The implementation of sensors in these areas can help better estimate water quality. Though, this is not widely applicable due to high costs. Despite these set backs, we believe that a machine learning algorithm can be built to predict levels of turbidity in coastal areas using satellite images. We focused on building a machine learning algorithm that can predict the levels of turbidity near Zeke's Basin, a site sensor location in the Carolina Beach River. We chose this location to focus on as both satellite and sensor data are available. As such, we can train the model on satellite data and then validate it with in-situ water quality readings.

This paper is organized as follows: in section 2 background information on water quality, our location of interest, and the data bases used are discussed. Section 3 reviews related works on water quality estimations using satellite images. Section 4 details the methodology and experimental design, as well as how we created the dataset. In section 5, results of the various models we built trained and tested are reported. Finally, in section 6 a summation of works is presented and future works are discussed.

2 Background

Water is interconnected to everything within an ecosystem. It constantly cycles through the atmosphere and is intimately tied to all life; as such, its quality is a universally important variable. Parameters used to quantify water quality include: nitrate levels, pH, color, oxygen content, and turbidity [9]. In this paper we focus on turbidity as a measure of water quality. Turbidity is the measure of suspended sediments in the water column [4]. This is an important measure to consider when investigating water quality as it has wide ranging effects. Increased turbidity levels could irritate the gills of fish, decrease primary productivity through light attenuation, and possibly transport contaminants [4].

As such it would be incredibly beneficial to increase the accuracy of turbidity estimates.

3 Related Work

Tracking water quality with satellite data has been commonplace for some time now, but coastal and estuarine ecosystems remain a challenge due to their shallowness and variability. Previous works including satellite imagery and water quality estimations are reviewed here.

Multiple linear regression (MEL) and gene expression programming (GEP) have been used to measure the level of turbidity in the Tseng-Wen and Nan-Hwa reservoir in Taiwan. The dataset was composed of Landsat 8 spectrum imagery, and in-situ turbidity readings. The GEP outperformed the MEL in accuracy, and had a lower root mean squared error(RMSE). Though the performance of both were considered acceptable [8].

Supervised ML algorithms have been utilized to create and train a model that predicted Chlorophyll-a and suspended solid levels. Additional visual data beyond satellite imagery pulled from Sentinel-2 was used, and their model is specific to only two bodies of water. An artificial neural network, random forest, and K nearest neighbor produced the lowest mean squared error when predicting Chlorophyll-a and total suspended solids. The random forest model produced the best results overall [12].

A support vector machine algorithm (SVM) achieved a 97.01% accuracy when predicting water quality index. Non-linear autoregressor neural networks and long short term memory were also able to produce satisfactory accuracy, though not as high as the SVM. These models were trained on water quality readings at certain locations in India from 2005 to 2014 [2].

Through the use of satellite remote sensing reflectance paired with in-situ Chlorophyll a readings concentrations of chlorophyll in St. Lawrence's Estuary and Gulf waters were estimated. However they only achieved moderate accuracy. Their methodology was based on empirical orthogonal functions and did make improvements to existing accuracies [6].

Landsat 8 data has been used to predict turbidity in Ramganga River in the Ganges Basin, India. The bands focused on were 1-7. Water quality measurements were also taken to validate results. The band pixels were scale and a regression model was used and produced satisfactory results [3].

Ground truth measurements paired with VNREDSat-1A multi-spectral data from remote sensing imagery has been used to train a regression model and estimate suspended sediment concentrations in the Red River, Hanoi, Vietnam with little difference between predicted and actual levels [13].

THEOS satellite data has also been used to predict total suspended solids

within a coastal environment. The algorithm used two bands, red and blue, and produced quality estimates [7].

Finally, turbidity levels have been estimated using the correlation of in-situ measurements and the red band of Landsat 8 OLI. Patterns of turbidity fluctuation were successfully identified and turbidity levels within the patterns were accurately estimated [10].

3.1 Zeke's Basin

Zeke's Basin is a wetland ecosystem equipped with a water quality sensor and is located in Fort Fisher, North Carolina. The exact location of Zeke's Basin is (-77.93500, 33.95470). The sensor at Zeke's Basin is one of three original components of NOAA's reserve system and it covers around 1,165 acres [5]. Zeke's Basin is open to the public and accessible though the Fort Fisher State Recreation Area. Its sensor is funded by the NOAA's National Estuarine Research Reserve or NERRS. This organization will be discussed later in the paper. The sensor at this location collects water quality measurements every 15 minutes.

3.2 NOAA's National Estuaries Research Reserve System

The National Estuarine Research Reserve System was created as a result of the Coastal Zone Management Act. The program receives it funds and guidance from The National Oceanic and Atmospheric Administration [1]. NOAA works with states to ensure sensors are placed correctly and maintained. Through the collaboration of NOAA and individual states, the National Estuarine Research Reserve System established 29 coastal protected sites that facilitate research. In total the protected and monitored land equals about 1.3 million acres of coastal and estuarine environment [1]. The NERRs sensors that are placed around the coast collect data every 15 minutes. Variables monitored include levels of turbidity, water temperature, and the levels of chlorophyll [1]. The sensor data collected at Zeke's Basin provided training and validation sets for the models.

3.3 Landsat 8 Satellite Image Data

Landsat 8 data was used to create our dataset; the specific subsection of data we used is USGS Landsat 8 Surface Reflectance Tier 1. We first pulled data from atmosphere reflectance, but found that surface reflectance was better suited for our goals. This switch allowed us to better compare the images of Zeke's Basin and discern water quality with less weight from atmosphere conditions.

Landsat 8 Surface Relectance includes 9 spectral bands. A spectral band isolates specific information from the image that can be used to gather additional insights. Spectral Bands one through seven isolate specific ranges of light and are measured in wavelengths, while bands 10 through 11 measure the heat reflected from the earth in Kelvins (bands eight and nine are not used). Band one (B1) represents ultra-blue surface reflectance within a range of 0.435 to 0.451 μm. Band two (B2) represents blue surface reflectance within a range of 0.452 to 0.512 μm. Band three (B3) is green surface reflectance within the range 0.533 to 0.590 μm. Band four (B4) represents red surface reflectance with a range of 0.636 and 0.673 μm. Band five (B5) represents near infrared surface reflectance within a range of 0.851 to 0.879 μm. Band six (B6) is shortwave infrared one and has a range of 1.566 to 1.651 μm. and seven (B7) represents shortwave infrared two within a range of 2.107 and 2.295 μm. Band ten (B10) represents brightness temperature within a range of 10.6 to 11.19 Kelvin. Band eleven (B11) represents brightness temperature as well, within a range of 11.50 to 12.51 Kelvins. Bands 10 and 11 were originally collected with 100m/ pixel resolution, but Landsat 8 Surface Reflectance Tier one scaled it down to 30m/ pixel.

4 Methodology

Our aim was to develop a machine learning algorithm capable of accurately predicting the levels of turbidity in coastal areas using satellite images. To create our dataset we combined NERRs in-situ sensor data with Landsat 8 satellite images obtained from Google Earth Engine image collection. The NERRs sensor data collects water quality readings every 15 minutes. In our experiments we used only turbidity and timestamp value from the NERRS sensor data. The Landsat 8 satellite images captures images in Zeke's Basin every 16 days. To maximize the accuracy of our dataset, we used the timestamps from both datasets to merge them accordingly. Since the sensor data is collected every 15 minutes, we were able to take the average of the turbidity for that day to create a new column in the dataset. Taking the average of the turbidity did help the accuracy of the models and allowed for duplicate days to be dropped from the dataset. This ensured there was one entry for each day.

We used the Landsat 8 surface reflectance image collection which does not include bands 8 and 9; however, bands 1 to 7 along with bands 10 and 11 were used to predict the levels of turbidity. We pulled both sensor and Landsat 8 data from 2013 to present. The sensor data will be used to determine the accuracy of the machine learning algorithm.

After combining the sensor data with the satellite image data, we ran a correlation analysis to find any existing features that would be a good predictor

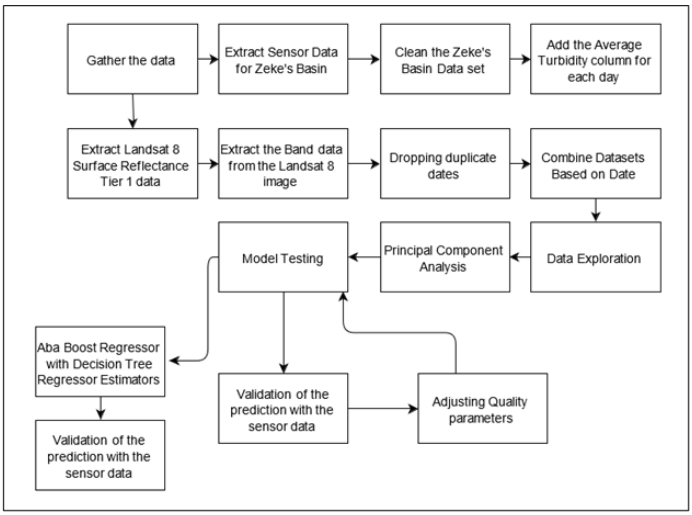


Figure 1: Workflow and methodology

of turbidity. Our analysis did not show any strong correlation between our target feature (turbidity) and other features. After checking for correlation with our target feature, we did a principal component analysis using the raw images, top of atmosphere reflectance and surface reflectance.

Following data exploration, we normalized the data using a standard scaler. We then split the data into testing and training sets, keeping 30% of the data for validation. Following this, multiple models to train the surface reflectance data and the data generated by the principal component analysis. Models synthesized include a Decision Tree Regressor, AdaBoost Regressor, Bagging Regressor, Random Forest Regressor. The workflow and design of methodology is shown in Figure 1. We used Python to pull and clean the data, create our dataset, create the models, and train and test them.

5 Experimental Results

We evaluated our models performance using root mean squared error (RMSE) and r-squared values. The r-squared values of the models trained using principal component analysis were much worse than the surface reflectance. As such, we chose not to refine those models. Regarding the models trained using surface reflectance data, the AdaBoost Regressor with a Decision Tree Regressor estimator produced the highest r-squared value, 97.51%. This model achieved a train RMSE of 24.14 and a test RMSE of 29.44. Out of all the bands used,

the most relevant in predicting turbidity within this model were bands 5 and 11. The weight of all bands used can be seen in Figure 2.

