The Journal of Computing Sciences in Colleges

Papers of the 31st Annual CCSC Rocky Mountain Conference

October 21st-22nd, 2022
Utah Valley University
Orem, UT

Bin Peng, Associate Editor
Park University

Pam Smallwood, Regional Editor
Regis University

Volume 38, Number 2 **October 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:

Serving the CCSC: These members are serving in positions as indicated:
Bin Peng, Associate Editor, bin.peng@park.edu, Park University - Department of Computer Science and Information Systems, 8700 NW River Park Drive, Parkville, MO 64152.
Ed Lindoo, Associate Treasurer & UPE Liaison, elindoo@regis.edu, Anderson College of Business and Computing, Regis University, 3333 Regis Boulevard, Denver, CO 80221.
George Dimitoglou, Comptroller, dimitoglou@hood.edu, Department of Computer Science, Hood college, 401 Rosemont Ave. Frederick, MD 21701.
Megan Thomas, Membership System Administrator, mthomas@cs.csustan.edu, Department of Computer Science, California State University Stanislaus, One University Circle, Turlock, CA 95382.
Karina Assiter, National Partners Chair, karinaassiter@landmark.edu.
Deborah Hwang, Webmaster, hwangdjh@acm.org.

CCSC National Partners

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

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 Rocky Mountain Conference

Welcome to the 31st annual conference of the Rocky Mountain (RM) Region of the Consortium for Computing Sciences in Colleges. This year we are running our first in person conference after being virtual for two years due to the COVID-19 pandemic. The CCSC RM region board members are grateful for the authors, presenters, speakers, attendees, and student participating in this year's conference.

This year we received 13 paper submissions on a variety of topics, of which 9 papers were accepted for presentation in the conference. Multiple reviewers, using a double-blind paper review process, reviewed all submitted papers for the conference. The review process resulted in an acceptance rate of 69%. In addition to the paper presentations, there will be two peer reviewed tutorials/workshops, as well as workshops from Google for Education. We truly appreciate the time and effort put forth into the reviewing process by all the reviewers. A special thank you goes to co-Submission chair Karina Assister who worked with co-chair, Mohamed Lotfy. Without their dedicated effort, none of this would be possible.

The CCSC RM region board would like to thank our national partners: Turing's Craft, Google for Education, GitHub, the National Science Foundation (NSF), Codio, zyBooks, the National Center for Women Information Technology (NCWIT), TERADATA University Network, Mercury Learning and Information, Mercy College, and the Association for Computing Machinery in-cooperation with SIGCSE.

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

Troy Taysom
Utah Valey University
Conference Chair

2022 CCSC Rocky Mountain Conference Steering Committee

Troy Taysom, Conference chair	Utah Valley University, UT
Troy Taysom, Site chair	Utah Valley University, UT
Mohamed Lotfy, Program Chair	Utah Valley University, UT
Karina Assiter, Submission Co-chair	Landmark College, VT
Mohamed Lotfy, Submission Co-chair	Utah Valley University, UT
Pam Smallwood, Editor	Regis University, CO
Dan McDonald, Webmaster	Utah Valley University, UT
Sayeed Sajal, Publicity chair	Utah Valley University, UT
Michael Leverington, Student Posters chair	Northern Arizona University, AZ

Regional Board — 2022 CCSC Rocky Mountain Region

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

Reviewers — 2022 CCSC Rocky Mountain Conference

Hands-on Working Examples of How to Perform Advanced Systems Administration on a Virtualized Environment *

Conference Tutorial

Mohamed Lotfy and Christian Fredrickson
Utah Valley University
Orem, Utah 84058
{MohamedL, fredrich}@uvu.edu

In order to prepare IT graduates for current and future practices of virtualized computing resources which integrate IT systems and services as well as the virtualization of networks and network services, DevOps and microservices, system administration courses need to introduce the different tools used for continuous integration and deployment like Git, containers, etc.. Current system administration tools like yum, rpm, dpkg, Git, Kubernetes, and Docker should be introduced and applied in the different hands-on activities thus allowing students to gain the needed knowledge of current best practices in system deployment, virtualization and services. Connecting the systems and data networks knowledge with current best practices, like microservices and containers within a specific deployment context, enables students to develop and gain current professional system administration competency.

Tutorial Description

In this tutorial we will provide hands-on working examples of how to perform advanced systems administration on a virtualized environment. We will introduce containers using Docker and deploy different containerized servers to replicate current organization infrastructure. We will demonstrate the tutorial tasks using VMware Workstation Pro or Oracle VM VirtualBox CentOS VMs on personal laptops or PCs[1, 2, 3]. In addition, we will share the current virtual infrastructure and environment utilized by Utah Valley University.

In the tutorial we will illustrate and explain the following:

1. How to deploy containers using Docker
2. How to create containerized virtul networks using Docker
3. Installing a containerized Bind Daemon and Named Services using Docker
4. Installing a Postfix mail server using containers
5. Deploy and run a Docker swarm of nginx Web servers

Tutorial program

Step-by-step deployment and management of containerized services in a virtual environment to mimic current systems administration practices.

Expected outcomes

Attendees will exit the tutorial with a working VMware or VirtualBox environment and learn how to perform some current advanced systems administration tasks on CentOS.

Target audience

Any faculty who desires to incorporate a virtual environment and use it to apply current systems administration tasks that could be used in a advanced systems administration course.

Prerequisites

Attendees should be familiar with Linux, networking, and programming knowledge (Java, C++, Python, etc.). It is highly recommended that attendees bring their own laptops with VMware or VirtualBox and four CentOS VMs installed on a NAT network.

References

[1] CentOS. CentOS Repositories, 2022. `https://centos.pkgs.org/`.

[2] ORACLE. ORACLE VM VirtualBox, 2022. `https://www.virtualbox.org/`.

[3] VMware. VMware Workstation Pro, 2022. `https://www.vmware.com/products/workstation-pro.html`.

Event Log Analysis for Threat Hunting & Detecting Suspicious/Malicious Behavior*

Conference Tutorial

Basil Hamdan
Department of Information Systems & Technology
Utah Valley University
Orem, UT 84058
basil.hamdan@uvu.edu

This workshop aims to introduce participants to the basics of Event Log Analysis in a Windows environment. Attendees interested in cybersecurity in general and in log analysis in particular will be introduced to Event Logs and the Event Viewer tool as the default logging capability on a Windows-based system. The workshop will discuss and demonstrate several use cases where log analysis is used to find evidence of events such as successful/failed login attempts, starting and stopping of services, and brute-force attacks. Given the limitation of the Event Viewer tool, the workshop will then discuss how to utilize Sysmon, a Sysinternals tool, for threat hunting and detecting malicious behavior such as lateral movement and process injection. Participants will be provided with event log files with pre-generated events. To maximize their learning, it is recommended that the attendees bring laptops with virtualization software (e.g., VirtualBox) pre-installed on them. This will allow the participants to import pre-built Windows 10 machines and work directly with the log files using Event Viewer and Sysmon.

Teaching API Development Using Scaffolding-inspired Techniques*

Thyago Mota
Department of Computer Sciences
Metropolitan State University of Denver
Denver, CO 80217
tmota@msudenver.edu

Abstract

APIs have become one of the most important building blocks of modern software applications. Developing an API is a complex task that requires different skills and abilities, many of which students (in undergraduate CS programs) are still developing. Scaffolding is an interactive learning technique where instructors provide conceptual support to help the learners develop skills to be able to work independently. Scaffolding learning is beneficial because it teaches students how to complete tasks that they are not very familiar with. This paper proposes the use of scaffolding to teach API development by breaking learning material into progressively more challenging tasks. Throughout the scaffolding learning process students are given a consistent support structure based on a model-driven development approach.

1 Introduction

APIs (Application Programming Interfaces) have become one of the most important building blocks of modern software applications. Through APIs, access to web content is not limited to human users interacting via a web browser, but has expanded to other systems and applications. APIs provide a straightforward mechanism of communication between applications through a set of well-defined commands and standardized data exchange formats [8]. These

API characteristics have allowed companies and organizations to provide innovative products and services with a global reach. APIs have quickly become one of the most popular mechanisms for integrating systems and applications in our connected world.

Due to the relevance of APIs, it is crucial that computer science bachelor's degree programs provide their students, in particular those with an emphasis in software engineering, with practice in API design and development during their education. It is not surprising that the ACM's 2020 Computing Curricula lists "Design and implement an API using an object-oriented language and extended libraries, including parameterization and generics on a small project" as one of its software engineering competencies [5]. Encouraging students to work on projects that not only involve the use of APIs, but also their design and implementation, brings numerous benefits. For example, while working on API-related projects students can put to test the knowledge and skills they acquired throughout their education, in areas such as software engineering, database, programming, and computer networks.

Scaffolding is an interactive learning technique where instructors provide conceptual support to help students develop skills that qualify them to work independently in the future. Maybin et al. [10] describe scaffolding as that initial help to *enable a learner to accomplish a task which they would not have been quite able to manage on their own* ... and *intended to bring the learner closer to a state of competence which will enable them eventually to complete such a task on their own.* It is an interactive process because it involves gradually reducing the instructor's assistance as students increase their understanding of the task.

Model-driven development (MDD) is a software development methodology that emphasizes the construction of domain models from which specific applications are derived [17]. It is generally described as the use of a domain-specific language to formulate a model, followed by model transformations. For example, the OpenAPI Specification (OAS) [12] is a commonly accepted language used in describing APIs. There is a plethora of tools available which can perform simple model transformations that generate code in the programming language of choice directly from OAS. One of the most direct benefits of MDD is the potential for increased productivity in software development. The use of formalized and tested structures also contributes to gains in software quality. In addition, studies suggest that novice developers are more comfortable with MDD compared to code-centric development approaches and that the code produced from MDD is less prone to errors [2] [7] [9].

Developing an API is a complex task that requires different skills and abilities, many of which students are still developing. Scaffolding learning is beneficial because it teaches students how to complete tasks that they are not very familiar with. This paper proposes the use of scaffolding to teach API development by breaking learning material into progressively more challenging tasks. Throughout the scaffolding learning process students are given a consistent support structure based on a model-driven development approach.

The rest of the paper is structured as follows. Section 2 proposes a model for API development based on the MDD approach. Section 3 illustrates how the model proposed can be used in API development with progressive levels of difficulty. Section 4 discusses the contributions of this work and points out future developments.

2 A Model for API Development

The proposed model for API development consists of three layers: design, transformation, and implementation. The model was conceived to be independent of the programming language, database, or platform used in the API's implementation. It was created to give students interested in learning API development an abstract way of conceptualizing the various steps involved in the process.

Figure 1 describes the proposed model for API development by means of a flowchart. The design layer consists of a data model and an API specification. The data model describes the API's data entities that are required to be in a database for persistency. This includes different types of information needed by the API and the relationships between them. Examples of data models are Entity Relationship Diagrams (ERD), relational schemas, and JSON-based dynamic-schemas. The API specification is a document that describes the API using a formal language. For example, OAS is a formal language that can be used to describe APIs in YAML or JSON.

In a simplified way, APIs can be described by schema data types and paths to schema objects (sometimes called resources). To illustrate, consider the simple "Quotes API" described in OAS-YAML and shown in Code Listing 1. The "Quotes API" defines only one schema data type named "Quote" with the following attributes: id, text, author, popularity, category, and tags. The "/quotes/0" path allows users to retrieve a JSON document containing a (random) "Quote" object, with "0" being used to represent any quote id.

The proposed model for API development identifies three types of (model) transformations: DB modeling, DB introspection, and code generation. DB modeling generates a database schema from a given data model. A database schema consists of one or more statements that are used to create a database. For example, tools like ERDPlus [3] can be used to transform an ERD or a relational schema into SQL statements. DB introspection extracts database metadata information and uses it to create schema data types for the API

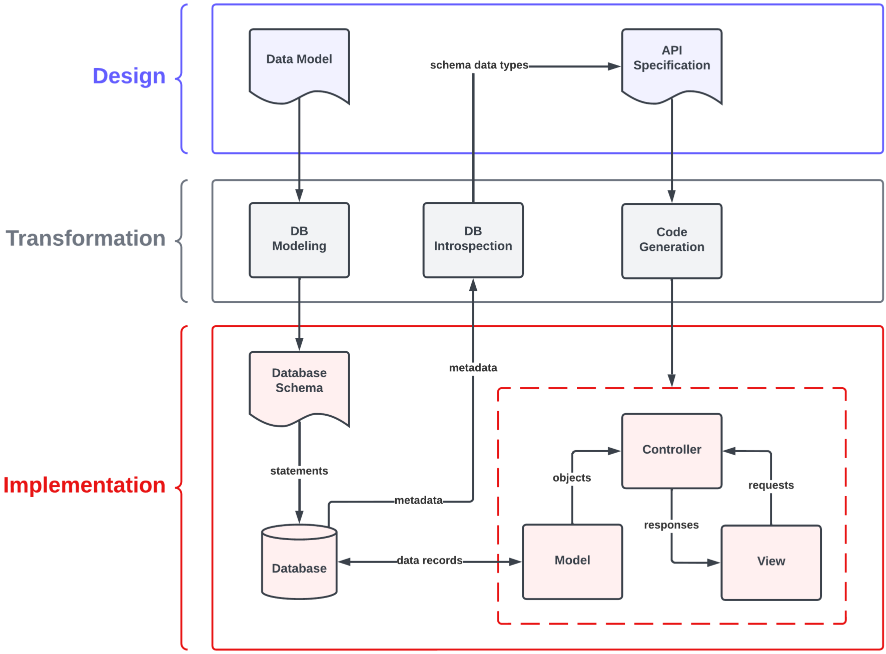


Figure 1: API Development Model.

specification. Since it is not unusual to start the development of an API from a database that already exists, this type of model transformation can help speed up the API design process. Code generation, as the name implies, generates a complete (or semi-complete) code based on the API's specification. Ideally, a code generator would split the API's code into Model, View, and Controller (MVC) components, adding desirable design features to the API implementation, like the separation of concern, for example.

Listing 1: "Quotes API" OAS Specification

```
openapi: 3.0.0
info:
  version: "1.0"
  title: Quotes API
components:
  schemas:
    Quote:
      required:
        - id
        - text
        - author
      properties:
        id:
          type: integer
        text:
          type: string
        author:
          type: string
        popularity:
          type: number
        category:
          type: string
        tags:
          type: array
          items:
            type: string
paths:
  /quotes/0:
    get:
      summary: Returns a random quote
      responses:
        200:
          description: A random quote
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/Quote"
```

The lowest layer of the proposed model is associated with aspects related to implementation, such as data persistency, object-entity mapping, and processing user requests. When it comes to implementing an API there are many options to choose from, like the database, programming language, or web framework. The proposed model intentionally leaves those choices open so that it can be more easily adapted to what instructors decide to use in their teaching activities. The next section exemplifies some of those choices.

3 Instructional Scaffolding Examples

After students are introduced to the proposed model for API development, they can be challenged to implement APIs with progressing levels of difficulty. This section suggests a few lab activities, variations of the "Quotes AP" introduced earlier, that can be adapted for specific instructional needs. These lab activities were carefully designed to incorporate features commonly found in real-world APIs, such as data persistency, parameterization, pagination, and authentication. Each lab is described in detail to facilitate students with no previous experience in API development. Code and detailed documentation for the labs can be found at the “api-labs” GitHub repository [11].

All the lab APIs were specified in OAS 3.0.0 (YAML) and implemented in Python 3.8.9 together with FastAPI [4], a popular framework for building APIs. FastAPI uses Starlette [18], a lightweight ASGI framework. The labs described in this section use Uvicorn [19] as the ASGI-compatible web server that runs the example APIs. Code Listing 2 illustrates how to create a simple web application using FastAPI.

Listing 2: A FastAPI Web Application

```
from fastapi import FastAPI
app = FastAPI()
@app.get('/')
async def root():
    return {"message": "Hello World"}
```

All labs begin by asking students to configure a Python virtual environment, which creates isolated and reproducible build environments. Once the virtual environment is activated, students are instructed to install all the library dependencies listed in a “requirements.txt” file.

3.1 Lab-00 - Quotes API (baseline)

The goal of this lab is to implement the "Quotes API," a simple API that returns a random quote and described in Code Listing 1. The quotes used in this lab were based on Kaggle’s Quotes Dataset [6]. The lab begins by asking the students to use fastapi-code-generator [1], a FastAPI code generator from an API specification in OAS. Code Listing 3 and 4 show key parts of the code generated by the fastapi-code-generator tool and that correspond to the “Quotes API” model and controller, respectively.

The “Quotes API” model defines the “Quote” class based on the API specification schema with the same name. The “Quote” class is specialized from the Pydantic’s “BaseModel” class. Pydantic [13] is a Python package that can help parse and validate data. The web application created for the “Quotes API” redirects GET requests on path “/quotes/0” to function “get_quotes_0,” which in turn responds with a “Quote” object. Code Listing 5 shows the updated code for the controller.

Listing 3: "Quotes API" Model

```
class Quote(BaseModel):
    id: int
    text: str
    author: str
    popularity: Optional[float] = None
    category: Optional[str] = None
    tags: Optional[List[str]] = None
```

Listing 4: "Quotes API" Controller

```
@app.get('/quotes/0', response_model=Quote)
def get_quotes_0(response: Response) -> Quote:
  """
  Returns a random quote
  """
  pass
```

Listing 5: "Quotes API" (updated) Controller

```
quotes = [...] # hardcoded Quote objects (not shown due to space constraints)
@app.get('/quotes/0', response_model=Quote)
def get_quotes_0(response: Response) -> Quote:
    """
    Returns a random quote
    """
    response.status_code = 200
    raw_json = quotes[random.randint(0, len(quotes))]
    quote = Quote(
        id=raw_json['Id'],
        text=raw_json['Quote'],
        author=raw_json['Author'],
        category=raw_json['Category'],
        popularity=raw_json['Popularity'],
        tags=raw_json['Tags'])
    return quote
```

In keeping with scaffolding learning practice, we find it beneficial for the instructor to conduct a review session with their students at the end of each activity/lab. Due to space limitation for this paper, the other labs are described only briefly. More details can be found at the "api-labs" GitHub repository mentioned earlier and available at [11].

3.2 Lab-01 - Quotes API + SQL Database

In this lab, students are challenged to incorporate an SQL database to the "Quotes API" developed in Lab-00. An initialization script is made available for students to be able to prepare an SQLite database with 10K quotes. Yamlgen is an in-house Python script that creates schema data types in OAS-YAML format by extracting metadata from an SQLite database [16]. Open-source DB introspection tools like yamlgen can be adapted to work with different databases as needed. Sqlacodegen [14] also uses DB introspection but this time to generate SQLAlchemy [15] model code. SQLAlchemy is a popular choice for object-relational mapping in Python. The controller for this lab is updated to use the new SQLite database. FastAPI requires its response models to be pydantic-compatible. However, the "Quote" class in this lab is no longer a pydantic model, but instead, an SQLAlchemy model. One way around this problem is to change the value of the "response_model" annotation parameter in “get_quotes_0” from “Quotes” to “dict” or to simply remove the parameter (loosing FastAPI type checks). The pros and cons of each approach should be discussed with students.

3.3 Lab-02 - Quotes API + SQL Database + Path Parameter

Previous labs required the path to a random quote to be specified as “/quotes/0.” In Lab-02 users can request a specific quote based on the quote’s id indicated in the path. Because the requested quote might not exist, a 404 “Not Found” response is added to the API’s specification.

3.4 Lab-03 - Quotes API + SQL Database + Path & Query Parameters

This lab makes the quote’s id parameter optional by adding a “/quotes” new path aimed to return a list of quotes that satisfy a set of criteria informed using (optional) query parameters.

3.5 Lab-04 - Quotes API + SQL Database + Path & Query Parameters + Pagination

API pagination gives users the ability to select which group of objects of the same type to return. This lab implements a limit-offset pagination of quotes that fit a given search criteria. Two more (optional) parameters were added to the ones described in the previous lab: “limit” (the maximum number of quotes to be returned) and “offset” (how many quotes should be skipped).

3.6 Lab-05 - Quotes API + SQL Database + Path & Query Parameters + Pagination + Authentication

A key parameter is added to “Quotes API” to provide a simple authentication mechanism.

4 Conclusions and Future Work

This paper presents a set of well documented lab activities that can be used for teaching API development. In line with the scaffolding learning approach, the lab activities provide increasingly challenge examples based on a proposed model for API development. The proposed model not only covers aspects related to design, but also the application of model transformations to help the actual implementation of the APIs. The author plans to extend this work by creating more lab activities that include the use of non-relational databases and the additional of rate-control and more sophisticated authentication mechanisms commonly found in APIs.

References

[1] Koudai Aono. FastAPI code generator, 2022. `https://github.com/koxudaxi/fastapi-code-generator`.

[2] África Domingo, Jorge Echeverría, Óscar Pastor, and Carlos Cetina. Evaluating the benefits of model-driven development. In Schahram Dustdar, Eric Yu, Camille Salinesi, Dominique Rieu, and Vik Pant, editors, *Advanced Information Systems Engineering*, pages 353–367, Cham, 2020. Springer International Publishing.

[3] ERDPlus. A database modeling tool for creating entity relationship diagrams, relational schemas, star schemas, and SQL DDL statements, 2022. `https://erdplus.com`.

[4] FastAPI. The FastAPI framework, 2022. `https://fastapi.tiangolo.com`.

[5] Association for Computing Machinery (ACM) and IEEE Computer Society (IEEE-CS). Computing curricula 2020: Paradigms for global computing education, 2020. `https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2020.pdf`.

[6] Kaggle. Quotes dataset, 2022. `https://www.kaggle.com/datasets/akmittal/quotes-dataset`.

[7] Timos Kapteijns, Slinger Jansen, Sjaak Brinkkemper, Henry Houët, and R. Barendse. A comparative case study of model driven development vs traditional development : The tortoise or the hare. 2010.

[8] Arnaud Lauret. *The Design of Web APIs*. Manning, 2019.

[9] Yulkeidi Martínez, Cristina Cachero, and Santiago Meliá. MDD vs. traditional software development: A practitioner's subjective perspective. *Information and Software Technology*, 55(2):189–200, 2013. Special Section: Component-Based Software Engineering (CBSE), 2011.

[10] Janet Maybin, Neil Mercer, and Barry Stierer. Scaffolding: Learning in the classroom. *Thinking Voices: The work of the National Oracy Project*, pages 186–195, 1992.

[11] Thyago Mota. API-labs, 2022. https://github.com/thyagomota/api-labs.

[12] OpenAPI-Initiative. OpenAPI specification v3.1.0, 2022. https://spec.openapis.org/oas/latest.html.

[13] Pydantic. Data validation and settings management using Python type annotations, 2022. https://pydantic-docs.helpmanual.io.

[14] SQLacodegen. A tool that generates SQLAlchemy model code, 2022. https://pypi.org/project/sqlacodegen.

