Intelligence System:

Reinforcement Learning in Machine Learning

សាកលវិទ្យាល័យន័រតុន
NORTON UNIVERSITY

Intelligence System Development

2024 – 2025
Y4E1 – DCS – NU

By: SEK SOCHEAT

Advisor to DCS and Lecturer

Mobile: 017 879 967

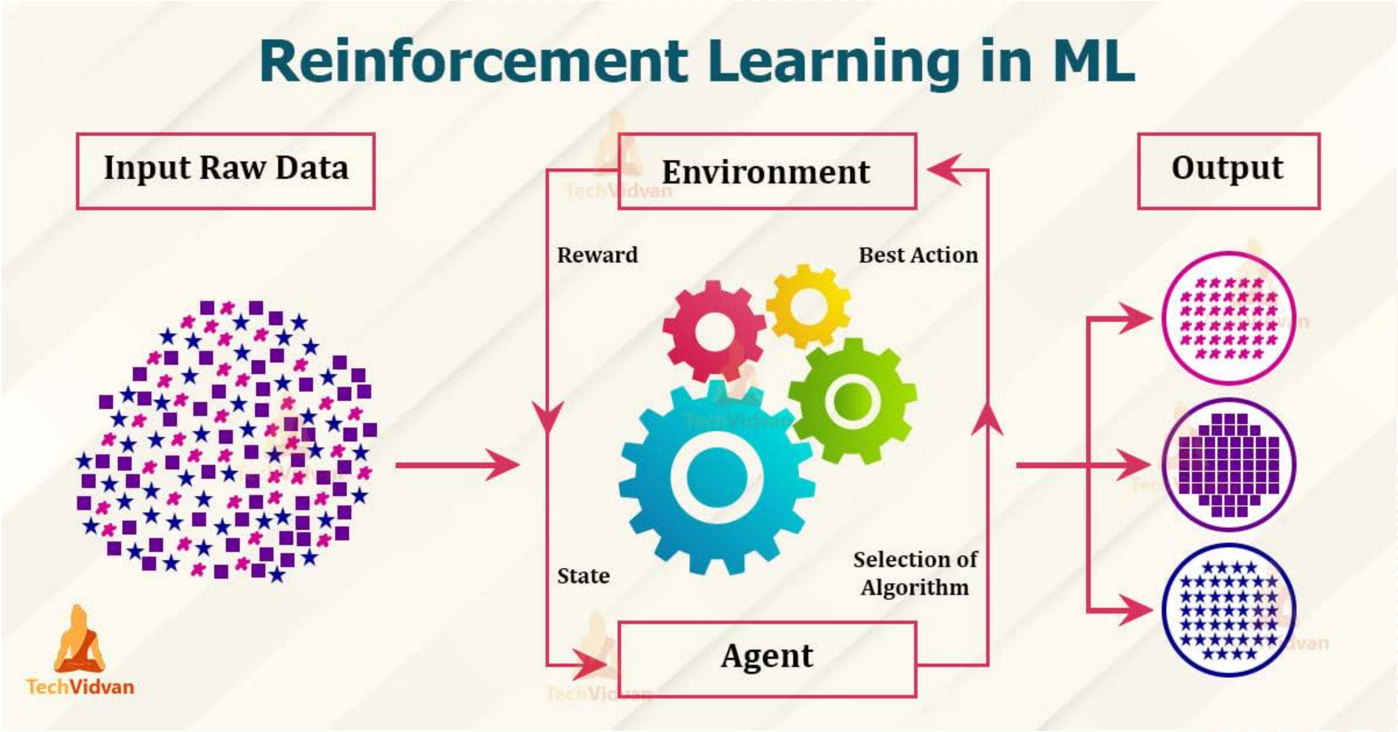
Email: socheat.sek@gmail.com

Table of Contents

Introduction to Reinforcement Learning

Reinforcement learning (RL) is a field within machine learning focused on how agents can learn optimal behaviors through interactions with their environment.

The goal is to maximize cumulative rewards over time. It emphasizes decision-making, where an agent learns by receiving feedback in the form of rewards or penalties in response to its actions, rather than being explicitly told what to do.



Key Characteristics of Reinforcement Learning

1. Agent-Environment Interaction: The agent interacts with the environment, taking actions and receiving feedback in the form of rewards.

2. Trial and Error Learning: RL relies on learning through exploration (trying new actions) and exploitation (using known actions).

3. Delayed Rewards: Actions may lead to long-term benefits rather than immediate rewards, requiring planning.

4. Policy: A strategy or mapping from states to actions that the agent learns to maximize rewards.

5. Value Functions: These estimate the future rewards from a given state or action, guiding the agent's decisions.

Key Characteristics of Reinforcement Learning

6. Exploration vs. Exploitation: A balance is needed between exploring unknown actions and exploiting known ones for rewards.