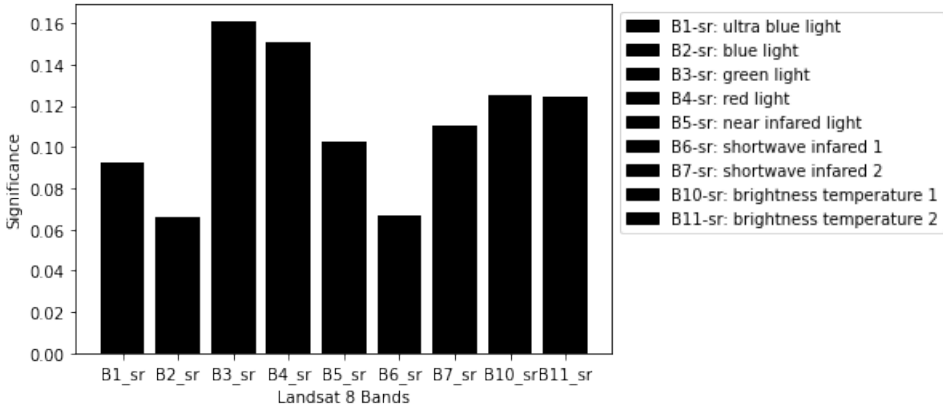


Figure 2: The significance of each band used within the AdaBoost Regressor

The decision tree regressor achieved a 37.29 train RMSE and a 29.44 test RMSE. The models r-squared value was 96.08%. The baggin regressor had the poorest performance of all the models. It attained a train RMSE of 61.32 and a test RMSE of 46.52. The r-squared value of the baggin regressor was 83.19%. The random forest regressor achieved a train RMSE of 31.48 and a test RMSE of 50.99. It's r-squared value was 92.53%. Table 1 has the results for each respective model trained.

Table 1: Results from Model

Model	Train RMSE	Test RMSE	R-Squared
AdaBoost Regressor	24.14	29.44	97.51%
Decision Tree Regressor	37.29	36.93	96.08%
Bagging Regressor	61.32	46.52	83.19%
Random Forest Regressor	31.481	50.99	92.53%

6 Conclusion and Future Work

The main goal of this project was to build a machine learning algorithm that would predict turbidity in coastal and estuarine environments using Landsat 8 Satellite images. With the combination of in-situ NERRs data and Landsat 8 data, we accomplished an r-squared value of 97.53 in predicting turbidity

levels at Zeke's Basin. All models achieved an acceptable accuracy, though the AdaBoost Regressor with a Decision Tree Regressor estimator model out preformed the others. However, improvements can always be made.

Future works should explore different datasets and different locations in order to expand our models applicability. Additionally, we would like to look into using the medium turbidity per day versus the average turbidity, as the medium helps remove outliers in the dataset. Lastly, we would like to modify the model to predict Chlorophyll-a levels as well as turbidity. Through exploring different datasets and adding different quantification's of water quality, we hope the future model will be able to predict turbidity in large range of coastal and estuarine areas.

As anthropogenic activities disturb ecosystem functionality and increase the frequency of natural disasters, it will be incredibly beneficial to be able to predict water quality. In summary, our machine learning algorithm successfully predicted turbidity levels in Zeke's Basin and shows promise for growth and larger application.

7 Acknowledgement

This work is supported by University of North Carolina Wilmington Pedagogy Initiative Grant.

References

[1] National estuarine research reserve: Overview. *NERRs: Overview*.

[2] Theyazn HH Aldhyani, Mohammed Al-Yaari, Hasan Alkahtani, and Mashael Maashi. Water quality prediction using artificial intelligence algorithms. *Applied Bionics and Biomechanics*, 2020, 2020.

[3] Ali Allam and Qingyan Mohd. Retrieval of turbidity on a spatio-temporal scale using landsat 8 sr: A case study of the ramganga river in the ganges basin, india. *Applied Sciences*, 10(11):3702, 2020.

[4] Davies-Colley, R. J. and Smith, D. G. Turbidity suspended sediment, and water clarity: a review. *JAWRA Journal of the American Water Resources Association*, 37:1085–1101, 2001.

[5] J Fear. A comprehensive site profile for the north carolina national estuarine research reserve. *National Estuarine Research Reserve System, Office for Coastal Management, NOAA Ocean Service, Silver Spring, MD*, 2008.

[6] Julien Laliberté, Pierre Larouche, Emmanuel Devred, and Susanne Craig. Chlorophyll-a concentration retrieval in the optically complex waters of the st. lawrence estuary and gulf using principal component analysis. *Remote Sensing*, 10(2), 2018.

[7] Lim, H. S., MatJafri, M. Z., Abdullah, K., and Asadpour, R. A two-band algorithm for total suspended solid concentration mapping using theos data. *Journal of Coastal Research*, 29:624–630, 2013.

[8] Liu, Li-Wei and Wang, Yu-Min. Modelling reservoir turbidity using landsat 8 satellite imagery by gene expression programming. *Water*, 11(7):1479, 2019.

[9] McCutcheon, Steve C. , Martin, James L., and Barnwell Jr., Thomas O. Water quality. *Handbook of Hydrology*, pages 11–73, 1992.

[10] Quang, N. H., Sasaki, J., Higa, H., and Huan, N. H. Spatiotemporal variation of turbidity based on landsat 8 OLI in Cam Ranh Bay and Thuy Trieu Lagoon, Vietnam. *Water*, 9:570, 2017.

[11] Florida Keys National Marine Sanctuary. Water quality describes the condition of the water, including chemical, physical, and biological characteristics, usually with respect to its suitability for a particular purpose such as drinking or swimming, Apr 2011.

[12] Lucas Silveira Kupssinskü, Tainá Thomassim Guimarães, Eniuce Menezes de Souza, Daniel C Zanotta, Mauricio Roberto Veronez, Luiz Gonzaga, and Frederico Fabio Mauad. A method for chlorophyll-a and suspended solids prediction through remote sensing and machine learning. *Sensors*, 20(7):2125, 2020.

[13] Zablotskii, V. R., Le, T. G., Dinh, T. T. H., Le, T. T., Trinh, T. T., and Nguyen, T. T. N. Estimation of suspended sediment concentration using vnredsat–1a multispectral data, a case study in red river, hanoi, vietnam. *Geography, Environment, Sustainability*, 11:49–60, 2018.

Owl Checker: Identifying Misinformation Through the Help of Wikipedia*

Vishal Fenn[1] and Sam Henry[2]
[1]Department of Physics
[2]Computer Science and Engineering
Christopher Newport University
{vishal.fenn.18, samuel.henry}@cnu.edu

Abstract

Twitter has become a hub for people from all areas of life to discuss current events. Unfortunately, it has unintentionally led to an increasing spread of misinformation. In this paper, we present OWL Checker, a machine learning system that can identify if a Tweet contains misinformation or not. Owl Checker first cleans the text of a Tweet, and then queries Wikipedia to retrieve articles relevant to the Tweet. Those articles and the Tweet are passed into a machine learning classifier which determines if the Tweet contains misinformation or not. Owl Checker performs on par with other misinformation detection systems, but since it uses Wikipedia as a source of credible information it can generalize to new and emerging topics.

1 Introduction

There is no question about the enormous impact social media has on our society. These impacts consist of communication, friendships, community, and

advertisement. While there are many positive impacts from social media, there are also negative impacts. With the rise of social technologies, the ease with which misinformation is disseminated has rapidly increased [28]. Misinformation is usually more attractive to people than facts and reasonable arguments. Suciu and Pelfrey state that "Human nature is to simply turn away when things get confusing. For example, if the government is telling you to get vaccinated, and social media is telling you that vaccines and COVID are a hoax, then the easiest thing to do is ignore all of it" [22]. Social media platforms such as Twitter and Facebook have allowed untrue and misleading content to reach wide audiences without credible review [3]. Many people are not aware whether a claim is factual and do not look into the claim themselves, so they are unaware of how to spot false information [21].

One popular social media platform is Twitter, for which users communicate by posting Tweets. Tweets are usually short messages containing text, and possibly hyperlinks, and images. It has been a challenge to identify misinformation on Twitter. Past approaches have relied on trained human fact-checkers to sort out misinformed content, but the amount of information posted on social media has overwhelmed our ability to manually control its spread, and automated methods are needed. Simple automated methods such as identifying and whitelisting credible sources are not always reliable because the credibility of sources can change over time. In order for the content of a Tweet to be considered factual, it should be verified from multiple credible sources. However, finding multiple credible sources can be challenging since many sources will not address all claims.

Wikipedia is the largest online encyclopedia and its contributors actively fight disinformation on Wikipedia [23] [26] [12]. Although Wikipedia is user-created, it has been found to be as accurate as other encyclopedias [7]. Furthermore, Wikipedia contains up-to-date, current information, and most Wikipedia articles contain citations linking information to the source material. These citations often come from multiple sources. Wikipedia is therefore a single trustworthy source that aggregates information from multiple credible sources and is able to address a variety of claims [13]. It therefore serves as an excellent source of information for fact-checking Tweets.

In this paper, we describe the Observing while learning (Owl) Checker, an algorithm that fact-checks Tweets using Wikipedia as a repository of credible information. The algorithm uses a binary classification model to predict whether a Tweet contains misinformation or not. The model performs on par with other systems and has the added benefit of using an external knowledge source to verify the truthfulness of the Tweet. Therefore we believe it will

be more generalize to new categories of news. Code for the model is publicly available [1].

2 Related Works

In this section, we review the previous research related to online misinformation. Memon and Carley [18] identified COVID-19-related misinformation by collecting and classifying social network data in terms of their network structure and community. Donzelli et al [6] identified vaccine misinformation by analyzing videos on YouTube about the link between vaccines and autism or other serious side effects in children. Rather than looking at the contents of a post, recent research has focused on determining if users are likely to post misinformation or not. Huang and Carley [10] categorized and identified misinformed users, based on where they are from and what are their political orientations. Forati et al [8] examined the origin and nature of misinformation and its relationship with the COVID-19 incidence rate using a geodatabase of all geotagged COVID-19 related Tweets. Sharma et al [25] identified unreliable and misleading Tweets based on fact-checking sources. They examined the narratives promoted in Tweets, along with the conversations of the Tweet.

There has been research focused on the algorithms implemented to combat misinformation not just on social media, but on websites and articles as well. Shao et al [24] developed Hoaxy, a web platform for tracking social news sharing. Hoaxy focuses on the temporal relationship between the spread of misinformation and the implementation of fact checking in response to the misinformation. They found that misinformation is produced in a much larger quantity than fact-checking content. Ahmed et al [1] presented a fake news detection model that uses n-grams to represent the context of a document. They evaluated their model using a dataset compiled from real and fake news websites. Patil [19] used several machine learning models such as Decision Trees, Random Forests, and an Extra Tree Classifier to evaluate the performance of a majority voting approach to detect fake news articles. They concluded that the Majority Voting technique better results than the individual learning techniques. Mandical et al [17] experiment with three machine learning algorithms: Multinomial Naive Bayes, Passive-aggressive classifier and deep neural network on eight different datasets acquired from various source to classify fake news. Vo and Kyumin [27] analyzed linguistic characteristics of fact-checking Tweets and built a deep learning framework based on Seq2Seq models with attention mechanisms to automatically generate fact-checking responses for fact-checkers. Conroy et al [4] used the info box of Wikipedia to verify the

[1] https://bitbucket.org/vkf63516/capstone-resources/src/master/

credibility of claim. The fake news detection system was a combination of linguistics and machine learning with network-based behavioral data. Patwa and Sharma [20], the creators of a dataset used in this paper, compared four machine learning algorithms: Decision Tree, SVM, Gradient Boost and Logistic Regression. They trained the models by learning patterns specific to COVID-19. Das and Basak [5] use a heuristic ensemble approach on the same dataset where the network takes into account the effect of important aspects of news items as statistical features.