[15] SQLAlchemy. The Python SQL toolkit and object relational mapper, 2022. https://www.sqlalchemy.org.

[16] SQLite. A C-language library that implements an SQL database engine, 2022. https://www.sqlite.org.

[17] Thomas Stahl and Markus Volter. *Model-Driven Software Developmen*. Wiley, 2006.

[18] Starlette. The little ASGI framework that shines, 2022. https://www.starlette.io.

[19] Uvicorn. An ASGI web server for Python, 2022. https://www.uvicorn.org.

Teaching Case Study: Introducing Quantitative Risk Assessments in a Cybersecurity Risk Management Course*

Basil Hamdan
Department of Information Systems & Technology
Utah Valley University
Orem, UT 84058
basil.hamdan@uvu.edu

Abstract

Despite the recent development of rigorous quantitative approaches for cybersecurity risk assessment, much of the focus in the pedagogical materials remains on teaching qualitative and semi-quantitative assessment approaches. To help fill this gap, this paper provides a scenario-based teaching case that introduces students in a Cybersecurity Risk Management course to FAIR; an advanced quantitative framework for risk assessment. The case study utilizes a fictitious company, for which a risk assessment is underway, and requires the students to use the FAIR framework to determine the risk exposure that the company faces from a threat scenario against one of its mission-critical information resources.

1 Background

In the context of Cybersecurity Risk Management, risk assessment is defined as "the process of identifying, estimating, and prioritizing risks to organizational operations (including mission, functions, image, reputation), organizational assets, individuals, other organizations, and the Nation, resulting from the operation of an information system" [4]. Among other activities, this process incorporates asset identification and prioritization, threat assessment, vulnerability analysis, and risk determination. Here, risk is defined as "a measure of the extent to which an entity is threatened by a potential circumstance or event, and typically is a function of: (i) the adverse impact, or magnitude of harm, that would arise if the circumstance or event occurs; and (ii) the likelihood of occurrence" [2]. A similar definition is offered in [3], where risk is defined as "a function of the likelihood of a given threat source's exercising a particular potential vulnerability, and the resulting impact of that adverse event on the organization". Mathematically, risk is computed by the following equation:

$$Risk = Likelihood * Impact$$

Risk and its contributing factors can be assessed in a variety of ways, including quantitatively, qualitatively, or semi-quantitatively. Table 1 provides short descriptions of each approach [3].

Table 1: Risk Assessment Approaches

Approach	**Description**
Qualitative	Employ a set of methods, principles, or rules for assessing risk, based on nonnumerical categories or levels (e.g., low, moderate, high).
Quantitative	Employ a set of methods, principles, or rules for assessing risk, based on the use of numbers (e.g., 1, 2, 3).
Semi-Quantitative	Employ a set of methods, principles, or rules for assessing risk using bins (e.g., 0-15, 16-35, 36-70, 71-85, 86-100) or scales (e.g., 1-10) that translate easily into qualitative terms (e.g., a score of 95 can be interpreted as very high).

While each risk assessment approach has advantages and disadvantages, much of the focus in the pedagogical materials on risk assessment has been on the qualitative and semi-quantitative approaches. This is largely because these scales are much easier to work with. For example, it is easier to qualify the adverse impact of a threat scenario (e.g., denial of service against an online store) as significant or severe than it is to compute a monetary value (e.g., $500,000 in revenue losses.) Other reasons include the complex nature of the computations and the uncertainty surrounding the determination of the values of the likelihood and the adverse impact magnitude. Another reason that hinders the adoption of quantitative assessments in academic pedagogical materials and also in practice is that there is not much in the way of historical data based on which cybersecurity risk analysts can assign probabilities and compute losses.

More recently, there has been a new effort to develop risk assessment methodologies with quantitative and computational orientation. One such methodology is offered in the FAIR[1] (short for Factor Analysis of Information Risk) framework. Figure 1 depicts high-level risk taxonomy[2] abstractions within the framework.

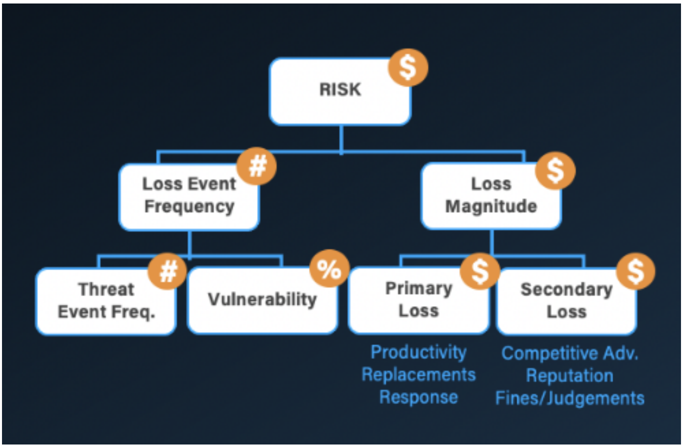


Figure 1: High-Level Risk Taxonomy

Table 2 presents definitions of the high-level factors that derive the risk.

Table 2: Risk Component Definitions

Component	**Definition**
Risk	The probable frequency and probable magnitude of future loss.
Loss Event Frequency	The occurrence, within a given timeframe, that a threat agent will inflict harm upon an asset.
Threat Event Frequency	The occurrence, within a given timeframe, that a threat agent will act against an asset.
Vulnerability	The probability that an asset will be unable to resist the actions of a threat agent.
Probable Loss Magnitude	The likely outcome of a threat event.

[1]https://www.fairinstitute.org/fair-risk-management
[2]Source: https://www.risklens.com/cyber-risk-quantification/the-fair-standard

Table 3 lists and defines the six forms of loss per the FAIR framework[1].

Table 3: Forms of Loss

Loss Form	**Definition**
Productivity	The reduction in an organization's ability to generate its primary value proposition (e.g., income, goods, services, etc.).
Response	Expenses associated with managing a loss event (e.g., internal or external person-hours, logistical expenses, etc.).
Replacement	The intrinsic value of an asset. Typically represented as the capital expense associated with replacing lost/damaged assets (e.g., rebuilding a facility, purchasing a replacement laptop).
Fines and judgments	Legal or regulatory actions levied against an organization. This includes bail for any organization members who are arrested.
Competitive advantage	Losses associated with diminished competitive advantage.
Reputation	Losses associated with an external perception that an organization's leadership is incompetent, criminal, or unethical.

This paper aims to incorporate quantitative assessment approaches that are presented in the FAIR methodology into a Cybersecurity Risk Management course. This goal will be achieved through a scenario-based teaching case study, which utilizes a fictitious company and requires students to assume the role of a cybersecurity risk team to compute the risk exposure that the company faces from a threat scenario against one of its mission-critical information resources.

2 Case Study

2.1 Case Scenario

Furniture Essentials is a fast-growing e-Commerce company that sells furniture and home decor items. Despite its relatively short age, having been in business for only 10 years, the company has experienced significant growth and has quickly become one of its industry leaders. The company employs approximately 1,500 employees and generates approximately $60 million of sales revenue per year.

Most recently, the company saw an exponential growth in its sales. While the increase in revenue was received as welcome news, it also alerted the company to the cybersecurity risk of doing business online.

To manage the information security risk to Furniture Essentials, the Cybersecurity Department, with the blessing and support of the top management team, instituted a formal Cybersecurity Risk Management Program. Consistent with industry standards, the program encompasses the supporting processes to manage information security risk to Furniture Essentials' organizational operation. This includes establishing the context for risk-related activities, assessing risk, responding to risk once determined, and monitoring risk over time.

The company's Cybersecurity Risk Team has just embarked on a new round of assessing its cybersecurity risk exposure. So far, the team completed two major activities; asset identification and threat assessment.

For **asset identification**, the Cybersecurity Risk Team, led by the Chief Information Security Officer (CISO) met with C-Level executives to identify the information assets that are most critical to the company's operations and whose protection from cyber-attacks should receive high priority.

Both the Chief Financial Officer (CFO) and the Chief Operations Officer (COO) were particularly worried about the eCommerce Website which customers use to order the products that Furniture Essentials sells. They were also concerned about the backend internal database that stores customer data and order data. Not surprisingly, these two assets ranked top two mission-critical assets in a subsequent asset scoping workshop that was attended by C-Level executives and department heads.

During the **threat assessment**, the Cybersecurity Risk Team identified several threat actors that could launch cyber attacks against Furniture Essentials. Given that Furniture Essentials has not been targeted by nation-state actors or advanced cybercriminals, the team elected to mainly focus on external malicious hackers. Subsequently, the team identified several threat actions that these hackers can potentially carry out against Furniture Essentials.

After careful analysis of the threat scenarios, the Cybersecurity Risk Team found DDoS to be the top attack vector by which malicious hackers could compromise the availability of the eCommerce Website. Additionally, they found phishing as the top attack vector by which malicious hackers could gain unauthorized access to Furniture Essentials' systems and to breach the confidentiality of customer data in the internal database.

Having identified the top threat scenarios, the team is now ready to conduct a risk analysis in order to determine the probability of occurrence and the impact for each scenario. However, a key determination that the team must make is deciding on the assessment approach to use going forward.

Over the 10 years that Furniture Essentials has been in business and up until now, the Cybersecurity Risk Team used both qualitative and semi-quantitative approaches to determine the risk exposure. This time around, however, the CISO who was under continuous pressure from the CEO and CFO to justify the cybersecurity budget, asked the Cybersecurity Risk Team to adopt a quantitative methodology for risk assessment; one that would allow for computing the risk exposure in terms of dollar amounts. In fact, the CISO had just returned from a Cybersecurity Conference in San Francisco where he attended several presentations on cybersecurity risk management and assessment including a presentation on the FAIR framework. The CISO shared soft copies of a Technical Standard for Risk Taxonomy per the FAIR Framework [1] and asked the team to perform the risk analysis per the methodology presented in the framework.

For the time being, however, and given that the team had no previous experience working with the FAIR Framework, a collective decision was made to limit the FAIR-based risk analysis to the DDoS attack scenario.

2.2 Data Collection

The Cybersecurity Risk Team had several communications (phone calls, meetings, emails, document requests, etc.) with various teams across the company. Table 4 presents the information that the team obtained. While the team obtained far more information, the table presents a summary of the data that was deemed relevant to deriving the risk exposure.

2.3 Risk Analysis

Armed with the data it obtained from various teams across the company, the Cybersecurity Risk Team set out to determine the likelihood that malicious hackers would launch a DDoS attack against the eCommerce website and estimate the loss magnitude associated with a successful attack.

Table 5 and Table 6 summarize the primary loss and the secondary loss as computed[3] from their relevant forms.

Based on data from the Incident Response Team and the Network Security Team, the Cybersecurity Risk Team estimated the Threat Event Frequency to be between 15 and 30 a year, with the most likely value of 24 events per year. Further, the team estimated the percentage of the threat events that will become loss events (hence, the vulnerability) to be between 1% and 7%, with the most likely value of 4%. The team then derived the Loss Event Frequency and computed the Total Loss Exposure as presented in Table 7 below.

2.4 Discussion

Based on the risk analysis, Furniture Essentials is poised to lose between $28,000 and $3,731,000, with the most likely value of $901,000. The total

[3]For space limitation and to maintain the sanctity of the case, the detailed computations were omitted. Contact the author for detailed computations of the all risk components.

Table 4: Data Collection

Source	**Data Summary**
Incident Response	Over the past 5 years, the eCommerce website was hit by 15-30 DDoS attacks per year. In the event of a sophisticated DDoS attack, an incident response team of 6-10 members will be assembled and deployed. Depending on the scope of the attack, the team is expected to work overtime for 10-40 hours. The average loaded hourly wage is $100 per hour. In the event of a successful DDoS attack, a cybersecurity company would be contracted to assist with the response and investigation for an average cost of $150,000.
Network Security	The eCommerce website is remotely scanned 5 to 10 times a day. On average, one malicious activity per quarter is detected on the network. A DDoS mitigation solution is in place. On average, the solution records 24 DDoS attempts per year. While no successful DDoS attack has ever occurred against the eCommerce website, it is estimated that 1 in 25 DDoS attacks will overcome the protection by the DDoS mitigation solution.
Sales Management	The company's eCommerce website generates approximately $60 million of revenue per year from a customer base of 50,000 active customers. The company estimates the customer lifetime value at $300 per customer. 300 employees use the order fulfillment solution for the Website. The average loaded hourly wage is $80 per employee.
Marketing & Public Relations	In the event of an outage, it is estimated that only 5% of the customers would stop purchasing products from Furniture Essentials and switch to the competition.
Business Continuity	The eCommerce website has gone down only once in the past 5 years. This incident was not the result of a DDoS attack but due to an internal software development error. The Website was restored in 45 minutes. The recovery time objective (RTO) for eCommerce website is 2 hours.
Customer Service	In the event of a downtime between 30 minutes and 3 hours, it is estimated that the Customer Service would experience an increase of 100-500 calls with an average cost of $2.50 per call.
Regulatory Compliance	Industry data indicates that regulators have rarely sued organizations for outages caused by an external threat.

loss exposure is significantly affected by the secondary loss, mainly from reputational damage. This underscores the importance of running an effective public relations campaign in the event of a DDoS.

Table 5: Primary Loss

Loss Type	Minimum	Most Likely	Maximum
Primary Productivity	$6,000	$18,000	$36,000
Primary Response	$106,000	$170,000	$240,000
Total Primary Loss	**$112,000**	**$188,000**	**$276,000**

Table 6: Secondary Loss

Loss Type	Minimum	Most Likely	Maximum
Secondary Response	$125	$375	$625
Secondary Reputation	$75,000	$750,000	$1,500,000
Total Secondary Loss	**$75,125**	**$750,375**	**$1,500,625**

Table 7: Risk Analysis Results

Risk Factor	Minimum	Most Likely	Maximum
Primary Loss Event/Year	0.15	0.96	2.1
Primary Loss Magnitude	$112,000	$188,000	$276,000
Secondary Loss Event/Year	0.15	0.96	2.1
Secondary Loss Magnitude	$75,125	$750,375	$1,500,625
Total Loss Exposure	**$28,069**	**$900,840**	**$3,730,913**

Considering this analysis, Furniture Essentials needs to establish risk tolerance levels. For example, is the company willing to accept a 99% chance of losing $30,000 or more in a given year? Is the company willing to accept a 5% chance of losing $500,000 or more in a given year? and so on. Depending on the company's risk appetite, the company may accept the risk, implement countermeasures to reduce the exposure to an acceptable limit, or transfer the risk by purchasing a cyber insurance.

2.5 Next Steps

Impressed with the level of details and rigor, the CISO asked the Cybersecurity Risk Team to perform a similar analysis for the threat scenario involving a phishing attack against the confidentiality of the customer data in the internal database. This time, however, the team hopes to re-use most of the data it had obtained and the computations it had performed.

3 Conclusion

This paper introduced students in a Cybersecurity Risk Management course to advanced quantitative risk assessment approaches. The FAIR framework was chosen given its heavy quantitative orientation and its growing attraction within the cybersecurity risk management community. The case study approach and the meticulous analysis involved should enhance the students overall learning of risk assessment and contribute to their marketability as future cybersecurity professionals.

References

[1] The Open GROUP. Technical standard risk taxonomy. `https://pubs.opengroup.org/onlinepubs/9699919899/toc.pdf`.

[2] Office of Management and Budget (OMB). Circular A-130, managing information as a strategic resource. `https://csrc.nist.gov/Topics/Laws-and-Regulations/executive-documents/OMB-A-130`.

[3] National Institute of Standards and Technology. NIST SP 800-30 rev. 1, guide for conducting risk assessments. `https://csrc.nist.gov/publications/detail/sp/800-30/rev-1/final`.

[4] National Institute of Standards and Technology. NIST SP 800-39, managing information security risk: Organization, mission, and information system view. `https://csrc.nist.gov/publications/detail/sp/800-39/final`.

The Structure and Delivery of an Advanced Systems Administration IT Course*

Mohamed Lotfy and Christian Fredrickson
Utah Valley University
Orem, Utah 84058
{MohamedL, fredrich}@uvu.edu

Abstract

IT program courses should allow students to acquire the needed IT skills to enable them to be job ready by graduation. To prepare the IT graduates to current and future practices of virtualized computing resources, which integrate IT systems and services, DevOps and microservices, systems administration courses need to introduce students to current tools used to administer and manage continuous integration and deployment of infrastructure and services. In this paper, we provide the structure, components, hands-on assignments, and the virtual environment of a senior-level competency-based advanced Unix/Linux systems administration course that is delivered face-to-face and online. The course introduces current organizational practices of continuous integration and delivery of services. Student course evaluation and what helped them learn the most are also presented and discussed.

1 Introduction

Courses in IT programs should allow students to acquire the needed IT skills to enable them to be job ready by graduation. According to IT2017 and CC2020 curricula guidelines, information technology (IT) education should move from knowledge-based to competency-based education. Competency-based education connect the knowledge (know-what), skills (know-how), and dispositions (know-why) dimensions regarding a specific context to accomplish a task [5, 8]. IT degree programs need to be organized around the knowledge, skills, and dispositions dimensions to enable student's career readiness. IT2017 mentioned that IT programs should adopt a curriculum that integrates learning of professional practice through the courses. In order for IT programs graduates to be job ready, the educational course work should mirror the computing technologies in the work environment[5]. Since IT degree programs should cover the Systems Paradigms knowledge area (an essential IT domain area) as well as Virtual Systems and Services (domain platforms), systems administration courses need to expose students to current methodologies including DevOps, microservices and continuous integration and deployment.

Cummings and Janicki, 2020, conducted a study to identify the demanded IT/IS technological knowledge, concepts, and skills from IT/IS college graduates. The study goal was to help universities prepare students by providing and adjusting the curriculum and course offerings that develop the appropriate set of IT/IS skills required by market. In the study, 2,500 U.S. IT professionals were surveyed to identify the current and future organizational technologies and their importance to the IT/IS field, the skills needed by IT/IS role, and IS/IT employment growth. The survey results showed that database and systems administration, tied as number one, were the most needed skills in incoming IT employees. Also, cloud and virtualization skills were ranked as the eighth most needed skill in incoming IT employees [3].

Redondo (2018) argued that in order to provide the students a more realistic feel about administering a real server, it requires a pre-existing infrastructure that cannot be replicated by students and must be always available[10]. According to Pike and Brown (2019), the availability of virtualized computing resources which integrate IT systems and services as well as the virtualization of networks and network services, require IT/IS professionals to have substantial systems administration skills to manage and optimize IT infrastructure components[9]. Bonders and Slihte (2018) provided the structure of an information systems management course. In addition to regular administration topics, they included secure network administration (DNS and VPNs), network file systems and storage administration (Samba and NFS), and cloud solution administration where students set up virtual machine, configure remote access and firewall, create project, resource groups and plans[1]. Hassan (2020) outlined similar systems administration skills but made students use practical assignments giving them the opportunity to practice real network and systems administration scenarios. The results of the end of course questionnaire in [6] showed that 80% of the students agreed that the course assignments and assessments mirrored real network and systems administration scenarios. In addition 88% of the students see the assignments and assessments as authentic that replicate real-world challenges [6].

In the last decade many organizations adopted the DevOps practice. In DevOps, organizational multidisciplinary teams collaborate to continuously deliver, integrate, and deploy reliable and correct organizational applications[2, 4, 7, 13]. Organizational applications are developed around the business processes using containerized microservices. Dividing the application into modules, each module is implemented and deployed as a microservice. Using containers, which are lightweight virtual machines that are faster to boot and use less memory, enable the deployment of many containerized microservices on the host operating system(OS)[12]. Each container encapsulates the data, schema, and objects of the microservice thus enabling secure computing. Using containers enable portability and scalability of delivery and deployment [7, 11, 13]. To enable continuous integration, delivery and deployment of microservices, Git, yum, Kubernetes, Jenkins, and Docker are used [7, 11].

In order to prepare the IT graduates for current and future practices of virtualized computing resources that integrate IT systems and services as well as the virtualization of networks and network services, DevOps and microservices, systems administration courses need to introduce the different tools used for continuous integration and deployment like Git, containers, etc.. Current systems administration tools like yum, rpm, dpkg, Git, Kubernetes, and Docker should be introduced and applied in the different hands-on activities thus allowing students to gain the needed knowledge of current best practices in systems deployment, virtualization and services. Connecting the systems and data networks knowledge with current best practices, like microservices and containers within a specific deployment context, enables students to develop and gain current professional systems administration competency.

At Utah Valley University (UVU), the IT3510 Advanced Systems Administration -Linux/Unix competency-based course has been taught face-to-face and online in a 16-week semester format. The course explores enterprise systems administration using a CentOS operating system. Students learn advanced administrative tasks including server installation, network configuration, file management, network services deployment, server security, shell scripting, source compilation, performance monitoring and tuning. In the course, students apply advanced systems administration concepts using hands-on real-world techniques with current tools, virtualized computing resources, virtualization of networks and network services as well as microservices.

In the remaining sections, the structure of the Advanced Systems Administration -Linux/Unix course as well as the course assignments, the used tools, and the virtual environment infrastructure are presented. Student evaluation and feedback of the course are shared followed by a discussion on what helped the students learn the most are also presented and discussed.

2 Course Structure

The IT3510 Advanced Systems Administration -Linux/Unix competency-based course explores enterprise systems administration using the CentOS operating system. Students learn advanced administrative concepts and tasks through an applied viewpoint using a hands-on application of real-world techniques and the use of current tools. The following are the course learning outcomes.

Upon successful completion of the course, students should be able to:

1. Interpret and edit scripts for Linux service installation.
2. Identify and implement Linux daemons.
3. Manage a web server daemon in a container.
4. Learn BIND, Postfix and other daemons.
5. Setup a central authentication system while learning about common practices.
6. Manage several daemons via a Docker container and create a Docker swarm.

Students attending the course are seniors in the IT program as well as students in the CS program who are using the course as an elective. Students should have completed and passed the Computer Programming I for IS IT, Data Communication Fundamentals, and the Introduction to Systems Administration–Linux/UNIX courses or equivalent. The course activities include hands-on assignments, readings, a final paper/project, and a two-part final exam. Each week students are required to find an article that includes a topic covered in the current week's hands-on assignment then submit the URL to the article, a summary of the article, and what the student learned from reading the article. The reading report is submitted in the Canvas course shell. Students are required to complete a two-part final exam in the Canvas course shell during finals week. The first part is a comprehensive exam covering the assigned readings and labs. The second part is a hands-on activity to demonstrate the acquired systems administration skills.

2.1 Hands-on Assignments

The hands-on assignments are constructed to allow the student to use and apply the knowledge from the class presentations, readings, recorded videos, online resources, etc., to develop the different current advanced systems administration skills within different contexts. Students use current systems administration tools to administer their own CentOS 8 VM. Deployment and management of containers using Docker are introduced and practiced throughout the second half of the course. Table 1 shows the structure and software/tools used in the course.

