

Learning and Education of Machine Learning for Undergraduate Robotics

A Major Qualifying Project (MQP) Report
Submitted to the Faculty of
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements
for the Degree of Bachelor of Science in

Robotics Engineering,
Computer Science

By:

Ashe Andrews
Andrew McKeen
Tuomas Pyorre

Project Advisors:

Prof. Matthew Ahrens, Computer Science
Prof. Kevin Leahy, Robotics Engineering
Prof. Gregory Lewin, Robotics Engineering

Date: April 2024

This report represents work of WPI undergraduate students submitted to the faculty as evidence of a degree requirement. WPI routinely publishes these reports on its website without editorial or peer review. For more information about the projects program at WPI, see <http://www.wpi.edu/Academics/Projects>.

Abstract

Despite an undergraduate curriculum focused on in-demand robotics topics, Worcester Polytechnic Institute (WPI) does not offer sufficient courses for students to learn about machine learning (ML) in relation to robotics. We developed educational modules where students apply ML theory, including transfer learning, optimization techniques, and reinforcement learning to robotic systems to develop potential course material. The modules were modified into workshops for testing with students. We evaluated the modules' effectiveness by comparing students' perceived confidence and ability in select topics before and after completing a workshop. After completion, students felt more confident in their understanding of higher-level concepts in ML, but still required support to apply them.

Acknowledgements

We would like to thank Professors Ahrens, Leahy, and Lewin for all their guidance and insight for this project. We would also like to thank all the professors who met with us to help us define our problem and all of the students who participated in our workshops to test our solution.

Contents

1	Introduction	1
2	Background	3
2.1	Overview of Machine Learning	3
2.1.1	Deep Learning	3
2.1.2	Machine Learning Optimization	4
2.1.3	Reinforcement Learning	4
2.2	Problem	5
2.2.1	A Growing Field	5
2.2.2	What Professors Say	5
2.2.3	What WPI Offers	6
2.3	Solution	7
2.3.1	Justification and Related Works	7
2.3.2	Machine Learning Techniques	8
2.3.3	Final Product	9
3	Methods	10
3.1	Professor Interviews	10
3.2	Choosing Material to Teach	12
3.3	Module Development	12
3.4	Hardware Testing	14
3.5	Student Workshops	15
4	Findings	17
4.1	Professor Interviews	17
4.2	Student Workshops	18
4.2.1	Introduction to PyTorch and Transfer Learning	18
4.2.2	Introduction to Optimization	19
4.2.3	Introduction to Reinforcement Learning	22
4.2.4	Student Interests in the Field of Machine Learning	23
5	Future Work	24
5.1	Final Project	24
5.2	Improvement on Materials	24
6	Conclusion	25
	References	27
	Appendices	30
A	Project Deliverable	30
B	Professor Interview Questions	31
B.1	The Claims	31

B.2	Questions	31
B.3	ML Questions	32
B.4	CV Questions	32
B.5	Module - ML and CV Synthesis	33
	B.5.1 Assumed Background	33
	B.5.2 The Deliverable	33
	B.5.3 Student Skills Upon Completion	33
B.6	Questions About the Module	34
C	Workshop Posters	35
D	Workshop Surveys	37

List of Tables

1	Amount of assistance participants felt they would need to complete a project using PyTorch before and after the workshop. All seven participants completed the pre-survey, while only five completed the post-survey.	19
2	Participants' comfort in explaining quantization and pruning to a peer. 0 indicates telling the peer to ask a professor, 1 indicates showing the peer resources and documentation, and 2 indicates trying to explain the topic in their own words.	20
3	Participants' response when asked if they could explain how pruning and quantization impact the size and performance of an ML model. The response options were "No", "Maybe", and "Yes".	20
4	Number of resources needed by participants to complete a project using quantization before and after our workshop. Five resources were listed and included Stack Overflow, tutorials, professors assistance, peer assistance, and documentation.	21
5	Number of resources needed by participants to complete a project using pruning before and after our workshop. Five resources were listed and included Stack Overflow, tutorials, professors assistance, peer assistance, and documentation.	21
6	Participants' comfort in explaining Q-learning and reinforcement learning to a peer. 0 indicates telling the peer to ask a professor, 1 indicates showing the peer resources and documentation, and 2 indicates trying to explain the topic in their own words.	22
7	Number of resources needed by participants to complete a project using Q-learning before and after our workshop. Five resources were listed and included Stack Overflow, tutorials, professors assistance, peer assistance, and documentation.	23
8	Number of resources needed by participants to complete a project using reinforcement learning before and after our workshop. Five resources were listed and included Stack Overflow, tutorials, professors assistance, peer assistance, and documentation.	23

List of Figures

1	The configuration of the Romi, Raspberry Pi, and camera.	10
2	Dependency graph of skills in machine learning and computer vision, which provided a basis for grouping related skills into modules and determining an order such that each module has the appropriate foundation.	13
3	Small lecture presented at the Introduction to Optimization workshop.	17

1 Introduction

The Worcester Polytechnic Institute (WPI) Robotics Engineering undergraduate program gives students an introduction to topics across the field of robotics with several interdisciplinary tie-ins. The curriculum is intended to teach students skills that are in demand in industry, prepare them to think like engineers, create innovative and effective solutions, and introduce them to a variety of topics that the students can specialize in [1]. Both academia and industry are increasingly looking for graduates with skills in artificial intelligence (AI) and machine learning (ML). These skills include using common ML frameworks such as PyTorch or TensorFlow, debugging large ML models, and using GPUs for developing and training models. Additionally, other universities are offering a broad range of courses in ML, including ones that apply ML models to robotic platforms [2].

While current WPI courses in AI and ML teach important theoretical foundations, they do not sufficiently teach necessary real-world skills applicable to robotics. Additionally, these courses are mainly offered at the graduate level. By contrast, the undergraduate program has only one course in ML that is offered by the Computer Science department and an introductory AI course offered by both the Robotics Engineering and Computer Science departments. At the undergraduate level, usage of common ML frameworks or developing an understanding of which models to use for a given task are not covered. While undergraduate robotics courses address the complications posed by robotic systems, such as sensor noise and limited computational resources, neither these courses nor the ML courses adequately address the impact of these complications on applying ML to robotic systems. As such, undergraduate robotics students at WPI have insufficient skills in ML to apply the concepts to real-world robotics problems.

To fill this gap, we present a course that covers necessary topics in the intersection of robotics and ML. The objective of the project is to build modules that are able to teach the students the following skills:

- Integrate an ML model with a robotic system such that the model impacts the decision-making of the robot
- Mitigate the impacts of hardware limitations from a robotic system on the performance of an ML model and vice versa
- Use different types of hardware such as GPUs and microcontrollers for training and deploying ML models
- Determine whether ML is an appropriate solution to a technical problem

To meet these objectives, we developed an undergraduate-level course curriculum with hands-on lab exercises to teach the skills presented. These exercises were developed as educational modules that teach one to two skills at a time. The modules were developed to build on one another up to a final project, similar to the structure of current undergraduate robotics courses at WPI. These modules include activities that work with computer simulations and with a physical robot with a camera system. These modules, along with notes for debugging, have been uploaded to a Canvas page and are ready to be used by professors for existing courses or as a new course.

We developed this solution by testing hardware options and creating a list of desired outcome skills for students to guide module development. To test hardware options, we gathered available Raspberry Pis and cameras and benchmarked them in our primary performance-intensive use case, running ML models for classification using video input. For our robotics platform, we are using a Pololu Romi with a Romi 32U4 control board. The Robotics Engineering department already has an abundance of these robots for the sophomore-level Unified Robotics courses, and they are easy to use and customize. Our work is mainly done on the Raspberry Pis that are connected to the Romis via the I2C bus, so we do not need an advanced robotics platform to test the modules. To develop the list of skills, we compared the outcome skills of courses currently taught at WPI and other schools. We also interviewed professors at WPI to determine what skills they believe students should have in this field. To test the effectiveness of our solution, we hosted several student workshops using the developed modules. We collected and aggregated metrics from the workshops to determine how much students learned from our materials. Finally, we used these metrics to inform revisions of our modules into their final forms.

