Introduction to Reinforcement Learning

Dr. Jules Raymond Kala

@

The 3rd Google TensorFlow College Outreach Bootcamp and FEDGEN Mini-Workshop

Date: 11th to 13th December, 2023

Sponsored

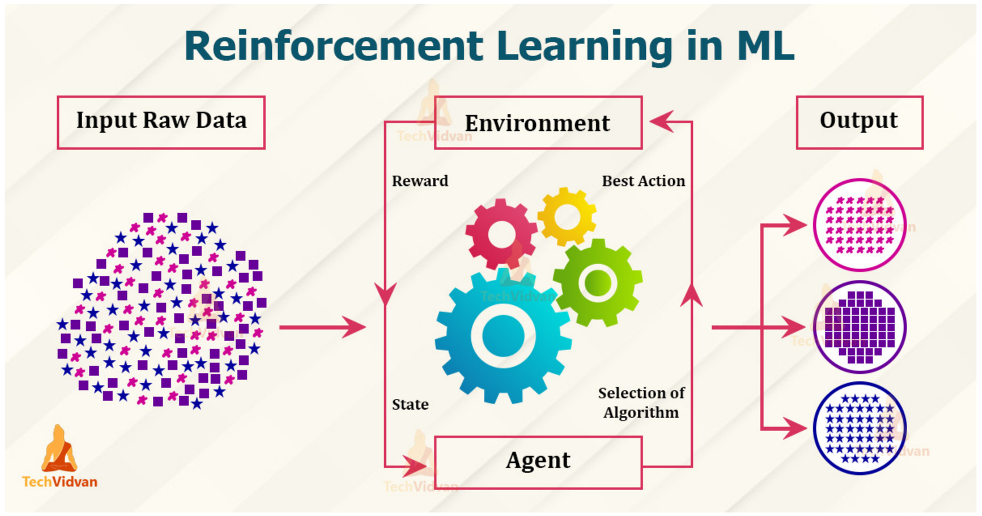
By

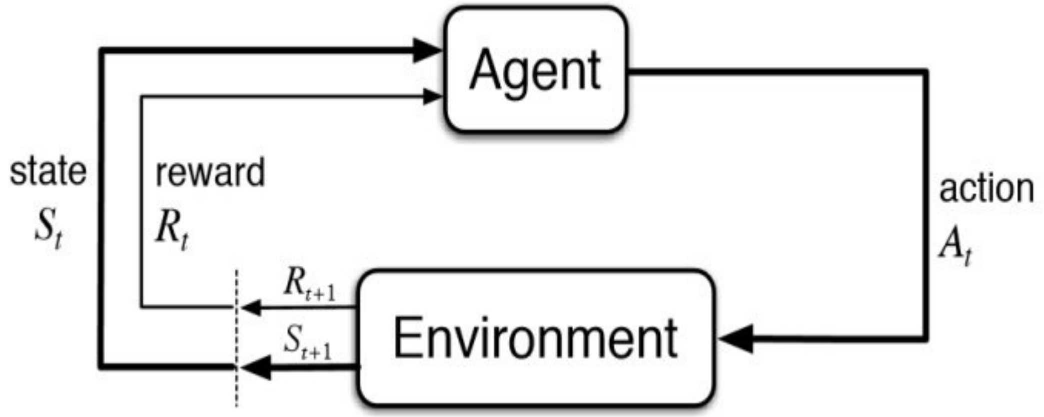
Summary

I. Introduction

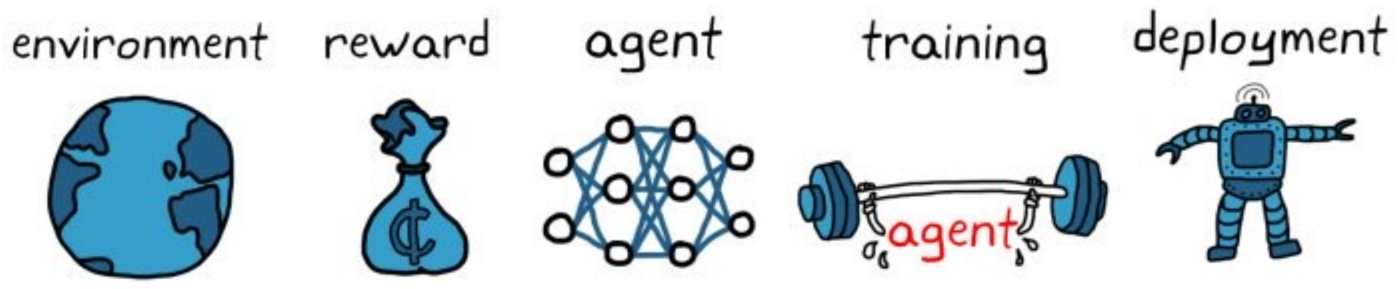
- Reinforcement Learning (RL) is a paradigm in artificial intelligence focused on training agents to make sequential decisions by interacting with an environment.
- RL relies on trial and error, with agents receiving feedback in the form of rewards or penalties based on their actions.
- RL draws inspiration from how humans and animals learn by interacting with the world

- **Components:**
  - *Agent:* Makes decisions and takes actions.
  - *Environment:* External system in which the agent operates.
  - *Actions:* Choices available to the agent.
  - *Rewards:* Numerical signals indicating the desirability of the agent's actions.