Table 1: Course Hands-on Assignments

	Assignment	Software/Tools used
1	Shell scripting	bash filtering functions
2	Storage Administration	parted and mkfs
3	Installing Packages and Package Management	rpm, dpkg, yum, apt git, systemctl, and service
4	Networking	netstat, lshw, route firewall-cmd, and gpg
5	Network Sharing	nfs4-acl-tools, nfs-utils, exportfs, samba, and cifs-utils
6	Bind Daemon and Named Service	podman, dnf, and docker
7	Mail Services	docker and postfix
8	Web Services	nginx, docker swarm, and docker stack
9	Central Authentication Services	FreeIPA and ipa-client
10	Time/Date Services	chronyc, hostnamectl, and timedatectl
11	Tuning and Update	sysctl, tuned, dnf and grubby

The course assignments introduce current organizational systems administration practices. For example, in the application installation assignment students learn how to use Git repositories and the git command to implement application acquisition and deployment. Containers implementation and deployment using Docker are introduced in week 7 of the course. To allow students to experience current DevOps and microservices practices, containers are used to deploy an authoritative BIND DNS server, a Postfix mail server, and a swarm of nginx web servers. The concept of containerized virtual networks is also introduced and practiced. The practices of secure systems administration are explored and applied in the networking assignments, installation and management of authentication services using FreeIPA, and systems tuning and updating.

Each assignment, provided in the Canvas assignment module/page, is very detailed with descriptions and instructions to enable the student to perform the different tasks. Each assignment module/page includes four areas, the purpose and goals of the assignment, the needed tools, the tasks that should be performed, and the expected deliverables. The deliverables are written documents or technical reports showing screenshots of accomplished tasks and all used commands, with a reflection on the lessons learned and issues encountered while performing the assignment.

2.2 Individual Project or Paper

For students to demonstrate the acquirement of the advanced systems administration skills, each student must submit a project or paper. The project allows students to challenge their hands-on skills. It enabled them to acquire the needed resources, or it can be part of their professional work. The paper can include hands-on work or a guide for performing work. Some of the students chose projects like developing a magic mirror, gaming console, Linux based drones, Plex server, automated home systems using Linux, smart home systems, and central authentication/authorization system. The papers ranged from comparing different Linux distros, comparing Linux to other OSs, developing a user guide for installing a Linux distro or an application, and any other Linux topic they want to investigate and learn more about. Rubrics were provided describing the grading criteria for each component.

2.3 Course Delivery

Both the face-to-face and online course shells are delivered using the Canvas learning management systems. Each course Canvas shell includes video recordings, presentations, demonstrations, hands-on learning activities, and assignments. The course Canvas shell includes the course information, course syllabus, grade book, calendar, course materials/modules, reading assignments, hands-on labs, exams, and a final project/paper. To keep students on track, the course calendar is populated with all the assignments and their due dates. Recorded video lectures showing why and how to use the different tools to conduct the systems administration tasks are posted in the course media folder.

A Q&A discussion forum is available for the students in the course shell to ask questions and clarifications about assignments, course materials, and/or assessments. Students are encouraged to use the course Microsoft Teams channel and the weekly discussion Q&A forum to answer each other's questions and provide help as long as it is not exam related. To enable students to acquire the needed competency and achieve the course outcomes, the faculty provides detailed feedback on each graded assignment. The feedback explains what the student did well, what they missed, how they used the tools to meet the assignment/lab requirements, and any additional resources or tools that should have been used.

2.4 The Virtual environment

To conduct the hands-on assignments, each student is provided a VM running CentOS 8 and has administrator privileges. The VM can be accessed off campus using PuTTY or SSH after authenticating the student and using a VPN client to access the academic network. The faculty is provided four VMs, a CentOS 8

VM like the students VM, a CentOS 7 Samba/NFS server with Samba services installed, a CentOS 7 NTP server with ntpd service installed, and a CentOS 7 IPA server with FreeIPA service installed. In both face-to-face and online delivery of the course, students must install the Cisco AnyConnect VPN client and PuTTY on their laptops or PCs to access their own VM and complete the required assignments/labs.

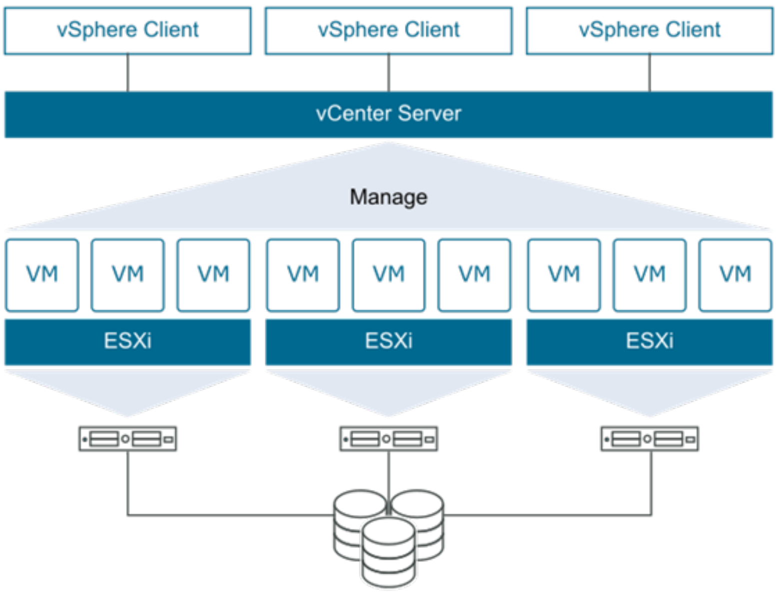


Figure 1: UVU Virtual Machine Environment.

All class VMs, faculty and students, are on a specific class C virtual LAN and provisioned on UVU's College of Engineering and Technology (CET) VM environment. The VM environment that CET utilizes runs across multiple ESXI server clusters, a type 1 hypervisor, which have the VMware ESXI server OS installed on them. The physical server environment is maintained by the University System Operations team in UVU's data center. The CET IT administrators use vSphere to interact with the servers and create virtual machines that students can use for their classes. When a Windows VM is created, the student can access it via RDP. If it is a Linux VM, the student can access it via SSH. Typically, each class has around 30 students, so scripts were developed that will create the virtual machines based off templates. The VMs are created and assigned to a specific student according to Microsoft Active Directory. This saves the CET IT administrators lots of time, so they don't have to create each VM manually. Figure 1 shows the Utah Valley University virtual machine environment.

3 Student Course Evaluation and Feedback

At the end of each course, students are provided an online course evaluation form. The overall course evaluation area used a five-point Likert scale to answer the following questions:

Q1: I learned more about the subject as a result of taking this class.
Q2: I learned how this subject can be used to address issues outside of the classroom.
Q3: This class challenged me to think in new ways.
Q4: I developed one or more essential skills as a result of this class.

Table 2 shows the results for the IT3510 Fall 2021 and Spring 2022 overall course evaluation.

Table 2: Overall Course Evaluation Results

	SA(%)	A(%)	N(%)	D(%)	SD(%)	Avg	StdDev	95% CI
Fall 2021 (N= 28)								
Q1	89	7	4	-	-	4.86	0.44	(4.69, 5.02)
Q2	86	11	4	-	-	4.82	0.47	(4.65, 4.99)
Q3	86	11	4	-	-	4.82	0.47	(4.65, 4.99)
Q4	68	29	4	-	-	4.64	0.55	(4.44, 4.85)
Spring 2022 (N=24)								
Q1	79	17	4	-	-	4.75	0.52	(4.54, 4.96)
Q2	83	13	4	-	-	4.79	0.50	(4.59, 4.99)
Q3	75	21	4	-	-	4.71	0.54	(4.49, 4.92)
Q4	71	21	8	-	-	4.63	0.63	(4.37, 4.88)

Students could also answer the following open-ended question "what helped you learn the most". The following were the written responses provided by the students who opted to answer the open-ended question. Each bullet represents a students' response.

Fall 2021

- "Applying our assignments and in class labs to the real world"
- "Very real-world focus and application-oriented."
- "I learned the most by the project that I worked on throughout the semester which I also really enjoyed doing. The hands on assignments really helped me learn in this class also."
- "I really liked the way we were taught. Learning the content through different assignments and seeing how the commands were applied was really nice."
- "The hands-on nature of this course, and creating the project"
- "The lecture videos"

- "The video recorded lectures helped a lot."
- "The walkthrough videos were incredible. They were thorough and explained in detail what needed to be done."
- "When the professor walked us step by step through how to do things and explained thoroughly why we are doing it the way we are, problems we may encounter, etc."
- "Working through assignments in class in person and the ability to ask questions and work through issues together."
- "Working together in class"

Spring 2022

- "Being able to do the assignments with your help as you explained why we did each part."
- "Class labs (writeups and lecture videos)."
- "Hands on experience using VM's."
- "I thought the class content was pretty good."
- "Lab assignments."
- "The assignments."
- "The assignments we had to in Linux"
- "The hand on small projects every week helped me a lot to learn the concepts we were learning."
- "The labs helped me to learn."
- "The walkthrough for the assignments helped me a lot. Also doing my own reading to look more into the topic helped me because it forced me to learn more about the topic. Also the final project was nice because I got to apply the knowledge I learned in class to the project I chose."
- "Clear hands-on exercises."
- "Doing the hands on labs to apply what the lesson was on."

The fall 2021 and the spring 2022 course sections used the same course content, structure, hands-on assignments, virtual environment, and paper/project requirements and rubrics. Different faculty taught and managed the fall 2021 and spring 2022 courses.

4 Discussion

To prepare the IT graduates to current and future practices of virtualized computing resources which integrate IT systems and services, DevOps and microservices, systems administration courses need to introduce students to current tools used to administer and manage continuous integration and deployment of infrastructure and services. The submitted graded student work showed that the students applied the learned knowledge and gained the needed skills to perform current advanced systems administration tasks. Also, student graded work showed that the students understood how each tool was used and why the tool usage fit the systems administration task within the provided scenario/context, therefore allowing them to acquire the needed IT skills and enabling them to be job ready by graduation.

Student response to the course evaluation questions, presented in Table 2, showed that most students in the fall 2021 and spring 2022 courses developed one or more essential advanced systems administration skills and learned more about advanced systems administration because of taking the course. Also, most of the students learned how systems administration can be used to address outside the class issues. Lastly, most students agreed that the course challenged them to think in new ways. Figure 2 provides a summary of percent of agreement to the course evaluation questions.

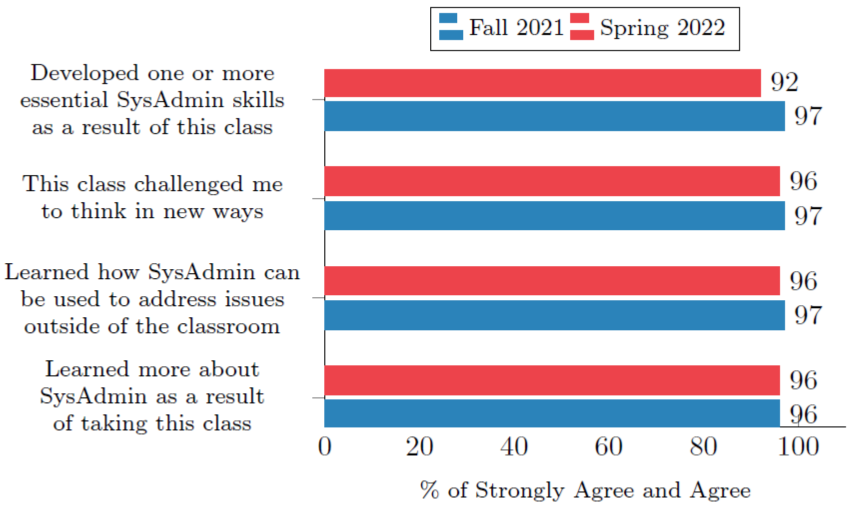


Figure 2: Agreement to the Course Evaluation Questions.

All students managed to access their VMs from their laptops and PCs and complete all the assignments/labs. A very small number of students corrupted their VMs causing the VMs to be recreated by the CET IT administrators. Student written comments answering the open-ended question, shared in section 3 above, showed that the hands-on experience in their own virtual environment enabled them to learn the different advanced systems administration skills. Some students mentioned that the weekly provided walk-through videos helped them to learn the most. Most students highlighted the value of the hands-on assignments on their learning. In addition, some students found the practicality of the course content was another element that allowed them to learn the most. Step-by-step classroom instructions coupled with recorded walk-though videos facilitated learning and acquiring the different advanced systems administration skills.

5 Conclusions

Advanced systems administration courses need to be an integral component in current IT education. The courses should provide advanced concepts through an applied viewpoint using a hands-on application of real-world practices of continuous integration and delivery techniques (Git, containers, Docker, etc.) using current systems administration tools that mimic real organizational infrastructures. Allowing students to practice current systems administration scenarios enables them to develop advanced systems administration skills and IT job readiness. Analysis of student overall course evaluation and qualitative responses showed that having a personal provisioned VM, hands-on experience with current systems administration tools facilitated with walk-through videos enabled the students to acquire the different advanced systems administration skills.

References

[1] Martins Bonders and Jans Slihte. A practical approach to teaching information technology infrastructure management. In *2018 59th International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)*, pages 1–5. IEEE, 2018.

[2] Matt Callanan and Alexandra Spillane. DevOps: making it easy to do the right thing. *Ieee Software*, 33(3):53–59, 2016.

[3] Jeff Cummings and Thomas N Janicki. What skills do students need? a multi-year study of IT/IS knowledge and skills in demand by employers. *Journal of Information Systems Education*, 31(3):208, 2020.

[4] Christof Ebert, Gorka Gallardo, Josune Hernantes, and Nicolas Serrano. DevOps. *Ieee Software*, 33(3):94–100, 2016.

[5] CC2020 Task Force. *Computing Curricula 2020: Paradigms for Global Computing Education.* Association for Computing Machinery, New York, NY, USA, 2020.

[6] Ismail Hassan. Automated authentic assessment: Applied to an undergraduate course in network and system administration. In *2020 IEEE Global Engineering Education Conference (EDUCON)*, pages 183–191. IEEE, 2020.

[7] Leonardo Leite, Carla Rocha, Fabio Kon, Dejan Milojicic, and Paulo Meirelles. A survey of DevOps concepts and challenges. *ACM Computing Surveys (CSUR)*, 52(6):1–35, 2019.

[8] Task Group on Information Technology Curricula. *Information Technology Curricula 2017: Curriculum Guidelines for Baccalaureate Degree Programs in Information Technology.* Association for Computing Machinery, New York, NY, USA, 2017.

[9] Ron Pike and Brandon Ray Brown. IT infrastructure strategy in an undergraduate course. *Information Systems Education Journal*, 17(5):17, 2019.

[10] Jose Manuel Redondo. Improving student assessment of a server administration course promoting flexibility and competitiveness. *IEEE Transactions on Education*, 62(1):19–26, 2018.

[11] Dawei Yang, Daojiang Wang, Dongming Yang, Qiwen Dong, Ye Wang, Huan Zhou, and Daocheng Hong. DevOps in practice for education management information system at ECNU. *Procedia Computer Science*, 176:1382–1391, 2020.

[12] Hong Zhu and Ian Bayley. If docker is the answer, what is the question? In *2018 IEEE Symposium on Service-Oriented System Engineering (SOSE)*, pages 152–163. IEEE, 2018.

[13] Liming Zhu, Len Bass, and George Champlin-Scharff. DevOps and its practices. *IEEE Software*, 33(3):32–34, 2016.

Designing a Real-World Deep Learning Project for Undergraduate Students*

Jingsai Liang and Wynter Mindnich
Department of Computer Science
Westminster College
Salt Lake City, UT 84105
JLiang@westminstercollege.edu
wk.mindnich@gmail.com

Abstract

While there are plenty of deep learning projects that we can adapt for senior undergraduate students, students will benefit more from working on a project that can have an impact in the real world. In this paper, we present a designing process of a real-world deep learning project cooperated with Great Salt Lake Institute (GSLI) and Utah Division of Wildlife Resources. The first goal of the project is to count the number of pelicans in photos automatically captured by stationary cameras located on Gunnison Island in Great Salt Lake, as part of efforts to study and observe the population, migration, breeding, and survival rate of pelicans on the island. The project was designed in a cycle mode with five steps: image annotation, image augmentation, model training, hyperparameters tuning, and model testing. The project has been delivered in various types of courses in the last three years. In a regular deep learning course, students built models in the last weeks or month and evaluated their models on a testing dataset at the end of the semester. In a capstone course or summer research, students followed a similar framework with more flexibility on a larger dataset. Students typically worked on the project in groups on Colab.

1 Introduction

Deep learning has undeniable success in numerous fields [10, 7] and surpasses classical machine learning on many applications such as computer vision [12], image processing [5], and natural language processing [8], thus teaching deep learning in machine learning, artificial intelligence, or a standalone deep learning course receives high demand from our undergraduate students. A hands-on project that wraps up basic deep learning lectures students have learned is an essential component of the deep learning courses. We can tailor projects from open sources or textbooks based on course contents and the backgrounds of students [4, 2, 6].

However, for these kinds of projects, students will simply replicate them and usually fail to compete with the state-of-art performance of them due to the constraints of computational resources, funding in our department, and available time students have in one semester. If we could design a real-world project cooperated with other institutions, students would work more actively since they could make a contribution in real world using the knowledge and skills they just learned in class. Students work under supervision from both course instructors and mentors from outside and do still need competition on the performance of their models but with their classmates only. They also would have the opportunity to interact with or present their projects to mentors from outside.

The task of the project was crafted to meet the learning goals of the course and fulfill requirements from outside mentors. The project was released in several steps to lead students to complete it on time in groups. Though some groups may not achieve good results at the end, all groups complete the whole process of the project.

2 Background

The institutions we cooperated with are Great Salt Lake Institute (GSLI) and Utah Division of Wildlife Resources on the Great Salt Lake PELI Project (Partnership for Education and Longitudinal Investigation of American White Pelicans), which studies and observes the population, migration, breeding, and the survival rate of pelicans on Gunnison Island, Great Salt Lake.

Stationary cameras were installed on the island in early March 2017 to take pictures of American White Pelicans on their breeding grounds every three minutes. An example of such images is shown in Figure 1. Over time, the cameras yield a massive collection of images with not only pelicans, but also coyotes and other animals. The first phase of processing the massive images kept being produced is to automatically count the number of pelicans on images, so that we can estimate the population and migration patterns of pelicans based on the size of colonies and the time/date of the images.

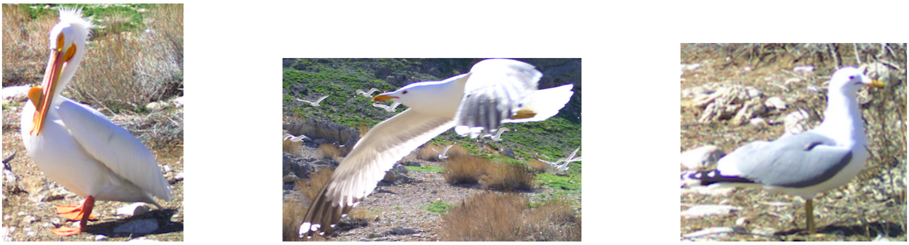
Figure 1: An example of pelican images.

Given the rising successes of applying deep learning in the object detection field, we decided to analyze the pelican images using deep learning models years ago. We have delivered this project on various occasions including a capstone course in spring 2020, a summer research in 2021 summer, and a regular course in 2022 summer.

Students used free GPU of Colab to develop their models at first and upgraded the Colab to pro version if they needed faster GPU, more memory, or fewer disconnection times. Still, due to the constraints of the computational resources, they would only build models on a subset of the whole image dataset. They finally produced a demo product that can be scaled to handle larger dataset.

3 Project

The project was designed in five steps: image annotation, image augmentation, model training, hyperparameters tuning, and model testing. In step 1, students manually annotated training and validating images using self-defined labels. In step 2, labeled images were augmented with various transformations to obtain a larger training set without more labeling work. In step 3, students called different choices of models to train the data and were able to see initial results. In step 4, students tuned hyperparameters to get better results on the validation sets. If the results were still not satisfying, students could go back



(a) beak (b) flying (c) no beak

Figure 2: Annotation of pelican images.

to step 1 to repeat the process. Finally, in step 5, students tested their models on a testing dataset and reported their results.

3.1 Image Annotation

The first step is a very time-consuming but important step that can significantly affect the performance of the model. Basically, students were required to manually annotate pelicans on images one by one based on the design of the models. There are many tools as well as formats students could choose to annotate the images. The default choices we recommended are labelImg and YOLO format [11]. The name of labels could be either of three choices below based on the specific task students would like to train the model:

- binary labels: present or not present of pelicans
- number of pelicans: no pelican, moderate number of pelicans, large amount of pelicans
- posture of pelicans: flying, beak, no beak

The simplest model is using binary annotation, but it could not tell the number of pelicans at all. Practically, labeling based on the posture of pelicans is better than labeling based on the number of pelicans. Here, we labeled the pelicans into three categories: pelicans with long yellow beaks (Figure 2a), pelicans flying in the air (Figure 2b), and pelicans sitting on the ground without significant beaks (Figure 2c).

3.2 Image Augmentation

Students typically labeled 400 images as training set and 100 images as validation set in step 1. Image augmentation provides a way that students can

(a) Horizontal Flipping

(b) Random Brightness Contrast

(c) Hue Saturation Value

(d) Random Rotation

Figure 3: Augmentation of pelican images.

quickly generate more images from existing images by artificially altering the pixel-level or spatial-level information, such as brightness, shifting, rotating, clipping, and adding noises. In Figure 3, we listed four transformations of Figure 1: horizontal flipping (a), random brightness and contrast (b), random change of hue, saturation, and value (c), and random rotation (d). The package we recommended students use in this step is albumentations [3].

3.3 Model Training

Numerous deep learning models were invented in the field of object detection in the last decade. Two models among them, Faster R-CNN [9] and YOLO [1], were primarily deployed in our project. Students could build the model from scratch using Keras and Tensorflow or build the model simply by calling the YOLO package under Pytorch. One hurdle in this step is that students needed to wait for half of an hour or even longer to finish the training of their models on Colab.

Students could use Tensorboard to track and compare a sequence of models in terms of mAP, precision, and recall. In Figure 4, a curve shows the relationship between the number of epochs on the horizontal axis and mAP@0.5 on the vertical axis. Based on this figure, students should reduce the number of epochs to around 20 in the next training since mAP stays on the highest value without improvement after 15 epochs.