2 Background

2.1 Overview of Machine Learning

Machine learning (ML) is a branch of artificial intelligence (AI) that allows engineers and scientists to create models from observed data. ML is becoming increasingly popular as a tool for a variety of modern problems, such as detection of objects, faces, noise, and chemicals; learning patterns of behavior; dynamic control; advanced path planning and strategy; and predictive tasks such as trajectory prediction [3]. ML models learn to do these tasks by automatically adjusting a large number of coefficient values, called weights, to match statistical properties of input data. However, the intersection of ML and robotics poses its own set of challenges. Normally, large ML models require a lot of high-quality data for training, but in robotics this type of data can be sparse, and the robot needs to be able to adapt to new environments with unreliable and noisy sensor data. In addition, there are often physical limitations of the hardware and mechanical aspects of the robot imposed by cost, size, power, and other design constraints. For example, if we want a robot that is able to fit into small spaces with a camera that handles sensitive data, we will be limited in our options for processors because of size constraints. We also cannot rely on using an API to access a large model hosted on the web because we may not want to store sensitive information online.

The following subsections cover ML techniques used in this project. We introduce more specific topics in ML as they relate to our course material. Broadly, the ML applications we focus on fall into three categories: advanced perception techniques, such as image classification; optimization techniques for using such models on performance-constrained embedded systems; and applications of reinforcement learning (RL) in advanced path planning and robot control.

2.1.1 Deep Learning

Deep Learning (DL) is a branch of ML used in robotics that focuses on learning complicated concepts by breaking them down into simpler ones. This is done with multiple layers of functions and computation that distinguish simple features from data or even find structure in unlabeled data. DL techniques often rely on pre-structured data and take a lot of time to become highly accurate [4]. DL models are also notoriously large, requiring billions of parameters to achieve high-accuracy results [5]. In robotics, DL applications have found success in perception, manipulation, and navigation, among other fields. However, they come with a drawback of unpredictability and high computational cost due to their size and complexity [6].

2.1.2 Machine Learning Optimization

Optimization in ML is an important concept that ranges from optimizing model learning for accuracy to optimizing model size and speed on embedded hardware. In robotics, we are more interested in how the models work on hardware and how we can optimize them for resource-constrained robots. Many modern ML models have billions of parameters that the hardware is required to compute quickly in order to make the model usable. The two optimization methods covered in our modules are model quantization and model pruning. Our specific application of these techniques is discussed later, but the goal of both techniques is to make DL models simpler to run for computers. These techniques accomplish this goal by using less computationally-intensive mathematical operations and by making models smaller by removing parameters in a strategic way to minimize performance loss.

2.1.3 Reinforcement Learning

Reinforcement learning is a branch of ML used in robotics that trains an agent using trial-and-error learning to make decisions in an environment that gives feedback based on action outcomes. The goal of RL is to map situations to actions and maximize some reward provided by the environment. RL has several applications in robotics and has led to important progress in the field [7]. An example application is dynamic control: control the voltage input to motors in a robotic arm, then select and tune parameters for complex, untuned controls algorithms. Other applications include path planning and decision-making using a variety of different models.

Often in robotics using RL means a lot of trial and error to learn a set of actions or parameter values. However, this repetition is not always feasible with physical robots. For example, to tune parameters for a control algorithm to fly a quadcopter drone, we cannot deploy a real drone 500 times to check the tuning. This leaves the drone susceptible to failure, and the process is time consuming compared to a simulation that can be run without any risk to the hardware [8]. However, we can perform RL for a robot by simulating an environment the robot will experience, then running it repeatedly until the robot has learned some pattern of behavior. We can then apply this to the physical robot for some task. Perhaps the largest benefit of RL as applied to robotics in a sim-to-real pipeline is the ability to let the agent train over a large number of iterations without requiring time spent on a physical robot or large sets of labeled training data. Additionally, RL can allow agents to learn patterns of behavior, such as in task and motion planning, that can be much more nuanced and adaptable than traditional, non-learning-based algorithms. However, transferring a model trained in simulation to a physical robot brings its own challenges that will be discussed

later.

2.2 Problem

2.2.1 A Growing Field

There is currently a boom in interest in ML and its surrounding topics. From large language models such as ChatGPT to AI visual creation like DALL-E to autonomous cars from Tesla and Google, we hear about these advances from all angles in the news and social media. Additionally, researchers are increasingly interested in ML. The number of publications in ML topics in different technical fields has increased every year for the past five years [9]. From 2016 to 2020, the number of publications related to ML increased drastically from 3,886 to 16,339, more than 400% [9].

Machine learning has also become a field of interest for robotics. Making robots that are autonomous and intelligent has become more popular in recent years, and the boom in ML has only contributed to this popularity [10]. For example, ML, in combination with computer vision, has been integrated with robotic grippers to move stock through retail warehouses [11].

The recent fervor in the ML field, combined with its forecasted long-term viability, has led to an increased demand for workers with skills in AI and ML [12]. The World Economic Forum’s 2020 “Future of Jobs” report listed AI and ML specialists as the role with the second-most increasing demand [13]. This demand can be seen in a variety of sectors, from agriculture to healthcare, and it requires a variety of roles, including software developers with expertise in the field [12][14]. As AI and ML become prevalent in the job market, universities are adapting and evolving their curricula to better prepare their students. This includes teaching theory as well as application to real-world problems [12].

2.2.2 What Professors Say

As part of our problem exploration, we interviewed professors in the fields of ML and robotics to better understand the gap in ML for the robotics curriculum. The results of these interviews are discussed in detail in section 4.1. Below are the key highlights of what professors say about the intersection of ML and robotics at the undergraduate level at WPI:

- Students do not typically struggle with ML theory, but rather they struggle with practical applications of ML.

- For roboticists, bridging the gap between computer science theory and robotics applications is an invaluable skill.
- RL is important in robotics, but it is not covered even by graduate courses at WPI.
- Transferring ML/ RL models trained in simulation to a physical robot is a challenging but critical skill.
- Most professors we interviewed would want students to have some experience training ML models on GPUs before researching in their labs.

2.2.3 What WPI Offers

WPI offers courses at both the undergraduate and graduate level for ML and further graduate courses in DL. However, these courses are predominantly taught by the Computer Science department and focus on general topics that are universally applicable. They neglect the nuances presented by robotic systems, such as sensor noise, computational limitations, and dynamic environments. Higher-level graduate courses in ML cover some of these issues, but not all interested robotics engineering (RBE) students can fit all of these courses into a four-year plan.

The following is a breakdown of skills foundational to ML for robotics that are taught to RBE undergraduate students in required courses [15]. These are general skills that provide a foundation for applying ML to robotic systems.

- Basic robot navigation control, such as turning and line following
- Embedded system applications and programming
- Math, particularly probability, linear algebra, and differential calculus

Below is a breakdown of skills related to ML for robotics that are taught in undergraduate courses available to RBE undergraduate students [15]. These skills are more specific to ML and do not consider robotics applications.

- Statistical methods
- Theoretical introduction to AI and ML techniques
- Introduction to RL and DL concepts

Additionally there are a number of useful ML skills, some of which relate more directly to robotics applications, taught in graduate level courses. Undergraduate students may be able to fit some, but likely not all, of these courses into a four-year schedule. Additionally, some concepts are taught only in special topics courses, and the content of these courses are professor-dependent. The aim of this project is to present topics from this list in a way that is accessible to undergraduate students and specific to embedded robotics applications [15][16].

- Multilayer feedforward networks
- Variations of neural networks (e.g. convolutional and recurrent neural networks)
- RL theory, such as value function approximation, actor-critics, and policy gradient methods
- DL techniques for embedded systems, such as model compression and optimization techniques, on-device transfer learning, and tinyML

The required and available courses at the undergraduate level cover important theoretical foundations in ML, but only courses at the graduate level begin to address how to mitigate the impact of hardware limitations on performance of ML. None of the courses adequately address integrating an ML model with a robotic system to direct a robot’s decision-making or determine whether ML is an appropriate solution to a problem. This presents a gap for this project to fill. This project aims to develop an undergraduate course that teaches students when to use ML and how to integrate an ML model effectively with a robotic system while mitigating the constraints posed by that system on model performance.

2.3 Solution

We propose the creation of a course specifically dedicated to teaching students how to integrate an ML model with a robotic system, how to mitigate the impacts of hardware limitations on model performance, and how to determine whether ML is an appropriate solution to a problem. This course assumes some knowledge from an introductory AI or ML course, and it furthers that knowledge with applications to physical robots. Our course would introduce tools such as PyTorch, teach the fundamentals of RL and object detection with ML, and teach optimization techniques to make ML models perform better on resource-limited hardware.

2.3.1 Justification and Related Works

But why make a course on ML as applied to robotics? We believe it is necessary to dedicate a course on the topic, rather than introduce it at the sophomore and junior level robotics courses as another concept.