3 Methodology

Owl Checker consists of three main steps. First, the query formulation step inputs some text (e.g. a Tweet) and performs some text cleaning to formulate a query. That query is passed to the information retrieval module, which retrieves relevant factual information from Wikipedia. The cleaned text and the retrieved factual information are passed into a machine-learning-based classifier that predicts whether the Tweet contains misinformation or not. We explain each of these components in the next few subsections.

3.1 Text Cleaning and Query Formulation

The purpose of the query formulation step is to form the search string used in the information retrieval step. To do this, we collect relevant words from a Tweet and concatenate them into a single space-separated search string. To collect these relevant words, we first clean the text, then perform part-of-speech (POS) tagging. For text cleaning we use Tweet-Preprocessor[2] to remove URLs, hashtags, mentions, and emojis. For part of speech tagging we use the POS tagger implemented in TextBlob [16], which is an extension of the NLP library Natural Language Processing Toolkit (NLTK). We identify adjectives, singular nouns, and proper nouns. We briefly experimented with other POS tags, but found their inclusion negatively impacted the quality of the search results. We found that verbs in particular increased the amount of irrelevant pages returned. In the rare scenario where a Tweet contains no adjectives, singular nouns, or proper nouns, we extract noun phrases from that Tweet. We form a search query by combining each of the POS tagged words into a single space-separated string.

[2] `https://github.com/s/preprocessor`

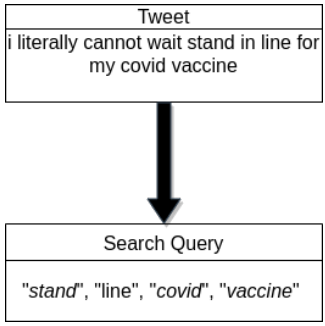


Figure 1: In the text cleaning and query formulation step, a Tweet is input, the text is cleaned, and adjectives, singular nouns, and proper nouns are extracted and form the search query.

3.2 Information Retrieval

The next step in Owl Checker is the information retrieval step in which we retrieve relevant Wikipedia articles. We input the query into the Wikipedia search engine using the Wikipedia-API [3], which returns a ranked list of articles. For simplicity, we retrieve only the top-ranked article. In come cases, the search query may be too specific andno articles may be retrieved. In this case, we use only the first three words and search again. In the rare situation where there are still no articles retrieved, we assign the Wikipedia information as "blank" for that Tweet.

The articles returned by the Wikipedia-API contain more than just the text of the article. Each article contains information divided into categories such as references, page sections, and the summary. The summary section contains a brief, generally paragraph-long summary of the contents of the whole article. We found that the summary typically contains enough information to determine if a Tweet is factual or not, so we extract only the summary section of the article.

3.3 Machine Learning Classifier

The last step in Owl Checker is a machine-learning classifier. This classifier model predicts whether the Tweet contains misinformation or not. It takes as

[3]https://github.com/martin-majlis/Wikipedia-API/

Search Query
good, news, property, coronavirus, long-term, vaccine, possible
Article Summary
A COVID-19 vaccine is a vaccine intended to provide acquired immunity against severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2), the virus that causes coronavirus disease 2019 (COVID-19). Prior to the COVID-19 pandemic, an established body of knowledge existed about the structure and function of coronaviruses causing diseases like severe acute respiratory syndrome (SARS) and Middle East respiratory syndrome (MERS). On 10 January 2020, the SARS-CoV-2 genetic sequence data was shared through GISAID, and by 19 March, the global pharmaceutical industry announced a major commitment to address COVID-19. Many countries have implemented phased distribution plans that prioritize those at highest risk of complications, such as the elderly, and those at high risk of exposure and transmission, such as healthcare workers.

Figure 2: Article summary retrieved using the search query

input the cleaned text of a Tweet (text without urls, hashtags, mentions, and emojis) and a reduced form of the article summary. The classifier is limited by the number of words that can be input into it, which is often less than the length of the article summary. Therefore, we extract only the most relevant content from the article summary. We apply the POS tagger to identify sentences containing more than 10 words which are adjectives, verbs, singular nouns, or proper nouns. We believe this eliminates uninformative sentence fragments and selects the most relevant sentences. These sentences are then passed to the next step. The original text of the Tweet is concatenated with the sentences from Wikipedia as data for the model.

Figure 3 shows the architecture of the model. In it, the texts are concatenated and plugged into a Twitter-roBERTa language model [2] [15]. Our system uses the language model to convert the input text (a sequence of tokens) into a numeric representation (an embedding matrix) indicating its meaning. The RoBERTa model can take a maximum input length of 768 and it outputs an embeddings matrix which consists of contextualized vector representations of each token in the input sequence. In the next layer of the model, the embedding matrix is input into a Bidirectional Long Short-Term Memory (BiLSTM) layer. BiLSTMs are extensions of sequential LSTMs that can improve model performance on sequential classification problems [11], but take into account dependencies in both the forward and backward direction. The BiLSTM layer outputs a single vector representing the meaning of the Tweet and Wikipedia Text as a whole. This vector is passed into a 3 layer densely connected neural network. The layers contains 256, 128, and 64 neurons respectively, each with

Gaussian Error Linear Unit (GELU) [9] activation functions. Dropout with a rate of 0.7 is used between each layer to prevent overfitting. Lastly, an output layer is densely connected to the network, which consists of a single neuron and a sigmoid activation function. The final output is a probability between 0 and 1, which indicates the likelihood that the Tweet is credible. To train the model, we used the Adam optimizer [14] and a learning rate of 0.01.

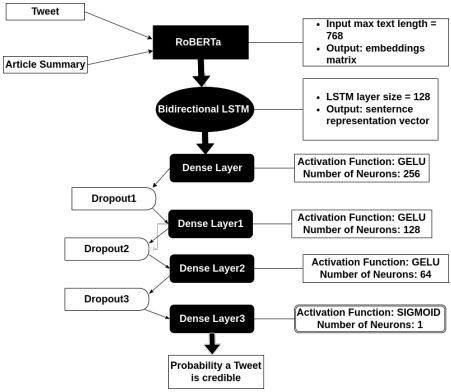


Figure 3: Architecture of the binary classifier. A Tweet and Wikipedia article summary are input and the system outputs the probability that the Tweet is credible.

4 Experimental Details

To evaluate the effectiveness of Owl Checker, We use the Covid Fake News dataset, which is available online [4] [5]. This dataset contains Tweets related to the topic of COVID-19. The data was collected in February 2021 [20]. In total, the training dataset contains 2140 English Tweets. 1120 of which contain factual information, and 1020 of which contain misinformation.

We randomly sample the dataset to divide it into training, validation, and test sets. We reserved 70 percent of the dataset as training data. We split the remaining 30 percent in half, with one half used for the validation set and the other half used for the test set. We used the training set to train the model weights. We used the validation set to tune hyperparameters (learning

[4] https://github.com/diptamath/covid_fake_news/blob/main/data/english_test_with_labels.csv

rate, dropout rate, batch size) and monitor generalization performance during training. We stop training when validation loss stops decreasing for 5 epochs. We use the test set to evaluate the final model performance and withhold it during all training and hyper-parameter tuning. We use accuracy as our primary evaluation metric because it is a nearly balanced dataset.

5 Results and Conclusions

Our trained model achieved a test set accuracy of 0.9221. This outperforms, or performs on par with the models proposed by the creators of the dataset [20], whose models achieved accuracies of 0.8523 (Decision Tree), 0.9196 (Logistic Regression), 0.8696 (Gradient Boost), and 0.9332 (SVM). It performs similarly to Das et al [5] who achieved a performance of 0.972 using a roBERTa-based model. These other use authors don't use an external knowledge source in their models. The patterns learned during training relate the language of credible Tweets to uncredible Tweets present in the training set. Since the training set contains only samples related to COVID-19, these patterns will therefore find patterns of language use specific to COVID-19. In contrast, Owl Checker uses an external knowledge source and learns patterns related to how a Tweet and a truthful article relate to one another. These patterns are not necessarily specific to COVID-19, and for this reason we believe our model is more generalizable to new domains, and emerging topics where no training data is yet available. For example other systems may learn that Tweets containing the text "COVID vaccines are not effective" are misinformation, but our system will learn that if there is contradictory information in the Tweet and article, then the Tweet contains misinformation.

Through our research, we have demonstrated that Wikipedia can be used for machine learning to identify a Tweet as misinformation. There were limitations. Since Owl Checker is a binary classification model, it does not take into account Tweets that are neutral, meaning that there is no information to support or deny the claim. There are Tweets that contain sarcasm but are classified as misinformation. For example, "it would be for an empathy implant for those humans who lack any, like Trump and the GOP," was marked as misinformation because it was assumed to be taken literally. Also, the information from Wikipedia is only collected from the summary and does not account for sections in the article which may contain more information in order to classify a Tweet as misinformation or not misinformation correctly. For future work, it would be interesting to include more search results from the search engine to take into account the information from other articles. It would be intriguing to develop a multi class classifier to identify misinformation and be able to take into account sarcasm.

References

[1] Hadeer Ahmed, Issa Traore, and Sherif Saad. Detection of online fake news using n-gram analysis and machine learning techniques. In *International conference on intelligent, secure, and dependable systems in distributed and cloud environments*, pages 127–138. Springer, 2017.

[2] Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa-Anke, and Leonardo Neves. TweetEval:Unified Benchmark and Comparative Evaluation for Tweet Classification. In *Proceedings of Findings of EMNLP*, 2020.

[3] Cody Buntain and Jennifer Golbeck. Automatically identifying fake news in popular twitter threads. In *2017 IEEE International Conference on Smart Cloud (SmartCloud)*, pages 208–215. IEEE, 2017.

[4] Nadia K Conroy, Victoria L Rubin, and Yimin Chen. Automatic deception detection: Methods for finding fake news. *Proceedings of the association for information science and technology*, 52(1):1–4, 2015.

[5] Sourya Dipta Das, Ayan Basak, and Saikat Dutta. A heuristic-driven ensemble framework for covid-19 fake news detection. *arXiv preprint arXiv:2101.03545*, 2021.

[6] Gabriele Donzelli, Giacomo Palomba, Ileana Federigi, Francesco Aquino, Lorenzo Cioni, Marco Verani, Annalaura Carducci, and Pierluigi Lopalco. Misinformation on vaccination: A quantitative analysis of youtube videos. *Human vaccines & immunotherapeutics*, 14(7):1654–1659, 2018.

[7] Don Fallis. Toward an epistemology of wikipedia. *Journal of the American Society for Information science and Technology*, 59(10):1662–1674, 2008.

[8] Amir Masoud Forati and Rina Ghose. Geospatial analysis of misinformation in covid-19 related tweets. *Applied Geography*, 133:102473, 2021.

[9] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[10] Binxuan Huang and Kathleen M Carley. Disinformation and misinformation on twitter during the novel coronavirus outbreak. *arXiv preprint arXiv:2006.04278*, 2020.

[11] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[12] Rebecca Iannucci. What can fact-checkers learn from wikipedia? we asked the boss of its nonprofit owner, Jul 2017.

