Resources on Edge Computing Research for Undergraduate Students

Week Number	Topics	Description
1	Introduction to IoT	Discussion on different kinds of IoT devices, introduction to tools. A high-level overview of different components of IoT devices both at the software and hardware level.
2	Cloud Computing vs Edge Computing	Explain fundamental differences between cloud and edge computing. Understand the specific motivations behind developing these two techniques. Familiarize with the existing implementation platforms of these two schemes. **Hands-on:** Implement an end-to-end deep model in the cloud platform. Configure an edge device for deep model deployment and perform inference on it. Analyze the implementation disparities and performance evaluation.
3	Applications of Edge Computing	Learn how to use deep learning techniques to equip edge devices with intelligence using their own data. **Hands-on:** Collect data from different kinds of sensors i.e., IMU, camera integrated with edge devices. Develop a small-scale machine learning model which can take the collected data as input and get trained with it.
4	Computational Overheads in Edge Computing	Identify necessary parameters to measure the computational resource requirement, and their relative dependency on the experimental system design. **Hands-on:** ● Compute the computational complexity of a deep learning model. ● Implement the measurement procedure to benchmark the run time, memory requirement, power consumption, and inference time.
5	Model Compression Basics	Make a deep model implementable in resource-constrained edge devices, it is imperative to reduce the computation requirements of that model. Model compression can come in handy in this scenario. The basic workflow of a generalized model compression technique will be discussed.
6	Application Scenario of different Model Compression Techniques	About the motivation of originating various model compression techniques, their fundamental structural difference, their different application scenarios.

		Hands-on: Implement pruning, and quantization for two different applications (computer vision, and time-series). Discussion on the findings and possible strategies for result improvement.
7	Privacy of Edge Data: Federated Learning	Discussion on the motivation of federated learning and its applicability on managing valuable edge data with minimization of data breach threat. **Hands-on:** Familiarize students with different federated learning frameworks, implementation requirements. Implement a basic Federated averaging technique and understand the use of each building block.
8	Virtualization and Containerization	Theoretically, there is a need for a large number of clients' devices to run a federated learning experiment. Achieve this through containerization (docker) considering each container as one client. **Hands-on:** Set up Virtual machines, Docker containers, and write shell scripts.
9	Major Challenges in Implementing Federated Learning	Discuss some major research problems of federated learning i.e., data poisoning, model poisoning, etc. Gather knowledge on existing solutions, understand the challenges to find new research problems. **Hands-on:** Experiment on observing the effect of data and model poisoning attacks on federated training.
10	Merging Federated Learning with Model Compression	Formulate the pipeline to integrate model compression technique with federated learning approach to leverage the advantages of both resource efficiency and data security. **Hands-on:** Perform federated training experimentation with pruning, and quantization. Observe the effect on performance, training time, convergence trend after applying model compression. Empirically choose one suitable model compression technique with a federated learning scheme for one specific application scenario.

Resources:

Articles on Edge Computing:

Demystifying Edge Computing – Device Edge vs. Cloud Edge
What is edge computing?

Papers on Edge Computing:

- Fog Computing: A Platform for Internet of Things and Analytics
- The Emergence of Edge Computing

Federated Learning Implementation Tools:

- https://www.tensorflow.org/federated/tutorials/tutorials_overview
- https://flower.dev/

Model Compression Implementation Tools:

https://www.tensorflow.org/model_optimization/guide

Federated Learning Papers:

https://github.com/chaoyanghe/Awesome-Federated-Learning

Model Compression Papers:

https://github.com/chester256/Model-Compression-Papers

Special topics in Robotics: Robot Operating Systems (ROS) and Applications

The objective of this course is to provide students with a comprehensive understanding of Robot Operating Systems (ROS) and its application. Students will learn the fundamentals of ROS architecture and the underlying concepts such as nodes, topics, and messages. They will be able to develop and test ROS packages for various applications including robot navigation, localization, and manipulation. Additionally, students will gain knowledge about various tools and libraries available in ROS for developing robot software.