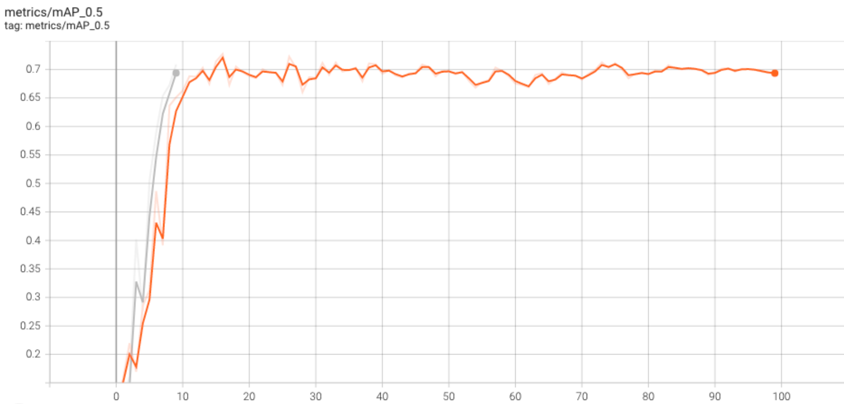


Figure 4: A curve between epochs and mAP@0.5.

3.4 Hyperparameters Tuning

There are many hyperparameters students can adjust to achieve higher performance on the validation set. Typically, students may try changing the following hyperparameters:

- image size to be resized during the training.
- batch size from 10 to 32.
- number of epochs from 10 to 100.
- different pretrained models of YOLO from yolov5s to yolov5l.
- learning rate, number of hidden layers, etc

3.5 Model Testing

The final step is to test the model on a testing set that was never used in the previous four steps. A typical result is shown in Table 1, where we applied YOLO model with pretrained weight yolov5m, image resized as 960, batch size as 20, and epochs as 20. It is not a surprise that this model performs best in the beak category since the long, wide, and yellow beak significantly contrasted with the background should be the easiest feature to learn.

Students typically only compared the overall mAP@.5 across various type of models with different hyperparameters, but they were also required to understand other metrics like precision and recall.

Table 1: An example of testing results.

Class	Images	Labels	Precision	Recall	mAP@.5	mAP@.5:.95
all	122	950	0.788	0.793	0.825	0.539
flying	122	68	0.733	0.691	0.725	0.419
no_beak	122	633	0.751	0.807	0.828	0.503
beak	122	249	0.88	0.88	0.922	0.695

4 Results

Figure 5 shows an example of detecting pelicans on a testing image, where each detected pelican is bounded in a box with a label and confidence score. It is hard for the model to count every pelican in the images since pelicans could be appeared very tiny in the background, only a part of pelicans were captured, or the light is too strong or too weak.

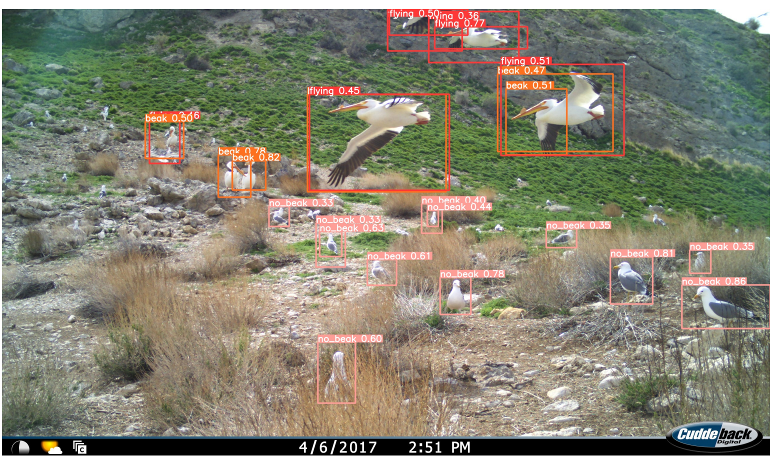


Figure 5: An example of prediction detection results.

Table 2 shows some examples of the testing results using models trained by students. Not all students completed their projects with expected results, typically because they did not accurately annotate enough pelicans in training images. The best overall result among them, 0.976 of mAP@.5, was achieved by a group who trained YOLO model on a large dataset with flying, beak, and no_beak as labels. It is safe to say that this model could be successfully used to automatically count the number of pelicans captured by the cameras on the island.

Table 2: Results of Students' Models.

Group	Annotation	mAP@.5	Accuracy
1	binary		90%
1	0, 1-39, 40+ pelicans		70%
2	flying, beak, no_beak	0.976	
3	flying, beak, no_beak	0.278	
4	fly, no_fly, beak	0.516	
5	flying, ground, beak	0.324	
6	flying, no_beak, beak	0.825	

5 Conclusion

Students who completed this project gained hands-on experience in building a deep learning model for detecting pelicans in images. Several groups presented their projects to the mentors from outside and received valuable feedback. Students benefited more from working on this real-world project than working on a project that we adapted from textbooks.

We will spend more time focusing on tuning hyperparameters since training deep learning network is very time-consuming and tuning hyperparameters wisely can help students save hours of running time. We will also recommend students to build models from scratch so that they will have more power to manipulate their models and get a better understanding of their models.

There are many future work students can do beyond counting the number of pelicans. For example, students failed to build a good migration pattern between time/date and the size of colonies, never detected abnormal object like human intrusion on the island, etc. We will continue exploring the pelican datasets in our future courses.

6 Acknowledgement

We would like to acknowledge support for this paper from Westminster College Great Salt Lake Institute (GSLI) and the Utah Division of Wildlife Resources.

References

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. YOLOv4: Optimal speed and accuracy of object detection, 2020.

[2] Jason Brownlee. *Deep learning for computer vision: image classification, object detection, and face recognition in Python*. Machine Learning Mastery, 2019.

[3] Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin. Albumentations: Fast and flexible image augmentations. *Information*, 11(2), 2020.

[4] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, Inc., 2019.

[5] D Jude Hemanth and V Vieira Estrela. *Deep learning for image processing applications*, volume 31. IOS Press, 2017.

[6] Anirudh Koul, Siddha Ganju, and Meher Kasam. *Practical Deep Learning for Cloud, Mobile, and Edge: Real-World AI & Computer-Vision Projects Using Python, Keras & TensorFlow*. O'Reilly Media, 2019.

[7] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[8] Marc Moreno Lopez and Jugal Kalita. Deep learning applied to NLP. *arXiv preprint arXiv:1703.03091*, 2017.

[9] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[10] Iqbal H Sarker. Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science*, 2(6):1–20, 2021.

[11] Tzutalin. LabelImg. `https://github.com/tzutalin/labelImg`.

[12] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018.

Integrating Diversity, Equity, and Inclusivity Curriculum into Introductory Computer Science Courses*

Jody Paul
Department of Computer Sciences
Metropolitan State University of Denver
Denver, CO 80217
jody@acm.org

Abstract

An existing undergraduate computer science curriculum was augmented with activities that address diversity, equity, and inclusion (DEI) in the discipline and associated professions. The effort was part of a larger project (Partnership for Equity:STEM) that seeks to cultivate inclusive professional identities in engineering and computer science students through curricular design. The development and implementation of such a curricular activity, used successfully in introductory computer science courses, is described along with qualitative analyses of student responses. The experience also revealed that inherent differences among institutions and across disciplines represent significant impediments to the transferability of curricular activities and the conduct of research in broader contexts.

1 Introduction and Challenges

Concerns and mitigations associated with the negative impacts of the lack of diversity in computer science education and computing professions have been well documented and publicized [5][4][3][8][7][6][13]. The NSF-funded Partnership for Equity:STEM (P4E:STEM) project[10][12] investigated developing professional identities in students through curricular activities that integrate discipline-relevant diversity, equity, and inclusion (DEI) content. This would better prepare students for the workplace by cultivating an understanding of the nature of careers in engineering and computing and instilling in those students: (1) an appreciation for how all kinds of diversity strengthen the disciplines, (2) knowledge of how to act in inclusive ways and create inclusive environments, and (3) consideration of diverse populations who are impacted by their professional practice. The project espouses a broad definition of diversity, which includes problem-solving approaches, personalities, as well as demographic, cognitive, social, and personal characteristics.

A precursor project[9][1][2] had developed curricular interventions (course activities and assignments) for first-year engineering majors at a research-emphasis land grant institution. That effort was aimed at increasing students' awareness of, and facility with, DEI concepts and practices through understanding the value of DEI in the engineering profession.

A five-year plan for the P4E:STEM project included the transfer of outcomes from the precursor project to other institutions and to the computer science discipline, and the development of curriculum for the sophomore and junior years in the undergraduate programs. Co-PIs from three research-oriented (R1/R2) institutions—Colorado State University (CSU), University of Denver (DU), West Virginia University (WVU)—initiated the P4E:STEM project at their respective campuses with the expectation that existing curricular interventions would easily transfer to other institutions and to the computer science discipline. At the close of the first year, reviewers recommended adding an additional institution that differed from the existing partners. The Computer Science program at Metropolitan State University of Denver (MSU Denver) was invited to join as a fourth partner and to provide additional diversity by representing an institution with an education-orientation (vs. research-orientation), a policy of open enrollment (vs. selective admission), and a diverse student body demographic (ages, ethnicities, socio-economics, ...).

Significant complications came to light when attempting to transfer the extant curricular interventions to Computer Science at MSU Denver. Similar difficulties arose with attempts to utilize the existing research instruments at this new partner institution. These included:

- presumption of a research-oriented, selective-admission, predominantly white institutional context
- presumption of common pedagogy across sections taught by different faculty
- disciplinary content tightly-coupled with the DEI content
- presumption of a single appellation for professionals in the field
- presumption of common background and motivation of students in the

first-year courses in the degree program

The difference in institutional contexts was quickly apparent in the set of anticipated responsibilities of the fourth partner institution/department which included, "Help develop, implement and collect data from teaching assistant trainings." The presumed ubiquity of teaching assistants (TAs) in the educational infrastructure was contradicted by the new partner institution having no "teaching assistants" or similar roles. Responsibilities typically delegated to TAs rested entirely with the faculty members themselves.

The lack of transferability of the project's existing course activities to computer science challenged another of the anticipated responsibilities. Based on the presumption that first-year course curriculum previously developed would transfer easily to other institutions and departments, the original specification called on developing new curriculum "especially for sophomore and junior level courses." Once the discrepancy with expectations of transferability was recognized, this was changed to developing new curriculum for the "first-year courses." The result was an opportunity to specify, design, and implement new DEI-relevant curricular activities for introductory computer science courses, as described in the body of this paper.

The need to customize research instruments impacted the ability to collect and compare research data across institutions and disciplines. Questionnaire items frequently employed the term "engineer" in reference to those in the profession associated with the degree, such as,

> "In general, being an engineer is an important part of my self-image."

A suggested adaptation was to simply replace "engineer" with "computer scientist", such as,

> "I have come to think of myself as 'an engineer'."
>
> "I have come to think of myself as 'a computer scientist'."

Although "engineer" describes persons in the profession and associates with engineering majors, "computer scientist" does not describe most persons in the profession who majored in computer science or those currently pursuing that major. In practice, use of the term "computer scientist" was anomalous to students working towards a Computer Science degree and was not associated with their professional pursuits.

There was no single appellation that served the same function as "engineer". "Computing professional" posed similar issues with ambiguity and identity, but was the preferred term when applied to a member of the profession.

> “What are some characteristics good engineers have?”
>
> “What are some characteristics good computer professionals have?”

When “engineer” was used to assess a sense of belonging, the replacement term used was “computer science student,” which served a similar contextual role but reduced comparability across disciplines from a data analysis perspective.

> “Being an engineer is an important reflection of who I am.”
>
> “Being a computer science student is an important reflection of who I am.”

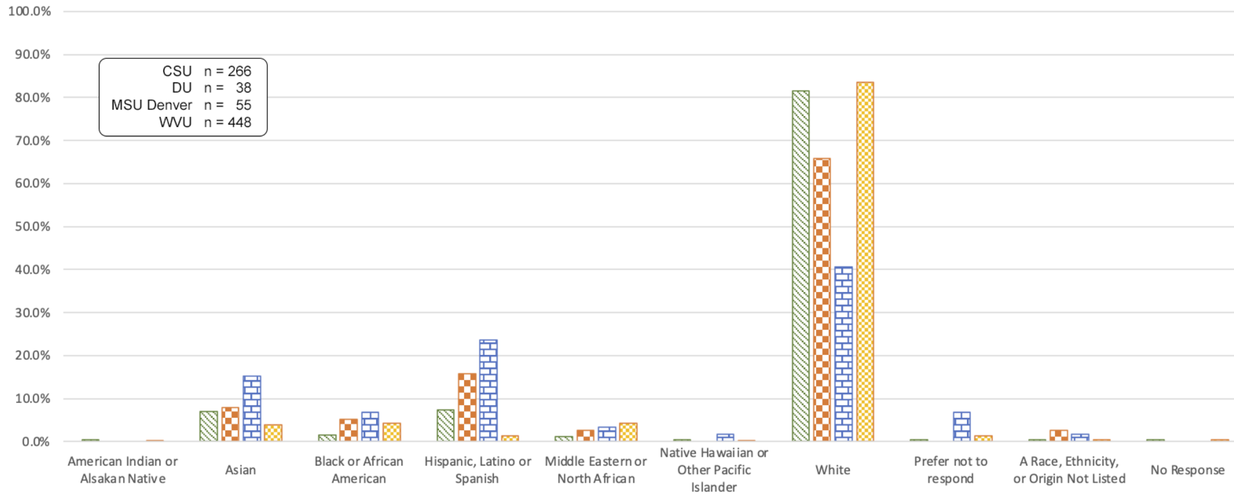
Some questionnaire items presumed that students enrolled in the disciplines’ introductory courses were entering college directly from high-school, generally 18-19 years of age, first time enrolling in higher education, and had chosen their major prior to taking the first-year course. Such was not the case at MSU Denver whose students are typically older and are often continuing their education after a hiatus. Enrollment in introductory computer science courses was not limited to those having declared a computer science major. Sample data from Spring 2020 show the ethnic demographics of students surveyed at the four partner institutions (see Figure 1).

2 Curricular Intervention Design and Implementation

Because the pre-existing curricular interventions were tightly-coupled with engineering content, a new DEI-relevant curriculum activity was needed to address students taking an introductory course in computer science at MSU Denver. The objective selected was raising awareness of factors contributing to the lack of representation of particular demographics in the population of computer science students and professionals. The activity delivery needed to address a student population characterized by breadth in ages, levels of preparation, ethnicities, and academic continuity (e.g., full-time, part-time, and occasional enrollment).

An “Identities in Computer Science” curricular intervention was developed and piloted at MSU Denver.[11] This activity transitions from the *recall of personal experiences* to *consideration of how individuals influence the experiences of others* with the intent of influencing future behavior. The activity, which has been modified after each of multiple administrations, is structured into three stages.

First is a preparatory stage, using prompts that stimulate recall of experiences and awareness of perceptions. This is aimed at encouraging participants to think about “how external environments affect students” and “conceptions of characteristics needed to be a successful computer science student.” Prompts in this stage include the following:



	American Indian or Alsakan Native	Asian	Black or African American	Hispanic, Latino or Spanish	Middle Eastern or North African	Native Hawaiian or Other Pacific Islander	White	Prefer not to respond	A Race, Ethnicity, or Origin Not Listed	No Response
CSU%	0.4%	7.0%	1.5%	7.4%	1.1%	0.4%	81.5%	0.4%	0.4%	0.4%
DU%	0.0%	7.9%	5.3%	15.8%	2.6%	0.0%	65.8%	0.0%	2.6%	0.0%
MSU Denver%	0.0%	15.3%	6.8%	23.7%	3.4%	1.7%	40.7%	6.8%	1.7%	0.0%
WVU%	0.2%	4.0%	4.2%	1.3%	4.2%	0.2%	83.6%	1.3%	0.4%	0.4%

Figure 1: Ethnicity of Intro Course Survey Takers

- P0: What do you want to be when you grow up?
  - P0a: How did you answer when you were first asked that question?
  - P0b: How has that changed over time and why?
- P1: What does it mean to be a Computer Science Student?
  - P1a: After your friends or family members learn that you are enrolled in a computer science course, how does that change how they see you?
  - P1b: In what ways do their comments or behaviors influence your own thinking and behaviors?
- P2: Who can be a Computer Scientist?
  - P2a: What "norms" can you think of that would lead you to consider someone a good candidate to become a computer scientist or other computing professional?

The second stage involves experiencing a TedX talk, "Why Do We Dance Around Diversity in Tech?" via a YouTube video.[14] In this talk, Benjamin Williams uses spoken word and dance to stimulate consideration of why there are so few people of color in computer science and how to improve inclusivity. Benjamin Williams is a software engineer, entrepreneur, and a professional hip-hop dancer. The talk opens with recollections of being asked, "What do you want to be when you grow up?" and closes with asking, "What type of people do we want to be when we grow up?"

The third stage is both reflective and forward-looking. The prompts are also used to prime engagement in group discussion. Prompts in this stage include the following:

- Q0a: How does the environment outside of educational institutions influence someone's choice of whether or not to pursue a career in computing?
- Q0b: What attributes of the classroom environment encourage someone to pursue a career in computing? What attributes discourage someone from such pursuits?
- Q1a: Explain what you think Benjamin Williams meant when he said, *"You are already characters in other people's stories."*
- Q1b: In what ways do you agree with that sentiment? In what ways do you disagree with it?
- Q2a: What character do you think you are in your classmates' stories?

- Q2b: What character would you like to be in the stories of other people?
- R: Take another moment to reflect on your experience with this module. Then describe your observations and any insights you may have gained from your experience.

3 Student Responses

Qualitative analysis of student responses (n=71) suggest that the activity engendered meaningful consideration of factors that influence the under-representation of specific demographic groups in computer science. The form of the activity was effective: very few student responses were off-topic. The activity elicited thoughtful contemplation: most responses were insightful, many demonstrating significant introspection.

The prompt evoking the most introspective responses was

> "What character would you like to be in the stories of other people?"

This question draws directly on a concluding thought in Benjamin Williams' TedX talk. Responses indicated students' desires to modify their future behaviors, expressed as explicit intents to "be more aware of" and "be actively welcoming towards" those who seem to be somewhat "outside" a dominant clique or just not as comfortable in the setting.

The activity initially afforded responding at a superficial level. Modifications of preparatory and reflection prompts reduced the proportion of superficial responses in subsequent versions of the activity.

Of particular note is refinement of prompt Q0b (attributes of the classroom environment) during group discussions to the more-specific and reflective:

> "Recall past class experiences. Where did you feel welcomed? What made you feel like you belonged? Where did you feel like an outsider?"

Responses commonly shared by students to the refined prompt are captured in the following observations of what made them feel unwelcome or anxious about being in a particular classroom or lab setting:

- Entering a classroom the first day and everyone but me had a laptop computer.
- The people in class didn't look like me. (dress, mannerisms, ethnicity, gender, . . .)
- The lab was all [a particular gender or ethnicity] and there wasn't a comfortable place to sit there.
- Everyone else in the class had more experience than me.

Regarding the activity more generally, students indicated that the questions/prompts were not explicit enough about *racial inequity*.

Student responses suggest that the kind of activity exemplified by "Identities in Computing" affords a way of empowering individuals to be agents of change. The design—moving from recalling personal experiences to how individuals influence the experiences of others—appeared to engender explicit intents to engage in more inclusive behaviors.

4 Conclusions

Although the objectives were the same across all partner institutions and programs, the specific implementation contexts mattered tremendously. Transferability of research instruments, processes, and products was not direct and less feasible than anticipated. In particular, the characteristics typical of research-oriented institutions are not necessarily in common with the larger set of higher education institutions. Even within any particular type of institution, the status of DEI awareness, activity, and policy at the institution and its representation in the existing curriculum vary greatly and significantly impact intervention effectiveness. This experience demonstrated one particular approach to DEI curricular integration that appeared to be effective, albeit with potentially low transferability to other disciplines. A future analysis of the collected set of curriculum activities may support discovery of discernable patterns of design that facilitate development of DEI curriculum integrations with high transferability.

References

[1] Rebecca A Atadero, Christina Paguyo, Karen E Rambo-Hernandez, and Heather Lysbeth Henderson. Promoting inclusive engineering identities in first-year engineering courses. In *2016 ASEE Annual Conference & Exposition*, 2016.

[2] Rebecca A Atadero, Christina H Paguyo, Karen E Rambo-Hernandez, and Heather L Henderson. Building inclusive engineering identities: implications for changing engineering culture. *European Journal of Engineering Education*, 43(3):378–398, 2018.

[3] Tracy Chou. Women in software engineering - data. 2019. `https://github.com/triketora/women-in-software-eng`.

[4] Tracy Chou. Where are the numbers? *Medium*, 11:47–61, 2013.

[5] Robyn Foyster. Workforce diversity: Why are we where we are? *Women Love Tech*, 8 March 2021. https://womenlovetech.com/workforce-diversity-why-are-we-where-we-are/.

[6] Rachel Gutman. The origins of diversity data in tech. *The Atlantic*, 3 February 2018. https://www.theatlantic.com/technology/archive/2018/02/the-origins-of-diversity-data-in-tech/552155/.

[7] Jane Margolis. *Stuck in the Shallow End, updated edition: Education, Race, and Computing*. MIT press, 2017.

[8] Jane Margolis and Allan Fisher. *Unlocking the clubhouse: Women in computing*. MIT press, 2002.

[9] Christina H Paguyo, Rebecca A Atadero, Karen E Rambo-Hernandez, and Jennifer Francis. Creating inclusive environments in first-year engineering classes to support student retention and learning. In *2015 ASEE Annual Conference & Exposition*, pages 26–418, 2015.

[10] Jody Paul, Christina Paguyo, Rebecca Atadero, and Karen Rambo-Hernandez. Partnership for Equity: STEM, 2022. https://partnership4equity.org.

[11] Jody Paul, Myke Walker, and Megan Horton. Identities in computing, 2020. https://partnership4equity.org/resources/Lesson-Plan_Identities-in-Computing.pdf.

[12] Blaine Austin Pedersen, Robin AM Hensel, Sumaia Ali Raisa, Rebecca A Atadero, AM Aramati Casper, Ronald R DeLyser, Christopher D Griffin, Scott T Leutenegger, Melissa Lynn Morris, Christina Paguyo, et al. Leveraging changes in engineering and computer science curricula to engender inclusive professional identities in students. In *2021 ASEE Virtual Annual Conference Content Access*, 2021.

[13] Rebecca Thomson. Why poor diversity is bad for the IT industry. *Computer Weekly*, 3 September 2009. https://www.theatlantic.com/technology/archive/2018/02/the-origins-of-diversity-data-in-tech/552155/.

[14] Benjamin Williams. Why do we dance around diversity in tech?, 2017. Accompanied by dancers Monyett Crump Jr. and Jasmin Williams. https://www.youtube.com/watch?v=b5n7my8KJ0o.