WPI undergraduate courses are run in 7-week terms as opposed to 14-week semesters, which greatly limits professors' ability to meaningfully teach ML for robotics in existing courses. Furthermore, as mentioned previously, academia and employers are looking for more skills in the fields of ML. However, not all students are interested in going down this path; instead, they may be interested in designing physical systems or integrating sensors into robotic systems. There should be a course to introduce students and give them the proper tools to be able to understand ML and incorporate it into robotic systems, which WPI is currently missing.

We investigated how other schools teach concepts in ML and how courses at WPI teach these topics. From this we were inspired to create a hands-on, lab-focused class, similar to the core Unified Robotics courses taught at WPI. Simultaneously, we gained inspiration of what the labs would look like by looking into courses at other schools that teach topics not covered at the undergraduate level at WPI. Three courses we investigated were F1Tenth at Carnegie Mellon University, EECE 5644: Introduction to Machine Learning and Pattern Recognition at Northeastern University, and ESE 3600 TinyML: Tiny Machine Learning for Embedded Systems at University of Pennsylvania. Objectives for these courses include the following:

- Convert ML model for hardware [17]
- Deploy ML model on microcontroller [17]
- Use a GPU to accelerate ML algorithms [2]
- Understand where and how to apply ML [18]

2.3.2 Machine Learning Techniques

A key ML technique we used in this project is transfer learning. Transfer learning is where a pre-trained model for one task is trained using new data to perform a new, but similar task [19]. This model has some existing weights that have been trained for one task, and now we can tune the weights to fit another task. For example, a model originally trained to recognize cars can be modified to recognize trucks. A key benefit of transfer learning that has made it popular in ML research is the reduced training time. The pre-trained model already has weights that extract broad features, so they are simply being re-trained for another task. Additionally, one can often use a smaller training set for transfer learning than for training a model from scratch [19]. This technique makes using ML for our project more accessible because we do not need large datasets to train sufficiently accurate models.

We also used pruning and quantization to reduce the size of ML models so that we can fit them

into our devices and improve inference (prediction) accuracy. Quantization in ML refers to discretizing a larger input into a smaller input [20]. A general example of this would be converting an RGB image, which has three values per pixel, to a black and white image, which only has one. In the PyTorch framework, quantization is done by converting 32-bit floating point weight values to 8-bit integers, which reduces model size by 400% and increases model speed by 200% to 400% [21]. Pruning reduces model size by removing weight connections in neural networks. Since neural networks are often over-parameterized, pruning increases model speed by removing unused weights: it can also improve model accuracy by reducing redundancy [22]. These two techniques decrease model size while improving efficiency and were essential in helping us use ML models on resource-constrained embedded hardware.

The last ML technique we used was RL for path planning and navigation. We specifically used Q-learning for navigating a grid-based environment. Q-learning is a model-free RL method, which means it implicitly learns the value of taking certain actions without a direct expectation of the future [23]. By contrast, model-based RL methods learn or have access to a model of the cause-and-effect outcomes of actions taken in the environment [23].

In general, model-free RL methods are easier to implement in code and train in a reasonable amount of time given the right problem or environment [23]. Q-learning is the most basic “starting point” for model-free RL algorithms. Its goal is to learn the expected immediate and future reward for each possible action in the environment. Q-learning is therefore ideal for state/action spaces that are not too large. Additionally, once a Q-learning model is trained, running the model merely requires using a lookup table the size of the state/action space. For a reasonably sized environment, this is easy to run on embedded hardware.

2.3.3 Final Product

Our final product is a Canvas page, linked in Appendix A, containing modules covering the following topics: Introduction to PyTorch and Raspberry Pi setup, Introduction to Transfer Learning, Introduction to Pruning and Quantization as Optimization Methods for Machine Learning, and Introduction to Reinforcement Learning. Within these modules are Python notebooks for student exercises, slideshows for instructions and content, and supporting code files for the Romi and the Raspberry Pi.

We elected to use Python notebooks because they supported the workshops we ran for testing our modules with students. Google Colab allows users to share notebooks that run using cloud-based computing resources, reducing required setup for the user. However, these modules can also be downloaded to a user’s personal machine and be run using their own hardware. The setup for a personal machine, in addition to

the setup for the robotics platform and Raspberry Pi is included in the setup module.

Within the notebooks are text explanations and images to teach students material for the given module. There is also code with fill-in-the-blank spots for students to apply the knowledge described in writing. As students read through material, they fill in blocks of code to complete a portion of the exercise. For the transfer learning and optimization modules, there are hardware-focused components within the notebooks where students will produce a model and then transfer it to the robotic system for testing. At the end of the notebook there are reflection questions for students to answer such that they think critically about the results of their work. For example, in the pruning and quantization module, students are asked to reflect on how each method impacted the performance of their ML models.

In addition to the notebooks that students use to learn material, there are separate code files for the Romi and Raspberry Pi. These files interface the two devices with one another such that the Raspberry Pi sends signals to the Romi, which controls its motors based on instructions from the Pi. To connect the Raspberry Pi to the Romi, we connected 5V and GND pins from the 32U4 board to the 5V and GND pins of the Raspberry Pi GPIO layout. We then used I2C to connect to the two devices such that the Pi controls the Romi. We connected the camera to this system via a ribbon cable to the Pi, and we 3D printed a mount to fix the camera to the Romi chassis (Figure 1).

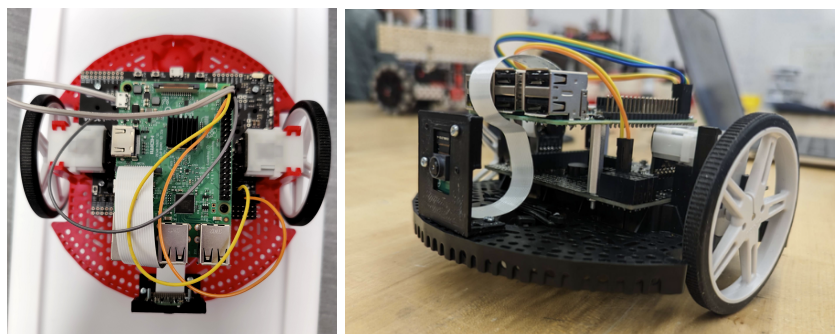


Figure 1: The configuration of the Romi, Raspberry Pi, and camera.

3 Methods

3.1 Professor Interviews

Before developing modules, we first had to understand which problems our project should focus on the most by performing a stakeholder analysis. A key component to formulating the objectives was understanding where student skills were lacking. Therefore, we interviewed professors who are active in AI and ML across

different departments at WPI to inform decisions regarding our course direction. We attempted to gather data from a diverse set of sources, so we interviewed professors with experience in ML with robotics and professors without robotics experience.

From these interviews, we aimed to understand the following:

- What outcomes should we have for students taking an undergraduate course in ML? What skills are assumed of the student before and after taking such a course?
- What topics should be covered in lectures to enable students to complete ML activities?
- What exercises would be valuable for students to complete to learn these skills?
- What kind of hardware should the students be familiar with while working on ML models?

We interviewed professors in semi-structured interviews to answer key questions for our project while allowing free flow of thought. We then used the key points across several interviews to determine which skills were most relevant and to inform our course structure.

We began each interview by explaining the goal of our project, and then we presented claims we wrote about the available ML curriculum at WPI. These claims and the responses to them were used to assess the need at WPI for the course we are developing. We then asked general questions about student abilities in ML at WPI to inform the development of a list of skills we should teach. We also focused on the challenges brought to ML by robotic systems, which further informed our list of skills and warned us of potential roadblocks we may face in this project when working with hardware.

After the questions, we presented an example module with ML and robotics elements. This example was presented as a lab exercise to be completed by students in the fourth or fifth week of our term-long course. We presented a description of the lab environment and assumed student background before describing the deliverable and reflection questions. We then presented the target student skills upon completion of the example module. After presenting the module, we asked professors for their general thoughts, such as the feasibility of the exercise for students and what content they would teach to support the exercise. We also asked professors what other activities or modules they would want students to complete in this kind of a course. This helped inform our module design with respect to what kinds of activities professors would want students to complete. The questions, claims, and module used in these interviews are presented in Appendix B.