Upon completion of this course, students will be able to:

- Understand the fundamental concepts of ROS architecture.
- Create and run ROS nodes, topics, and messages.
- Develop ROS packages for various applications including robot navigation, localization, and manipulation.
- Use various tools and libraries available in ROS for developing robot software.
- Apply the knowledge gained to develop robotic applications using ROS.

Overall, this course will equip students with the necessary knowledge and skills to work with Robot Operating Systems and apply it to various real-world scenarios in robotics.

Week Number	Topics	Description
1	Introduction to Robotics and ROS	• Overview of Robotics and its applications • Introduction to ROS architecture and its components • Setting up the ROS environment on a local machine
2	ROS Topics and Messages	• Understanding ROS topics and messages • Creating and publishing messages • Subscribing to and processing messages **Hands-on:** Create a simple publisher-subscriber scheme (preferably talker-listener) and grasp the concept of topics and messages from rqt_graph.
3	ROS Nodes and Launch Files	• Creating ROS nodes in Python • Launching multiple nodes using launch files • Managing node communication **Hands-on:** Follow the documentation from the official website of ROS to create ROS nodes for specific application scenarios. Also, implement a launch file which initiates multiple ROS nodes.

4	Robot Motion Control	• Overview of robot motion control • Robot kinematics and dynamics • Controlling robot motion using ROS **Hands-on:** Experiment with ROS-enabled robotic agents and learn about the physics behind the control and dynamics of the robot. Also, create a launch file and deploy it into the robot to control the robot manually through keyboard commands.
5	Robot Localization	• Overview of robot localization • ROS interfaces for localization sensors • Using ROS to localize a robot **Hands-on:** Integrate IMU, camera, and LiDAR sensors with a ROS-compatible robot. Create specific ROS-topics for each sensor and visualize the data on RViz. Train the robot to localize itself at a given place using the sensor data.
6	Robot Mapping and Navigation	• Introduction to robot mapping • Simultaneous Localization and Mapping (SLAM) • Using ROS for robot mapping • Overview of robot navigation • Path planning and obstacle avoidance • Using ROS for robot navigation **Hands-on:** Create a map for an enclosed area with the help of LiDAR data and using the saved map try to navigate the robot to any specific place of the area autonomously.
7	Robotic and Network Simulators	Learn about the working basics of Physics based simulators and Network simulators. **Hands-on:** • Install a physics and a network simulator under ROS environment and try to extract the topic, node, and messages generated by those simulators through ROS • Design a terrain and deploy simulated robots inside of it • Design a simple network topology using the network simulator
8	Addressing Synchronization Issue during Co-simulation	• Learn about the phenomena of synchronization issue while doing co-simulation of physics and network simulators • The student is expected to formulate the project on this topic **Hands-on:** Create a co-simulation environment without any synchronization effort to experience the data transmission mismatch between physics and network simulators. Implement existing approaches to solve the issue.

9	Exploring Features of ROS2	• Why is ROS2 important? • ROS1 to ROS2 • Fundamental differences between ROS1 and ROS2 **Hands-on:** Install ROS2 and learn about the fundamental difference with ROS1 at code level. Explore the impact of master-based and masterless architecture of data transmission rate, and packet loss.
10	ROS2 QoS Policy Design	Read about the components of ROS2 QoS policy design and learn about the impact of each one of them. **Hands-on:** Choose a specific communication scenario and design a suitable QoS policy for that scenario to increase packet delivery rate and decrease latency. Deploy that designed policy with actual robots and validate the simulation performance.

Resources:

Reference papers:

- Exploring the performance of ROS2
- Simulation tools for robotics research and assessment
- CPS-Sim: Co-Simulation for Cyber-Physical Systems with Accurate Time Synchronization
- FlyNetSim: An Open Source Synchronized UAV Network Simulator based on ns-3 and Ardupilot
- RoboNetSim: An integrated framework for multi-robot and network simulation
- ROS-NetSim: A Framework for the Integration of Robotic and Network Simulators

Helpful links on ROS:

- Official website
- Tutorials
- ROS index
- ROS discourse
- Getting involved

Implementation on Robots:

- Data collection with ROSbag and analysis
- Swarm robotics with Crazyflie
- Mapping and Navigation with Turtlebot

ROS compatible bots:

- Aerial
- Ground
- Marine

Reinforcement Learning Study Guidelines for REU Students

Objective: The objective of this study plan is to provide a comprehensive introduction to Reinforcement Learning (RL) concepts and techniques and prepare students to be able to apply these techniques to solve real-world problems. The plan will cover foundational RL concepts, explore various RL algorithms, and highlight recent advancements in the field. By the end of this study plan, REU students will have gained a deep understanding of RL and its applications and be equipped with the necessary skills to pursue research in RL.