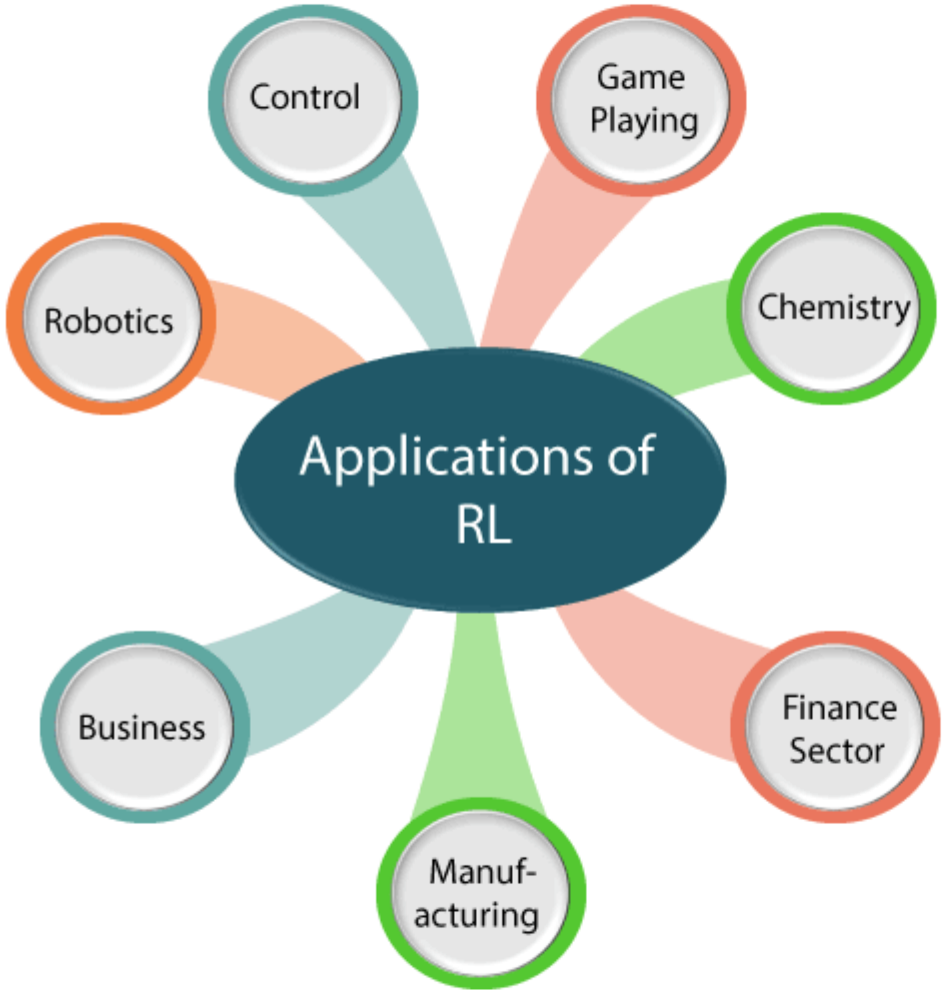


- **Objective:** Discovering an optimal policy— a strategy guiding the agent to take actions that lead to the highest cumulative reward.



Reinforcement Learning Framework

- **Applications:**
  - Robotics and autonomous systems
  - Game playing
  - Resource allocation
  - Finance
  - Healthcare



- **Achievements:** Notable successes include mastering complex games like Go and chess, as well as applications in optimizing resource allocation, finance, and healthcare.
- **Ongoing Developments:** Researchers are exploring innovative algorithms, strategies for real-world complexities, and methods to scale RL to more challenging problems (recent application to LLM).
- **Interdisciplinary Nature:** Reinforcement learning sits at the intersection of computer science, neuroscience, and control theory, offering promise for developing intelligent systems capable of autonomous decision-making in dynamic and uncertain environments.

Benefits of Reinforcement Learning

Reinforcement learning is applicable to a wide range of complex problems that cannot be tackled with other machine learning algorithms. RL is closer to artificial general intelligence (AGI), as it possesses the ability to seek a long-term goal while exploring various possibilities autonomously.

- **Focuses on the problem as a whole**.
- **Does not need a separate data collection step**.
- **Works in dynamic, uncertain environments**.

Challenges with Reinforcement Learning

While RL algorithms have been successful in solving complex problems in diverse simulated environments, their adoption in the real world has been slow.

RL agent needs extensive experience.

Delayed rewards.

Lack of interpretability.

Reinforcement Learning vs. Supervised Learning vs Unsupervised

- Supervised learning is a paradigm of machine learning that requires a knowledgeable supervisor to curate a labelled dataset and feed it to the learning algorithm.