3.2 Choosing Material to Teach

Based on our objectives and interviews with professors, we developed a list of skills to be taught in our course. To determine how to group skills into module activities, we created a dependency graph to find groups of related skills and their dependency on one another. This graph provided us with an order in which to develop the modules (Figure 2). The modules we developed are as follows:

- Setting up hardware and libraries
- Introduction to PyTorch
- Transfer learning
- Model optimization techniques for embedded systems
- Reinforcement learning

We chose to use PyTorch, a popular, Python-based framework for ML and its subfields [24]. We chose this framework because of its increasing relevance in ML research, its documentation and ease of use, and its capabilities [25]. The capabilities of PyTorch fully support our objectives to teach students how to use ML on a robotic platform. It also supports and has tutorials written for transfer learning and optimization for embedded systems, two topics we aimed to cover with our course materials.

3.3 Module Development

One of the common points made by professors we interviewed is that students need applied knowledge in addition to theoretical understanding. Additionally, professors largely appreciated that our example module was designed to be hands-on. This motivated us to focus on making interactive modules that apply concepts rather than focusing solely on theory. We began module development with “paper prototypes”, or low-fidelity prototypes that laid out the procedures of each module and what the results would be without any technical implementation. After reviewing these paper prototypes with our advisors, we then added technical details and code to the modules.

A challenge to module development was that we as a team had to learn the concepts that we chose to teach. This involved reading research papers and books on DL and RL, referring to lecture slides from previous graduate-level courses, and talking with our advisors. It also included considerable trial and error by writing code and debugging it with the help of forums and documentation.

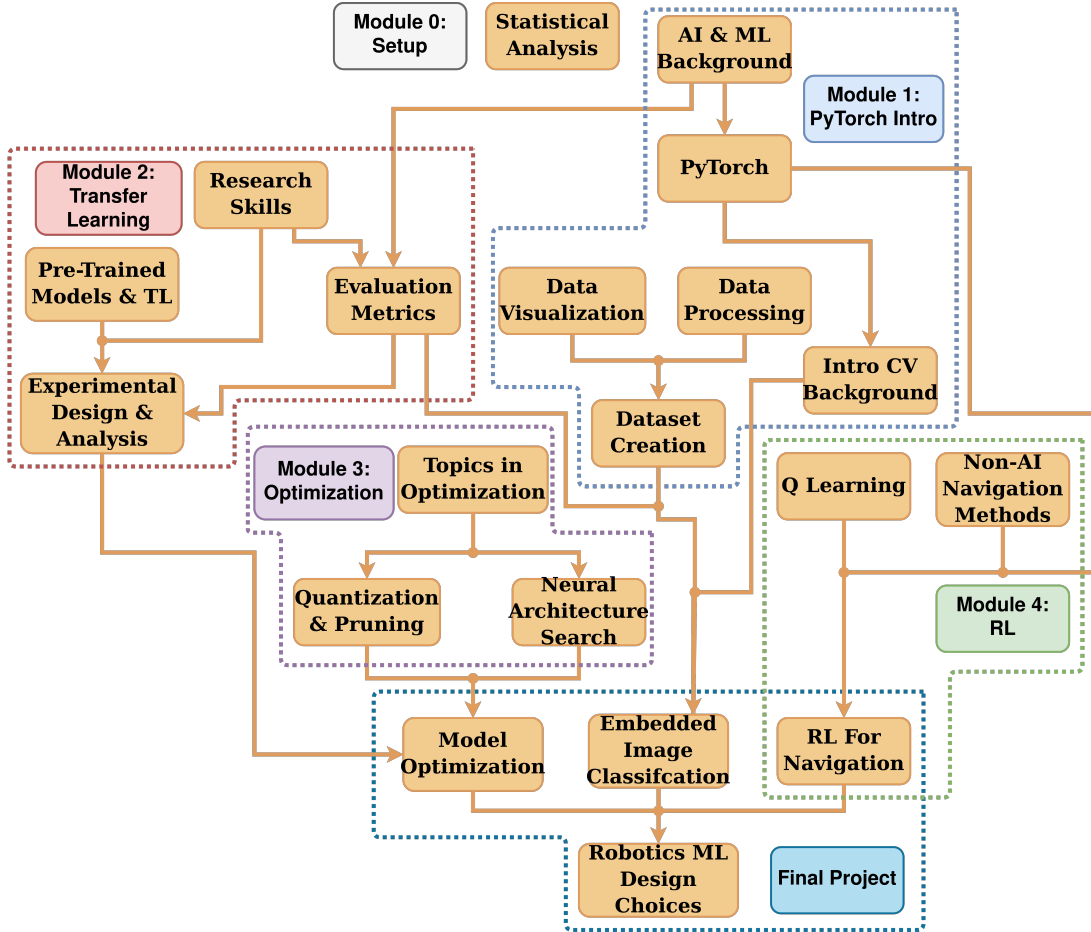


Figure 2: Dependency graph of skills in machine learning and computer vision, which provided a basis for grouping related skills into modules and determining an order such that each module has the appropriate foundation.

The first module is simply setting up the Raspberry Pi with the proper OS and Python libraries that are used in future modules. This module also covers setting up a personal computer with similar libraries. We chose Debian Bullseye as the OS for the Raspberry Pis because it is the most recent OS that still supports interfacing with Pi cameras using the Python picamera module [26].

The second module introduces students to PyTorch, which is used throughout the course material to develop DL and RL models. This module also teaches students how to use a GPU either on Google Colab or their personal computers and how it compares to using a CPU while training a model.

The third module covers transfer learning and dataset building, where students start with a provided, pre-trained model architecture and then train the model with their own data for their own task. This module meets the objective of introducing students to using pre-built models and modifying them for particular tasks. On top of this the students get exposure to how difficult and time consuming building a

custom dataset can be, especially with larger sets.

The fourth module expands upon the transfer learning module by introducing several optimization techniques that result in models that are easier to run on embedded systems. The main techniques used here include quantization and pruning, as they are the two main methods currently used to speed up models. This module meets the objective of using ML on embedded systems.

The fifth module introduces students to RL, which is relevant to robotic applications of ML. In this module, students implement Q-learning to navigate a frozen lake with holes in the ice in simulation. The students will learn about different types of RL algorithms that are currently popular, and they will learn how to build an optimal policy for a robot using RL. This module meets the objective of introducing students to RL.

3.4 Hardware Testing

Two of our objectives for our solution involve students using ML on robotic hardware, particularly hardware that is resource limited. For our robotic system, we elected to use the Pololu Romi for several reasons. Current robotics students use the platform in the 2000-level core robotics core courses, so they are already familiar with the hardware. This reduces the barrier of entry for integrating ML concepts on hardware. Additionally, the Romi 32U4 control board is designed to interface with a Raspberry Pi [27]. Finally, the robotics department has plenty of Romis readily available for use.

To pick an embedded device to use for ML models, we tested the following versions of the Raspberry Pi: 2 Model B, 3 Model B, Zero W, and Zero 2 W. We chose these models based on availability; some team members and project advisors already owned Raspberry Pis that were available for use in this project. However, at the time of this project, stock for some versions of the Pi were still extremely limited, while others were available for purchase at the time of testing [28].

The team performed device testing with the Raspberry Pi Camera Module 2 and the Arducam IMX519 16MP autofocus camera. We did not focus on testing the cameras specifically, but we tried these cameras to see if they both would work for our purposes. During device testing, we found the setup process for Raspberry Pi cameras was less prone to error than the process for Arducam cameras. The Arducam camera ran into many different technical problems at the beginning of testing. However, once those issues were passed, it performed the same as the Raspberry Pi camera module.

To select a Raspberry Pi model, we specifically were concerned with device cost, device availability, and frames per second (FPS) of ML models working with video data. Cost and availability of devices and

compatible cameras were relevant because the team had a limited budget for this project, and we needed multiple devices to run hardware-based workshops. Model FPS determined whether the Raspberry Pi had the processing power to output model inferences fast enough that a robotic system could react to them in real time. We ran this test using an unoptimized ResNet18 model and with a quantized ResNet18 model to compare un-optimized and optimized performance [29].

Device cost and availability for Raspberry Pis and compatible cameras were determined via internet research. To test the FPS, we developed a Python script that loaded a trained model and made inferences on images coming from the camera video feed. While the model made inferences, the code tracked camera frames read each second. The model FPS was printed to the console each second, and the team took an average of the FPS values over 30 seconds.

After benchmarking our available Raspberry Pis, we determined that the Raspberry Pi 3 Model B and Zero 2 W both performed satisfactorily in inference while deploying a quantized ML model. Though the Zero 2 W is significantly cheaper than the 3B, many shops currently limit its sales to one per customer. This meant that though we could not purchase large quantities of the Zero 2 W for our workshops, we could purchase a few and fill in the remaining supply with 3Bs. The Zero 1 W and 2B do not have the computational capability to run these models very well, so even though we could get a lot of them, they do not perform well enough in inference.