A Survey of Software Testing Tools in the Web Development Domain*

Abdullah Alenzi[1], *Waleed Alhumud*[1], *Renée Bryce*[1]
and Nasser Alshammari [2]
[1]*Computer Science and Engineering*
University of North Texas
Denton, TX 76203
{AbdullahAlenzi,WaleedAlhumud}@my.unt.edu
Renee.Bryce@unt.edu
[2]*College of Computer and Information Sciences*
Jouf University
Saudi Arabia
nashamri@ju.edu.sa

Abstract

Software testing is important to provide reliability, efficiency, and to minimize errors in the development process. There are many criteria to consider when choosing software testing tools. In this paper, we conducted a survey with professional developers to provide guidance to instructors on which criteria to adopt when selecting web application testing tools to use in their courses. As technology constantly changes, students may benefit from a foundation in which they understand such criteria, trade-offs of different tools, and gain experience with popular testing tools. The results show that the top five criteria are platform support, report generation, technical support, browser support and language support. Specific tool recommendations are provided with discussion of applicable domains and scenarios.

1 Introduction

Software Testing is an important part of the software development life cycle (SDLC) [8]. Detecting errors is not a trivial task and it takes time and effort. Software testing tools have been proven to save time and reduce boring activities. Many testing tools are supportive, helpful and facilitate finding errors in software. As we prepare students for software engineering careers, it's important for them to consider which criteria to use when selecting software testing tools and to know examples of popular tools. In this work, we focus on which criteria to select when developing web testing tools and solicit examples of tools from a group of 91 developers so that instructors may use this knowledge when they make choices for their courses.

The number of web applications have grown around the world in an unprecedented way. For web developers, having web applications free of bugs and errors is one of their priorities even though it is a challenge. Thus, there is a need to have software testing tools that allows the developers to detect and reduce bugs. Web applications are playing a significant role, nowadays. Many software testing tools are available for the web application domain [10]. Choosing an appropriate software testing tool requires knowledge, experience, and familiarity with the domain. There are a variety of testing tools and each too has its advantages and disadvantages based on the software domains such as system software, embedded software, web applications. In addition, type of testing is an effective factor in classification of testing tools. For instance, functional testing, performance testing, smoke testing, load testing, to mention a few. The motivation behind this research is to identify the most important criteria for selecting software testing tools and also to find out which software testing tools are most suitable for developers in the web development domain.

2 Background

Software testing tools and techniques are continuously advancing at a rapid pace with technology. This section reviews papers that discuss important features of tools and many comparisons that have been made over the years. Rigzin et al.[3] discussed how to identify the best tool in the Selenium suite. The authors studied and evaluated different testing tools in terms of performance then compared the best tool in Selenium suite with some selected tools which have the same function. They found that Selenium Webdriver is the best choice depending on some parameters.

Rabiya et al.[1] compared four automated testing tools applied in load testing. The comparison between these tools relied on some selected criteria such as scripts generation, plug-in support, reports, supported application, and cost to determine the best and more efficient tools. The targeted tools to be compared are Apache JMeter, Microsoft Visual Studio (TFS), LoadRunner, and Siege. They created a comparison table to differentiate between these tools and presented some benefits and drawbacks. Despite LoadRunner being a licensed testing tool, it is the best tool regarding performance. The testing tools are only used in load testing which supports users and developers who need to focus on load testing scenarios.

Mohamed Monier et al.[9] provided comparative study between different testing tools in web testing. They selected the tools based on the feature of recording and playing the script. Few criteria have been used in both studies [1] and [9] which limits the users' choices.

Dilara Ateset et al.[5] collected twenty-one automated tools and made a comparison between these tools based on their features and attributes. The comparison involved different features like record and reply, reusability, language support, operating system support, parallel test running, AI based object recognition, framework support, mobile and desktop application test. They found some distinctive and popular features such as GUI test, record and playback, and operating system support.

Heidilyn V. Gamido et al.[6] presented a comparative review of testing tools based on various characteristics such as platform compatibility, ease of use, report generation, data driven, cost, function, script language, record playback, and browsers compatibility. Based on their recommendation, Selenium is the best choice if the budget of the project is high. QTP/UFT as a commercial tool is recommended if the features: report generation, easy learning, and availability of support are needed. Multiple testing tools are classified based on variety and distinguished criteria as presented in [5] and [6] which allows for more options to choose the appropriate testing tool.

Saja Khalid et al.[2] presented the most common tools and categorized them based on different testing types to determine the best and more efficient tools. Maher et al.[7] presented the most important and required testing methodologies in web 2.0 applications. In addition, different testing tools were shown based on testing types. Karuturi et al.[12] defined many different kinds of software testing strategies "level" and approaches of testing. Khaled M. Mustafa et al.[10] classified a group of testing tools based on types of testing methodologies to be applied on three different types of software. It includes web application, application software, and network protocol to identify the testing tools that support such a type of testing and which type of testing has limitations in these tools. The core contribution of these works in [2], [10], [7], and [12] classified and categorized testing tools with respect to some characteristics and parameters. Their reliance was on testing types, methodologies, or level of testing. However, these classifications do not consider the developers' needs and requirements.

Sarmad Arif et al.[4] made a comparison between different types of testing and various testing tools. Their aim was to illustrate which testing tools are extensively used for mobile application testing and what types of testing that the developers and testers focus on in the mobile applications domain. As a result, the most significant testing types of mobile application domain were functional, performance, usability, compatibility, security, and interoperability testing. Based on quality assurance, different automated tools were presented and described such as Appium, Selendroid, Monkeyrunner, and Robotium. There are limitations in the number of testing tools in terms of integration testing and security testing. The number of the participants in this study was only 37 which might affect the results.

From the previous research studies, we identified the most popular software testing tools in the literature. The review of the previous studies shows that there are different criteria and parameters for evaluating software testing tools in the web development domain.

Our work differs in four major ways: (1) we conduct our survey with a larger sample of 91 developers, (2) focus on the web development domain, (3) analyze important criteria for tool selection, and (4) summarize specific examples of tools that meet such criteria. This work provides guidance to instructors that wish to prepare their students to review testing tools that are suited for different scenarios and provides concrete examples of tools that are favorably recommended by a large number of developers that completed our survey.

3 Methodology

The main goal of this research is to answer the following questions:

- **RQ1:** What are the most important criteria for choosing testing tools in the web development domain?
- **RQ2:** What are the most popularly reported testing tools in the web development domain?

The study surveys 91 software developers that have experience in the web development domain and was conducted online. The survey was sent to a targeted sample of participants who majored in computer science. The participants have at least a master's degree or currently are enrolled in a higher education program.

Table 1 shows the list of eight criteria for the developers to rank using a 5-point linear scale (Likert scale), including platform support, programming skills, technical support, browser support, language support, record and playback functionality, report generation, and a public license.

Table 1: Description of Selected Criteria

Criterion	Description
Platform Support	The capability of the software testing tool to support different operating systems.
Programming Skills	The tool requires a certain level of programming skills.
Technical Support	The availability of technical support for the users of the software testing tools.
Browser Support	The ability of the software testing tool to support different browsers.
Language Support	What programming languages does the software testing tool support.
Record and Playback	It means the tool is capable of recording scripts and playing them back.
Report Generation	The capability of tool to generate results and reports.
Publication License	It means that the testing tool is released under either an open source or commercial licence.

Based on the survey results, we assessed a list of popular web testing tools according to the criteria in the results.

4 Results

The results shown in Figure 1 show how the developers value the different criteria. The evaluation is based on Likert Scale of 1 (not important at all), 2 (slightly important), 3 (moderately important), 4 (very important) and 5 (extremely important). Figure 1 show the percentage of survey participants that report each of these scores for each criteria.

Four of the criteria were rated as 5 (extremely important) by more than 50% of the developers: platform support, browser support, technical support, and report generation. All other criteria was rated as 5 (extremely important) by more than 25% of the developers. The less important criteria (although still important to more than a quarter of the developers) include publication license, language support, record and playback, and programming skills.

Figure 2 shows the overall importance of the criteria based on the participants' responses. On average, the most important criterion is platform support while the developers see that record and playback is not important in comparison to the other criteria. In addition, the developers report that browser

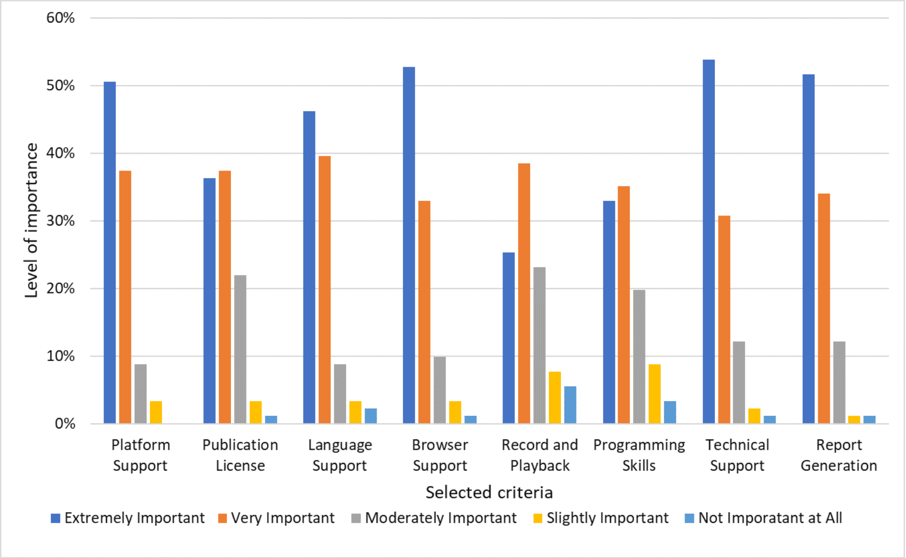


Figure 1: Individual rankings of criteria for web testing tool selection.

support criterion is also extremely important mostly as platform support. The result indicates that technical support and report generation criteria have similar interest with approximately 86% each. The participants' responses show that the programming skills criterion is less important than most other criteria due to solid programming background of the developers. A Pearson correlation test was performed on the data and reveals a significant relation between the level of experience in web development and the technical support criterion (r = 0.308, with a 2-tailed significance value of 0.003). On the other hand, there is a significant inverse correlation between the number of years of experience in software testing and the technical support criterion (r = - 0.216, with a 2-tailed significance value of 0.041). The participants' responses indicate that platform support, technical support, report generation, and browser support are the most important criteria respectively.

4.1 Software Testing Tools

This section presents software testing tools and discusses them in relation to the criteria that was ranked above: cross-platform, technical support, report generation and browser support.

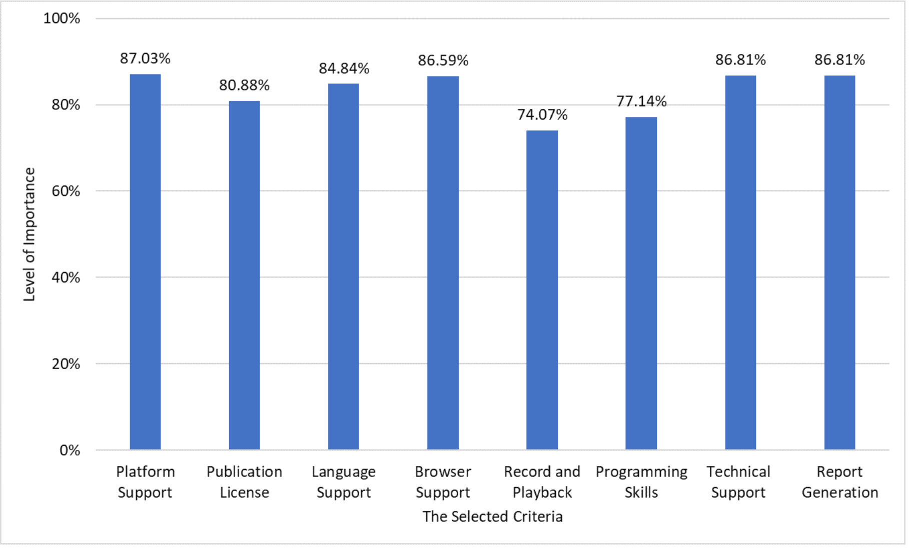


Figure 2: Overall results of each web testing tool criteria.

4.1.1 Selenium

Selenium is an open-source testing tool on the web application domain. It is one of the most common and popular automated testing tools that is widely used [7]. It supports various programming languages such as C#, Java, Groovy, Perl, PHP, Ruby, Python and Scala [12]. In addition, it is compatible with many web browsers like IE, Firefox, Opera, Safari. Selenium is a cross-platform which runs on different platforms such as Windows, Linux, and Mac. It supports record and replay features and it has a user-friendly interface [3]. However, using Selenium is not simple because it needs an experience. Selenium IDE supports report generation feature [11]. In spite of Selenium has a technical support, it is not official [8].

4.1.2 Watir

Watir stands for Web Application Testing in Ruby [7]. It is an open-source testing tool, and it can be run on many operating systems. It is considered a simple testing tool for using [6]. Also, it is compatible with multiple web browsers like Chrome, Firefox, Internet Explorer, Safari, Edge [5]. It supports Ruby Language and others like Java, C#, .net and it supports record playback features. However, it is an inappropriate testing tool for beginners in terms of programming skills because it requires a programming experience in Ruby Language [7]. Watir supports report generation where results can be represented as Htlm and Xml forms [6]. Watir offers limited online technical support for its users [9].

4.1.3 TestComplete

In 2018, TestComplete has been categorized as a leader in software testing automation [7]. TestComplete is an automated tool that supports different testing types. It supports many features such as record-playback, online-support, debugging support, and report generation [9]. It is a commercial software testing tool, and it supports multiple programming languages such as C#, C++, Jscript VBscript and Delphi [6]. Test Complete needs some experience and programming background and it is only compatible with Windows [6]. However, it supports multiple web browsers [9].

4.1.4 SahiPro

SahiPro is an testing tool that is used in the web application domain. It can generate reports in HTML, and it also supports record-playback [6]. It is simple to learn but it needs some experience of programming skills [6]. It supports different operating systems [7] and the basic features are free to use but it has advanced features which are paid including online support [9]. In addition, it is cross-browser and supports a few programming languages which are JavaScript and Ruby [9].

4.1.5 QTP

In regression testing and functional testing, Quick Test Professional (QTP) is considered one of the most common testing tools [8]. QTP has a unique feature which allows testers to test three different levels including the service, the interface, and the database level [7]. QTP is a commercial tool that runs only on Windows OS. It supports record-playback, data driven features [8]. Also, QTP is compatible with IE, Firefox and Chrome browsers. In addition, it supports VB script, Java, .NET, and Delphi. However, it needs some experience of programming skills and the online support is not free [9]. QTP users can generate reports in different forms such as Html and Xml [6].

4.1.6 JMeter

Apache JMeter was developed by Apache Software Foundation [2]. It is an open-source testing tool [13]. It is a cross platform and browser. In addition, it supports BeanShell and java [1]. Also, it provides a regression testing and supports performance testing [13]. However, JMeter does not require a high level of programming skills. JMeter can generate reports and it has the feature to record test scripts [13]. JMeter offers technical support via email.

4.1.7 LoadRunner

It provides two versions which are free and paid versions. It supports Windows, Linux and Mac OS and it is cross-browser. In addition, it supports VB, C, VBscript C# and JavaScript. It also provides report generation and record playback features. However, it needs experience to use it and partially needs programming skills. [6].

5 Threats to validity

This paper has threats to validity that we attempted to minimize. First, we noticed that many previous studies related to this work have very small sample sizes [4]. We tried to minimize this threat by using a sample size of 91 developers. Another limitation of this study is that technology advances at a rapid pace and some of the tools in this paper may eventually become obsolete. Therefore, we chose popular tools that allow us to focus on evaluating whether they support the criteria deemed important by developers so that readers have concrete examples of trade-offs among tools. The number of the selected criteria was limited. Thus, investigating more criteria will be considered. This paper may be updated in the future after new testing tools emerge. Indeed, a longitudinal study of preferences over time may provide more insight into the evolution of testing tools and importance of different tool selection criteria.

6 Conclusion

Software testing is critical in the tech industry. Different testing tools have trade-offs and users must consider their domain and specific scenarios when they select the best tools to use. Instructors strive to prepare their students for careers in the tech industry and often select testing tools for students to use on class projects. This work provides guidance from 91 developers that rank important criteria to consider in the selection of such software testing tools in the web domain. The results show that supported platform, technical support, report generation, browser support, and language support were the most important criteria. Based on these criteria, comparison and classification are presented between different software testing tools to help select proper tools for different scenarios.

References

[1] Rabiya Abbas, Zainab Sultan, and Shahid Nazir Bhatti. Comparative analysis of automated load testing tools: Apache JMeter, Microsoft Visual Studio (TFS), LoadRunner, Siege. In *2017 International Conference on Communication Technologies (ComTech)*, pages 39–44, 2017.

[2] Saja Khalid Alferidah and Shakeel Ahmed. Automated software testing tools. In *2020 International Conference on Computing and Information Technology (ICCIT-1441)*, pages 1–4, 2020.

[3] Rigzin Angmo and Monika Sharma. Performance evaluation of web based automation testing tools. In *2014 5th International Conference - Confluence The Next Generation Information Technology Summit (Confluence)*, pages 731–735, 2014.

[4] Khawaja Sarmad Arif and Usman Ali. Mobile application testing tools and their challenges: A comparative study. In *2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, pages 1–6, 2019.

[5] Dilara Ateşoğulları and Alok Mishra. Automation testing tools: A comparative view. *International Journal on Information Technologies & Security*, 12(4), 2020.

[6] Heidilyn V. Gamido and Marlon Viray Gamido. Comparative review of the features of automated software testing tools. *International Journal of Electrical and Computer Engineering (IJECE)*, 9:4473–4478, 2019.

[7] Maher Hayek, Peter Farhat, Youssef Yamout, Charbel Ghorra, and Ramzi A. Haraty. Web 2.0 testing tools: A compendium. In *2019 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)*, pages 1–6, 2019.

[8] Rakesh Kumar Lenka, Utkalika Satapathy, and Meenu Dey. Comparative analysis on automated testing of web-based application. In *2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*, pages 408–413, 2018.

[9] Mohamed Monier and Mahmoud El-mahdy. Evaluation of automated web testing tools. *International Journal of Computer Applications Technology and Research*, 4:405–408, 05 2015.

[10] Khaled M. Mustafa, Rafa E. Al-Qutaish, and Mohammad I. Muhairat. Classification of software testing tools based on the software testing methods. In *2009 Second International Conference on Computer and Electrical Engineering*, volume 1, pages 229–233, 2009.

[11] Khushboo Sawant, Reetu Tiwari, Swapnil Vyas, Pawan Sharma, Aryan Anand, and Shivani Soni. Implementation of selenium automation & report generation using selenium web driver & ATF. In *2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*, pages 1–6, 2021.

[12] Karuturi Sneha and Gowda M Malle. Research on software testing techniques and software automation testing tools. In *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, pages 77–81, 2017.

[13] Junmei Wang and Jihong Wu. Research on performance automation testing technology based on JMeter. In *2019 International Conference on Robots & Intelligent System (ICRIS)*, pages 55–58, 2019.

Dead Reckoning and Terrain Image Processing as basis for UAV Home-oriented Navigation under foreign GPS-denied Environments*

Christine Van Kirk[1], *Alice Chen*[2], *Saad Biaz*[3], *Richard Chapman*[3]
[1] *University of Notre Dame*
Notre Dame, IN 46556
cvankir2@nd.edu
[2] *The City College of New York*
New York, NY 10031
nianxc@gmail.com
[3] *Auburn University*
Auburn, AL 36849
{chapmro, biazsaa}@auburn.edu

Abstract

Unmanned Aerial Vehicle (UAV) navigation within GPS-denied environments has become increasingly critical in recent years due to modern heavy reliance on GPS and the rise of UAV flight within GPS-denied and GPS-unstable zones. GPS reliance marks on the of the largest weaknesses of most autonomous UAV systems, and the loss of GPS signal often renders many autonomous UAVs completely coordinately-impaired. To confront this issue, this paper presents the implementation of a navigational system based upon dead reckoning and terrain imaging for home-oriented navigation, which is viable even in foreign environments. It explores the possibility of utilizing various forms of image processing, such as feature extraction and matching (SIFT) and optical flow via template matching, in order to replicate similar data that would have otherwise been produced by GPS outputs (namely, confirmation of an estimated final location with real location through image feature comparisons, ground speed calculations from terrain distance movement). This paper uses traditional dead reckoning methods as a path basis for the initial UAV navigation, and it applies both physical implementation and formal methods to study the viability of this solution.

1 Introduction

Unmanned aerial vehicles (UAVs), such as quadrotors and fixed-wing drones, are used more frequently in recent years due to the combination of low cost and high user friendliness. The lightweight design and user setup has accelerated UAV growth, expanding both the user base and general purposes behind drone deployment. These growing applications in daily life include emergency medicine transport to remote locations, environmental monitoring, surveillance photography, terrain imaging, and countless other growing fields. Regardless of application, all autonomous UAVs should contain a system for returning to a source position if navigational processes fail.

Current and previous autonomous navigation systems for most UAV applications center primarily upon the utilization of Global Positioning Systems (GPS); this presents a significant challenge for current and future autonomous UAV flight, as this system of navigation is restricted by GPS-jamming devices, indoor environments, signal-denied zones, and increases in governmental legislation. Such signal-denied zones include dense tree regions, underground areas, and extraterrestrial landscape where the GPS signal is unable to penetrate solid walls and/or structures. Additionally, although the United States Department of Defense currently upkeeps the satellites used for GPS and provides their services free of cost, the Department reserves the right to deny services at any time, forming a dependency that is often undesirable for third-party companies, organizations, and individuals.

To explore other avenues of autonomous navigation outside of GPS implementations, this paper proposes combining traditional dead reckoning methods with various forms of terrain image processing, specifically for the return of the UAV to a source position within GPS-unstable zones. Aerial images taken from the underbelly of the UAV present an opportunity for computer vision techniques to yield similar flight data that would have otherwise been produced by GPS outputs, regardless of environment interference that could impair coordinate systems. The UAV takes a photo of its source location upon liftoff at the beginning of its flight. Upon the loss of GPS-signal, the UAV estimates its trajectory home via simple dead reckoning, and checks the current terrain images with the previously taken photo of its source location; if the images do not match and the dead reckoning estimation is off due to drift or miscalculation, then the UAV will fly in a spiral pattern until the source location is found. Although this presents a possibility for total loss, we expect to minimize this error associated with the dead reckoning methods through use of accumulated terrain imaging for real ground speed and drift estimations. It is notable that this method for autonomous navigation requires very little computational power and minor memory access during the original flight of the UAV (merely taking a photo of its initial environment for future reference), and the system only becomes truly active when GPS-signal is lost. This method is also suitable for completely foreign environments, as there is no pre-flight training, offline map access, or annotation of the environment required beforehand.