- RL is a separate paradigm of machine learning. RL does not require a supervisor or a pre-labelled dataset; instead, it acquires training data in the form of experience by interacting with the environment and observing its response.
- Unsupervised learning is a technique that determines patterns and associations in unlabeled data.

II. Exploration, Q-Learning and Policy Gradients

- Exploration
- Q-Learning
- Policy Gradients

Exploration

Exploration in RL is a crucial aspect that deals with how an agent should explore the environment to discover optimal or near-optimal policies.

Epsilon-Greedy:	The agent selects the action with the highest estimated value with probability $1-\epsilon$.
Softmax Exploration:	The probability of selecting an action is determined by the softmax function applied to the estimated action values.
Upper Confidence Bound (UCB):	The agent selects actions based on their upper confidence bounds, taking into account both the estimated value and the uncertainty in the estimate.
Thompson Sampling:	The agent maintains a distribution over possible models of the environment.
Bootstrapped DQN:	In deep reinforcement learning, multiple value estimates (Q-networks) are maintained.
Count-Based Exploration:	The agent maintains counts of how often each state or state-action pair has been visited.
Intrinsic Motivation:	The agent is equipped with an intrinsic reward signal that encourages it to explore novel or uncertain states.
Sparse Rewards and Shaping:	Modifying the reward structure to provide sparse rewards for accomplishing high-level goals can encourage exploration.
Parameter Noise:	Injecting noise into the parameters of the policy or value function during training can encourage the agent to explore different regions of the action space.
Model-Based Methods:	Instead of learning directly from interactions, the agent builds a model of the environment and plans in this model.