Timeline	Agenda
Week 1 **Introduction to Reinforcement Learning and Markov Decision Processes**	• Learn about the basic concepts of Reinforcement Learning (RL), such as agents, environments, states, actions, rewards, and policies. • Understand the difference between Model-Free and Model-Based RL. • Understand the difference between Off-Policy and On-Policy. • Learn about Markov Decision Processes (MDPs) and their mathematical framework for modeling RL problems. • Understand the concepts of value functions, Bellman equations, and the Bellman optimality principle. • Read chapters 1, 2, and 3 from Sutton's RL book. • Work on chapter 1, 2, and 3 Implementation (Check Python Implementation).
Week 2 **Policy Iteration and Dynamic Programming**	• Learn about Policy Iteration and its mathematical framework for solving RL problems. • Understand the concepts of dynamic programming, value iteration, and policy evaluation. • Read chapter 4 from Sutton's RL book. • Work on chapter 4 Implementation (Check Python Implementation)
Week 3 **Monte Carlo Methods and Temporal Difference**	• Learn about Monte Carlo methods for RL and their advantages over Dynamic Programming methods. • Understand the concepts of first visit and every-visit Monte Carlo algorithms. • Learn about Temporal Difference (TD) learning and its advantages over Monte Carlo methods.

	• Understand the concepts of TD (0) and TD (lambda) algorithms. • Read chapter 5 and 6 from Sutton's RL book. • Work on chapter 5 and 6 Implementation (Check Python Implementation)
Week 4 **Function Approximation and Policy Gradients**	• Learn about the challenges of RL in high-dimensional state and action spaces. • Understand the concepts of function approximation and feature engineering. • Learn about Policy Gradients (PG) methods and their advantages over Q-learning. • Understand the concepts of policy gradients, advantage functions, and the REINFORCE algorithm. • Read Paper Policy Gradient Methods for Reinforcement Learning with Function Approximation • Read chapters 9, 10, 11 and 12 from Sutton's RL book. • Work on chapters 9, 10, 11 and 12 Implementation (Check Python Implementation).
Week 5 **Deep Q-networks**	• Read the original paper on DQNs: Human-level control through deep reinforcement learning • Understand the challenges of DQNs, such as stability and exploration-exploitation trade-offs. • Implement a DQN algorithm for playing Atari games, using PyTorch and OpenAI Gym • Programming Example: Implementing DQN with PyTorch and OpenAI Gym
Week 6 **Actor-Critic Methods**	• Learn about Actor-Critic methods and their combination of value-based and policy-based RL. • Understand the concepts of Actor-Critic architectures, TD error, and the A2C algorithm. • Implement an A2C algorithm for playing Pong, using PyTorch and OpenAI Gym • Programming Example: Implementing A2C with PyTorch and OpenAI Gym

Week 7 **Fast Learning and Batch RL**	• Understand the concept of Fast Learning, Experience Replay, Sample Efficiency, Prioritized Experience Replay, Policy Distillation, and Batch RL. • Watch Stanford's Fast Learning and Batch RL lectures - CS234: Reinforcement Learning Winter 2021 • Read Paper Prioritized Experience Replay • Read Paper Off-Policy Deep Reinforcement Learning without Exploration
Week 8 **Multi-Agent RL**	• Learn about Multi-Agent RL (MARL) and its challenges and applications in games, robotics, and social dilemmas. • Understand the concepts of MARL, independent and joint learning, and the MADDPG algorithm. • Implement a MADDPG algorithm for playing a simple game with two agents, using PyTorch and OpenAI Gym • Programming Example: Implementing MADDPG with PyTorch and OpenAI Gym.
Week 9 **Recent Advances in RL**	• Learn about recent advances in RL research, such as Deep RL, Meta RL, Imitation Learning, Inverse RL, and Hierarchical RL • Read the selected RL papers (Check next page).
Week 10 **Work on RL Projects**	• Learn to design a project with problem formulation, algorithm selection, parameter selection, and representation design---fits together into a complete solution, and how to make appropriate choices when deploying RL in the real world. • A Complete Reinforcement Learning System (Capstone) | Coursera • Work on listed Robotics Projects.