2 Related Work

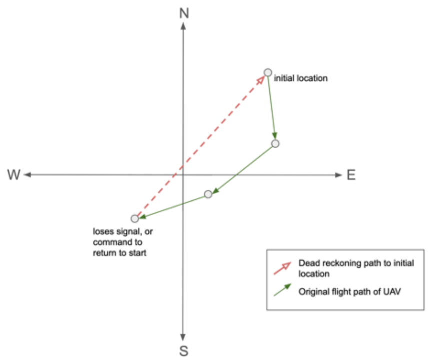
2.1 Traditional Dead Reckoning

Dead reckoning has historically been used as the default estimation method for the return towards a home destination by keeping a log of the previous relative distances and angles traveled. By updating a record of the relative angles shifted and the distances traveled for each angle position, a simple return direction and angle can be calculated, as shown in Figure 1. We propose using this method as the starting point for our UAV home-oriented navigation upon the loss of GPS signal.

In addition, we supplement these dead reckoning calculations with other estimations (wind speed, real ground speed, etc.) to oppose the effects wind and drift upon the UAV trajectory, also shown in Figure 1. Previous work demonstrated by Wang et al. (2017) explores methods of estimating UAV wind velocity with minimal sensor use through calculating an airspeed vector, based upon derivations of the wind disturbances between rotor speed and UAV acceleration[11]. Implementable methods for estimating UAV thrust, drag coefficient, and wind drag are also explored.

2.2 Visual Navigation Approaches

Computer vision techniques have shown considerable promise for the task of autonomous UAV navigation. Previous approaches have utilized Simultaneous Localization and Mapping (SLAM), convolutional neural networks (CNNs), and Visual Odometry, amongst others. We will highlight the fundamental methods for each of these previous approaches, as well as the associated strengths and weaknesses for each.



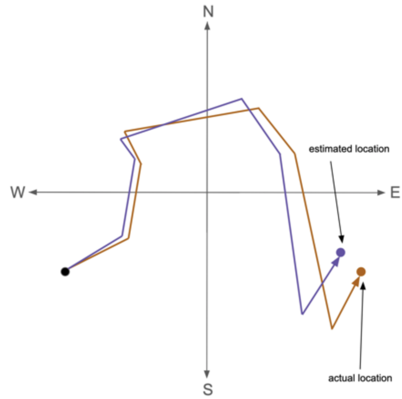


Figure 1: (left) demonstrates how traditional dead reckoning methods accumulates previous flight data to calculate simple director and distance to return to a source location. (right) demonstrates the aggregated error that is apparent over longer distances with dead reckoning approaches.

Simultaneous Localization and Mapping (SLAM) methods focus upon building and updating a map of an unknown environment while simultaneously updating the UAVs position within the map itself. Smith et al. (1986) defined the foundation of SLAM for general robotic implementations several decades ago, and it has since been applied to fields outside of the scope of the original paper[10]. Lopez et al. (2017) demonstrates a system of SLAM to record UAV localization in GPS-denied zones through integrated use of vision, laser, and inertial measurements[5]. This implementation yields experimental results that improves the estimated trajectory of the UAV compared to traditional baseline techniques.

Convolutional Neural Network approaches have also demonstrated promising results. The work of Shamer et al. (2019) outlines an approach for UAV path-following in GPS-denied environments[8]. This method inserts waypoints for the UAV to follow in simulation of a flight trajectory, and it trains a convolutional neural network to output the yaw angle delta based on visual input for each respective path between waypoints. Although it was met with considerable success with a final cross distance track of 2.88 meters, it is notable that this system requires an offline map of the proposed foreign environment for extraction of distinct landmarks, and it also requires pre-flight training for each respective sub path between the multiple waypoints.

2.3 Feature Matching Approaches

Work in the field of computer vision outside of strict navigational implementations includes methods for matching keypoint features between images, as well as methods for finding sub images within a second image. The most widely accepted implementations of feature matching between two images are Scale-Invariant Feature Transform (SIFT), Speeded-Up Robust Features (SURF), Binary Robust Invariant Scalable keypoints (BRISK), and Binary Robust Invariant Scalable keypoints (BRISK), amongst others. As a considerable portion of this paper focuses upon feature extraction and matching algorithms for determining if the UAV has correctly located the source position, an overview is given of these various methods in determining which is most suitable for specific purposes.

Lowe (2004) submitted ground-breaking work within the field of computer vision for his implementation of Scale-Invariant Feature Transform (SIFT), which is invariant to alterations in rotational orientation and scale[6]. This method uses gaussian blurring to reduce the noise of the image, a difference of gaussians approach to construct a scale space of the high contrast variations, and a threshold on minimum contrast to select the best keypoints. The corresponding keypoint descriptors are then created through the construction of a unique histogram of gradients for each keypoint. Lowe also proposes a method of selecting the best match between keypoints in two images by placing a maximum threshold on the ratio between the best- and second-best matches for each featured descriptor. As a high ratio indicates an ambiguous match, the paper suggests a threshold of maximum 0.8 to reject approximately 90 percent of the incorrect matches while losing less than 5 percent of the correct machines.

Following this publication, several others submitted work on new forms of scale invariant and rotationally invariant feature extraction and matching methods. Bay et al. (2007) submitted the outline for Speeded-Up Robust Features (SURF), which is much faster than previous scale and rotation invariant feature matching approaches due to its reliance on integral images for image convolutions, rather than the previous difference of Gaussians approach of SIFT[1]. Calonder et al. (2010) proposed the implementation of Binary Robust Independent Elementary Features (BRIEF), which contains more efficient feature descriptors through the implementation of binary strings[2]. Leutenegger et al. (2010) submitted the framework for Binary Robust Invariant Scalable keypoints (BRISK) for a feature extraction and matching approach that operates at a significantly lower computational cost[4].

Several studies have been conducted on the relative real-time merits and demerits of each of these various feature extraction and matching approaches. Darshana Misry et al. (2017) suggests that SURF implementations are computationally three times faster than SIFT implementations[7]. Maji et al. (2020)

compares SURF and BRISK algorithms in the terrain imaging space, specifically for the feature recognition rate in video image processing after ten frames, and concludes that SURF features yield a 30 percent recognition rate while BRISK features maintain a rate of approximately 50 percent[9]. The authors note that possible design-inherent factors of their experimentation, such as the experiment's study on moon-comparable surfaces with similar repeated texture, may have affected the outputs of their results, and they also note that the BRISK features may have surpassed initial expectations due to the circular pattern for BRISK description generation (rather than the rectangle patterns for SURF). Furthermore, Karami et al. (2017) found that in performance testing for feature extraction and matching for distorted images, SIFT generally outperformed SURF and BRIEF, although at an expense of greater computational complexity (as compared with SURF) and memory requirement (as compared with BRIEF)[3].

Due to the wide success for keypoint extraction and matching despite changes in rotation and scale, which are inherent to UAV imaging with constant modifications in UAV orientation and altitude, as well as general better performance, SIFT will be implemented in this approach for feature matching between aerial images. The SIFT template within OpenCV also provides easier user implementation and greater support documentation than other feature matching templates. This implementation of SIFT will be used in the confirmation of whether the final estimated location corresponds with its initial beginning location, verifying that the UAV is correctly oriented in its home position prior to landing.

3 Proposed Approaches

As mentioned previously, this paper explores (a) the use of feature matching to confirm the UAV location and (b) the use of template matching to estimate the UAV ground speed. These estimations can be used to supplement dead reckoning methods and reduce the margin of error that is associated with dead reckoning navigation over long distances.

This is organized as follows. First, an overview of traditional dead reckoning and its correlated equations are given. Next, an overview of the calculations for determining the ground speed between two overlapping aerial images are given. Thirdly, an overview of the calculations behind feature matching and the definitions of "good matches" are given for confirmation of whether two photos represent the same source location. The results of these proposed methods are given in the next section, Section 4: Experimental Results.

3.1 Dead Reckoning

In this section, the basics behind dead reckoning are covered. In moving between two fixed coordinates, the heading angle is defined as the angle orientation relative to North while moving in a straight line between the two points. In addition, the three angles that define an UAV's orientation are yaw (z-axis), pitch (y-axis), and roll (x-axis). A Triple-axis Accelerometer and Magnetometer can be used to measure roll, pitch and yaw angle. The roll and pitch angle can be estimated using rotation matrix.

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}_{Body} = \begin{bmatrix} cos\theta & 0 & -sin\theta \\ sin\theta sin\phi & cos\psi & cos\theta sin\phi \\ sin\theta cos\phi & -sin\psi & cos\theta cos\phi \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix}_{Level}$$

$$(1) a_x = gsin\theta$$

$$(2) a_y = -gcos\theta sin\phi$$

$$(3) a_z = -gcos\theta cos\phi$$

Roll and pitch formula using rotation matrix:

$$\phi = arctan(\frac{a_y}{a_z})$$

$$\theta = arcsin(\frac{a_x}{g})$$

In this UAV application, the yaw angle can be equivalent to the heading angle, should the UAV be oriented towards North when at the beginning of its path.

$$\psi = atan2(m_y, m_x)$$

These three angles reveal which direction the UAV is facing, and simply summing this current-facing angle with the heading angle demonstrates the desired turning angle.

In addition to the angle needed for flight, dead reckoning methods for UAV purposes typically include the Haversine formula. The Haversine formula is:

$$a = sin^2(\Delta\varphi/2) + cos\varphi_1 * cos\varphi_2 * sin^2(\Delta\lambda/2)$$

$$c = 2 * atan2(\sqrt{a}, \sqrt{(1-a)})$$

$$d = R * c$$

where φ is latitude, λ is longitude and R is the Radius of earth.

This formula yields the distance between two coordinates while taking into account the slight curvature of the Earth. Combining both the angle (direction) and the Haversine distance should allow the UAV to navigate successfully to its initial source location, should the UAV lose GPS singal or need to return to the source destination.

Additionally, although the UAV can calculate a relative direction and time of travel, external factors such as drift and additional wind may impact the true distance traveled relative to the estimated distance traveled. To confront this issue with computer vision instead of GPS-devices, additional measurements (see next subsections, 'Estimation of Ground Speed' and 'Location Confirmation through Image Feature Matching') are used to ensure that the UAV does not surpass or fail to reach the target (source) destination.

3.2 Estimation of Ground Speed

In this section, template matching algorithms and trigonometry are used to find the UAV's ground speed to supplement dead reckoning calculations, thereby reducing the accumulated error associated with long distances (as shown in Figure 2). Only the altitude, the field of view of the camera, and the time between two photographs are needed for this estimation. The general method of making this ground speed estimation is found via image processing by taking two overlapping photos, finding the terrain distance moved between the two photos (applying template matching), finding the scale of the image (converting the pixels moved into meters), and dividing by the time taken between the photos.

Template Matching refers to algorithms for finding one small image, referred to as "the template", within one larger image. This method is highly sensitive to changes in brightness, rotation/scale, intensity, and illumination; however, these sensitivities are manageable within this application as the UAV takes photos (a) merely seconds apart, (b) at the same altitude, (c) with the same camera, and (d) while flying in a relative straight path.

First, two overlapping images are taken by a camera on the underbelly the UAV, as shown in the left of Figure 2. A template is created from the first image from simple cropping, and this template represents the center of the first image. Template matching is applied between this template and the second photo, and the match coordinates of the template within the second image is returned. The Euclidean distance between the midpoint coordinates of the created template within the first image and the midpoint coordinates of the template match within the second image represents the amount of pixels moved by the UAV in the photo time-span. To convert this "pixels moved" into "distance moved," simple trigonometry is required to give the image a scale. An angle bisect of the camera field of view, the current altitude of the UAV, and half of the terrain distance of the photo width create a right triangle, as shown in the right of Figure 2. Multiplying the previously calculated pixels

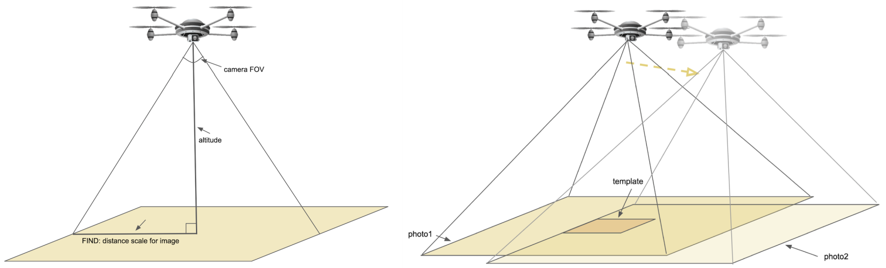


Figure 2: Method for ground speed estimation of UAV via images. (left) demonstrates using two consecutive overlapping photos to determine the distance moved by the UAV in a short time period. (right) demonstrates finding scale of the image (pixels/meter) by simple trigonometry with angle bisect of camera field of view and altitude of UAV.

moved with this scale of meters/pixel yields a distance moved, which can be divided by the time between the two photos to yield a final ground speed of meters/second, shown in Figure 3.

$$\frac{meters}{pixel} = \frac{tan(\frac{FOV}{2}) \times altitude}{(0.5 \times width)}$$

Figure 3: Formula for determining the scale of the image (meters/pixel), where FOV represents the field of view of the camera, the altitude is given in meters, and the width represents the dimensions of the photo output

3.3 Location Confirmation through Feature Matching

To verify that the UAV is indeed at its source location, the current image of the underlying terrain is compared with the original photo taken of its source location. These two photos will not be exactly the same due to changes in brightness, shadows, rotation, slight shifting in orientation, illumination, and other photo qualities. By directing the photos to recognize key features rather than simply overlaying the photos, these slight changes can be overlooked to still correctly identify locations regardless of some degree of photo variance.

In SIFT terminology, feature keypoints are the portions of the image that represent a high contrast point. Each feature keypoint has a corresponding keypoint descriptor, which is a vector representing its local surroundings. More formally, this vector represents a unique histogram of gradients that is associated with each keypoint. keypoints are connected between two different images by comparing its keypoint descriptors, and the probability that the match is correct is greater when there is a large difference between the first- and second-keypoint matches. For identification purposes, it is more important to correctly distinguish the correct matches from the incorrect matches, rather than simply extract a large number of ambiguous matches. For this reason, a maximum threshold of 0.7 was chosen for the ratio of the distances between the first- and second- greatest matches for each keypoint descriptor. If a match between two keypoints falls within this range, then there is a good probability that the match is correct and therefore a 'good match.'

To translate these varying keypoints, descriptors, and corresponding matches into a binary yes/no output of whether the UAV is correctly located above its source location, a simple ratio between the number of extracted keypoints and the number of distinguished 'good matches' is used. Through experimental trails (see next Section 4: Experimental Results), any two photos with a ratio greater than 0.15 is considered as close enough to represent the same location. In this scenario, the UAV will confirm that its underlying terrain matches the original photo of its source location, and if the two photos are considered close enough, then the UAV will land at its source location.

4 Experimental Results

4.1 Dead Reckoning

The UAV flight path via dead reckoning for basic navigation was implemented using a Parrot AR 2.0 drone and a Raspberry Pi 3. The Raspberry Pi was attached to the top of the drone for autonomous navigation, as shown below in Figure 4.

Figure 4: Raspberry Pi 3 atop Parrot AR 2.0 drone

Due to the close proximity of the battery to the Raspberry Pi, the magnetometer's readings are influenced by magnetic distortion and result in large amounts of drift and instability. Future work should employ a battery that does not greatly affect nearby magnetic fields.

4.2 Estimation of Ground Speed

In physical implementations, three trials were applied to find the optimal specifications for minimizing the margin of error with the dead reckoning calculations, with a final estimation that lies within a 1 percent margin of error from its true ground speed. In this testing, a DJI Phantom 3 Pro UAV and a camera with a 94-degree field of view were used. Data was collected with the UAV flying at a speed of 3.1 mph (1.386 m/s) at an altitude of 65 ft (19.8 m).

First, two images taken 2 seconds apart were analyzed with the previous template matching method (square difference of norms), as shown in Figure 5. The Euclidean distance between the two template midpoints was calculated as 376.198 pixels, and the scale was calculated to 0.00849 meters/pixel. A simple multiplication of this distance and scale yields 1.598 meters per second, which when compared to the original 1.386 meters per second demonstrates a 15 percent margin of error.

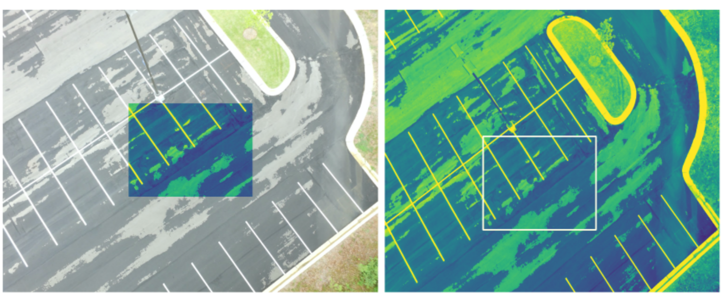


Figure 5: A template generated from the first photo taken by the UAV (left), and its corresponding match in the second photo taken 2 seconds later (right).

To rectify this error, two images that were taken a longer distance apart are applied, as the inherent margin of error with the template matching would be a smaller percentage of the overall pixel difference calculation. Shown in Figure 6, two images taken six seconds apart yields a template match that lies farther from the image center as compared to Figure 5. This calculation yielded a pixel difference of 911.76 pixels, a final estimation of 1.291 meters per second, and an overall margin of error of 6.8 percent.

In addition to optimizing this estimation with a longer distance between the images, this estimation is further improved by ensuring the template of the image contains an adequate number of unique features. Although the trial

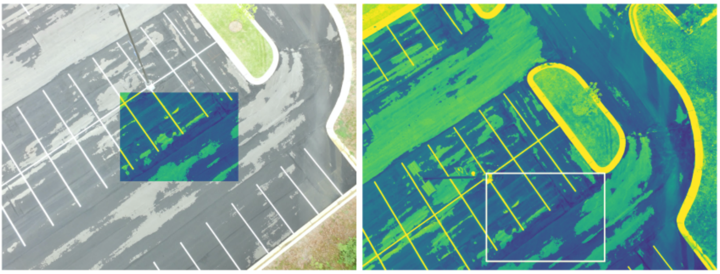


Figure 6: A template generated from the first photo taken by the UAV (left), and its corresponding match in the second photo taken 6 seconds later (right).

from Figure 6 matches the template correctly between photos, the match could be more precise if the template contained more unique features. To implement this hypothesis, the photos of the first and second trials were applied in reverse order, as the center of the second photo contains more unique features than the first; regardless of direction, the UAV maintains the same speed. In this scenario, the differences in the midpoints of the templates between the two images yielded a distance traveled of 989.117 pixels, and a final estimation of 1.386 meters per second. When compared to the original 1.386 meters per second of the recorded UAV flight, this represents a 1 percent margin of error. This trial is shown in Figure 7.

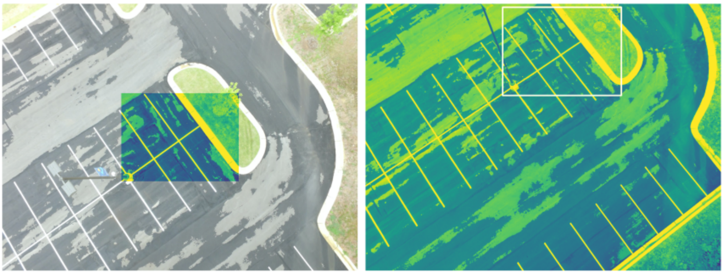


Figure 7: A template generated from the first photo taken by the UAV (left), and its corresponding match in the second photo with a 6 second time difference (right).

The previous data is aggregated into the table below within Figure 8.

trial	time between photos (s)	altitude (m)	calculated scale (m/pixel)	Euclidean distance moved (pixels)	Euclidean distance moved (meters)	estimated ground speed (m/s)	real ground speed (m/s)	margin of error
1	2	19.8	0.00849	376.198	3.197	**1.598**	**1.386**	0.15
2	6	19.8	0.00849	911.768	7.748	**1.291**	**1.386**	0.068
3	6	19.8	0.00849	989.117	8.406	**1.401**	**1.386**	0.0108

Figure 8: Aggregated data of the previous results into one table

4.3 Location Confirmation through Image Feature Matching

In this testing, a DJI Phantom 3 Pro UAV and a camera with a 94 degree field of view were used. Data was collected with the UAV flying at a speed of 3.1 mph (1.386 m/s) at an altitude of 65 ft (19.8 m).

Through experimental testing, images of differing locations yielded a deemed 'good matches' to extracted keypoint features ratio of less than 0.01, and images of the same location (with slight variances) yielded a matches to feature ratio of somewhere between 0.15 to 0.70. For simplicity purposes of determining whether two images with slight variances represent the same location, anything above a minimum threshold ratio of 0.15 was considered good enough for location confirmation, and anything less was considered as representing differing locations. Although this ratio may seem small, this is expected given the high degree of selectivity inherent in the definition of a 'good match,' setting the maximum threshold of best- and second-best matches at 0.7 within the SIFT algorithm for feature extraction. Setting a higher value for this maximum threshold would define more matches to be 'good matches', thereby increasing the ratio between the 'good matches' and the extracted features, but at the cost of more incorrect matches being classified as a 'good matches.'

In addition, although this minimum threshold of 0.15 is applicable to most situations, this can be further tailored to specific environments and camera specifications if desired. Areas with high numbers of contrast points, such as dense tree regions, result in lower ratios for images of the same location due to high numbers of extracted features within those areas, and ambiguity in matching is unlikely to produce a corresponding 'good match'; in these situations, the minimum threshold should be lowered to around 0.10 to account for this element. In addition, images with higher resolutions typically result in a greater number of extracted keypoints with a greater precision in each corresponding keypoint descriptor. This typically yields higher ratios of deemed 'good matches' to extracted keypoints. In these scenarios, the minimum threshold can be increased, although it is unlikely to make a significant difference in the final determination of whether the images represent the same location, as images of differing locations are unlikely to have ratios greater than 0.01 anyway.

For reference, Figures 9 and 10 represent two images of the same location with slight variances (the second image is taken 2 seconds later and has shifted slightly in orientation) correctly identifying its current location as its source location. As the ratio (0.3024 in this example) is greater than the minimum threshold of 0.15, then the UAV has confirmed its location and is safe to land. The green lines within Figure 10 represent the corresponding matches between keypoints, and the red circles represent each extracted keypoint.

Figure 9: Before feature extraction and matching with Figure 10

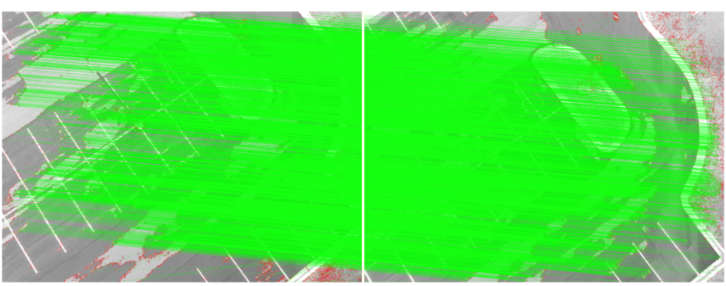


Figure 10: After feature extraction and matching: 9024 matches between the images were found to be 'good matches,' and a total of 29843 features were extracted. The final ratio of these two numbers was 0.3024 (0924/29843).