[13] Dariusz Jemielniak and Eduard Aibar. Bridging the gap between wikipedia and academia. *Journal of the Association for Information Science and Technology*, 67(7):1773–1776, 2016.

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[15] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[16] Steven Loria, P Keen, M Honnibal, R Yankovsky, D Karesh, E Dempsey, et al. Textblob: Simplified text processing [www document]. *URL https://textblob. readthedocs. org/en/dev/(accessed 4.7. 16)*, 2014.

[17] Rahul R Mandical, N Mamatha, N Shivakumar, R Monica, and AN Krishna. Identification of fake news using machine learning. In *2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, pages 1–6. IEEE, 2020.

[18] Shahan Ali Memon and Kathleen M. Carley. Characterizing covid-19 misinformation communities using a novel twitter dataset, 2020.

[19] Dharmaraj R Patil. Fake news detection using majority voting technique. *arXiv preprint arXiv:2203.09936*, 2022.

[20] Parth Patwa, Shivam Sharma, Srinivas Pykl, Vineeth Guptha, Gitanjali Kumari, Md Shad Akhtar, Asif Ekbal, Amitava Das, and Tanmoy Chakraborty. Fighting an infodemic: Covid-19 fake news dataset. In *International Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situation*, pages 21–29. Springer, 2021.

[21] Gordon Pennycook and David G Rand. The psychology of fake news. *Trends in cognitive sciences*, 25(5):388–402, 2021.

[22] William V. Pelfrey Peter Suciu. Spotting misinformation on social media is increasingly challenging. `https://www.forbes.com/sites/petersuciu/2021/08/02/spotting-misinformation-on-social-media-is-increasingly -challenging/?sh=2ddd49b72771`. Accessed: 2022-04-13.

[23] Pina Schipani. How wikipedia became the world's largest encyclopedia, Jul 2021.

[24] Chengcheng Shao, Giovanni Luca Ciampaglia, Alessandro Flammini, and Filippo Menczer. Hoaxy: A platform for tracking online misinformation. In *Proceedings of the 25th international conference companion on world wide web*, pages 745–750, 2016.

[25] Karishma Sharma, Sungyong Seo, Chuizheng Meng, Sirisha Rambhatla, Aastha Dua, and Yan Liu. Coronavirus on social media: Analyzing misinformation in twitter confafversations. *arXiv preprint arXiv:2003.12309*, 2020.

[26] Natalie Shoemaker. Using wikipedia to automatically fact-check the internet, Sep 2021.

[27] Nguyen Vo and Kyumin Lee. Learning from fact-checkers: analysis and generation of fact-checking language. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 335–344, 2019.

[28] DM West. How to combat fake news and disinformation. brookings (blog), 2017.

Location Based Assignments in Early CS Courses Using *BRIDGES* Engages Students*

Matthew Mcquaigue[1], Jay Strahler[1], Kalpathi Subramanian[1]
Erik Saule[1], Jamie Payton[2]
[1]Computer Science, The University of North Carolina
[2]Computer Science, Temple University
{mmcquaig, jstrahl1, krs, esaule}@uncc.edu, payton@temple.edu

Abstract

Freshmen and sophomore level courses in computer science are critical to long-term student development and success. At the same time, these courses, such as data structures and algorithms are usually challenging and require significant motivation to keep students engaged. In this work, we present through our *BRIDGES* system a set of *location* based assignments that can serve to reinforce core concepts and algorithms by placing them in more meaningful settings and applications, and demonstrate the relevance of computing in the early stages of a student's career. We performed a small pilot study using a subset of these assignments in a special topics course on algorithms, and conducted student surveys after each assignment. The surveys were unanimously positive, and the students enjoyed coding the algorithms as well as the datasets and visualizations associated with the assignments.

1 Introduction

While enrollments in computer science have grown significantly in recent years, future trends indicate a significant need for CS graduates to join the modern workforce [5]. Future jobs will increasingly be more technical, especially in areas such as AI, vision, machine learning, data science and security. Thus,

it is imperative that CS majors possess a strong foundation in their early years to ensure their long-term success; however, foundational courses can be rigorous and not very engaging, and the field has seen significant attrition, especially in the freshmen/sophomore years, and for groups traditionally underrepresented in computing. Engaging majors in the early stages of their CS program with rigorous, yet meaningful and engaging content can provide the means to demonstrate the relevance and possibilities of computing, and help retention.

As part of the *BRIDGES* system [6, 4], we have developed a repository of assignments for early courses in computer science [3]. We had also presented how these could be used in CS1/CS2 [2] and algorithm analysis courses [21]. The *BRIDGES* API (available in Java, C++ and Python) facilitates the use of real-world datasets in early CS courses and uses visualizations to demonstrate student generated work, while maintaining course rigor.

In this work we present a class of *location based assignments* that we have recently developed as part of the *BRIDGES* assignment repository; these assignments rely on data sources that have location (Lat/Long) information. These assignments have two advantages, (1) they continue to provide highly engaging assignments with visual output of classic CS concepts/algorithms, and (2) students' output can be customized to locations or concepts of their own interest. This gives more flexibility to assignments, and getting away from the 'one size fits all' characteristic of typical course assignments. We report on a pilot study of using these assignments in an algorithms course.

2 Related Work

Engagement Strategies. In recent years, a significant amount of effort has focused on strategies to engage and motivate students in early CS courses, in an effort to retain CS majors. Engagement can span many dimensions, involving skills, participation/interaction, emotional and/or performance [11]. Engagement strategies can be classified as either *content-based* or based on pedagogical activities. Content based engagement focuses on making the content of the course or associated activities (assignments, lectures, videos, etc) meaningful and relevant. Engagement strategies based on activities include those based on active learning techniques, which in turn could be based on any combination of lab-based instruction, flipped classroom settings, gamification, peer-learning, and use of multimedia content [18, 10, 12].

Learning Materials and Repositories. Repositories containing learning materials (slides, assignments, videos) are another mechanism to assist instructors find engaging content; these include Nifty Assignment collections [17], En-

gageCSEdu [13], ModelAI [1]. Some of these tend to include the 'fun' factor, such as those based on games [22].

Location Based Materials Many of the Nifty assignments are location based assignments, such as the N-body simulation [20], star chart display [19], voting districts and gerrymandering [25, 15], that lend themselves to nice visualizations that can be highly engaging. Using games and AI as part of assignments also lend themselves to using geometric spaces [7].

3 Location based Datasets

3.1 Earthquakes - US Geological Survey (USGS)

USGS records earthquake data [23] in real-time and makes it available in real time. We have built an interface to gather these earthquake records periodically and store the 10,000 most recent earthquakes in a database. Our *BRIDGES* system provides an API call to retrieve a desired number of these records as objects in C++, Java or Python. Each earthquake record contains the quake magnitude, location (lat/long and quake area), and epoch time.

3.2 OpenStreet Map - Street Networks, Amenity Data

OpenStreet Maps is a publicly available resource [16] providing street networks at multiple resolutions, as well as amenities (schools, hospitals, restaurants, fire stations, etc.). This data, while extremely rich, is also based on contributions by the public and does not guarantee uniform coverage of all areas.

We downloaded the entire map of North America from GeoFABRIK [8]. Additional processing included removing extraneous data to save storage. The data was held in 2 parts, one for the road and street data, and the second holds amenity data, such as restaurants, schools, etc.

In *BRIDGES*, we can retrieve the OSM street network data by specifying a Lat/Long bounding box; the system then returns the set of vertices and edges that make up the street network. Each vertex contains an id and lat/long information. Edges are specified by the source and destination vertex ids. Amenity data is retrieved through a separate query, by providing the amenity name within a rectangular Lat/Long region.

3.3 City/Country Data

The city/country data was obtained from GeoNames [9]. This site provides a single file with every city in the world. Currently, we use only cities in the United States. Each city instance is part of a collection that has a schema

containing an id, city, state, country, latitude, longitude, elevation, and population, and time zone attribute. Currently, *BRIDGES* allows querying this data using a combination of the above attributes; alternately, a Lat/Long bounding box can be specified to extract all the cities contained within the box.

3.4 Elevation Maps and Navigation

Elevation model data was obtained from NOAA [14] on per query basis, using a Lat/Long bounding box. The data is saved locally to the server for future requests. Utility tools are used to convert the raster data into a more human understandable form for use by end users. Note that the resolution varies depending on the queried region, and is specified as part of the query.

3.5 US Census

US Census data, published every 10 years, is available from the US Census bureau [24]. The data collects data on states, counties, census tracts, block groups and blocks. Basic demographic data such as race and sex are part of the data.[1]

4 Location Based Problems, Assignments, Applications

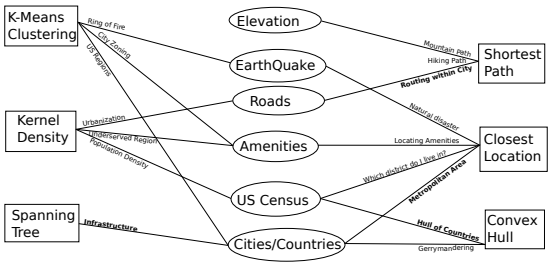


Figure 1: Location Based Assignments: Left and Right columns illustrate relevant algorithms that can take advantage of data sources that contain location information. Middle column illustrates examples of data sets that we have used to construct programming assignments using BRIDGES, all of which lend themselves to visual outputs. Edge labels indicate potential applications.

[1] we have only recently begun looking at this dataset for use with *BRIDGES*.

4.1 Kernel Density

In the kernel density problem, a space (here a 2D space) is discretized in cells. And there are points that may carry a weight. In the density problem, one counts for each cell the (weighted) number of points in the surrounding of that cell. One can count the number of points within the cell or in proximity of the cell. These maps are common in geography. Using different data lead to different analyses, while being an equivalent assignment from a learning objective standpoint:

Population Density. The simplest application of Kernel Density is to compute population density maps using US Census data. One can also use ethnical attribute or age attributes to get a finer picture of US demographics. The census bureau also provide voter registration data which can help further understand voting patterns in the US such as the urban/rural divide.

Underserved region. Using Amenity data collected by Open Street Map, one can compute density maps for public and private services, such as schools, hospitals, restaurants, grocery stores. This can enable identifying food deserts or communities underserved by local governments.

Urbanization. Using road data from Open Street Map, kernel density will provide maps of urbanization of a region.

4.2 Shortest Path

A classic assignment when using map based data is to compute shortest path. There are a number of shortest path algorithms such as Dijkstra's and Ford Bellman, that are applicable. Shortest Path is also appropriate to discuss artificial intelligence methods with algorithms like A*.

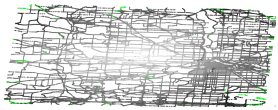


Figure 2: Dijkstra's shortest path algorithm on OpenStreeMap data of Minneapolis

Routing within City. GPS routing is a feature students use commonly and they can get the chance to implement their own. One can use the Open Street Map street level data to build a graph and compute shortest path on it. *BRIDGES* visual attributes can be used to show distance variation from the source, as can be see in Fig. 2.