Resources

- Reinforcement Learning: An Introduction" by Richard S. Sutton and Andrew G. Barto.
- Python Implementation of Reinforcement Learning: An Introduction
- Spinning Up in Deep RL!
- Key Papers in Deep RL — Spinning Up documentation
- Recent RL Papers from Conference

RL Libraries

- OpenAI Gym: A toolkit for developing and comparing reinforcement learning algorithms.
- Stable Baselines 3 is a learning library based on the Gym API. It is designed to cater to complete beginners in the field who want to start learning things quickly.
- RL Baselines3 Zoo builds upon SB3, containing optimal hyperparameters for Gym environments as well as code to easily find new ones.
- Tianshou is a learning library that's geared towards very experienced users and is designed to allow for ease in complex algorithm modifications.
- RLlib is a learning library that allows for distributed training and inferencing and supports an extraordinarily large number of features throughout the reinforcement learning space.
- PettingZoo is like Gym, but for environments with multiple agents.

RL Courses

- Reinforcement Learning Specialization
- DeepMind Reinforcement Learning Lecture Series 2021
- UC Berkley CS 294 Deep Reinforcement Learning, Spring 2017
- Stanford CS234: Reinforcement Learning | Winter 2019 - YouTube

Selected Papers to Read

- Playing Atari with Deep Reinforcement Learning
- Learning to Adapt in Dynamic, Real-World Environments through Meta-Reinforcement Learning
- Apprenticeship Learning via Inverse Reinforcement Learning
- Deep Reinforcement Learning from Self-Play in Imperfect-Information Games
- Proximal Policy Optimization Algorithms
- Deep Reinforcement Learning with Double Q-learning
- Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor
- Smith & Gasser, The Development of Embodied Cognition: Six Lessons from Babies
- Silver, Huang et al., Mastering the Game of Go with Deep Neural Networks and Tree Search
- Bojarski et al., *End to End Learning for Self-Driving Cars*
- Ross et al., Learning Monocular Reactive UAV Control in Cluttered Natural Environments
- Abbeel et al., Apprenticeship Learning via Inverse Reinforcement Learning
- Ziebart et al., Maximum Entropy Inverse Reinforcement Learning
- Lillicrap et al., Continuous control with deep reinforcement learning
- Lake et al., Building Machines That Learn and Think Like People
- Finn et al., Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks
- Silver et al., Mastering the Game of Go without Human Knowledge

Articles to Read

- Math Behind Reinforcement Learning, the Easy Way | by Ziad SALLOUM | Towards Data Science
- Hands-On Reinforcement Learning Course: Part 1 | by Pau Labarta Bajo | Towards Data Science
- States, Observation, and Action Spaces in Reinforcement Learning | by #Cban2020 | The Startup | Medium
- A hands-on introduction to deep reinforcement learning using Unity ML-Agents

Applying RL in Robotics

- Reinforcement Learning with ROS and Gazebo - Artificial Intelligence Research
- Benchmarking Reinforcement Learning Algorithms on Real-World Robots
- A ROS2-based framework for TurtleBot3 DRL autonomous navigation
- ML-Agents with Unity
- High Dimensional Planning and Learning for Off-Road Driving
- Deep Reinforcement Learning for Autonomous Driving: A Survey
- Using TurtleBot in Deep Reinforcement Learning

Important

If you are not familiar with Machine Learning, please consider building a foundation first, before diving into Reinforcement Learning.

- Machine Learning Specialization
- Mathematics for Machine Learning and Data Science Specialization