3.5 Student Workshops

To measure the effectiveness of our course materials and to measure interest, we opted to collect student feedback about the modules we developed and collect metrics about student interest in the field of ML. To accomplish this, we hosted several student workshops that were modified versions of our educational modules so they could be completed in the span of two hours with a large group of people.

From the workshops, we hope to understand the following:

- How did completing the workshop activity impact the students' perceived skill level in the given subject?
- What improvements could be made to our modules to improve the student experience and module effectiveness?
- Is there interest from the student population in ML for robotics applications?

The first two questions look at the effectiveness of our modules and what can be done to make

them better. The third is intended to gauge whether students would want to take a course on the subject of ML and its applications for robotics. This information may be useful to professors interested in using the materials made for this project.

We designed our workshops by modifying our transfer learning, optimization, and reinforcement learning modules to reduce the time required to complete the activities. This involved increasing the amount of starter code, providing hardware with all necessary libraries installed, and connecting and testing the hardware prior to the workshops. We chose these modules specifically because of their relevance in the ML field and their importance to ML for robotics applications. Transfer learning allows students to quickly make their own models and see the results. The optimization workshop gives students the ability to reduce the overhead of their models so that they run faster and more accurately on embedded devices like the Raspberry Pi. These two workshops involve working directly with the Raspberry Pi, camera, and Romi hardware in a hands-on fashion, which is a key part of our materials. The third workshop, which covers RL, was done entirely in simulation given the complexity of moving a RL model to a real-world system from simulation.

We then recruited workshop participants by advertising the workshops in the RBE Discord server and the undergraduate robotics laboratories using the flyers in Appendix C. At the very beginning of each workshop, students filled out a pre-survey indicating their background knowledge and current ability. We then presented a small, non-technical lecture on the subject to provide students with some background before completing the workshop activities (Figure 3). After the lecture, we provided students with a Google Colab containing instructions, starter code, and technical background information. Students then completed the code activities on their own, requesting help from members of the MQP team as necessary. For the transfer learning and optimization workshops, we provided hands-on assistance with connecting students' laptops to the Raspberry Pi.

During each workshop, we had a team member observe the workshop and take notes on the events of the workshop, such as issues that may impact student feedback. To collect student feedback, we used the before- and after-surveys in Appendix D to understand students' background knowledge, current ability, and how participating in the workshop affected their ability to use the material covered. We also collected general feedback about interest in ML topics and how our tools, such as Google Colab and the Raspberry Pi, impacted their learning. The data from these surveys was then analyzed and used to make modifications to the modules.



Figure 3: Small lecture presented at the Introduction to Optimization workshop.

4 Findings

4.1 Professor Interviews

The first key point from the professors was that students do not usually struggle with the theory in ML. Rather, the practical applications of ML and how we get a useful product from the theory is difficult for students. They have a hard time being innovative with ML and properly applying it to different tasks with different data. For example, a student who has only worked with images has a hard time transferring this knowledge to a text-based problem or even one with video data. There needs to be more focus on applying the concepts to different problems, using different data, and using different architectures so students are fully capable and confident in their skills.

When we interviewed professors specifically in robotics, they told us that bridging the gap of computer science theory to robotics applications is a very valuable skill. As we also noticed in our background for the problem, robotics students do not necessarily learn to apply ML and its theory to robotics. Rather, students learn to apply non-AI algorithms such as A* to robotics problems. ML is not covered in any of the required classes for Robotics Engineering majors at WPI.

Professors also mentioned how important RL is in robotics; for example, RL has been found to help robots autonomously learn to solve complex problems [30]. However, we found that even the graduate ML courses at WPI do not actually put any focus on this topic. We do not have RL courses at the undergraduate level either, and robotics students may not even touch RL before they leave WPI. While this is not important

for all students, roboticists looking to work with ML and robots would benefit from having the option to learn about RL in a course setting.

Furthermore, RL is often done in simulation first, and the learned behavior is then transferred to a physical robot. While robotics students learn some of the nuances of transferring algorithms from simulation to the real world in the Unified Robotics classes, they do not learn how to deal with these nuances in the case of ML or RL. Professors we interviewed pointed out that simulations do not fully simulate environmental factors such as physics, which would be relevant for an ML model intended to train a PID controller for a motor, for example. This then poses a problem for the real world performance of a model trained in simulation. While transferring models trained in simulation to real-world robots is challenging, professors argued that this skill is critically important in the intersection of ML and robotics.

Professors we interviewed agreed with our claim that students do not learn when and when not to apply ML to a given problem. They did argue that students should learn this skill. While ML has had positive impacts on the robotics field, not all problems in robotics require ML to be solved: in some instances, using ML can overcomplicate a solution. For example, path planning in a deterministic environment can be solved with a variety of existing algorithms, such as A*.

We also found that most of the professors we interviewed would want students to have some experience with training models on GPUs before researching in their lab. One professor also advocated for students being able to work with embedded hardware such as Raspberry Pis or Jetson Nanos before entering their lab. As we discussed earlier, this is a unique skill set that is not taught in traditional, theory-based courses.

4.2 Student Workshops

4.2.1 Introduction to PyTorch and Transfer Learning

For this workshop, we had a total of seven participants. Three had taken a course in AI, while four had not, and only one participant had any prior project work with PyTorch. A caveat to the following data is that this workshop used earlier versions of our pre- and post-surveys that did not fully reflect the questions we wanted to answer with these workshops. Additionally, participants' pre- and post-survey responses were not linked for these surveys.

Table 1 indicates how much assistance participants felt they would need to complete a project using PyTorch before and after the workshop. A caveat to these data is that only five of the seven participants

completed the post-survey. These responses show that while some participants felt more capable of using PyTorch after the workshop, others did not, and some felt less capable. These results suggest our materials may have been helpful to some participants, but not all.

	None	Minimal	Moderate	A Lot	A Great Deal
Amount of assistance required (before)	0	0	4	1	2
Amount of assistance required (after)	0	2	0	2	1

Table 1: Amount of assistance participants felt they would need to complete a project using PyTorch before and after the workshop. All seven participants completed the pre-survey, while only five completed the post-survey.

In the free response section of the post-survey, all five respondents indicated that they had learned some PyTorch, with two participants specifying that they had learned some basic PyTorch functions and skills. Additionally, three of the five respondents indicated they learned about transfer learning and how to use PyTorch for transfer learning. Similar to the numerical results, these responses indicate that participants learned some PyTorch and transfer learning, but at an introductory level.

This workshop made use of Google Colab and the Raspberry Pi running the Romi. The majority of respondents to the post-survey made at least one complaint about using Google Colab, most of which were focused on the clunky file transfer experience. However, one participant did note that the Google Colab for the workshop was easy to follow. Three respondents also commented positively on the use of a GPU through Google Colab. As for the Raspberry Pi, comments ranged from neutral to slightly positive, with two participants writing that using them was straightforward. Comments on the Romi also ranged from neutral to slightly positive. These responses to the tools we used suggest that while these tools are not perfect and can sometimes be a hindrance, they are overall palatable to participants and can benefit their learning.

4.2.2 Introduction to Optimization

For this workshop, we had seven participants, none of which had prior experience with pruning or quantization. Participants’ self-assessed experience in Python ranged from beginner to intermediate, and some participants had taken an introductory AI class, while others had not. Of seven participants, six completed both the pre- and post-survey. A potential factor in participants’ responses to our surveys for this workshop is the significant hardware issues we had with the Raspberry Pis and Romis during this workshop, which hindered some of the exercises we had planned.

Table 2 shows participants’ comfort level in explaining quantization and pruning to a peer before and after the workshop. The participant that indicated no improvement had no prior project work in either

subject, and they had not taken any coursework in AI or ML. The remaining five participants had either no prior coursework or had only taken an introductory AI class. These before and after responses suggest that for most participants, our materials were effective in improving participants’ perceived ability to explain quantization and pruning to a peer.

Participant	1	2	3	4	5	6	7
Comfort explaining quantization (before)	0	0	1	0	0	0	0
Comfort explaining quantization (after)	0	1	1	2	2	1	-
Comfort explaining pruning (before)	0	0	0	0	0	0	0
Comfort explaining pruning (after)	0	2	1	2	1	1	-

Table 2: Participants’ comfort in explaining quantization and pruning to a peer. 0 indicates telling the peer to ask a professor, 1 indicates showing the peer resources and documentation, and 2 indicates trying to explain the topic in their own words.