Mountain Path. The mountain path assignment as written in the Nifty assignment collection [17] works from an elevation map (a gray scale image) which *BRIDGES* makes available from NOAA. The task is to find the lowest cost path, from the left end of the image to the right end of the image, where the cost is defined as the sum of difference in elevation between consecutive pixels in the path. The Nifty assignment uses a simple greedy approach to find a path which is suitable in various classes from CS1 to Algorithms. A

visualization of such a path can be seen in Fig. 3. One can find the optimal path that only ever goes right using a dynamic programming algorithm which is suitable for an Algorithms course. One can also model the image as a graph where each pixel has 8 neighbors and use a shortest path algorithm.

Biking/Hiking Paths. One can combine both Open Street Map street level data and elevation data from NOAA to create a graph for cycling and hiking paths. The length of edges in the graph can be adjusted to account for the difficulty of going uphill and the ease of going downhill.

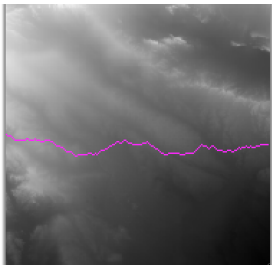


Figure 3: Mountain Path

4.3 K-Means Clustering

K-means clustering is a simple unsupervised method to find clusters in a spatial dataset. A number cluster centroids are distributed randomly initially. Each point in the database is allocated to the region represented by the closest centroid. Then the centroid is updated to be at the barycenter of the points in the region it represents. While k-means clustering is refined further in data mining courses, the naive scheme is suitable for implementation by CS1 students.

Ring of Fire. Computing K-means cluster on the location of recent earthquakes will highlight geological properties of the Earth; For instance, many earthquakes are located around the Pacific ocean and is known as the Ring of Fire.

US regions. Computing K-means on the US city map will highlight the structure of the country in a few main region. And it will highlight a known east coast/west coast divide.

US city zoning. Clustering commercial amenities (from OpenStreetMap) of a particular type will identify commercial zoning in most US city. Alternately, clustering the population of a city using census data will identify residential zoning. The two maps can be compared.

4.4 Closest Location Queries

Given a set of points in a (here 2D) space, a simple query is to find the closest point in the set to a particular location. This form of spatial query is suitable for different levels. At a CS1 level, one can simply put the set in a linear structure, such as a list or an array, and iterate over it to find the closest match. One could also use a uniform partition of the space to accelerate the search. This can be suitable for a project in CS1/CS2. At data structure level,

one could do adaptive partitioning such as quad-trees and K-D trees. Students could even implement different strategies and compare their performance.

Locating Amenities. With OpenStreetMaps data, one can answer questions such as 'Where is the restaurant that is closest to me?', or in a GPS routing scenario, "what is the street corner closest to my location?"

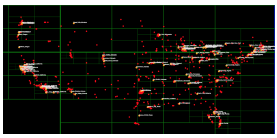


Figure 4: Quadtree Partitioning of 1000 cities

Natural Disaster Risk. With Earthquake data, one can compute for a particular location the closest large earthquake that has been reported.

Identifying metropolitan areas. Using the city location data, the assignment could aim to identify the closest city of a particular size of a particular location. Fig. 4 illustrates an example quadtree partitioning on a set of 1000 cities. This can be further used to approximate Voronoi decomposition of the country based on the 1000 largest cities.

Which census district do I live in? While census districting can be complicated, a student can use location query to identify the closest census district their home is part of and look at the demographics of that district.

4.5 Convex Hull

Location data can also enable a range of geometric problems. A classic one is to compute the convex hull of a set of points. This is useful for visualization purposes but also as a preprocessing step to compute collisions in a physics engine. Computing the convex hull can be done with a range of methods; Jarvis March is a brute force algorithm suitable for CS1; Divide and Conquer (similar in its analysis to merge sort) and Quick Hull (similar to quick sort) are suitable for algorithm courses.

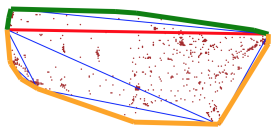


Figure 5: Convex Hull using divide and conquer alg. on US city data

Convex Hull of countries and states. One can use the City data to use as a the input to a convex hull algorithm. This enables the student to see how much their country or state looks like their convex hull. A convex hull of US cities can be seen in Fig. 5.

Visualizing gerrymander. Convex Hull can be used on congressional voting districts (extracted from US census data) as an approximation of gerrymandering.

4.6 Minimum Spanning Tree

Minimum Spanning Trees (MST) are classic chapters in algorithm books and courses with two primary greedy algorithms: Prim's and Kruskal's algorithm. Minimum Spanning tree is often a topic in Algorithms courses. A variant of spanning trees are Steiner Trees that are often used to plan infrastructure project such as train lines, power lines, water lines, optic fibers. However, Steiner trees are often not presented, as the problem is more complicated and does not have an optimal polynomial algorithm.

Figure 6: Prim's MST alg. on cities in North Carolina

Minimum Spanning Tree using City Data. Computing Minimum Spanning tree of city/country data serves as a approximation of infrastructure. Fig. 6 illustrates the MST of a set of cities in North Carolina.

5 Pilot Study: Student Reflections/Feedback

In the fall 2021, a special topics course entitled 'Real World Algorithms' was offered to Junior/Senior CS students at UNC Charlotte for the first time. The course required Data Structures as a prerequisite[2]. The course is being considered for a more permanent offering as part of our degree concentration. Four of the location based projects detailed in Section 4 and highlighted in Fig. 1 were included. The course had five students enrolled. Students completed a project/reflection survey after each module.

Students completed 3 project surveys in the course. The location based projects in the course were part of 2 of the surveys. In particular, it had 2 free form questions:

- Identify the essential concept(s) learned by completing the assignment, and indicate why you think that concept may be important to understanding and learning how to program.
- Comment on what aspects of the project were interesting, meaningful, engaging (or not)? Also, indicate why you liked or disliked using *BRIDGES*.

The surveys were unanimously positive. Students enjoyed coding the algorithms like MST, Convex Hull and Dijkstra's shortest path algorithms as well as the datasets used with the algorithm ('learning the functionality...how to implement Djikstra's algorithms...further increased my confidence in programming', 'greater understanding of geometric structures...how to manipulate them as a data structure'). All students commented positively on the visualization

[2]Our current program does not contain a separate algorithm analysis course

capabilities of *BRIDGES* ('enjoyed graphically displaying the Convex Hull', 'served as a robust visualization and debugging tool') as it helped them understand the output of these algorithms, which are naturally suited to be visually salient.

6 Conclusions

In this work, we presented a class of location based assignments that are highly engaging, visual and can be adopted across the freshmen and sophomore CS courses such as CS1, CS2, Data Structures and Algorithm Analysis. These assignments provide a vital connection to real world applications and problems that will be attractive to CS majors. More importantly, they provide the key advantage of making it *fit the interests of students by making it possible to explore such applications to the location or region of their interest*. We report preliminary results from an algorithm analysis course that tied classic computer science algorithms to real world applications. Specifically, it focused on coupling traditional algorithms to location based datasets, so as make the course content more meaningful to CS students. A small pilot study of an algorithms course was presented where some of these algorithms were assigned as part of the course. Feedback from students were unanimously positive, in terms of both the course content as well as the use of the *BRIDGES* system, which facilitated such real world content and visualization capabilities. However, this study had only five students and was not a required course, thus, results will need to be replicated in both a larger course and avoid self-selection bias. We plan to make this course a more regular offering as part of a degree concentration in the coming year and gather and analyze more formal feedback.

7 Acknowledgements

This work was supported in part by grants from the National Science Foundation (Nos. 1726809, 2142389).

References

[1] AAAI. Model AI Assignments, 2018.

[2] Allie Beckman, Matthew Mcquaigue, Alec Goncharow, David Burlinson, Kalpathi Subramanian, Erik Saule, and Jamie Payton. Engaging Early Programming Students with Modern Assignments Using BRIDGES. volume 35, page 74–83, Evansville, IN, USA, apr 2020. Consortium for Computing Sciences in Colleges.

[3] BRIDGES Development Team. BRIDGES Assignment Repository. `http://bridgesuncc.github.io/newassignments.html`, 2022.

[4] BRIDGES Development Team. BRIDGES Website. `http://bridgesuncc.github.io`, 2022.

[5] Bureau of Labor Statistics. Occupational outlook handbook. https://www.bls.gov/ooh/computer-and-information-technology/computer-and-information-research-scientists.htm#tab-6.

[6] David Burlinson, Mihai Mehedint, Chris Grafer, Kalpathi Subramanian, Jamie Payton, Paula Goolkasian, Michael Youngblood, and Robert Kosara. BRIDGES: A System to Enable Creation of Engaging Data Structures Assignments with Real-World Data and Visualizations. In *Proceedings of ACM SIGCSE 2016*, pages 18–23, 2016.

[7] Adrian A. de Freitas and Troy B. Weingart. I'm going to learn what?!? teaching artificial intelligence to freshmen in an introductory computer science course. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*, page 198–204, New York, NY, USA, 2021. Association for Computing Machinery.

[8] GeoFABRIK. Openstreetmap data extracts. https://download.geofabrik.de/.

[9] GeoNames. Geonames geographical database. http://www.geonames.org/.

[10] Mark Guzdial. A media computation course for non-majors. In *Proceedings of the ITICSE 2003*, pages 104–108, 2003.

[11] M.M. Hendelsman, W.L. Briggs, N. Sullivan, and A. Towler. A measure of college student course engagement. *Journal of Educational Research*, 98(3):166–175, 2005.

[12] Diane Horton, Michelle Craig, Jennifer Campbell, Paul Gries, and Daniel Zingaro. Comparing outcomes in inverted and traditional CS1. In *Proceedings of the ITICSE 2014*, pages 261–266, 2014.

[13] Alvaro Monge, Beth A. Quinn, and Cameron L. Fadjo. Engagecsedu: CS1 and CS2 materials for engaging and retaining undergraduate cs students. In *Proceedings of ACM SIGCSE*, SIGCSE '15, pages 271–271, 2015.

[14] howpublished = https://www.ngdc.noaa.gov/mgg/global/global.html NOAA, title = Digital Elevation Models.

[15] A. Obourn. Gerrymandering. http://nifty.stanford.edu/2019/obourn-gerrymandering/.

[16] OpenStreetMap contributors. Openstreemap. https://www.openstreetmap.org, 2021.

[17] Nick Parlante. Nifty assignments, 2018.

[18] Johanna Pirker, Maria Riffnaller-Schiefer, and Christian Gütl. Motivational active learning: Engaging university students in computer science education. In *Proceedings of ITICSE*, pages 297–302, 2014.

[19] K. Reid. Star charts and constellations. http://nifty.stanford.edu/2009/reid-starmap/.

[20] R. Sedgewick and K. Wayne. N-body simulation. http://nifty.stanford.edu/2017/wayne-nbody/.

[21] Jason Strahler, Matthew Mcquaigue, Alec Goncharow, David Burlinson, Kalpathi Subramanian, Erik Saule, and Jamie Payton. Real-world assignments at scale to reinforce the importance of algorithms and complexity. *J. Comput. Sci. Coll.*, 35(8):166–175, apr 2020.