Additionally, Figures 11 and 12 represent two images of differing locations correctly recognizing that the two images represent different locations, that the current position is not its source location. As the ratio (0.0045) is less than the minimum threshold of 0.15, then the UAV can fly in a spiral pattern until the source location is found and confirmed.

Figure 11: Before feature extraction and matching with Figure 12

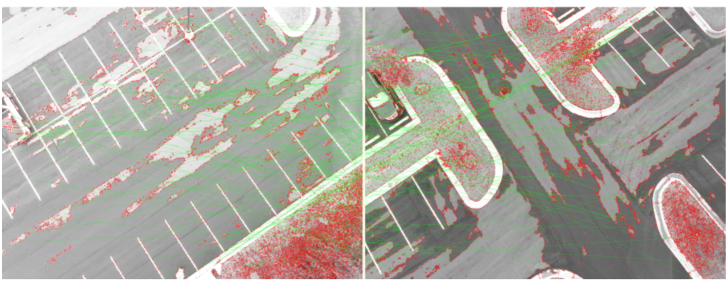


Figure 12: After feature extraction and matching: 184 matches between the images were found to be 'good matches,' and a total of 30646 matches were extracted. The final ratio of these two numbers was 0.0045 (184/40646).

5 Conclusion

In this work, computer vision methods for increasing the accuracy of the dead reckoning home-oriented navigation within GPS-denied environments are presented. Estimating the real ground speed of the UAV with two images can be calculated within a 1 percent margin of error, and factors that influence this margin of error (time between photos, uniqueness of template) are explored. Furthermore, location confirmation can be presumed through SIFT feature extraction and matching, and relative minimum thresholds for the ratio between these matches and features are explored for yes/no location confirmation.

Future work can include exploring calculations of ground speed through the slopes of the extracted feature matches between images (instead of relying upon image template matching), or explorations into estimations of drift that could also aid the GPS-denied dead reckoning process. Future work could also include the implementation of this system within real-time UAV navigation.

References

[1] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008. Similarity Matching in Computer Vision and Multimedia.

[2] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: Binary robust independent elementary features. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 778–792, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[3] Ebrahim Karami, Siva Prasad, and Mohamed Shehata. Image matching using SIFT, SURF, BRIEF and ORB: Performance comparison for distorted images, 2017.

[4] Stefan Leutenegger, Margarita Chli, and Y. Siegwart. brisk.

[5] Elena López, Sergio García, Rafael Barea, Luis M Bergasa, Eduardo J Molinos, Roberto Arroyo, Eduardo Romera, and Samuel Pardo. A multi-sensorial simultaneous localization and mapping (SLAM) system for low-cost micro aerial vehicles in GPS-denied environments. *Sensors*, 17(4), 2017.

[6] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, November 2004.

[7] Dr Mistry and Asim Banerjee. Comparison of feature detection and matching approaches: SIFT and SURF. *GRD Journals- Global Research and Development Journal for Engineering*, 2:7–13, 03 2017.

[8] M. Samy, K. Amer, M. Shaker, and M. ElHelw. Drone path-following in GPS-denied environments using convolutional networks, 2019.

[9] Andrew B. Simon, Manoranjan Majji, Carolina I. Restrepo, and Ronney Lovelace. *A Comparison of Feature Extraction Methods for Terrain Relative Navigation.*

[10] Randall Smith, Matthew Self, and Peter Cheeseman. Estimating uncertain spatial relationships in robotics. volume 1, pages 435–461, 01 1986.

[11] Jia-Ying Wang, Bing Luo, Ming Zeng, and Meng Hao. A wind estimation method with an unmanned rotorcraft for environmental monitoring tasks. *Sensors*, 18:4504, 12 2018.

Infusing Syntactic Knowledge into Deep Learning for Clinical Relationship Extraction*

Maxwell Wayne[1] *and Sam Henry*[2]
Department of Physics[1], *Computer Science and Engineering*[2]
Christopher Newport University
maxwn5@gmail.com
samuel.henry@cnu.edu

Abstract

Natural language processing (NLP) is a subset of computer science that is focused on enabling computers to understand human language. One core task of NLP is relationship extraction (RelEx), which aims to identify how concepts in a sentence are related. RelEx applied to medical documents enables computer systems to access information stored as narrative text. This facilitates more accurate search, retrieval, and analysis of medical records. There are two basic approaches to RelEx systems: context-based and syntax-based systems. Context-based systems use machine learning to find patterns of words that indicate relationships, whereas syntax-based systems use rules of English grammar to identify the relationships. These two approaches are complementary, and this paper explores methods to combine them.

1 Introduction

Natural language processing (NLP) is a subset of artificial intelligence that is focused on enabling computers to understand human language as it is spoken or written[3]. A core task of NLP is relationship extraction (RelEx). The goal of RelEx is to identify and label relationships between words in a sentence.

For example, the sentence "The patient was given Ibuprofen for a headache", describes several relationships between *the patient*, *ibuprofen*, and *a headache*. Among them that *ibuprofen was administered to treat a headache*.

Clinical NLP is a specialized domain of NLP, in which NLP is applied to clinical documents such as electronic health records (EHRs). The digitization of health records has become standard practice, and the vast majority of hospitals make use of EHRs[6]. EHRs support quality improvement, increase patient safety, and can be used to measure organization performance[17]. EHRs may also be used for clinical decision support systems[4]. A challenge of using EHRs is that most of the information contained within them takes the form of unstructured narrative text[14]. Thus, NLP techniques such as RelEx must be applied to extract information from them.

Most NLP systems are built in a pipelined approach, and RelEx is a low level task that enables upstream processing of text crucial for many different applications such as data mining[1], question answering[23], and automatic knowledge-base construction[10]. NLP has achieved amazing success, due in large part to the use of deep learning systems, which make use of Transformer architectures[21]. These systems identify relationships by generalizing patterns found in training data. Conceptually, the training data may contain the sentence: "Take Aspirin for a headache" labeled as containing a *treats* relationship. Based on this, the system may learn that the text pattern *for a* indicates a treats-type relationship. These systems use context (the text of the sentence) alone to identify relationships. Due to how they learn, these architectures, require large amounts of training data in the form of text annotated for a specific task. Within the clinical domain, annotated corpora are often unavailable due to patient privacy concerns, and because annotation requires expert knowledge and time[20]. Additionally, the structure of clinical records varies from institution to institution, so machine learning systems may struggle with adaptability[15].

An alternative to context-based systems are rule-based syntactic systems. These systems use knowledge about English grammar and sentence structure to identify relationships. They typically begin by identifying the part of speech (POS) (e.g. noun, verb) of the words in a sentence, then use English grammar rules to construct a dependency tree indicating the roles of each word in a sentence, and how they relate to one another. Because these systems are rule-based they do not require any training data, and they are more adaptable to new institutions. However, clinical records often lack typical grammatical structure[14], which makes their use in the clinical domain challenging.

These two approaches are therefore complementary. Context-based systems can find relationships in sentences which are grammatically unclear, but they require large amounts of training data representative of the target application.

Syntax-based systems require typical grammatical structure, but do not require training data. Therefore, in this work, we explore the effectiveness of incorporating syntactic-knowledge into a context-based system for clinical RelEx. We create three systems which incorporate varying degrees of knowledge and compare their results. Specifically, we create:

(1) A context-only system (context system), which incorporates no syntactic knowledge. (2) We found that the concept type information was particularly important for differentiating broad relationship types. Therefore, we created a concept-knowledge infused system (concept system), which incorporates information about the types of concepts present in a sentence. These concepts are problems, treatments, and tests, and are specific to the task we are completing. (3) We found that verbs and adverbs were particularly important for differentiating between fine-grained relationship types. Therefore, we created a concept and syntactic-knowledge infused system (syntax system), which incorporates information about both the types of concepts present in a sentence, and the verbs and adverbs in the sentence.

We evaluate performance of these systems on the i2b2 2010 dataset[20]. We find that the performance of the systems are statistically insignificant from one another, but they all achieve state of the art performance for the i2b2 2010 dataset.

2 Related Works

Relationship extraction systems may be either fully context-based, or fully rule-based. Models such as BlueBERT[18], BioClinicalBERT[2], and BioDischargeSummaryBERT[2] use a BERT-based language model and a single classification layer. They achieve excellent performance using context alone. Alternatively, SemRep[9] is a purely rule-based system that extracts relationships based on rules of English grammar. Developing methods that combine these two approaches is an emerging area of research [19]. Luo et al.[13] propose Segment Convolutional Neural networks (Seg-CNNs) which incorporate word order structure by using multiple convolution units to process specific segments of a sentence. Rather than learning a relation representation for an entire sentence, Seg-CNNs learn representations for sentence segments corresponding to the tokens before the first concept (preceding), the first concept (concept 1), tokens between the first and second concept (middle), the second concept (concept 2), and lastly tokens after the second concept (succeeding). Li et al.[11] developed a deep neural network that captures syntactic features to improve performance. Their architecture uses grammatical rules to generate a dependency tree and uses the shortest dependency path (SDP) within a sentence to extract relationships. The architecture has three modules. The first module uses a Bi-LSTM network for sentence sequence representation and capturing the features in the sentence sequence. The next module is the SDP representation module. This module implements a convolutional neural network (CNN) and Bi-LSTM network for capturing syntactic context of target entities using SDP information. The last module is the classification module, which utilizes a fully-connected layer to classify the relationships.

3 Methodology

We believe that deep learning systems can benefit by incorporating syntactic knowledge. To test this hypothesis, we create three systems which we describe below. All systems will take in a single sentence string as input and output an 8 dimensional binary vector where each index is mapped to a relationship defined by the i2b2 2010 dataset. A 1 indicates the relationship represented by that index is present and a 0 indicates that relationship is absent from the sentence. The code to replicate the experiment can be accessed on GitHub. [1]

3.1 Context System

The baseline context-only system (Context System) is similar to most state-of-the-art BERT-based systems. A raw sentence string is input into the system and passed into a BERT language model[5]. The language model outputs a single context vector (the CLS token embedding), which numerically represents the linguistic meaning of the sentence. This vector is passed into a classifier, which outputs a relationship vector indicating which relationships are present in the sentence. In our system, we construct the classifier portion of the system using four densely connected layers consisting of 256, 128, 64, and 8 neurons respectively. Each layer uses a GeLu activation function except for the last layer which uses a sigmoid activation function. Dropout of 0.2 is used between each of the classifier layers. We use the same classifier architecture for all of our systems with the exception of the input dimensionality which we increase to accommodate additional information.

3.2 Concept System

In our analysis of the results of the context system, we found a surprising number of sentences classified as containing relationships between concept types it did not contain. For example, if a sentence does not contain a *Test* concept type, then it cannot contain the relations *Test reveals Problem (TeRP)* or *Test given to investigate Problem (TeCP)*. Therefore, we create the Concept

[1] `https://github.com/nlp-cnu/i2b2_relex_with_syntax`

System, which incorporates information about the concept types present in a sentence. The identification of concepts and their type is a separate NLP task called Named Entity Recognition (NER). NER is typically performed prior to relationship extraction and therefore, obtaining the concept type information is possible (although potentially imperfect) for most RelEx systems. In our implementation we use the gold-standard concept information in the i2b2 2010 dataset. However, in practice a separate NER module could be used to provide this knowledge. To add the concept knowledge, we concatenate a three dimensional binary vector to the CLS token embedding output by the language model prior to being input into the classifier portion of the system. The first, second, and third indexes of the knowledge vector indicates if a treatment, test, or problem respectively are contained in the sentence. The theory is that this vector encodes knowledge to help the system determine if treatment-problem, test-problem, or problem-problem relationships are possible in the sentence.

3.3 Syntax System

We analyzed the results of our Concept System and found that there was still confusion among classes, but the confusion was now mainly between classes sharing concept-type pairs. For example, two sentences may contain *Treatment* and *Problem* concepts, but there was confusion between *Treatment Improves Problem* (TrIP) and *Treatment Worsens Problem* (TrWP) classes. In these cases, we found that adverbs and verbs were often key indicators of the correct class. For example, the verbs "worsened" or "improved" and the adverbs "positvely" and "negatively" indicate the outcome of the treatment. With this in mind, we created our Syntax System, which infuses knowledge about the concepts, verbs, and adverbs present in a sentence.

Figure 1 illustrates the architecture of this system. The architecture consists of two independent language models. Language Model 1 is identical to Context System and is responsible for generating a vector that models the meaning of the sentence as a whole. An entire sentence is input and the CLS token embedding is taken as its output. Language Model 2 is responsible for generating five different vectors, each modeling the meaning of all the verbs, adverbs, Tests, Treatments, and Problems present in the sentence. Each of these vectors is created sequentially. First the sentence is passed through the Natural Language Toolkit (NLTK)[12] part of speech (POS) tagger to identify the verbs and adverbs in a sentence. The verbs in a sentence are combined into a space-separated text which is passed through the language model, and the CLS token embedding is taken to model the meaning of the verbs in the sentence. Similarly, the adverbs are combined into a space-separated text, which is passed through the language model, and the CLS token embedding is taken to model the meaning of the adverbs in the sentence. The Problem, Treatment, and Test vectors are generated in a similar manner, except the terms are collected using gold standard concept annotations from the dataset. In practice, they would be identified using an NER system. If no verbs, adverbs, or concepts of a specific type exist in the sentence, then an empty string is passed through the language model in their place. Due to memory constraints, back-propagation through language model 2 is disabled.

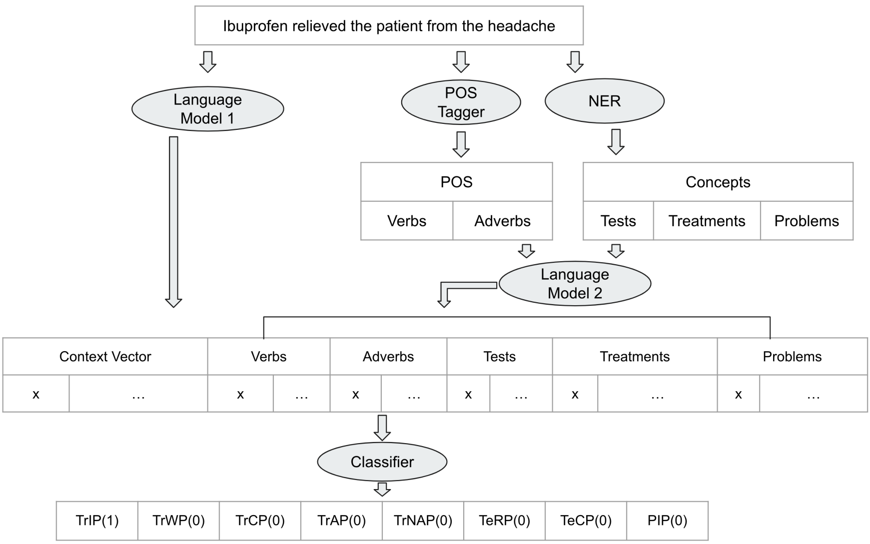


Figure 1: Syntax System which incorporates information about Tests, Treatments, Problems, verbs, and adverbs

In Figure 1, the sentence "ibuprofen relieved the patient from the headache", is passed into the first language model. The verb "relieved" is extracted via the POS tagger, and the concepts "ibuprofen" and "headache" are retrieved via NER. The second language model will then take in five inputs: "relieved" representing the verbs, an empty string representing the adverbs, an empty string representing tests, "Ibuprofen" representing the treatments, and lastly "headache" representing the problems. All outputs from both language models are concatenated, then input into the classifier. The classifier then predicts that the TrIP relationship is present in the sentence.

4 Results and Conclusions

All systems were implemented in Python using TensorFlow architecture and Keras API. We performed a hyperparameter search using combinations of the following hyperparameters: back propagation through the language model enabled and disabled; four language models: BaseBERT[5], BlueBERT[18], BioClinicalBERT[2], and BioDischargeSummaryBERT[2]; and seven learning rates from 1e-1 through 1e-6 inclusive in increments of 1e-1. We found that BlueBERT[18] with a learning rate of 1e-5 with the language model trainable enabled were the best hyperparameters for all systems.

We evaluate each system using the 2010 i2b2/VA challenge for clinical relationship extraction[20] (i2b2 2010 dataset). The i2b2 2010 dataset contains discharge summaries annotated for the presence of medical problems, treatments, and tests, and relationships between them. There are a total of 8 relationship types between these concepts grouped into three broad types. Treatment to problem relationships which include the types: Treatment improves or cures medical problem (TrIP), Treatment worsens or does not improve medical problem (TrWP), Treatment caused medical problem (TrCP), Treatment administered for medical problem, but outcome is not mentioned in the sentence (TrAP), and Treatment is not administered or discontinued because of a medical problem (TrNAP). Test to problem relationships which include the types: Test reveals a medical problem (TeRP), and Test given to investigate a medical problem (TeCP). Lastly, problem to problem relationships which include a single type: Medical problem indicates or reveals aspects of another medical problem (PIP). The i2b2 2010 dataset is split between test and training datasets, and originally contained 340 training documents and 256 test documents. However, since its initial release, the University of Pittsburgh Medical Center records have been removed, so it now contains 170 training documents and 256 test documents. Since the training set is smaller than the test set, and because the original dataset contained more samples, Wei et al.[22], combine all the documents into a single pool and perform their own train-test split. Other researchers report results using the original train-test split, and still other researchers report results with the reduced size train-test split. Due to this discrepancy, comparison between researchers can be difficult. We follow the work of Wei et al, and combine all data into a single pool, reserve 80% of the samples for training and 20% of the samples for testing. The samples reserved for training are further split, and 80% are used for training and 20% are used for validation. We withhold the test set from model training and hyperparameter tuning, and only use it to evaluate final model performance. Due to the large number of sentences which contain no relations, we remove sentences that contain less than two concepts in both the test and training sets.

	Context System F1	Concept System F1	Syntax System F1
Micro	0.81	0.81	0.80
Macro	0.72	0.69	0.69
TrIP	0.68	0.62	0.59
TrWP	0.51	0.46	0.55
TrCP	0.78	0.77	0.75
TrAP	0.86	0.86	0.86
TrNAP	0.69	0.61	0.59
TeRP	0.87	0.87	0.87
TeCP	0.57	0.60	0.56
PIP	0.77	0.77	0.75

Table 1: Test set metrics by class with optimal parameters

Table 1 summarizes the results of each of our systems. It shows the micro and macro averaged F1 as well as the F1 scores per class. Each of the systems performed similarly. Keeping in line with other research[7], we used approximate randomization[16] to determine if the differences in results were statistically significant. We used a significance threshold of 0.05 and 50,000 shuffles and found that the micro and macro F1 scores of each system were not statistically significantly different. Therefore, we conclude that adding knowledge neither increased or decreased performance over the Context System.

Despite this, our system with a micro-F1 score of 0.81 performed remarkably well. Wei et al [22] and Ji et al [8] used a similar train-test split. Wei et al achieved a micro-F1 of 0.7679 using their context-only deep learning system, and Ji et al achieved a micro-F1 of 0.4336 using their joint entity and relation extraction system. Li et al[11] combine the test and training data into a single set and use 5-fold cross validation to report an average validation micro-F1 of 0.7434 using their system which integrates the shortest dependency path.

Additionally, we wanted to compare our results to systems that use the original, but reduced size training-test split. So, we retrained our context system using the only officially provided reduced size training set and evaluated its performance on the officially provided reduced-size test set. With this setup, we achieved a micro-F1 of 0.75, and a macro-F1 of 0.57. This performs similar to Luo et al [13] who report a micro-F1 of 0.7420 with their Segment-CNN, and similar to Peng et al [18] who report a micro-F1 of 0.764 using BlueBERT.

In the future we hope to develop more sophisticated systems that incorporate syntactic knowledge. Ideas include incorporating an attention mechanism into the system over the contextualized word embeddings or over the concept and part of speech embeddings, or both. Additionally, it would also be interesting to develop an end-to-end system that handles both NER and RelEx. The systems in this paper assume flawless NER, and it would be interesting to see how errors in NER propogate through the RelEx system.

References

[1] Charu C Aggarwal and ChengXiang Zhai. *Mining text data.* Springer Science & Business Media, 2012.

[2] Emily Alsentzer, John R. Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew B. A. McDermott. Publicly available clinical BERT embeddings. *arXiv preprint arXiv:1904.03323 [cs.CL]*, 2019.

[3] Gobinda G. Chowdhury. Natural language processing. *Annual Review of Information Science and Technology*, 37:51–89, Jan 2003.

[4] Dina Demner-Fushman, Wendy W. Chapman, and Clement J. McDonald. What can natural language processing do for clinical decision support? *Journal of Biomedical Informatics*, 42:760–772, Aug 2009.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805 [cs.CL]*, 2018.

[6] J Henry, Yuriy Pylypchuk, Talisha Searcy, and Vaishali Patel. Adoption of electronic health record systems among US non-federal acute care hospitals: 2008–2015. *ONC Data Brief*, 35:1–9, 2016.

[7] Sam Henry, Kevin Buchan, Michele Filannino, Amber Stubbs, and Ozlem Uzuner. 2018 n2c2 shared task on adverse drug events and medication extraction in electronic health records. *Journal of the American Medical Informatics Association*, 27(1):3–12, 10 2020.

[8] Zongcheng Ji, Omid Ghiasvand, Stephen Wu, and Hua Xu. A discrete joint model for entity and relation extraction from clinical notes. *AMIA Summits on Translational Science Proceedings*, volume 2021:page 315, 2021.

[9] Halil Kilicoglu, Graciela Rosemblat, Marcelo Fiszman, and Dongwook Shin. Broad-coverage biomedical relation extraction with SemRep. *BMC Bioinformatics*, 21, 188(1):1–28, 2020.

[10] Halil Kilicoglu, Dongwook Shin, Marcelo Fiszman, Graciela Rosemblat, and Thomas C Rindflesch. SemMedDB: a PubMed-scale repository

of biomedical semantic predications. *Bioinformatics*, 28(23):3158–3160, 2012.

[11] Zhiheng Li, Zhihao Yang, Chen Shen, Jun Xu, Yaoyun Zhang, and Hua Xu. Integrating shortest dependency path and sentence sequence into a deep learning framework for relation extraction in clinical text. *BMC Medical Informatics and Decision Making*, 19, 22(1):1–8, 2019.

[12] Edward Loper and Steven Bird. NLTK: The natural language toolkit. *arXiv preprint arXiv:cs/0205028 [cs.CL]*, 2002.

[13] Yuan Luo, Yu Cheng, Özlem Uzuner, Peter Szolovits, and Justin Starren. Segment convolutional neural networks (Seg-CNNs) for classifying relations in clinical notes. *Journal of the American Medical Informatics Association*, 25(1):93–98, 2017.