Table 3 shows participants’ response when asked if they could explain how pruning and quantization impact the size and performance of an ML model. All participants felt more confident that they could after completing the workshop. Two participants improved from “No” to “Yes”, while the remaining four improved by one level. These responses suggest that our workshop was successful in demonstrating the differences between pruning and quantization and how they affect ML model performance.

Participant	1	2	3	4	5	6	7
Before	No	No	No	Maybe	No	Maybe	No
After	Maybe	Yes	Yes	Yes	Maybe	Yes	-

Table 3: Participants’ response when asked if they could explain how pruning and quantization impact the size and performance of an ML model. The response options were “No”, “Maybe”, and “Yes”.

Table 4 shows the number of resources participants felt they would need to complete a project using quantization before and after completing the workshop. Two participants indicated needing the same number of resources and the same kinds of resources to complete a project with quantization after completing the workshop. Two participants indicated needing additional resources after the workshop, with one participant indicating the need for tutorials and one indicating need for assistance from a professor. The remaining two participants indicated needing fewer resources, with one dropping the need for Stack Overflow and the other dropping the need for both peer assistance and Stack Overflow. These responses suggest that participants overall may or may not have felt more capable in using quantization in a project after completing our project despite typically feeling more able to explain it to a peer.

Table 5 shows the number of resources participants felt they would need to complete a project using pruning before and after completing the workshop. Two participants indicated needing the same number

Participant	1	2	3	4	5	6	7
# of resources needed (before)	3	5	2	4	5	3	5
# of resources needed (after)	3	3	3	5	5	2	-

Table 4: Number of resources needed by participants to complete a project using quantization before and after our workshop. Five resources were listed and included Stack Overflow, tutorials, professors assistance, peer assistance, and documentation.

of resources and the same kinds of resources to complete a project with quantization after completing the workshop. Only one participant indicated needing more resources by adding the need for professor assistance. The remaining three participants indicated needing fewer resources, with one dropping Stack Overflow and peer assistance, another dropping tutorials, and the third dropping Stack Overflow. These responses once again suggest participants overall may or may not have felt more capable in using pruning in a project after completing our project. However, there was greater improvement in perceived ability to use pruning than quantization, which suggests we may have better explained pruning than quantization.

Participant	1	2	3	4	5	6	7
# of resources needed (before)	3	5	3	4	5	3	5
# of resources needed (after)	3	3	2	5	5	2	-

Table 5: Number of resources needed by participants to complete a project using pruning before and after our workshop. Five resources were listed and included Stack Overflow, tutorials, professors assistance, peer assistance, and documentation.

When asked what they learned as a result of the workshop, participants’ responses were varied. Several participants’ learning was more conceptual than practical: they indicated that they understood the benefits of quantization and pruning and could explain the concepts, but they noted that they didn’t feel confident about implementing them without help. Another participant indicated that they had learned where to start researching if they wanted to reduce the size/ computational cost of an ML model. These responses align with earlier responses, which overall indicates that while our module was successful in explaining concepts to students, it was less successful in teaching them how to implement them in practice.

This workshop made use of Google Colab and the Raspberry Pi running the Romi. Four participants indicated that the use of Google Colab either somewhat or very much assisted their learning, while one indicated it had no significant impact, and one indicated it somewhat hindered their learning. A key point of feedback here was that the Colab text explanations were dense in places and could benefit from the addition of pictures to explain concepts. Another comment was that the directions for filling in blank code areas could be clearer.

For the Raspberry Pi, three participants indicated it somewhat assisted their learning, while two indicated it somewhat hindered their learning. One participant said the Raspberry Pi had no significant impact. These results are likely influenced by the hardware issues we had during the workshop where most of the Raspberry Pis had issues connecting to the Romi, which caused the script used to run participants’ model to fail. Some participants indicated a preference for using only simulation, in part because of the hardware failures.

4.2.3 Introduction to Reinforcement Learning

For this workshop, we had four participants, one of which noted prior project experience with Q-learning and RL in their personal projects. One participant had taken an introductory AI course, while none of the other participants had taken classes in AI or ML. Python experienced ranged from novice to intermediate and from less than one year to more than two years. Of four participants, three completed both the pre- and post-survey.

Table 6 shows participants’ comfort level in explaining Q-learning and RL to a peer before and after the workshop. The participant that indicated they had prior experience with RL indicated they felt most comfortable explaining the topics in their own words on both the pre- and post-survey. The participant that indicated they felt most comfortable telling a peer to ask a professor both before and after the workshop had no prior experience in RL and had not taken any courses in AI or ML. The results, in combination with participants’ background, suggest that our RL workshop did not negatively impact student understanding. It did improve it in some, but not all cases.

Participant	1	2	3	4
Comfort explaining Q-learning (before)	0	2	0	0
Comfort explaining Q-learning (after)	1	2	0	-
Comfort explaining reinforcement learning (before)	0	2	0	0
Comfort explaining reinforcement learning (after)	1	2	0	-

Table 6: Participants’ comfort in explaining Q-learning and reinforcement learning to a peer. 0 indicates telling the peer to ask a professor, 1 indicates showing the peer resources and documentation, and 2 indicates trying to explain the topic in their own words.

Table 7 shows the number of resources participants felt they would need to complete a project using Q-learning before and after completing the workshop. The participant with prior experience with Q-learning and RL indicated needing the same number of resources and the same kinds of resources before and after completing the workshop. The remaining two participants indicated needing fewer resources after the workshop, with one dropping the need for Stack Overflow and the other dropping the need for both

peer assistance. This is a small positive change towards needing fewer resources to complete a project using Q-learning, and dropping the need for peer assistance indicates greater independence.

Participant	1	2	3	4
# of resources needed (before)	4	3	4	5
# of resources needed (after)	3	3	3	-

Table 7: Number of resources needed by participants to complete a project using Q-learning before and after our workshop. Five resources were listed and included Stack Overflow, tutorials, professors assistance, peer assistance, and documentation.

Table 8 shows the number of resources participants felt they would need to complete a project using RL before and after completing the workshop. The participant with prior experience with Q-learning and RL indicated needing the same number of resources and the same kinds of resources before and after completing the workshop. The participant that indicated needing fewer resources after the workshop dropped the need for peer assistance, which suggests greater independence when working with RL.

Participant	1	2	3	4
# of resources needed (before)	4	3	3	5
# of resources needed (after)	3	3	3	-

Table 8: Number of resources needed by participants to complete a project using reinforcement learning before and after our workshop. Five resources were listed and included Stack Overflow, tutorials, professors assistance, peer assistance, and documentation.

This workshop did not require any hardware, which participants commented on positively in the post-survey. All participants indicated that the use of Google Colab as a teaching tool neither assisted nor hindered their learning, unlike prior workshops where participants had a wider spread of opinions. Additionally, one participant commented positively on not having to deal with transferring code and models to hardware. These responses, in combination with surveys from prior workshops, suggest that Google Colab may be an adequate teaching tool for our modules when used on its own but becomes less manageable once hardware is involved.

4.2.4 Student Interests in the Field of Machine Learning

In each post-survey for all of our workshops, we asked participants what they would want to learn more about in the field of ML. One of the more popular responses to this question was RL, which was mentioned by participants in all three workshops. Additionally, participants in the PyTorch and optimization workshops indicated they wanted to learn more about integrating ML with robotic systems: for example, one participant

wanted to learn more about mapping outputs of a neural network to movements of the Romi. Another participant wanted to learn more about training a more complex model that could identify an object in an image and then track it. Participants also indicated being interested in learning more about the subjects covered by our workshops.

5 Future Work

5.1 Final Project

Due to time and space constraints, we were not able to implement a final project-style module to complement the other modules we developed. Further work on this project could involve outlining, developing, and demonstrating a feasible final project that uses elements of transfer learning, optimization, and RL to dictate the actions of a physical robot.

The final project module could be flexible to allow an instructor to tailor it and the other modules to the interests of a class or to their area of research. If an instructor wanted to focus on how the robot should behave, like playing a game or tuning controls, they could make the final project focused on RL and have the robot learn an optimal policy while traveling in a maze. Or, if the professor wanted to focus more on computer vision-related tasks, the final project could have an emphasis on deploying various ML models to find objects in a field.

5.2 Improvement on Materials

The workshops helped students understand the theory and topics in ML but did not teach them how to make the models and functionality from scratch. For all the workshops, we gave students most of the code to complete the assignment, and the students simply added or modified some lines to see how their contributions affected the models. This problem largely stems from the fact that our workshops were only two hours long: we did not have the time to go over every detail of how a model is built.