[22] Kelvin Sung, Rebecca Rosenberg, Michael Panitz, and Ruth Anderson. Assessing game-themed programming assignments for CS1/2 courses. In *Proceedings of GDCSE*, GDCSE '08, pages 51–55, 2008.

[23] US Geological Survey. USGS earthquakes.

[24] United States Census Bureau. 2020 census state redistricting data (public law 94-171) summary file technical documentation. Technical report, August 2021.

[25] K. Wayne. Purple america. http://nifty.stanford.edu/2014/wayne-purple-america/.

A Pilot Study of a Virtual Informal Experiential Learning Activity during COVID-19 *

Mohammad Q Azhar[1] and Ada Haynes[2]
[1]Computer and Information Systems
BMCC, The City University of New York
New York, NY 10007
mazhar@bmcc.cuny.edu
[2] Sociology and Political Science
Tennessee Tech University
Cookeville, TN 38505
AHaynes@tntech.edu

Abstract

During COVID-19, undergraduate students in the urban college in New York City lost opportunities for almost all co-curricular informal learning outside the classroom. This paper describes the design, implementation, and preliminary evaluation of an "Virtual MAKE-A-THON" during 2020 at an urban HSI community college to provide underrepresented minority students a collaborative experiential co-curricular learning activity. Given today's rapid changes in technology, informal learning activities are important tools to bridge gaps with the newly emerging technologies for our undergraduate students. Informal learning activities can be designed to expose students to the new technologies that are not easily accessible within the traditional curriculum of structured, formal classroom learning activities while simultaneously developing their computational thinking, critical thinking, creativity, and soft skills. In

addition, little research has been conducted on remote informal experiential learning environments for community college students that incorporate the students into collaborative teams solving challenging real world problems. In this paper, we share the successful implementation and evaluation of a virtual informal collaborative experiential learning activity called MAKE-A-THON during COVID-19 where community college students work in a team to create practical innovative solutions to global problems with mentor support from both academia and industry. The goal of the proposed informal learning model is to provide students real-world career preparation and technical education across disciplines through collaborative project activities while experiencing cutting-edge technologies. This paper will also include the pilot study of the impact on diverse students' computational thinking, critical thinking, creativity, and self-efficacy. The results are from surveys from 31 students participating in this month-long virtual MAKE-A-THON during Fall 2020.

1 Introduction

The fact that COVID-19 disrupted formal learning at an unprecedented level is evident. Less attention had been paid, however, to the equally important losses in informal experiential learning. The learning culture of informal settings is especially important for underrepresented minority (URM) students, who have been among the hardest hit by COVID-19 and its deleterious effects on their educational access. Overburdened educators struggling to cope and adapt their formal learning environments to the virtual setting need tools for also adapting their informal learning practices to promote remote learning. The urgent need for innovative tools for remote informal learning to combat learning losses, lack of engagement, and high dropout rates for URM and community college students has been exacerbated by the COVID pandemic. As such, the solutions we are developing to these challenges should be put to good use by informing how we frame ongoing and future educational initiatives. The tools students are using to engage with remote learning can be developed to incorporate experiential learning into the classroom. In addition, little research has been conducted on informal, experiential learning remote challenge environments for community college students. The project's two objectives encompass program design and evaluation. For the first objective, URM teams of students from community college are engaged in a remote interdisciplinary informal collaborative learning environment to implement computing for social good in projects to design culturally sensitive, real-world, data-driven solutions to societal issues using AI and data science. The second objective is to conduct research on this informal learning environment to study its impact on students' self-efficacy and on their knowledge and comfort with computational thinking,

critical thinking, problem solving, and STEM. The two main objectives of this project are: (1) design a remote interdisciplinary informal collaborative learning environment to engage URM teams of students from community college to apply computing for social good, and (2) conduct research on our informal learning challenge environment to study students' self-efficacy and its impact on their knowledge and comfort of computational thinking, critical thinking, problem solving, and STEM. Computing for social good includes designing culturally sensitive, real-world, data-driven solutions to address societal issues using AI and data science. This paper presents a virtual, informal, experiential co-curricular learning activity designed to engage URM students to derive solutions to address societal issues. The paper also presents results from a quantitative analysis of data from 31 students participating in these month-long virtual informal experiential learning activities during Fall 2020.

2 Related Work

Reinholz and Andrews' theory of change [12] hypothesizes that active small-group structured learning will result in greater student persistence. Our interventions are "scaffolded" in a way that builds student confidence as content becomes more complex [10]. By including industry partners and links with the STEM community utilizing the designed competition events, students can experience their own ability to make meaningful changes in the world around them [13]. Beir et. al, found that including in the first four semesters a project-based course had positive effects including on perceptions of the purpose of participating in STEM courses, higher career aspirations, and increased skill building [5]. Incorporating project-based learning activities to engage students in STEM in an informal environment has been demonstrated to increase STEM engagement and positively expose students to a variety of STEM career options [14]. This supportive learning structure can give students evidence of their strengths, replacing a deficit approach that amplifies weaknesses [10], to prepare students to remain committed to graduation and prepare them for the job market where they will be expected to collaborate across disciplines [18]. Overall, the project combed evidence-based approaches including project-based learning ([6], [8]), STEM-CIS and non-CIS membership, and authentic research experiences [5] to reach students where they are and bring them to a new level of expertise, they may not have thought possible. The project created an environment that affirms the value of students' culturally diverse backgrounds by implementing a Culturally Responsive-Sustaining (CR-S) Framework grounded by culturally responsive pedagogy ([9], [11]) with a critical lens toward inequality. Calling on prior experience with informal Hack-A-thon programs, the project design created an online setting where students can practice mu-

tual respect and collaborate across differences in the activities in which they engage.

3 Designing a virtual informal experiential co-curricular learning environment

We designed an experiential co-curricular learning activity, MAKE-A-THON, to engage interdisciplinary student teams in a longer-term collaborative Zoom synchronous activity over the span of one month supported by faculty, student peer mentors, and industry professional mentors. MAKE-A-THON collaborative projects for social good are designed to be scaffolded [1] and culturally responsive to encourage and support students' passion for being engaged to solve problems relevant to their world. The MAKE-A-THON collaborative projects for social good incorporated a global Call for Code challenge. Call for Code is a global initiative led by IBM to apply crowd-sourced coding solutions to address societal issues [1]. The challenge required students to collaborate virtually to design a new or speculative product to solve real-world problems using ideas drawn from the field of AI/data science and produce a website and video to communicate the significance of the product. The MAKE-A-THON had the following phases: Phase 1 - Issue a MAKE-A-THON challenge for social good (e.g., COVID-19, racial injustice, and United Nations Sustainable Goals [3]. Our industry collaborators, faculty, and student peer mentors provided supplemental virtual workshops about Computational Thinking (CT) in the workplace, problem-solving, design-thinking, GitHub collaborative platform, entrepreneurship, leadership, and team building to prepare them to work on a collaborative project. Students were invited to participate through various outreach activities involving faculty, public affair and programming club. Student teams were formed across disciplines and work on the project was incentivized by comprising a component of course grade. Courses in which students were enrolled who participated in this project were remote. The team work on the project occurred using virtual platforms. In future iterations of the project, we will implement in-person team activities in order to evaluate differences in student responses. We provided all MAKE-A-THON students with hands-on virtual tech talks in the IBM AI chatbot. Phase 2 - Students choose one topic in which to pursue a badge credential (i.e., micro-credential) from Amazon [7] and IBM's credential program [2]. Badges are linked to the LinkedIn profile to aid them in future employment. Each team of students was able to choose a category of change of social good projects from which to develop their intervention within. These categories were predetermined by the professors of the courses involved. In each category projects were evaluated by industry panelists who chose top winners. Students had freedom to design

their solution creatively within the category they chose. Our students learned how to problem solve collaboratively while incorporating CT with the support of industry mentors and professors. Students used virtual platforms to meet and develop solutions to challenges across computer science, marketing and sociology disciplines, mirroring the approaches they will need to use in the real world. Phase 3 - teams of students worked on their chosen collaborative project with support from student peer mentors, industry mentors, and faculty. The MAKE-A-THON Collaborative Projects for social good ended with a showcase in which each student team presented their project were evaluated by a panel of judges, comprised of the industry representatives. The top projects include innovations addressing concerns such as "student anxiety", which addresses **good health and Wellness**, and "food scarcity", which responds to **Zero Hunger**, each of which are United Nations Sustainable Development Goals.

4 Research Methods

This research project is a pilot study and employs mixed methods of quantitative and qualitative research. This paper presents only the results from quantitative analysis. Future papers will include qualitative analysis. The research was conducted over two semesters at an urban, HSI, community college. It was conducted to explore the impact of an informal experiential learning activity, MAKE-A-THON, in a virtual environment.

4.1 Human Subjects

All human subjects' guidelines were followed in this study. This includes submission and approval as exempt research from the local Institutional Review Board.

4.2 Survey

Students participating in the MAKE-A-THON completed a survey that asked questions about their knowledge, comfort, and self-efficacy prior to and after their MAKE-A-THON experience. This survey consisted of 25 questions. The survey questions were modified from surveys by [14]. These questions included demographic information (e.g., gender identification, their field of study, why they selected their major, parents' highest level of education, student classification, and race and ethnicity). Students also completed Likert questions related to their knowledge (5-point), comfort (5-point), and self-efficacy (6-point) prior to the MAKE-A-THON and after participating in the MAKE-A-THON. Since students were not necessarily familiar with the term computational thinking, this term was defined for them on the survey as "Computational thinking is

the type of thinking used in a MAKE-A-THON. It involves thinking logically to solve problems and abstracting principles and applying them in other situations." Students also completed qualitative questions such as "what did you gain from participating in the MAKE-A-THON?".

5 Results

Students participating in the MAKE-A-THON completed one survey after the project which asked them to compare and contrast their level of comfort and competency with the materials before the project as part of an evaluation. A total of 31 students of 31 participants completed the survey. Of these 21 (68%) identified as female and 10 (32%) identified as male. No students selected the third option "prefer not to answer" (See Figure 1).