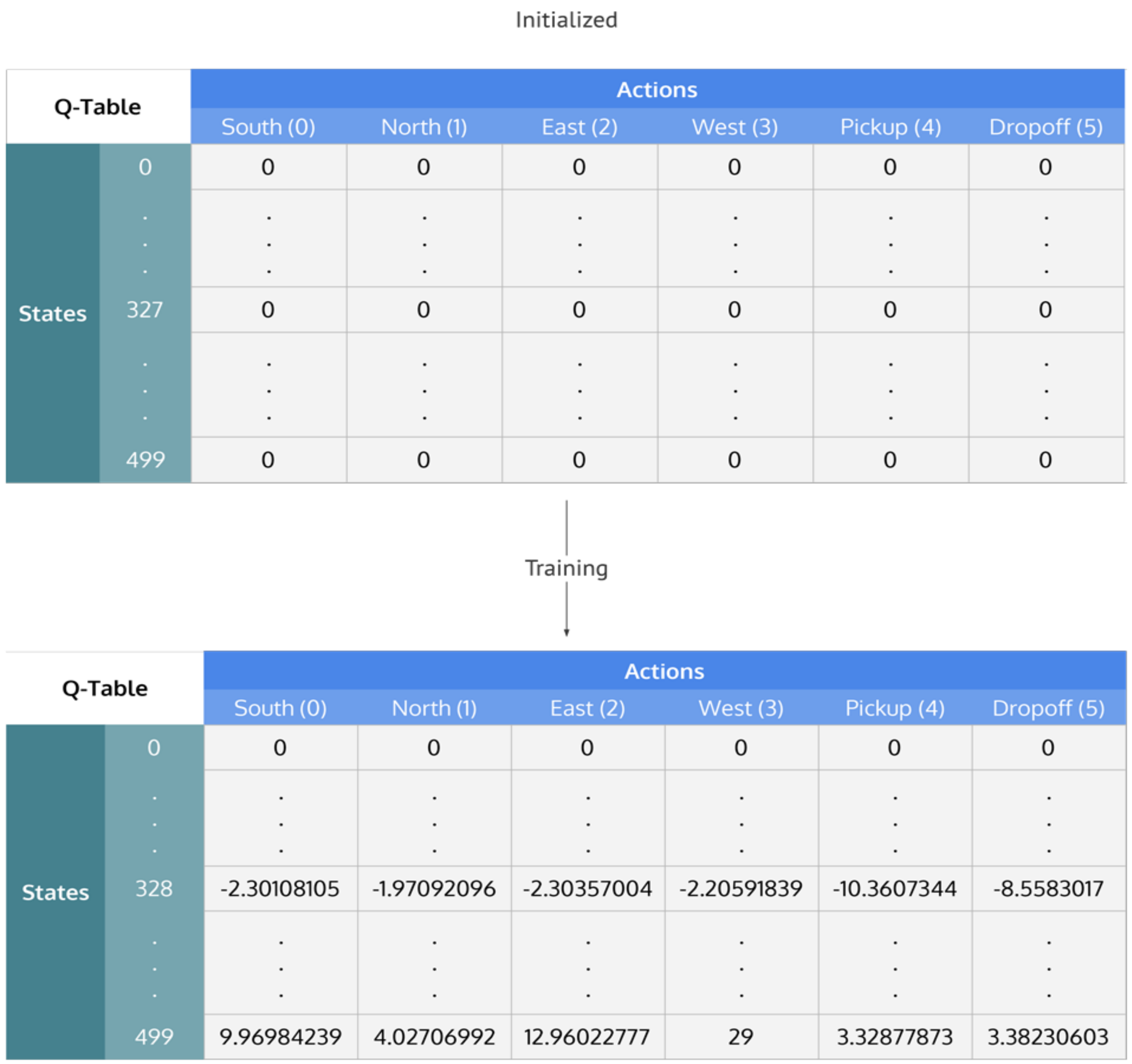
Q-Learning

- **Objective:**
  - Q-learning aims to learn the optimal action-value function, $Q(s, a)$, which represents the expected cumulative reward of taking action a in state s and following the optimal policy thereafter.
- **Update Rule:**
  - The Q-learning update rule is based on the Bellman optimality equation:
    $Q(s,a) \leftarrow Q(s,a) + \alpha\left[R + \gamma \max_{a'} Q(s',a') - Q(s,a)\right]$
    where R is the immediate reward, γ is the discount factor, s' is the next state, and α is the learning rate.
- **Exploration:**
  - Q-learning typically uses an epsilon-greedy strategy for exploration. With probability ϵ, the agent chooses a random action, and with probability $1 - \epsilon$, it chooses the action with the maximum Q-value.
- **Off-Policy:**
  - Q-learning is an off-policy algorithm, meaning it updates its Q-values based on the maximum Q-value of the next state regardless of the action taken.