Per our workshop survey results, students who participated in our workshops felt more confident in their high-level understanding of the concepts after completing the workshop activities. However, this increase in confidence was not universal among participants. Additionally, the participants' confidence in their ability to apply the concepts in a project showed minimal to no improvement. This, along with written feedback from participants, suggests the need to improve the educational effectiveness of the module

materials. We were able to make some improvements, including the following: adding images to illustrate concepts, reducing text density, and increasing guidance while reducing provided code. Improvements to make in the future could include the following:

- Adding lecture materials and example code
- More exercises that repeat and reinforce material

Given the limited attendance of our workshops, our data cannot conclusively attest to the effectiveness of our materials: more participants would be required to determine any conclusions about the modules. Additionally, we only tested our first iteration of the modules with students, so further work on this project would require testing the updated modules with students.

Additionally, we had inconsistent issues with the robot hardware we used to develop these modules, including IO errors between the Romi 32u4 and Raspberry Pi and sufficiently powering the Raspberry Pi from the 32u4 board. Future work on this project could explore using other hardware solutions and modifying the module materials to work with this hardware.

Finally, our materials are not a comprehensive education in learning ML and applying it to robotics applications. For example, our modules do not cover determining whether ML is an appropriate solution to a problem. Further work on this project could develop modules focused on other ML techniques and other challenges that come with using ML on a robotic platform.

6 Conclusion

Machine learning is an increasingly relevant topic in the robotics field. While WPI offers undergraduate courses that teach ML theory, these courses do not cover specifics relevant to using ML on robotic systems. We gathered data from professors through interviews, which revealed that students do well learning basic theory yet struggle to make the jump to real-world applications. Students also lack skills in working with hardware and ML together. Therefore, there is a gap in student understanding of how to make ML models that run on robotic systems, how to use tools that are popular in the ML field, and how different hardware can affect ML training and the models run on them. We attempt to fix this gap by making an undergraduate-level course to teach ML applications for robotics. For this project, we developed course materials to teach integration of an ML model with a robotic system and mitigation of the impacts of hardware limitations on model performance.

After developing our modules, we ran multiple workshops covering different topics to test the effectiveness of our materials. Even though during some workshops we ran into issues, the students who attended typically learned something from the workshops. Students often felt more capable to explain materials to a peer and more able to navigate a project involving the workshop topics. They also learned where to start looking if they want to further their knowledge in any of the workshops that we ran. Future work on this project to improve effectiveness could continue to develop more materials to teach students how to apply ML to robotics and/ or pursue alternative hardware solutions.

References

- [1] “Robotics engineering.” <https://www.wpi.edu/academics/departments/robotics-engineering>.
- [2] “F1tenth - cmu,” 2021. <https://courses.f1tenth.org/courses/course-v1:Carnegie-Mellon-University+15weeks+2021-01/about>.
- [3] M. Soori, B. Arezoo, and R. Dastres, “Artificial intelligence, machine learning and deep learning in advanced robotics, a review,” *Cognitive Robotics*, vol. 3, pp. 54–70, 2023.
- [4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [5] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, *Dive into Deep Learning*. Cambridge University Press, 2023. <https://D2L.ai>.
- [6] A. I. Károly, P. Galambos, J. Kuti, and I. J. Rudas, “Deep learning in robotics: Survey on model structures and training strategies,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 266–279, 2021.
- [7] A. Shevlyakov, “Using reinforced learning methods to control cube robots,” in *2019 Twelfth International Conference “Management of large-scale system development” (MLSD)*, pp. 1–4, 2019.
- [8] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, 2018. <https://www.andrew.cmu.edu/course/10-703/textbook/BartoSutton.pdf>.
- [9] R. Pugliese, S. Regondi, and R. Marini, “Machine learning-based approach: global trends, research directions, and regulatory standpoints,” *Data Science and Management*, vol. 4, pp. 19–29, 2021.
- [10] M. Mainampati and B. Chandrasekaran, “Evolution of machine learning algorithms on autonomous robots,” in *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 0737–0741, 2020.
- [11] J. Smith, “Robotic arms are using machine learning to reach deeper into distribution,” *The Wall Street Journal*, 2022.
- [12] O. Idiakhova, “Ai talent gold rush: The increasing demand for machine learning skills,” *Medium*, 2024. <https://medium.com/@oluwafemidiakhova/ai-talent-gold-rush-the-increasing-demand-for-machine-learning-skills-c5840d6bbcca>.

- [13] “The future of jobs,” 2020. https://www3.weforum.org/docs/WEF_Future_of_Jobs_2020.pdf.
- [14] F. Borgonovi, F. Calvino, C. Criscuolo, L. Samek, H. Seitz, J. Nania, J. Nitschke, and L. O’Kane, “Emerging trends in ai skill demand across 14 oecd countries,” *OECD Artificial Intelligence Papers*, no. 2, 2023.
- [15] “Undergraduate catalog 2023-2024,” 2023. <https://wpi.cleancatalog.net/classes>.
- [16] “2023-2024 catalog,” 2023. https://wpi-grad.cleancatalog.net/sites/default/files/pdf/pdf_generator/graduate-catalog-202324.pdf.
- [17] “Electrical & systems engineering (ese),” 2023. <https://catalog.upenn.edu/courses/ese/>.
- [18] “Academic catalog 2023-2024,” 2023. <https://catalog.northeastern.edu/search/?P=EECE%205644>.
- [19] V. Chouhan, “What is transfer learning,” 2023. <https://www.geeksforgeeks.org/ml-introduction-to-transfer-learning/>.
- [20] J. Hou, “Here’s why quantization matters for ai,” 2019. <https://www.qualcomm.com/news/onq/2019/03/heres-why-quantization-matters-ai>.
- [21] “Quantization,” 2023. <https://pytorch.org/docs/stable/quantization.html>.
- [22] “What is pruning in machine learning,” 2020. <https://opendatascience.com/what-is-pruning-in-machine-learning/>.
- [23] “Spinning up introduction to rl part 2: Kinds of rl algorithms,” 2018. https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html.
- [24] “Features — pytorch.” <https://pytorch.org/features/>.
- [25] R. O’Connor, “Pytorch vs tensorflow in 2023,” 2023. <https://www.assemblyai.com/blog/pytorch-vs-tensorflow-in-2023/>.
- [26] “Getting started with the camera module.” <https://projects.raspberrypi.org/en/projects/getting-started-with-picamera>.
- [27] “Romi robot kit for first.” <https://www.pololu.com/product/4022>.
- [28] J. Loynds, “Raspberry pi says its stock issues will end in 2023, but there’s a catch,” *Dexterto*, 2022. <https://www.dexerto.com/tech/raspberry-pi-says-its-stock-issues-will-end-in-2023-but-theres-a-catch-2009103/>.

- [29] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” 2015.
- [30] P. Shengguang, “Overview of meta-reinforcement learning research,” in *2020 2nd International Conference on Information Technology and Computer Application (ITCA)*, pp. 54–57, 2020.

Appendices

A Project Deliverable

Here is the link to the Canvas page containing the final modules produced by the project team: <https://canvas.wpi.edu/courses/62551>

B Professor Interview Questions

For our MQP, we're looking to develop an undergraduate course for juniors and seniors who have completed the main robotics course sequence and want to learn to apply computer vision and machine learning concepts to a physical robot.

We'll first read you some claims about the existing curriculum and current student skills and ask your opinions on those claims. Afterwards, we've designed a lab activity that students in our course would do: we'll ask you your thoughts on the activity itself and how you would fit it into a course.

With the claims, as you may not have experience with some of them, we can just assume them to be true for the sake of the interview, or we can explain more about the claim.

If at any point you have questions or comments, feel free to say your thoughts at any point of the interview.

B.1 The Claims

1. There are no courses at WPI synthesizing computer vision and machine learning in the undergraduate curriculum.
2. There are no courses at WPI that introduce machine learning concepts that relate to robotics systems in the undergraduate curriculum.
3. In the core undergraduate RBE curriculum, students are not introduced to machine learning concepts that relate to robotic systems.
4. On par with the previous claim, students do not learn to differentiate between problems where machine learning is the right tool and those where other methods are better.
5. There are no course options for learning computer vision with machine learning in the undergraduate curriculum in either the CS department nor the RBE department.
6. Understanding and applying machine learning and computer vision concepts is a good skill to have for research labs (at WPI and elsewhere) and for industry.

