Deep Reinforcement Learning Workshop: One-Page Overview

Workshop Title:
From Q-Learning to Multi-Agent Systems: A Hands-On Deep RL Journey

Duration: 3 - 4 Hours
Audience: ML engineers, game developers, robotics enthusiasts, AI researchers
Focus: Practical implementation of RL algorithms from basics to state-of-the-art

Workshop Structure & Units

Unit	Topic	Type	Duration
Foundations			
1	Introduction to Deep RL (Concepts & Theory)	Lecture	45 min
Bonus 1	Hands-on: Train Huggy (First RL Agent)	Lab	60 min
Live 1	Course Mechanics, Q&A, Huggy Showcase	Interactive	30 min
2	Q-Learning Fundamentals	Lecture + Code	90 min
Deep Q-Learning			
3	Deep Q-Learning with Atari Games	Lab	120 min
Bonus 2	Hyperparameter Tuning with Optuna	Lab	60 min
4	Policy Gradient Methods with PyTorch	Lecture + Code	90 min
Advanced & Multi-Agent			
5	Unity ML-Agents Introduction	Lab	90 min
6	Actor-Critic Methods in Robotics	Lecture + Lab	90 min
7	Multi-Agent RL and AI vs AI	Lecture + Demo	60 min
8.1	Proximal Policy Optimization (PPO) Theory	Lecture	45 min
8.2	PPO with Doom (Hands-on Implementation)	Lab	90 min

Unit	Topic	Type	Duration
Bonus 3	Advanced RL Topics (RLHF, Model-Based RL)	Lecture	45 min
Bonus 5	Imitation Learning with Godot RL Agents	Lab	60 min

Key Learning Objectives

- Master the RL problem formulation: agent, environment, rewards, policies
- Implement value-based methods (Q-Learning, DQN) from scratch
- Build policy gradient agents (REINFORCE, PPO) using PyTorch
- Train agents in complex environments (Atari, Unity, Robotics simulators)
- Understand actor-critic architectures and their advantages
- Deploy multi-agent systems and competitive AI
- Apply modern RL to real-world problems (games, robotics, LLMs)
- Debug and optimize RL training pipelines

Core Concepts by Track

Track 1: Value-Based Methods (Units 1-3)

Fundamentals (Unit 1):

- MDP framework: States, actions, rewards, transitions
- Exploration vs exploitation trade-off
- Bellman equations and value functions
- Deep RL = RL + Deep Neural Networks

Q-Learning (Unit 2):

- Temporal Difference learning
- Q-table and Q-function approximation
- Epsilon-greedy exploration strategy
- Hands-on: Taxi-v3, FrozenLake environments

Deep Q-Networks (Unit 3):

- Function approximation with neural networks
- Experience replay buffer
- Target networks and stability tricks
- Hands-on: Atari games (Pong, Breakout, Space Invaders)
- Double DQN, Dueling DQN extensions

Track 2: Policy-Based Methods (Units 4, 8)

Policy Gradients (Unit 4):

- Direct policy optimization
- REINFORCE algorithm and Monte Carlo sampling
- Baseline functions and variance reduction
- PyTorch implementation from scratch
- Hands-on: CartPole, LunarLander

Proximal Policy Optimization (Unit 8):

- Trust region methods and clipping
- PPO objective function breakdown
- Advantage estimation (GAE)
- Hands-on: Train agents in VizDoom
- Comparison to TRPO and other policy methods

Track 3: Actor-Critic & Advanced (Units 6, Bonus 3)

Actor-Critic Methods (Unit 6):

- Combining value and policy learning
- A2C (Advantage Actor-Critic)
- A3C (Asynchronous A3C)
- Hands-on: Robotic arm manipulation, locomotion tasks
- Continuous action spaces

Advanced Topics (Bonus 3):

- Model-based RL (world models, planning)
- Offline RL and conservative methods
- RLHF for LLMs (connection to Unit 8 content)
- Hierarchical RL and options framework
- Curiosity-driven exploration

Track 4: Applications & Ecosystems (Units 5, 7, Bonus 1, 5)

Unity ML-Agents (Unit 5):

- Setting up Unity + ML-Agents toolkit
- Training agents in 3D environments
- Curriculum learning strategies

- Building custom Unity environments
- Hands-on: Train agents in Unity games

Multi-Agent Systems (Unit 7):

- Cooperative vs competitive settings
- Self-play and population-based training
- Nash equilibria in game theory
- Hands-on: Soccer, Sumo, competitive games
- Emergent behaviors from multi-agent training

Huggy Environment (Bonus 1):

- First hands-on experience with RL
- Training a cute dog agent to fetch sticks
- Reward shaping and environment design
- Visual debugging and training monitoring

Godot + Imitation Learning (Bonus 5):

- Godot RL Agents framework
- Behavioral cloning from demonstrations
- Combining imitation + RL (DAgger, GAIL)
- When to use imitation vs pure RL

Practical Implementation Stack

Core Libraries:

- Stable-Baselines3 (high-level RL algorithms)
- Gymnasium/Gym (environment interfaces)
- PyTorch (neural network implementation)
- TensorBoard (training visualization)

Environments:

- OpenAI Gym classic control
- Atari 2600 games (ALE)
- Unity ML-Agents (3D simulations)
- PyBullet (robotics)
- VizDoom (FPS game)
- Godot (game engine)

Tools:

- Optuna (hyperparameter optimization)
- Weights & Biases (experiment tracking)
- Hugging Face Hub (model sharing)

Hands-On Projects

1. **Train Huggy** (Bonus 1): Your first RL agent in 30 minutes
2. **Atari Mastery** (Unit 3): DQN agent beating classic games
3. **Lunar Lander** (Unit 4): Policy gradients for continuous control
4. **Robot Arm** (Unit 6): Actor-critic for manipulation tasks
5. **AI vs AI Soccer** (Unit 7): Multi-agent competitive training
6. **Doom Agent** (Unit 8): PPO in complex 3D environment
7. **Custom Unity Game** (Unit 5): Build and train your own environment

Hyperparameter Tuning Workshop (Bonus 2)

Why Tuning Matters:

- RL is notoriously sensitive to hyperparameters
- Learning rate, batch size, gamma, entropy coefficient
- Manual tuning is time-consuming and suboptimal

Optuna Introduction:

- Bayesian optimization for RL
- Defining search spaces (continuous, categorical)
- Pruning unpromising trials early
- Hands-on: Tune PPO on CartPole, compare results

Best Practices:

- What to tune vs what to keep fixed
- Sample-efficiency considerations
- Parallel trial execution strategies

Workshop Deliverables

For Each Participant:

- Complete RL agent implementations (Q-Learning, DQN, PPO, A2C)
- Portfolio of trained agents with demo videos
- Hugging Face Hub profile with shared models

- Jupyter notebooks with annotations
- Custom environment (Unity or Godot)

Certification Projects (Choose 1):

- Train agent to superhuman performance on Atari game
- Multi-agent system with emergent cooperative behavior
- Robotics manipulation task with actor-critic
- Custom game with self-play trained AI opponent

Prerequisites & Setup

Required Knowledge:

- Python programming (intermediate level)
- Basic neural networks and backpropagation
- NumPy, basic linear algebra
- PyTorch fundamentals (or willingness to learn)

Pre-Workshop Setup:

- Python 3.8+ environment
- Install: `stable-baselines3`, `gymnasium`, `pytorch`
- GPU recommended (Colab free tier acceptable)
- Unity Hub + ML-Agents (for Unit 5)
- GitHub account + Hugging Face account

Optional Reading:

- Sutton & Barto: "Reinforcement Learning: An Introduction"
- DeepMind blog posts on AlphaGo, MuZero
- OpenAI Spinning Up documentation

The Deep RL Landscape (Covered in Workshop)

When to Use Each Method:

Algorithm	Best For	Limitations
Q-Learning	Discrete actions, simple spaces	Doesn't scale to continuous
DQN	Atari games, discrete control	Sample inefficient
Policy Gradient	Continuous control, stochastic policies	High variance

---

Algorithm	Best For	Limitations
A2C/A3C	Fast training, parallel envs	Sensitive to hyperparameters
PPO	General purpose, robust	Slower than A3C
SAC	Continuous, sample-efficient	Complex implementation

RL vs Supervised Learning:

- No labeled data, only rewards
- Credit assignment problem (delayed rewards)
- Exploration-exploitation dilemma
- Non-stationary data distribution

Success Stories & Applications

Games:

- AlphaGo, AlphaZero (superhuman Go/Chess)
- OpenAI Five (Dota 2)
- AlphaStar (StarCraft II)

Robotics:

- Robotic manipulation and grasping
- Quadruped locomotion
- Drone navigation

Real-World Systems:

- Data center cooling optimization (Google)
- Traffic signal control
- Autonomous driving (simulation to reality transfer)
- LLM fine-tuning (ChatGPT, Claude with RLHF)

Resources & Community

During Workshop:

- Discord server for real-time help
- Daily office hours with instructors
- Shared Colab notebooks for each unit
- Recording of all lectures

Post-Workshop:

- Hugging Face Deep RL Course (free, self-paced continuation)
- Weekly study group sessions
- Project showcase and feedback
- Alumni network and job referrals

Recommended Follows:

- OpenAI research blog
- DeepMind publications
- Spinning Up in Deep RL (OpenAI)
- /r/reinforcementlearning subreddit

Main Source: https://huggingface.co/learn/deep-rl-course/unit0/introduction

Contact & Registration

Instructor: Mehdi Maleki
Email: mosioc79@gmail.com
Format: In-person (preferred) or virtual with live coding sessions

The RL Mindset

> "Reinforcement Learning is the closest thing we have to general intelligence. Unlike supervised learning where we show the correct answer, in RL the agent must discover what works through trial and error, just like humans and animals learn."