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\Bigg(\underbrace{\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}_{\text{new value (temporal difference target)}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \Bigg)}^{\text{temporal difference}}$$

Breaking it down into steps, we get

- Initialize the Q-table by all zeros.
- Start exploring actions: For each state, select any one among all possible actions for the current state (S).
- Travel to the next state (S') as a result of that action (a).
- For all possible actions from the state (S') select the one with the highest Q-value.
- Update Q-table values using the equation.
- Set the next state as the current state.
- If goal state is reached, then end and repeat the process.


Initialized
Training

Q-Table		Actions					
		South (0)	North (1)	East (2)	West (3)	Pickup (4)	Dropoff (5)
States	0	0	0	0	0	0	0
	⋮	⋮	⋮	⋮	⋮	⋮	⋮
	327	0	0	0	0	0	0
	⋮	⋮	⋮	⋮	⋮	⋮	⋮
	499	0	0	0	0	0	0

Q-Table		Actions					
		South (0)	North (1)	East (2)	West (3)	Pickup (4)	Dropoff (5)
States	0	0	0	0	0	0	0
	⋮	⋮	⋮	⋮	⋮	⋮	⋮
	328	-2.30108105	-1.97092096	-2.30357004	-2.20591839	-10.3607344	-8.5583017
	⋮	⋮	⋮	⋮	⋮	⋮	⋮
	499	9.96984239	4.02706992	12.96022777	29	3.32877873	3.38230603

Exercise:

Consider a simple 3×3 grid world where an agent starts at the top-left corner (state S) and has to reach the goal at the bottom-right corner (state G). The agent can take actions: "Up," "Down," "Left," and "Right." Each action moves the agent one step in the corresponding direction. The environment has a reward of -1 for each step, and reaching the goal state has a reward of +10.

Initialize a Q-table for this environment and apply Q-learning to find the optimal policy. Assume a discount factor (γ) of 0.9 and a learning rate (α) of 0.1. Perform Q-learning for a few iterations and update the Q-values accordingly.

Exercise 2.1

For each of the following scenarios, state if it is an example of supervised or unsupervised learning. Explain your answers. In cases of ambiguity, pick one and explain why you picked it.

a. A recommendation system on a social network that recommends potential friends to a user
b. A system in a news site that divides the news into topics
c. The Google autocomplete feature for sentences
d. A recommendation system on an online retailer that recommends to users what to buy based on their past purchasing history
e. A system in a credit card company that captures fraudulent transactions

Policy Gradients

Policy gradients are a class of algorithms used in reinforcement learning (RL) to learn the policy directly, which is the strategy that the agent uses to select actions in different states of the environment. Instead of estimating the value function (as in value-based methods like Q-learning), policy gradient methods optimize the policy by directly adjusting its parameters.

The following elements are the components of policy gradients in reinforcement learning.