7. Markov Decision Process (MDP): RL often assumes the problem can be modeled as an MDP, with defined states, actions, rewards, and transitions.

8. No Supervised Labels: Unlike supervised learning, RL does not rely on labeled input-output pairs but learns through reward signals.

9. Feedback-driven: The agent learns dynamically from rewards and penalties, improving iteratively.

Types of Reinforcement Learning

Reinforcement Learning (RL) can be categorized into two main types based on how the agent learns to interact with its environment:

1. Model-Based Reinforcement Learning

2. Model-Free Reinforcement Learning

Types of Reinforcement Learning

1. Model-Based Reinforcement Learning

- **Definition:** The agent builds a model of the environment, including the transition probabilities and reward structure.
- **Purpose:** Enables planning by simulating potential future states and actions.
- **Advantages:** Efficient in solving problems where a model can be accurately learned; allows leveraging planning algorithms like dynamic programming.
- **Challenges:** Difficult when the environment is complex or uncertain.

Types of Reinforcement Learning

2. Model-Free Reinforcement Learning

- **Definition:** The agent learns directly from interactions without building a model of the environment.
- **Purpose:** Focuses on learning the optimal policy or value functions.
- **Common Algorithms:**
  - ***Q-Learning:*** Learns the value of state-action pairs to guide decision-making.
  - ***SARSA:*** Learns the value of taking specific actions under the current policy.
- **Advantages:** Simpler to implement and often effective for complex environments.
- **Challenges:** May require more interactions with the environment, leading to inefficiency in some cases.

Popular Algorithms in Reinforcement Learning

1. Model-Free Algorithms

Q-Learning

- Off-policy algorithm.
- Learns the optimal action-value function without requiring a model of the environment.
- Updates are based on maximum estimated future rewards.

SARSA (State-Action-Reward-State-Action)

- On-policy algorithm.
- Updates based on the actual actions taken by the current policy, making it more sensitive to the policy's behavior.

Deep Q-Networks (DQN)

- Combines Q-learning with deep neural networks to handle high-dimensional state spaces.
- Introduces experience replay and target networks for stability.

Popular Algorithms in Reinforcement Learning

2. Policy Gradient Methods

REINFORCE

- Directly learns the policy by maximizing the expected cumulative reward using gradient ascent.

Actor-Critic

- Combines policy learning (actor) and value function learning (critic) for stability and efficiency.

Proximal Policy Optimization (PPO)

- Simplifies and stabilizes policy updates, ensuring bounded changes during training.

Trust Region Policy Optimization (TRPO)

- Improves stability by constraining the policy updates within a trust region.

Popular Algorithms in Reinforcement Learning

3. Model-Based Algorithms

Dyna-Q

- Combines learning and planning, using a model of the environment for faster convergence.

Monte Carlo Tree Search (MCTS)

- Used in games like chess and Go, it builds a search tree of possible actions and rewards using a model.

Popular Algorithms in Reinforcement Learning

4. Other Advanced Methods

Temporal-Difference (TD) Learning

- Combines ideas from Monte Carlo methods and dynamic programming to update values incrementally.

Double Q-Learning

- Addresses overestimation bias in Q-learning by using two Q-functions for updates.

Asynchronous Advantage Actor-Critic (A3C)

- Leverages multiple threads for faster and more stable learning.

These algorithms form the backbone of RL advancements, and their choice depends on the problem complexity and requirements.

Steps in Reinforcement Learning

Reinforcement Learning (RL) typically follows these key steps:

1. Define the Environment:

- Specify the environment where the agent will interact.
- Define states, actions, rewards, and transitions (usually modeled as a Markov Decision Process).

2. Initialize the Agent:

- Set up the agent's initial policy (how it selects actions) and value functions (if applicable).
- Initialize parameters for learning, such as learning rate and exploration strategy (e.g., epsilon-greedy).

Steps in Reinforcement Learning

Reinforcement Learning (RL) typically follows these key steps:

3. Agent-Environment Interaction:

- The agent observes the current state of the environment.
- Based on its policy, the agent selects an action.
- The environment responds with a reward and the next state.

4. Update the Policy and/or Value Function:

- Use the reward and the observed state transition to update the agent's knowledge.
- Methods include:
- Value-based updates (e.g., Q-Learning).
- Policy-based updates (e.g., Policy Gradient).
- Combined updates (e.g., Actor-Critic).