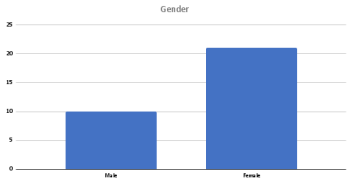


Figure 1: Gender

This contrasts with women only earning 18% of the degrees in computer science nationwide [4]. Students were asked about their field of study. All but one student (in Liberal Arts) were STEM students. Predominantly (90%), students were in computer science, computer information systems, or computer network technology. The remaining students were in math or engineering. Students were asked why they selected their major. They were able to select multiple reasons. The figure below reflects their responses by the percentage of people of each gender selecting the response. Three interesting findings are exhibited in this graph. Students who identified as female, as opposed to identifying as male, were more likely to select their field of study based upon the influence of others (e.g., teachers, mentors, counselors) or say that they always thought that they would study in this field. For both groups, the number one reason for selecting the field was that they were interested in the subject. Interestingly, the prospect of making a difference was the third highest-rated choice for females and the fourth highest-rated for males. Selection bias may have played a role in the high number of students choosing to make a difference as their reason for participation because topics of the MAKE-A-THON were all categorized as social good. Nevertheless, this is still important to note that many students in computer science want to make a difference. Having opportunities like this can help close the gender gap in STEM and computer science more specifically. This is supported by the MAKE-A-THON having a disproportionate attendance of females compared to the composition of computer science at this institution and nationwide. A

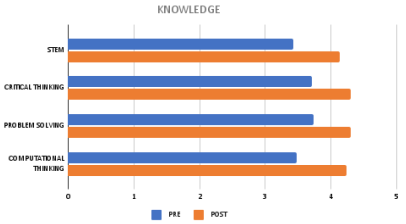


Figure 2: KNOWLEDGE Pre & Post

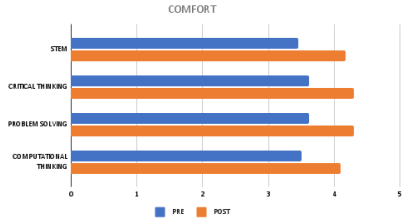


Figure 3: COMFORT Pre & Post

large percentage of the students participating were first-generation college students. Approximately 39% of students reported that neither of their parents had attended college. We are currently analyzing our qualitative surveys to investigate how events like this can help attract and retain first-generation college students and will include in the future publication. Paired t-tests were conducted to assess changes in students' knowledge, comfort, and self-efficacy prior to the MAKE-A-THON and after the MAKE-A-THON. Data were eliminated pairwise for each analysis. Statistically significant gains in knowledge of computational thinking (t=-4.490, p<.001), problem solving (t=-3.798, p <.002), critical thinking (t=-3.844, p<.002), and STEM (t=-4.583, p<.001) were found (See Table 1 and Figure 2).

Table 1: Paired t-tests statistics for Knowledge

		Mean	N	Standard Deviation	Std. of Mean	T	P
Computational Thinking	Pre	3.47	30	1.008	.184	-4.490	<.001
	Post	4.23	30	.679	.124		
Problem Solving	Pre	3.73	30	1.015	.185	-3.798	<.002
	Post	4.30	30	.750	.137		
Critical Thinking	Pre	3.70	30	.952	.174	-3.844	<.002
	Post	4.30	30	.651	.119		
STEM	Pre	3.43	30	1.073	.196	-4.583	<.001
	Post	4.13	30	.860	.157		

Paired t-tests statistics were also conducted for students' comfort level in each of these areas. Again, significant gains in comfort were between pre and post conditions for computational thinking (t=-4.039, p, <.001), problem solving (t=-3.768, p<.002), critical thinking (t=-3.931, p<.002), and STEM (t=-4.372, p<.001) (See Table 2 and Figure 3).

Paired t-tests were similarly conducted to measure students' self-efficacy related to their confidence in their ability to solve real-world problems related to STEM, do computational thinking, and make meaningful changes in the world around them. Again, students had significant gains in their self-efficacy in each of these areas: solve real-world problems related to STEM (t=-3.465,

Table 2: Paired t-tests statistics for Comfort							
		Mean	N	Standard Deviation	Std. of Mean	T	P
Computational Thinking	Pre	3.50	30	1.075	.196	-4.039	<.001
	Post	4.10	30	.712	.130		
Problem Solving	Pre	3.62	29	1.237	.230	-3.768	<.002
	Post	4.28	29	.702	.130		
Critical Thinking	Pre	3.62	29	1.147	.213	-3.931	<.002
	Post	4.28	29	.649	.121		
STEM	Pre	3.45	29	1.213	.225	-4.372	<.001
	Post	4.14	29	.789	.147		

Table 3: Paired t-tests statistics for Self-Efficacy							
		Mean	N	Standard Deviation	Std. of Mean	T	P
Solve Real World Problems Related STEM	Pre	3.90	30	1.647	.301	-3.465	<.003
	Post	4.80	30	1.297	.237		
Computational Thinking	Pre	4.23	30	1.569	.286	-2.947	<.007
	Post	4.97	30	1.159	.212		
Meaningful Change	Pre	4.23	30	1.675	.306	-2.673	<.02
	Post	5.10	30	1.296	.237		

p<.003), computational thinking (t=-2.947, p<.007), and meaningful changes (t=-2.673, p<.02) (See Table 3 and Figure 4).

The results from the pre and post paired t-tests indicate that the MAKE-A-THON successfully improved students' knowledge, comfort, and self-efficacy. All gains were significant with students making almost an entire point gain in their self-efficacy in solving real-world problems related to STEM students were given the opportunity to respond to an open-ended question about what they gained from participating in the MAKE-A-THON. The responses were overwhelmingly positive. To additionally support these results, a student team from the MAKE-A-THON participated in NSF ATE STEM Innovation Challenge. This team won a prize for designing a virtual reality learning environment for Autistic children.

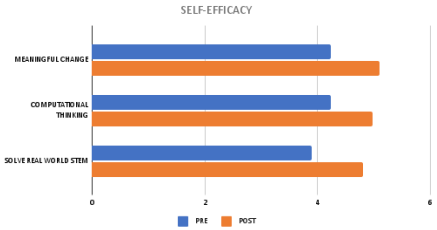


Figure 4: SELF-EFFICACY Pre and Post

6 Conclusion and Future Work

The study findings of our informal virtual co-curricular learning "MAKE-A-THON" activities support our contention that informal learning and exposure

to CT skills within a project-based setting reduces intimidation and increases self-efficacy among participants, regardless of previous experience with or comfort with STEM. If these findings are supported by more research in the future, it could provide an avenue toward broadening STEM participation. Initial results from the quantitative data and student comments from students who completed their MAKE-A-THON projects point strongly in this direction, making a clear case for scaling this research to the activities described in this proposal, to create generalizable models to share with other institutions. In this pilot study we explore only self-reported data, in future works we will look at student retention data and other data to complement the self-reported data. Over the course of this project, we will iterate this intervention incorporating changes and evaluating the efficacy of each. For example, the concept of a semester long MAKE-A-THON has given us the opportunity to contrast student outcomes to the format of a daylong Hack-A-thon. Future iterations will incorporate changes to incentives such as proportion of course grade, incorporating the MAKE-A-THON into a longer-term internship, and involving course sections from new departments into the interdisciplinary student teams. For future endeavors there is a need to further expand these activities to include more non-computer science students. Further research can explore the impact of race, ethnicity, and gender on these informal virtual co-curricular remote learning activities. For example, does providing these kinds of virtual informal co-curricular learning activities promote self-efficacy for women and other minorities and help retain them in STEM as this research suggests? Additional research needs to be conducted on similar informal co-curricular learning experiences over time and across different types of informal learning experiences. Long term studies need to be conducted to see if activities such as these experiences impact beyond students' self-reporting knowledge, comfort, and self-efficacy. It would be interesting to explore if these experiences improve students' retention and success.

References

[1] Call for code. `https://callforcode.org/`. Last accessed, May 12th, 2022.

[2] Skillsbuild. `https://skillsbuild.org/students`. Last accessed, May 12th, 2022.

[3] United nations' sustainable goals. `https://sdgs.un.org/goals`. Last accessed, May 12th, 2022.

[4] Women in computer science. https://www.computerscience.org/resources/women-in-computer-science. Last accessed, May 12th, 2022.

[5] Margaret E Beier, Michelle H Kim, Ann Saterbak, Veronica Leautaud, Sandra Bishnoi, and Jaqueline M Gilberto. The effect of authentic project-based learning on attitudes and career aspirations in stem. *Journal of Research in Science Teaching*, 56(1):3–23, 2019.

[6] Stephanie Bell. Project-based learning for the 21st century: Skills for the future. *The clearing house*, 83(2):39–43, 2010.

[7] David Gibson, Nathaniel Ostashewski, Kim Flintoff, Sheryl Grant, and Erin Knight. Digital badges in education. *Education and Information Technologies*, 20(2):403–410, 2015.

[8] Cindy E Hmelo-Silver, Ravit Golan Duncan, and Clark A Chinn. Scaffolding and achievement in problem-based and inquiry learning: a response to kirschner, sweller, and. *Educational psychologist*, 42(2):99–107, 2007.

[9] Gloria Ladson-Billings. Culturally relevant pedagogy 2.0: aka the remix. *Harvard educational review*, 84(1):74–84, 2014.

[10] Julie P Martin, Nathan Hyungsok Choe, Jared Halter, Margaret Foster, Jeffrey Froyd, Maura Borrego, and Erica R Winterer. Interventions supporting baccalaureate achievement of latinx stem students matriculating at 2-year institutions: A systematic review. *Journal of Research in Science Teaching*, 56(4):440–464, 2019.

[11] Django Paris. Culturally sustaining pedagogy: A needed change in stance, terminology, and practice. *Educational researcher*, 41(3):93–97, 2012.

[12] Daniel L Reinholz and Tessa C Andrews. Change theory and theory of change: what's the difference anyway? *International Journal of STEM Education*, 7(1):1–12, 2020.

[13] Julie K Silver, David S Binder, Nevena Zubcevik, and Ross D Zafonte. Healthcare hackathons provide educational and innovation opportunities: a case study and best practice recommendations. *Journal of medical systems*, 40(7):1–7, 2016.

[14] Katherine N Vela, Rachelle M Pedersen, and Macie N Baucum. Improving perceptions of stem careers through informal learning environments. *Journal of Research in Innovative Teaching & Learning*, 13(1):103–113, 2020.

Tutorial on Automating Configuring Parallel Compute Environments*

Conference Tutorial

Bryan Dixon
Computer Science Department
California State University - Chico
Chico CA, 95929
bcdixon@csuchico.edu

We presented work at a recent CCSC on how we automated the creation of small compute clusters for students [5]. This work discussed how we provided a guide to students to purchase two Jetson Nano boards and a suite of Ansible playbooks for the students to automate setting their boards up as a compute cluster [4][1]. We mentioned in this work an unintended consequence: students were using the playbooks to create multi-node clusters on local and cloud VMs instead of buying Jetson Nano boards.

These playbooks can work on more than just Jetson Nanos, which led to our idea for this tutorial. Since we have these ansible playbooks that could work to set up two or more nodes as a compute cluster on pretty much any Debian-based nodes. We thought it would be worthwhile walking interested faculty through step by step how to get a two-node cloud-hosted compute cluster setup using these playbooks.

This is a hands on tutorial will have the following:

1. Introduce the basic concept, end goal, and applications
2. Get everyone Google Cloud coupons and help them apply them
3. Walk through setting up two GCP virtual machine instances to be used for our example
4. Get tutorial repository onto a GCP VM instance.
5. Get dependencies for Ansible setup
6. Configure Ansible machine inventory for the VM instances created
7. Run playbooks

8. Use provided test code to validate our cluster works

Computers and the internet will be necessary to engage in hands-on activities. Google Cloud's EDU grant folks will be helping to facilitate coupons for attendees to use so that you can do the tutorial at no direct costs[3]. Google for Education is a platinum national partner for CCSC [2].

Biography

Dr. Bryan Dixon is an Associate Professor of Computer Science and has served students of diverse backgrounds for over ten years through various courses, from sophomore to graduate level. He spent a large part of his recent professional growth and teaching building systems and tools to help other faculty teach complex systems or DevOps concepts. He helped found the ACM-W student chapter at California State University Chico in 2020 and has continued to promote computing education and career paths for young women.