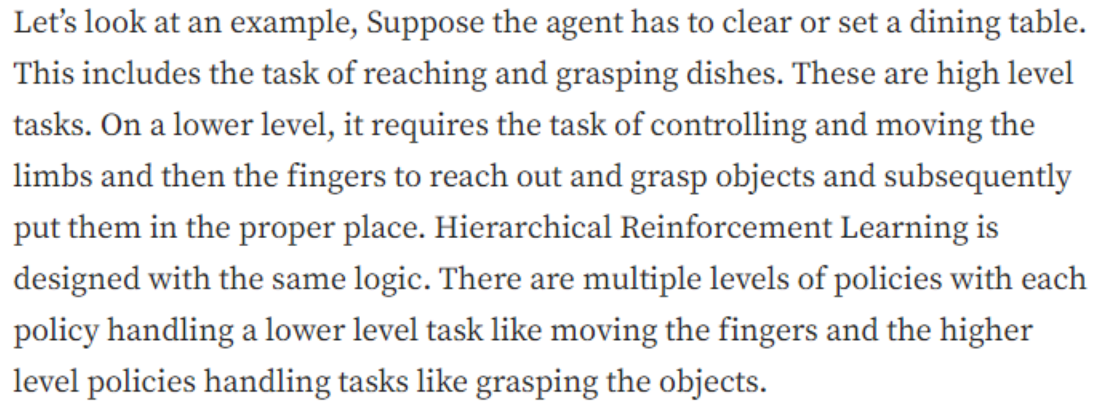
- **Policy Representation:**
- **Objective Function:**
- **Policy Gradient Theorem:**
- **REINFORCE Algorithm:**
- **Actor-Critic Methods:**
- **Policy Parameterization:**
- **Entropy Regularization:**
- **Batch Policy Gradient Methods:**
- **Trust Region Policy Optimization (TRPO) and Proximal Policy Optimization (PPO):**

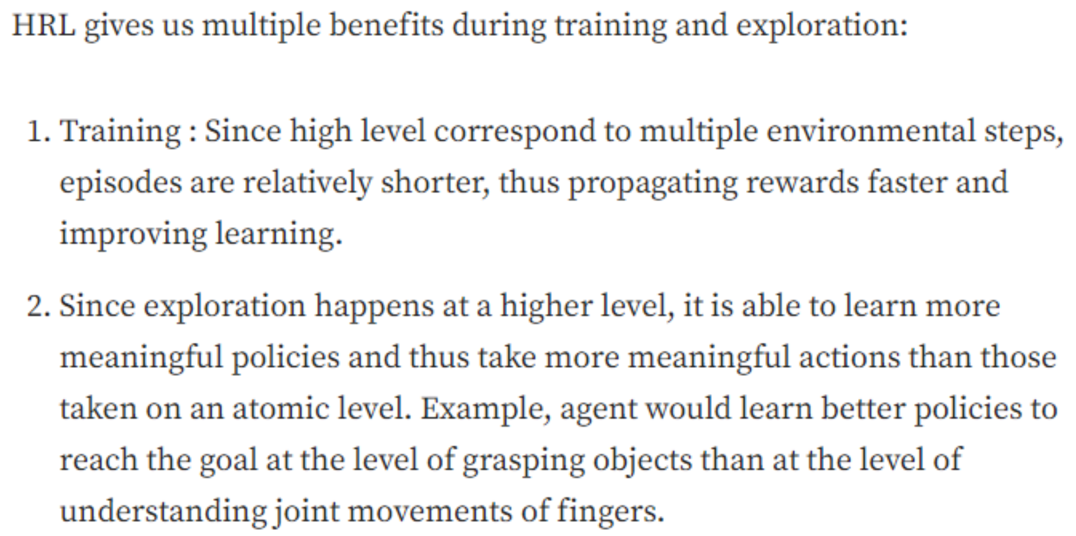
Evaluating Reinforcement Learning Algorithms

- How can we evaluate a reinforcement learning algorithm?
- Is it possible to have overfitting with reinforcement learning?

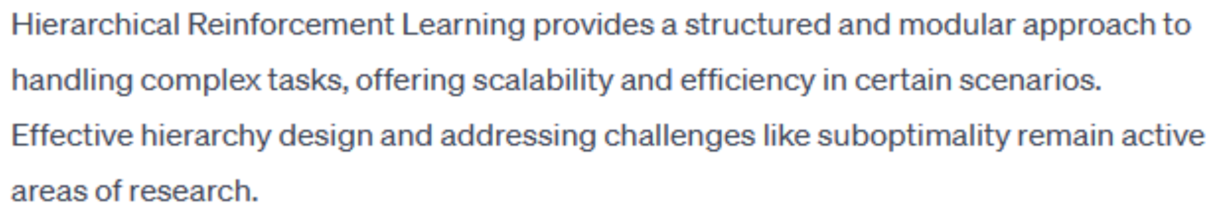
III. Heirarchical Reinforcement Learning

Goal: Hierarchical Reinforcement Learning decomposes long horizon decision making process into simpler sub-tasks.