[14] S. M. Meystre, G. K. Savova, K. C. Kipper-Schuler, and J. F. Hurdle. Extracting information from textual documents in the electronic health record: a review of recent research. *Yearbook of Medical Informatics*, pages 128–144, 2009. `https://pubmed.ncbi.nlm.nih.gov/18660887/`.

[15] Prakash M Nadkarni, Lucila Ohno-Machado, and Wendy W Chapman. Natural language processing: an introduction. *Journal of the American Medical Informatics Association*, 18(5):544–551, 09 2011.

[16] Eric W Noreen. *Computer-intensive methods for testing hypotheses*. Wiley New York, 1989.

[17] Sonal Parasrampuria and Jawanna Henry. Hospitals' use of electronic health records data, 2015-2017. *ONC Data Brief*, 46:1–13, 2019.

[18] Yifan Peng, Shankai Yan, and Zhiyong Lu. Transfer learning in biomedical natural language processing: An evaluation of BERT and ELMo on ten benchmarking datasets. *arXiv preprint arXiv:1906.05474 [cs.CL]*, 2019.

[19] Amit Sheth, Manas Gaur, Ugur Kursuncu, and Ruwan Wickramarachchi. Shades of knowledge-infused learning for enhancing deep learning. *IEEE Internet Computing*, 23(6):54–63, 2019.

[20] Özlem Uzuner, Brett R South, Shuying Shen, and Scott L DuVall. 2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association*, 18(5):552–556, 2011.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[22] Qiang Wei, Zongcheng Ji, Yuqi Si, Jingcheng Du, Jingqi Wang, Firat Tiryaki, Stephen Wu, Cui Tao, Kirk Roberts, Hua Xu, and et al. Relation extraction from clinical narratives using pre-trained language models. In *AMIA annual symposium proceedings*, pages 1236—-1245. American Medical Informatics Association, 2019. `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7153059/`.

[23] Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. Question answering on freebase via relation extraction and textual evidence. *arXiv preprint arXiv:1603.00957 [cs.CL]*, 2016.

Course Delivery Preferences of Students Post-COVID 2022*

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

Abstract

The COVID-19 world pandemic moved many activities online. With many activities now having returned to normal, we seek insight into student course and career modality preferences. Given their forced experiences with online courses and online work, would students want work and/or class to continue online? We surveyed 12 classes at a large public university and got 186 responses. For course work, 48% of students preferred online, 38% preferred hybrid, and 14% preferred face-to-face. In contrast, for work, 62% preferred hybrid work mode, with 21% preferring online, and 17% preferring face-to-face. Students at this large university seem to prefer efficiency and flexibility when it came to course work and did not view the lack of social connection as a significant disadvantage to online courses.

1 Introduction

The global pandemic of COVID-19 temporarily changed many things about the way we interact with others. In many states, face-to-face schooling was cancelled. Many companies had their employees work remotely from home. Previously in person experiences, such as religious worship and doctor visits, moved to online video formats. This widespread adoption of video meetings and online school gave many a first-hand experience of remote learning and working. With many activities back to a face-to-face format, we seek insight into student's preferred learning and working modalities going forward.

2 Literature Review

Since online learning has been around for some time, there are many studies regarding different learning modalities. In particular, combining elements of online learning and face-to-face learning has shown to benefit students. Hybrid courses fill the requirements lacking in online courses, specifically a high level of teaching presence[2]. At the same time, hybrid courses retain the main benefit of online courses, which is time flexibility for the student[1]. The existence of hybrid courses has made a difference in graduation rates. For some student groups, hybrid courses have been a key factor in the completing their education rather than dropping out[7]. Lacking the in-person reminders of face-to-face courses, time management skills and discipline are needed to be successful in hybrid courses[4].

The preference for a hybrid online/in-person environment also exists in the workforce. A survey by Boston Consulting Group found most Australian workers want to work in the office two to three days a week. They like the flexibility to choose when to work. The oldest group of workers (60 and older) favored working from home between 81-100% of the time. Younger employees (18-30) were most enthusiastic about working in the office (66%). Three reasons people want to return to the office are: informal social interaction, formal collaboration, and better work environment than home[8].

A survey conducted by Envoy, a workplace platform that helps teams manage hybrid work, found that 70% of employees would choose a hybrid work model, 16% would prefer to work in the office full-time, and only 12% would prefer to work fully remote. Of those who prefer hybrid, about half would like to work on-site most of the week and the other half would like to work mostly remote. They want flexibility. The top four concerns about returning to the office were exposure to COVID and interaction with non-vaccinated co-workers, along with commute time and costs, and limited time flexibility. Advantages to working at the office were being able to separate home from work life, collaborating in-person, and seeing work friends[3].

3 Methodology

We created an online student survey with eight questions. The first three questions dealt with student school and work modality preferences. The next five questions asked students to rate advantages and disadvantages of taking face-to-face classes, advantages and disadvantages of taking online courses, and then finally the advantages of taking hybrid courses. The survey was sent out to approximately 350 students across 12 courses. The survey went out in April 2022 and the recipients were Information Systems and Information Technology

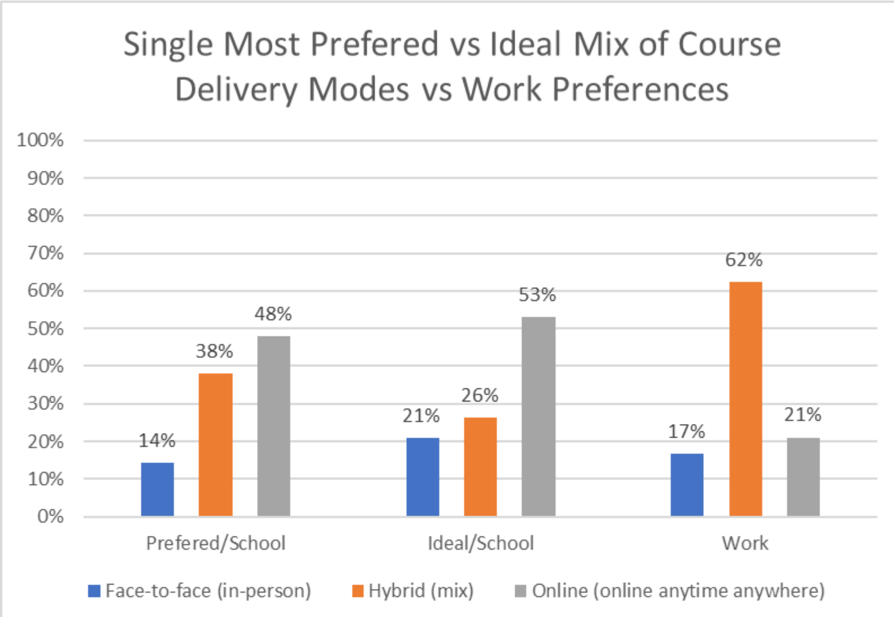


Figure 1: Preference of course delivery modes vs work modes.

majors at Utah Valley University. There was a total of 186 respondents.

4 Results

When asked for an overall single preference of a course delivery model, most students (48%) prefer online courses, 38% prefer hybrid courses (mix of one face-to-face class per week combined with online learning), and only 14% prefer face-to-face in-person at school courses (as shown in Figure 1).

We also asked students their ideal mix. The ideal percentage mix of course delivery modes was 53% online, 26% hybrid, and 21% traditional face-to-face. This is like the online single preference above at about 50%, however the ideal mix has hybrid and traditional face-to-face closer to each other at 26% and 21% versus 38% and 14% in the single preference (as shown in Figure 1).

We also asked about work mode preferences to compare to school preferences. Most students' single overall preference of a work mode is a hybrid mix of some days at home and some at work (62%), with 21% preferring online work and 17% preferring in-person work (as shown in Figure 1). These findings are interesting in the context of the Boston Consulting Group finding

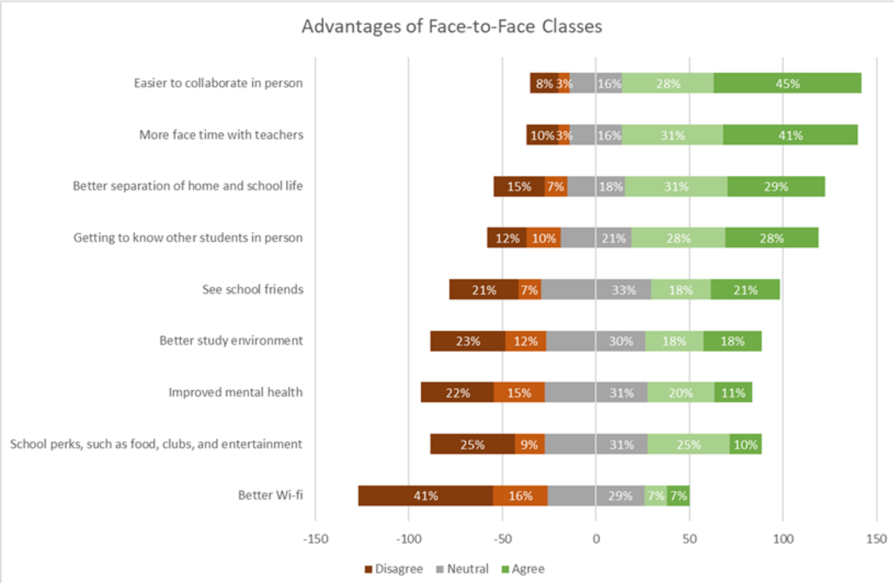


Figure 2: Advantages of face-to-face classes.

that younger people prefer to work in the office about 66% of the time. More research is needed to determine if worker preferences are shifting as a result of the pandemic or if there is something specific to our dataset.

Next, we asked students about the advantages and disadvantages of traditional face-to-face classes. Student ratings of advantages of traditional face-to-face classes are seen in Figure 2 below.

The biggest advantages of traditional classes were easier to collaborate in-person (73% agreed or somewhat agreed), more facetime with teachers (72% agreed or somewhat agreed), better separation of home and school life (60% agreed or somewhat agreed), and getting to know other students in person (56% agreed or somewhat agreed).

The secondary group of advantages with which students agreed or somewhat agreed were see school friends (39%), better study environment (36%), school perks such as food, clubs, and entertainment (35%), and improved mental health (31%), with better Wi-Fi coming in last at 14%.

Student ratings of disadvantages of traditional face-to-face classes are seen in figure 3 below.

When looking at the disadvantages of traditional face-to-face classes students thought that the main disadvantages were little choice in when you go

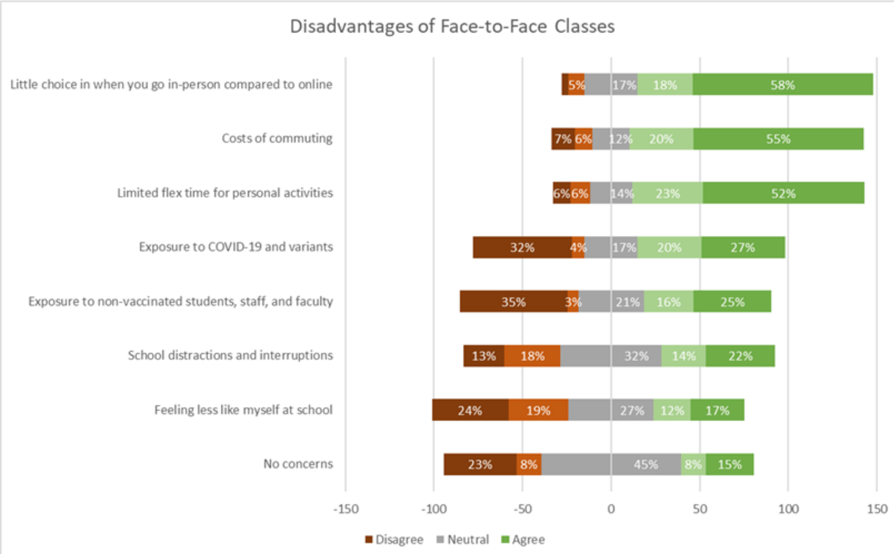


Figure 3: Disadvantages of face-to-face classes.

in-person compared to online (76% agreed or somewhat agreed), costs of commuting (75% agreed or somewhat agreed), and limited flex-time for personal activities (75% agreed or somewhat agreed).

The secondary group of disadvantages with which students agreed or somewhat agreed were exposure of COVID-19 and variants (47%), exposure to non-vaccinated students/staff/faculty (41%), school distractions and interruptions (36%), feeling less like myself at school (29%), and lastly no concerns (23%).

Next, we asked students about the advantages and disadvantages of online classes. Student rating of advantages of online classes are seen in Figure 4 below.

Overall, students rated online classes more positively than traditional face-to-face classes. The main advantages of online classes were flexibility to maximize time (94% agreed or somewhat agreed), reduced costs of commuting (87% agreed or somewhat agreed), less time wasted on tangents and repeated questions (71% agreed or somewhat agreed), easier to register for the course (63% agreed or somewhat agreed), course material is more focused on what is important (59% agreed or somewhat agreed), and finally course expectations are clearer (53% agreed or somewhat agreed). The only advantage listed with below 50% agreement was learning more online than in person (44% agreed or somewhat agreed).

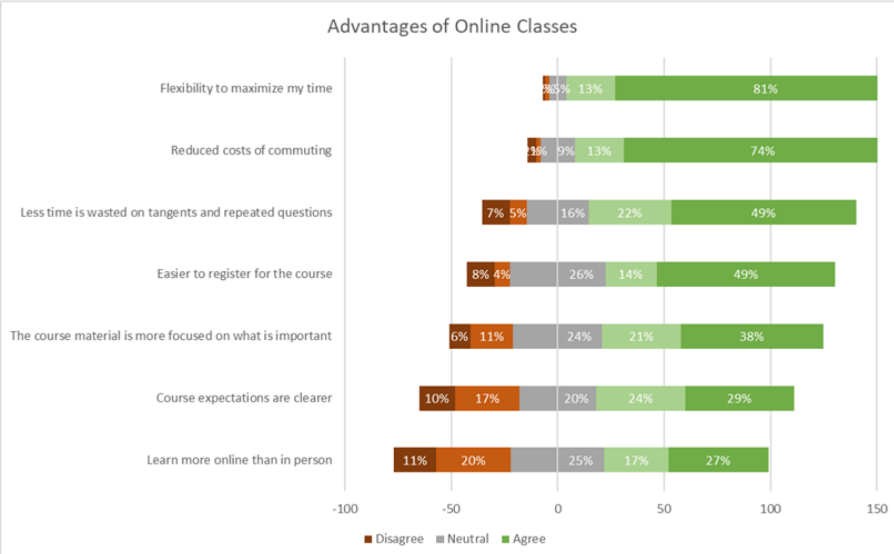


Figure 4: Advantages of online classes.

Student ratings of disadvantages of online classes are seen in Figure 5 below.

Overall, students rated the disadvantages of online classes more negatively than the disadvantages of traditional face-to-face classes. The main disadvantages of online classes were harder to collaborate compared to in-person (63% agreed or somewhat agreed), and less face time with teachers (57% agreed or somewhat agreed).

A secondary group of disadvantages with which students agreed or somewhat agreed were less separation of home and school life (42%), miss getting to know student in person (36%), don't see friends (28%), decreased mental health (24%), miss school perks such as food/clubs/entertainment (24%), and worse study environment compared to school (16%).

Finally, we asked students about the advantages of hybrid classes. Student rating of advantages of hybrid classes are seen in Figure 6 below.

Similar to online class ratings, students rated hybrid classes more positively than traditional face-to-face classes. The main advantages of hybrid classes were reduced costs of commuting (78% agreed or somewhat agreed), flexibility to maximize my time (73% agreed or somewhat agreed), face time with teachers (65% agreed or somewhat agreed), and easier to collaborate in person (64% agreed or somewhat agreed).

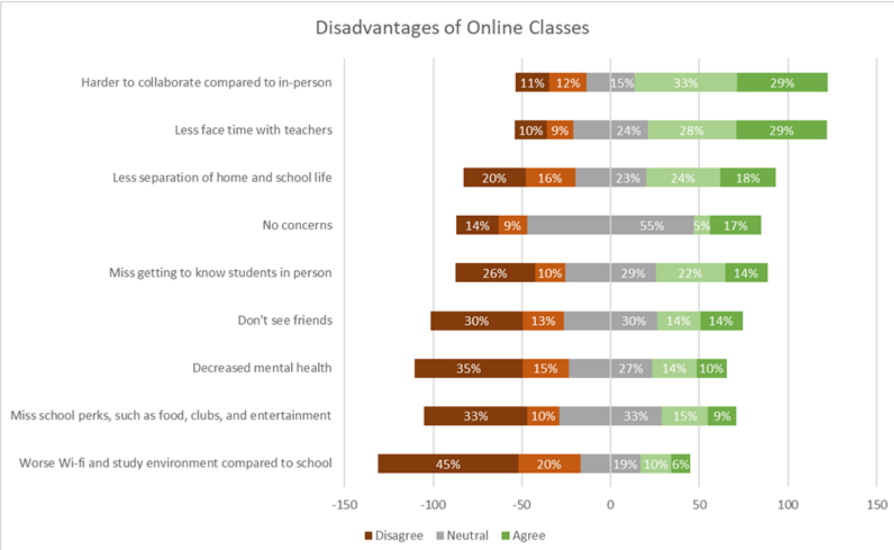


Figure 5: Disadvantages of online classes.

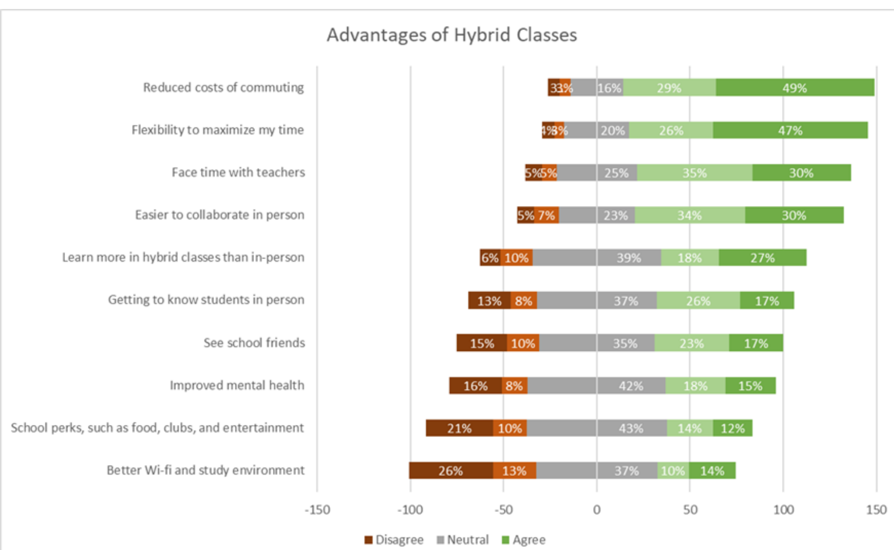


Figure 6: Advantages of hybrid classes.

A secondary group of advantages for hybrid classes with which students agreed or somewhat agreed were learn more in hybrid than in person (43%), getting to know students in person (43%), see school friends (40%), improved mental health (33%), school perks (25%), and a better study environment (24%).

5 Discussion

Authors were surprised that the most preferred course delivery model was online at 48% and the least was traditional face-to-face at 14%. It appears students want a change of delivery method. A student preference for a hybrid modality was strong at 38%. Currently UVU offers 29.5% of its courses online, 50% of its courses face-to-face, and 20.5% of its courses in a hybrid modality [6]. Students want to see about an equal percent increase in online and hybrid classes and a corresponding decrease in face-to-face classes.

It is interesting to note that students prefer hybrid work modes, but online classes. In 2020, UVU reported that 81% of students are employed [5]. With such a high employment rate, it is easy to see why students prefer the flexibility of online classes. Perhaps, as well, employee interactions at work help meet students need for social connection, which mitigate a major disadvantage of online classes.

Students at UVU are non-traditional. Most students work (81%) and many are married (38%) with children (17%)[5]. The reasons students prefer online course were flexibility to maximize time, reduced costs of commuting, less time wasted on tangents and repeated questions, easier to register for the course, course material is more focused on what is important, and finally course expectations are clearer. The advantages of online classes point to efficiency and ease of use.

The main disadvantages of online courses (harder to collaborate compared to in-person and less face time with teachers) seem to be remedied with hybrid courses main advantages (face time with teachers and easier to collaborate in person). Hybrid also incorporates the efficiency advantages of online courses (flexibility to maximize my time and reduced costs of commuting).

In addition to the non-traditional nature of the students, UVU does not have on-campus housing. All students have at least some commute to campus from home or work to take classes. The findings of this study should be interpreted with this in mind. The findings may be different at a more traditional campus where a much larger percentage of students live and work (to the extent that they work) on or near campus. Additional research should examine the generalizability of our findings for universities with a large in-residence student population.

In our survey, given to students at a largely commuter school, the social aspects of course delivery are not compelling. It could be that students' time is spread across more areas including work and family that help meet their social and mental health needs. The mental health aspects of course delivery is inconclusive. Also, most students disagree that face-to-face classes have a health risk disadvantage. Finally, most students disagree that school has a better study environment.

6 Conclusions

In a world where in the past few years we experienced many new activities online, it appears students want to continue having largely online classes, while transitioning work and careers to a hybrid mode. Students are juggling work, school, and families. Given their mix of activities, students appreciate efficiency and ease of use when it comes to higher education. Many of students' social needs appear to be met in other areas of their lives. Also, given the relative lack of online courses, the strong preference for online courses may be an effort to move the needle closer to the preferred student mix.

References

[1] J. Barker. Benefits of hybrid classes in community colleges. *Contemporary Issues in Education Research*, 8(3):143–146, 2015.

[2] C. Crawford and C. Persaud. Community colleges online. *Journal of College Teaching & Learning*, 10(1), 2013.

[3] Envoy. New workplace study reveals growing health concerns, preference for hybrid work among U.S. employees who have returned to the office, 2021. `https://www.businesswire.com/news/home/20210826005442/en/`.

[4] S. Lei and R. Gupta. College distance education courses: Evaluating benefits and costs from institutional, faculty and students' perspectives. *Education*, 130(4), 2010.

[5] Utah Valley University. Utah valley university is still largest public university in the state with nearly 41,000 students, 2020. `https://www.uvu.edu/news/2020/10/10072020_enrollment.html`.

[6] Utah Valley University. Utah valley university fall 2021 enrollment tops 41,000 students, 2021. `https://www.uvu.edu/news/2021/10/2021_10_26_fall_enrollment.html`.

[7] E. Yudko, R. Hirokawa, and R. Chi. Attitudes, beliefs, and attendance in a hybrid course. *Computers & Education*, 50(4):1217–1227, 2008.

[8] Daniel Ziffer. Most workers want 'hybrid' jobs at the office and at home after coronavirus, study finds, 2020. `https://www.abc.net.au/news/2020-06-23/most-workers-want-hybrid-of-home-and-office-post-coronavirus/12381318`.