References

[1] Buy the latest jetson products. `https://developer.nvidia.com/buy-jetson/`, Oct 2020.

[2] Google cloud ccsc national partner. `https://www.ccsc.org/partners/google/`, Apr 2022.

[3] Google cloud higher education programs. `https://cloud.google.com/edu`, Apr 2022.

[4] Red Hat Ansible. `https://www.ansible.com`.

[5] Bryan Dixon. Automating configuring parallel compute environments for students. *Journal of Computing Sciences in Colleges*, 37(4):25–29, 2021.

Reflective Curriculum Review for Liberal Arts Computing Programs*

Conference Tutorial

Jakob Barnard[1], Grant Braught[2], Janet Davis[3], Amanda Holland-Minkley[4], David Reed[5], Karl Schmitt[6], Andrea Tartaro[7], James Teresco[8]

[1]University of Jamestown, Jamestown, ND 58405
Jakob.Barnard@uj.edu

[2]Dickinson College, Carlisle, PA 17013
braught@dickinson.edu

[3]Whitman College, Walla Walla, WA 99362
davisj@whitman.edu

[4]Washington & Jefferson College, Washington, PA 15317
ahollandminkley@washjeff.edu

[5]Creighton University, Omaha, NE 68178
DaveReed@creighton.edu

[6]Trinity Christian College, Palos Heights, IL 60463
Karl.Schmitt@trnty.edu

[7]Furman University, Greenville, SC 29690
andrea.tartaro@furman.edu

[8]Siena College, Loudonville, NY 12211
jteresco@siena.edu

The ACM/IEEE-CS/AAAI curricula task force is currently developing an updated set of Computer Science Curricula guidelines, referred to as CS202X (since the release date is not yet determined). Information about the task force and preliminary drafts of the Knowledge Areas that will be included in the guidelines can be found online at http://csed.acm.org. To assist institutions in applying the new guidelines, CS202X will also publish a Cur-

ricular Practices Volume. This volume will include an article by the SIGCSE Committee on Computing Education in Liberal Arts Colleges (SIGCSE-LAC Committee) that will focus on designing or revising CS curricula in liberal arts contexts. Liberal arts colleges, and smaller colleges in general, face unique challenges when designing curricula. Small faculty sizes, limits on the number of courses that can be required for a major and the need for flexibility in student programs of study constrain designs. However, these environments also provide the opportunity to craft distinctive curricula fitted to institutional mission, departmental strengths, locale, student populations and unique academic experiences. These challenges and opportunities, combined with the size of prior curricular recommendations, have often forced smaller programs to assess trade-offs between achieving full coverage of curricular recommendations and their other priorities.

The SIGCSE-LAC Committee has heard from many faculty that their institutional and departmental contexts have indeed complicated the adoption of prior curricular guidelines. While the CS2013 and upcoming CS202X recommendations provide some flexibility for curriculum designers by dividing content into core and supplemental categories, smaller colleges still face challenges selecting content and packaging it into coherent curricula. To assist in this process, the committee is developing guidance for effectively integrating CS202X as a part of the design, evaluation and revision of computer science and related programs in the liberal arts. This guidance will encourage faculty to reflect on their programs and the role of CS202X, beginning with their institutional and departmental priorities, opportunities and constraints. Ultimately, this guidance will be presented in the committee's article in the CS202X Curricular Practices volume.

This session will open with an overview and brief discussion of the current CS202X draft. Participants will then begin working through a preliminary version of the committees' reflective assessment process. This process is framed by a series of scaffolding questions that begin from institutional and departmental missions, identities, contexts, priorities, initiatives, opportunities, and constraints. From there, participants will be led to identify design principles for guiding their curricular choices including the CS202X recommendations. Participants will leave the session with a better understanding of how CS202X can impact their programs and a jumpstart on the reflective assessment process. Feedback on the process and this session are welcome and will be used to refine the committee's guidance prior to its publication in the CS202X Curricular Practices volume.

Presenter Biography

One of the eight co-authors of this session plans to serve as presenter. **Andrea Tartaro** is an Associate Professor of Computer Science at Furman University. Her computer science education research focuses on the intersections and reciprocal contributions of computer science and the liberal arts, with a focus on broadening participation. She is a member of the SIGCSE-LAC Committee, and has published and presented in venues including the CCSC and the SIGCSE Technical Symposium.

Other Author Biographies

Jakob Barnard is Chair and Assistant Professor of Computer Science & Technology at the University of Jamestown. He is a member of the SIGCSE-LAC Committee and his research involves how curricula has been integrated into Liberal Arts Technology programs. **Grant Braught** is a Professor of Computer Science at Dickinson College. He is a facilitating member of the SIGCSE-LAC Committee, has organized committee events focused on curricula and has published widely on issues related to CS education, particularly within the liberal arts. **Janet Davis** is Microsoft Chair and Associate Professor of Computer Science at Whitman College, where she serves as the department's founding chair. She co-organized SIGCSE pre-symposium events in 2020 and 2021 on behalf of the SIGCSE-LAC Committee. **Amanda Holland-Minkley** is Chair and Professor of Computing and Information Studies at Washington & Jefferson College. Her research explores novel applications of problem-based pedagogies to CS education at the course and curricular level. She is a facilitating member of the SIGCSE-LAC Committee. **David Reed** is a Professor of Computer Science and Chair of the Department of Computer Science, Design & Journalism at Creighton University. He has published widely in CS education, including the text *A Balanced Introduction to Computer Science*, and served on the CS2013 Computer Science Curricula Task Force. **Karl Schmitt** is Chair and Assistant Professor of Computing and Data Analytics at Trinity Christian College. He has served on the ACM Data Science Task Force and various Computing, Technology, Mathematics Education related committees for the MAA and ASA. His interests explore data science education, and interdisciplinary education between computing, mathematics, data, and other fields. **Jim Teresco** is a Professor of Computer Science at Siena College. He has been involved in CCSC Northeastern for almost 20 years and currently serves as regional board chair, and has been involved with the SIGCSE-LAC Committee for 3 years. His research involves map-based algorithm visualization.

Markov Bot: Automated Text Generation*

Joe Meehean
Department of Computer Science
University of Lynchburg
Lynchburg, VA 24501
meehean.j@lynchburg.edu

Markov Bot is a project which uses regular expressions, Markov chains, and n-grams to generate text in the style of a specific author. This is a moderately difficult project which I assign as a capstone for CS2. It uses several data structures and provides nice scaffolding to support students in developing a reasonable complex project

Automated text generators are AI programs that generate text. They are used for a wide variety of applications, from twitter bots to automated journalism. In this project, students develop a simple automated text generator that can generate sentences in the style of a given author. As an example, given *Alice in Wonderland* as input, my implementation of this project generated the following sentence: *'Why, SHE, of course,' said the Dodo, pointing to Alice with one finger; and the whole party swam to the shore.*

This project generates sentences using a combination of Markov chains and n-grams. A Markov chain is created for a given corpus and the chain is traversed to generate new text in the style of the corpus. One problem with this simple type of Markov chain text generation is that the sentences generated seem like complete gibberish, like a word salad. Sentences can run on for a long time and the words may appear in the right proportion, but they are not thematically connected. This project improves on the naive approach by adding n-grams to our model, which greatly improves the grammar and structure of the generated sentences.

Markov Bot exposes the students to several important topics. They use regular expressions to parse the corpus into words/tokens. Markov chains demonstrate the power of randomized/lottery algorithms and provide practice generating and traversing graphs. N-grams are a simple, but powerful, tool and this project highlights that. Correctly programming this project requires several data structures, including: maps, queues, sets, and lists.

Students are given three weeks to complete this project at the end of CS2. They can earn up to a C for the simplified unigram implementation and up to an A for the full n-gram implementation.

This project has several Nifty aspects or strengths. First, it makes a good capstone project for CS2 because in several places it requires students to select the correct data structures without an explicit prompt. This is not a "list project" or a "map project"; rather it requires lots of different data structures throughout. There are even advantages to selecting hashmaps for some parts of the project and treemaps in others. Second, this project can be broken into milestones that scaffold. The first milestone is the lexical parsing. The second is the naive implementation of a Markov chain without n-grams. And finally, introducing n-grams drastically improves the text generation. Further, these milestones are easy to test which speeds grading and allows early feedback to keep students on the right track.

Giving students the appropriate level of assistance in developing projects can be difficult. The third strength of this project is that I can give students lots of starter files without making the project too easy. I typically give students the results of parsing an example corpus and a collection of sentences generated given that corpus and specific random seed. This lets students check their work without making the project too easy.

Markov Bot has some weaknesses. The primary weakness is that there are a number of libraries and software projects that use similar techniques to create Twitter or Discord chat bots. However, for students inclined to cheat, the large amounts of code in these projects dedicated to connecting to Discord or Twitter will likely overwhelm them. A secondary weakness is that there is lots of overlap between the unigram and n-gram text generating components, but not a lot of opportunity for code reuse. This leads to copy and pasting between these components which, as a general rule, I try to avoid.

This project depends on finding good, simple corpi which can be difficult. However, Project Gutenberg (`https://www.gutenberg.org/`) has over 60,000 public-domain eBooks, many in ASCII or UTF-8.

It would be easy to create variants or extensions to this project for interested, engaged students. The obvious choice is to connect the program to Twitter or Discord to collect a corpus of messages and then periodically post newly generated messages. Making the Markov chain persistent using a database or the file system would also provide an interesting challenge.

Reviewers — 2022 CCSC Southeastern Conference

Achee Bonnie Southeastern ..... Louisiana University
Alhashim Amin Ghalib ..... Arizona State University
Alvin Chris ..... Furman University
August Stephanie Elizabeth ..... Loyola Marymount University
Barlowe Scott ..... Western Carolina University
Bennett Brian ..... East Tennessee State University
Besmer Andrew ..... Winthrop University
D'Antonio Lawrence ..... Ramapo College
Dasgupta Anurag ..... Valdosta State University
Ensari Tolga ..... Arkansas Tech University
Goddard Wayne ..... Clemson University
Gunay Cengiz ..... Georgia Gwinnett College
Hamid Fahmida ..... New College of Florida
Holliday Mark ..... Western Carolina University
Hong Gongbing ..... Georgia College and State University
Hutchings Dugald Ralph ..... Elon University
Johnson Erin ..... CodeCrew
Lindoo Ed ..... Regis University, Denver
Liu Yi ..... Georgia College State University
Marcolino Anderson da Silva ..... Federal University of Paraná
Ngo Linh ..... West Chester University
North Sarah ..... Kennesaw State University
Ormond Pat ..... Utah Valley University
Plank James ..... University of Tennessee
Roach Jeff ..... East Tennessee State University
Robertson Cindy ..... Georgia Gwinnett College
Sanft Kevin ..... University of North Carolina Asheville
Shemroske Kenneth ..... University of Southern Indiana
Spurlock Scott ..... Elon University
Verdicchio Michael ..... The Citadel
Works Karen ..... Florida State University