Let's look at an example, Suppose the agent has to clear or set a dining table. This includes the task of reaching and grasping dishes. These are high level tasks. On a lower level, it requires the task of controlling and moving the limbs and then the fingers to reach out and grasp objects and subsequently put them in the proper place. Hierarchical Reinforcement Learning is designed with the same logic. There are multiple levels of policies with each policy handling a lower level task like moving the fingers and the higher level policies handling tasks like grasping the objects.


HRL gives us multiple benefits during training and exploration:

1. Training : Since high level correspond to multiple environmental steps, episodes are relatively shorter, thus propagating rewards faster and improving learning.
2. Since exploration happens at a higher level, it is able to learn more meaningful policies and thus take more meaningful actions than those taken on an atomic level. Example, agent would learn better policies to reach the goal at the level of grasping objects than at the level of understanding joint movements of fingers.


Hierarchical Reinforcement Learning provides a structured and modular approach to handling complex tasks, offering scalability and efficiency in certain scenarios. Effective hierarchy design and addressing challenges like suboptimality remain active areas of research.

Hierarchy Levels:	High-Level Policy (HL): Responsible for selecting among different subtasks or options. Low-Level Policy (LL): Executes the chosen subtask and provides detailed actions at a lower temporal scale.
Options:	Each option encapsulates a series of primitive actions, creating reusable, semi-autonomous subtasks.
Intra-Option Policies:	The intra-option policy provides detailed actions during the execution of the option.
Termination Conditions:	Options have termination conditions indicating when the option should end and control should return to the high-level policy.
Learning at Multiple Levels:	High-level policy learns to select options based on the current state and overall task. Low-level policy learns to execute selected options, focusing on detailed subtask accomplishment.
Intrinsic and Extrinsic Rewards:	Intrinsic rewards associated with successful subtask completion or progress within options. Extrinsic rewards are external rewards received from the environment based on overall task performance.
Learning Objectives:	High-level policy learning aims to make effective choices of options. Low-level policy learning focuses on executing options optimally to achieve subtasks.
Feasibility and Suboptimality:	HRL offers a more feasible exploration strategy in high-dimensional state spaces. Trade-off exists between suboptimality introduced by hierarchy and potential efficiency gains.
Transfer Learning:	HRL facilitates transfer learning, allowing knowledge from solving one task to expedite learning in related tasks.
Applications	Successfully applied in robotics, natural language processing, and game playing where tasks involve hierarchies of subtasks.
Challenges	Designing effective hierarchies is challenging; incorrect designs may lead to suboptimal solutions. Learning to switch between options at the right time is a non-trivial problem.
Algorithms	Various algorithms address hierarchical RL, including options framework, MAXQ, Feudal Networks, and deep reinforcement learning techniques for both high-level and low-level policies.

IV. Model Based Reinforcement Learning

- **Objective:**
  - Focuses on learning an explicit model of the environment, capturing dynamics and state transitions.
- **Components:**
  - Involves learning a transition model (how the environment evolves over time) and/or a reward model.
- **Planning:**
  - After learning the model, planning algorithms can be used to simulate and optimize future actions.
- **Sample Efficiency:**
  - Can be more sample-efficient compared to model-free methods in situations where real-world interactions are costly.
- **Advantages:**
  - Effective in situations where the environment is well-understood and modeling is feasible.
  - Suitable for scenarios where real-world interactions are expensive or limited.