B.2 Questions

1. What parts of the claims do you agree or disagree with?

2. What value is there in a course that addresses this claim?
3. What topics do you believe WPI should cover more in depth, or does not teach at all in the undergraduate curriculum in ML or CV?
4. How often do you see students struggling with ML and CV concepts? What are the most difficult concepts in ML that you see students struggling with?
 - (a) How are students filling in these skill gaps, if they exist?
5. What skills do you want from your students before they enter your lab?
 - (a) Do you feel that the students adequately have these skills?
 - (b) What kinds of gaps in knowledge do you see in these skills?
6. What kind of hardware would you want your students to be familiar with, if any, before entering your lab?
7. What kind of hardware would you want a course on ML and CV in the undergraduate level to work with?

B.3 ML Questions

1. What do you think are the most important considerations to make when using ML on an embedded system versus a regular computer?
2. What challenges do physical robots bring to ML?
3. What specific skills in ML do you find to be the most important for a robotics engineer?

B.4 CV Questions

1. What do you think are the most important considerations to make when using CV on an embedded system versus a regular computer?
2. What challenges do physical robots bring to CV?
3. What specific skills in CV do you find to be the most important for a robotics engineer?

Now we will go over a module we have developed for the interviews as an example of what the labs in the course could look like.

B.5 Module - ML and CV Synthesis

For prior context about this module, this lab is intended to take place in the middle of the course, around week 4 or 5. The robot students would use for this course would be relatively small—about the size of a cereal bowl: as such, the robot would navigate a scaled-down obstacle course during the activity.

B.5.1 Assumed Background

- Understanding of multiple machine learning models such as YOLO, SSD, Fast R-CNN
- How to train a machine learning model for object detection
- At least two algorithms for object detection with computer vision
- Programming a differential drive robot to drive in a circle

B.5.2 The Deliverable

Students will pick two object detection algorithms discussed in class (e.g. YOLO or SSD) to train to detect a robot and a stuffed animal, as these are obstacles their robot will encounter in its world. To train the model, students will use provided training data, though they are welcome to increase the size of the training set using image augmentation. Students should then upload each trained model to their camera and test their accuracy (1). Once students are confident in the reliability of both their models, they will write a program for their robot such that it drives in a circle, stopping and reversing direction when it detects either a robot or a stuffed animal through the camera. Students will test the reliability of this program with both detection models (2).

1. How often is the camera able to detect a given object? At what angles/ distances/ lighting does it struggle in, and how could you fix this?
2. Compare and contrast the performance of the two models. What issues are there in inference? False positives and negatives?
3. How do these models perform against objects that appear similar to the items we have targeted?

B.5.3 Student Skills Upon Completion

- Compare and contrast how different models perform on a mobile robotic system

- Use a machine learning model on a live camera feed to make decisions
- Program a robot to navigate an environment using a camera to detect certain obstacles

B.6 Questions About the Module

1. What are your thoughts on the module?
2. What other types of modules would you want to see be introduced in this kind of a course?

C Workshop Posters



Robots & ML Workshop

CC Mid-Century Room, 01/30/24, 3-5pm, and SL105, 02/02/24, 4-6pm

First part of three upcoming workshops.

Come and learn how you can apply ML models to your own robots! This first workshop will cover topics in transfer learning, utilizing PyTorch, and building your own model for classification. Afterward we will showcase how you can apply this model to a simple robotics platform.

We expect participants to have some basic Python and AI knowledge.




Optimization Workshop

SL 104, 02/16/24, 4-6pm

Come and learn ML optimization!

Learn how you can distill and apply ML models to your own robots! We are pruning and making the models much smaller to be able to fit them into a Raspberry Pi. We will also explore how to make the models more accurate, while making them tinier.

Come have fun and learn about making ML models smaller and more effective on hardware!

Contact gr-lemurmqp@wpi.edu with any questions!

RSVP Link





Reinforcement Learning

CC Mid-Century Room, 02/20/24, 3-5pm, and SL 104, 02/23/24, 4-6pm

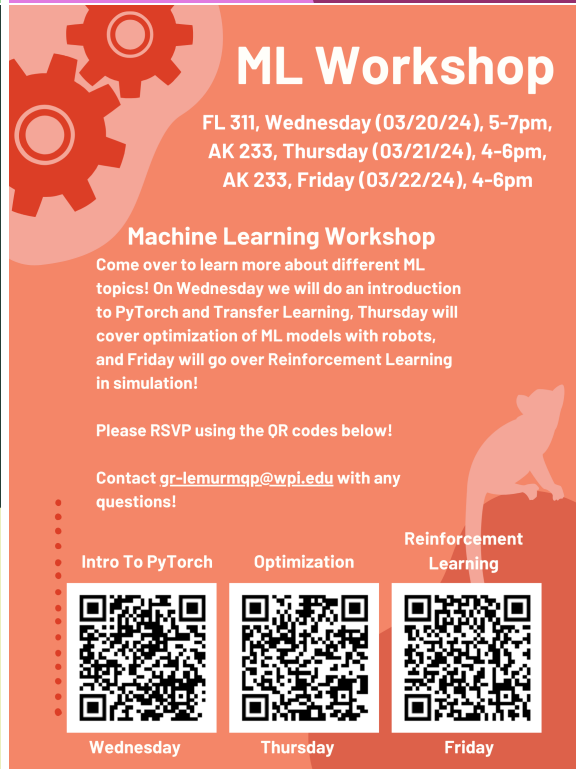
RL Workshop

Come over to learn more about Reinforcement Learning! In this workshop we will go over Q-learning and how to visualize agent-based decision making in a given map. We will go over the classic frozen lake game and using the gym library for building out an RL model.

Please RSVP using the QR codes below! Contact gr-lemurmqp@wpi.edu with any questions!

Tuesday RSVP 

Friday RSVP 

ML Workshop




FL 311, Wednesday (03/20/24), 5-7pm, AK 233, Thursday (03/21/24), 4-6pm, AK 233, Friday (03/22/24), 4-6pm


Machine Learning Workshop

Come over to learn more about different ML topics! On Wednesday we will do an introduction to PyTorch and Transfer Learning, Thursday will cover optimization of ML models with robots, and Friday will go over Reinforcement Learning in simulation!

Please RSVP using the QR codes below!

Contact gr-lemurmqp@wpi.edu with any questions!

Intro To PyTorch	Optimization	Reinforcement Learning
		
Wednesday	Thursday	Friday





Intro to Machine Learning Workshop

FL 311, Wednesday (03/27/24), 5-7pm

Workshop Topics

In this workshop we will cover how to use PyTorch and Transfer Learning to make your own model to classify whatever you want! We will then put the model on a robot with a camera, and attempt classifying objects in real time. The workshop will provide the robots and pi's that we will use in the workshop.

Participants are not expected to have AI/ML experience, and only minimal python experience.

Please RSVP using the QR code below!
Contact gr-lemurmqp@wpi.edu with any questions!

RSVP



D Workshop Surveys

Here are the links to previews of our workshop surveys:

- Introduction to PyTorch and Transfer Learning (Pre): https://wpi.yul1.qualtrics.com/jfe/preview/previewId/e3a35234-2390-402a-968f-4d585abdceff6/SV_0jpAZBFuAJIWdtY?Q_CHL=preview&Q_SurveyVersionID=current
- Introduction to PyTorch and Transfer Learning (Post): https://wpi.yul1.qualtrics.com/jfe/preview/previewId/256dabe7-d169-445b-9d72-80af9225be98/SV_0q4mmAHjANRopBc?Q_CHL=preview&Q_SurveyVersionID=current
- Introduction to Optimization (Pre): https://wpi.yul1.qualtrics.com/jfe/preview/previewId/7b06d082-06d2-4084-b2bc-6bb6c16c86b9/SV_8xnB6OqxNz87PVk?Q_CHL=preview&Q_SurveyVersionID=current
- Introduction to Optimization (Post): https://wpi.yul1.qualtrics.com/jfe/preview/previewId/806b6b1a-231c-4634-9de5-3dfae8cb4e05/SV_0SqmRCeqQuEtBNc?Q_CHL=preview&Q_SurveyVersionID=current
- Introduction to Reinforcement Learning (Pre): https://wpi.yul1.qualtrics.com/jfe/preview/previewId/67d73411-09b8-49c4-aa38-566d67b299c7/SV_8j1ERTDIMHT3JSm?Q_CHL=preview&Q_SurveyVersionID=current
- Introduction to Reinforcement Learning (Post): https://wpi.yul1.qualtrics.com/article/jfe/preview/previewId/2ccf8085-0fad-466d-94ad-ee2842a754ef/SV_2i8nMfy5YaoEbUq?Q_CHL=preview&Q_SurveyVersionID=current