- **Challenges:**
  - The accuracy of the learned model is critical for successful planning.
  - Planning using a model can be computationally expensive, especially in large state spaces.
- **Applications:**
  - Often used in robotics, control systems, and scenarios where a precise understanding of the environment is crucial.
- **Methods:**
  - Includes algorithms like Model Predictive Control (MPC) and various approaches for learning transition models and reward models.
- **Trade-offs:**
  - Balances the trade-off between the computational cost of planning and the efficiency gained through accurate modeling.
- **Combination with Policy/Value Methods:**
  - Can be combined with policy-based or value-based methods to enhance learning efficiency.
- **Suitability:**
  - Suitable for problems where understanding the underlying dynamics is essential and where planning with a model is computationally viable.

Questions

- What is model based reinforcement learning?
- What is a model?
- How is the model used?

V. Multi Agent Reinforcement Learning

- **Definition:**
  - In Multi-Agent Reinforcement Learning (MARL), multiple agents interact with each other and their environment, aiming to learn policies that collectively optimize a shared or individual objective.
- **Agents:**
  - Multiple autonomous agents, each making decisions independently based on its observations and possibly the observed actions of other agents.
- **Joint and Individual Objectives:**
  - Agents may have a shared global objective or individual objectives, leading to a spectrum of collaborative and competitive scenarios.
- **Observations and Actions:**
  - Each agent receives observations from the environment and selects actions based on its policy.
- **Interaction:**
  - Agents' actions influence the state of the environment, and the environment's state influences the observations of all agents.

- **Learning Paradigms:**
  - Cooperative: Agents collaborate to achieve a common goal.
  - Competitive: Agents have conflicting objectives, leading to competition.
  - Mixed: A combination of cooperative and competitive elements.
- **Communication:**
  - Agents may communicate with each other to share information, coordinate actions, or improve learning.
- **Exploration-Exploitation:**
  - Agents face the challenge of balancing exploration and exploitation, considering the actions and strategies of other agents.
- **Centralized vs. Decentralized Learning:**
  - Centralized learning involves a global learner that observes the actions and states of all agents.
  - Decentralized learning involves agents learning independently, often without direct knowledge of the global state.
- **Joint and Decentralized Policies:**
  - Joint policies represent a single policy for all agents.
  - Decentralized policies allow each agent to have its own policy.

- **Emergent Behavior:**
  - The interaction of multiple agents can lead to emergent behaviors that may be unpredictable based on individual agent policies.
- **Algorithms:**
  - Various algorithms address MARL, including independent Q-learning, multi-agent deep reinforcement learning, and algorithms specific to cooperative or competitive scenarios.
- **Communication Strategies:**
  - Explicit Communication: Agents communicate explicitly through messages.
  - Implicit Communication: Agents influence each other's behavior without explicit communication.
- **Applications:**
  - Used in various domains such as robotics, traffic management, game playing, and multi-robot systems.
- **Challenges:**
  - Non-stationarity: The environment is dynamic due to the actions of other learning agents.
  - Scalability: Handling a large number of agents can be computationally challenging.

- **Social Dilemmas:**
  - Agents may face social dilemmas where individual rationality conflicts with group rationality.
- **Learning from Others:**
  - Agents may learn from the policies and behaviors of other agents, leading to the possibility of knowledge transfer.

Multi-Agent Reinforcement Learning is a field that explores the dynamics and challenges that arise when multiple learning agents interact. It finds applications in scenarios where multiple decision-makers influence each other and the overall system behavior.

Implementation Requirements

- Tensorflow 2.10.1
- Keras-rl
- Gym
- Pettingzoo
- Dopamine-rl

References

- Morales M. Grokking deep reinforcement learning. Manning Publications; 2020 Nov 10.
- Lapan M. Deep Reinforcement Learning Hands-On: Apply modern RL methods, with deep Q-networks, value iteration, policy gradients, TRPO, AlphaGo Zero and more. Packt Publishing Ltd; 2018 Jun 21.
- Bertsekas D. Reinforcement learning and optimal control. Athena Scientific; 2019 Jul 1.