Steps in Reinforcement Learning

Reinforcement Learning (RL) typically follows these key steps:

5. Balance Exploration and Exploitation:

- Adjust the exploration strategy to ensure a good balance:
- Explore new actions to find potentially better rewards.
- Exploit known actions to maximize cumulative rewards.

6. Repeat Until Convergence:

- Repeat the agent-environment interaction and updates until the policy converges or performance stabilizes.
- Monitor performance metrics such as total reward or accuracy.

Steps in Reinforcement Learning

Reinforcement Learning (RL) typically follows these key steps:

7. Test the Learned Policy:

- Evaluate the agent in the environment to ensure it has learned the desired behavior.
- Adjust parameters or re-train if needed.

8. Optimize and Deploy:

- Fine-tune the model for efficiency.
- Deploy the agent in the real-world environment or integrate it into the application.

Applications of Reinforcement Learning

Reinforcement Learning (RL) has diverse applications across industries and research areas due to its ability to optimize sequential decision-making problems. Here are notable applications:

1. Gaming:

- ***Video Games:*** Training AI to play games like Atari, Dota 2, and StarCraft.
- ***Board Games:*** Solving complex games like Chess and Go (e.g., AlphaGo).
- ***Game Testing:*** Automating and optimizing game testing processes.

2. Robotics:

- ***Robot Control:*** Teaching robots to walk, run, or perform tasks.
- ***Industrial Automation:*** Optimizing robotic arms for manufacturing and assembly lines.
- ***Drone Navigation:*** Training drones for autonomous flight and obstacle avoidance.

Applications of Reinforcement Learning

Here are notable applications:

3. Healthcare:

- ***Treatment Plans:*** Personalized medical treatment (e.g., optimizing chemotherapy schedules).
- ***Drug Discovery:*** Identifying optimal chemical reactions and compounds.
- ***Healthcare Operations:*** Streamlining patient flow in hospitals.

4. Autonomous Vehicles:

- ***Self-Driving Cars:*** Training cars to navigate traffic, obey rules, and make real-time decisions.
- ***Fleet Optimization:*** Routing and scheduling of autonomous delivery vehicles.

Applications of Reinforcement Learning

Here are notable applications:

5. Finance:

- ***Portfolio Optimization:*** Maximizing returns by balancing risk and rewards in investments.
- ***Algorithmic Trading:*** Developing RL-based trading bots for financial markets.
- ***Fraud Detection:*** Dynamic adaptation to detect and prevent fraudulent activities.

6. Energy Systems:

- ***Grid Management:*** Optimizing energy distribution and consumption in smart grids.
- ***Renewable Energy:*** Managing resources like solar or wind energy for efficiency.
- ***HVAC Systems:*** Controlling heating and cooling systems in buildings to save energy.

Applications of Reinforcement Learning

Here are notable applications:

7. Natural Language Processing (NLP):

- ***Conversational AI:*** Improving chatbots and virtual assistants for better dialogue handling.
- ***Text Summarization:*** Using RL for coherent and concise summaries.
- ***Language Translation:*** Enhancing the accuracy of machine translation models.

8. E-commerce and Marketing:

- ***Dynamic Pricing:*** Optimizing product prices based on demand and competition.
- ***Recommendation Systems:*** Personalizing product or content suggestions for users.
- ***Ad Placement:*** Optimizing online advertisements to maximize revenue.

Challenges in Reinforcement Learning

Here are some key challenges in Reinforcement Learning (RL) summarized:

1. Sample Efficiency:

- ***Problem:*** RL algorithms often require a large number of interactions with the environment to learn effectively.
- ***Impact:*** Makes RL impractical in real-world scenarios where interactions are costly or time-consuming (e.g., robotics, healthcare).

2. Exploration vs. Exploitation Trade-off:

- ***Problem:*** Balancing the need to explore new actions with exploiting known actions for rewards is non-trivial.
- ***Impact:*** Poor exploration can lead to suboptimal policies; excessive exploration wastes resources.

Challenges in Reinforcement Learning

Here are some key challenges in Reinforcement Learning (RL) summarized:

3. Sparse and Delayed Rewards:

- ***Problem:*** Many environments provide rewards infrequently or after long sequences of actions.
- ***Impact:*** Makes it difficult for the agent to learn which actions are beneficial.

4. High-Dimensional State and Action Spaces:

- ***Problem:*** Real-world problems often involve complex environments with numerous states and actions.
- ***Impact:*** Increases computational and memory requirements, making learning inefficient or infeasible.

Challenges in Reinforcement Learning

Here are some key challenges in Reinforcement Learning (RL) summarized:

5. Non-Stationary Environments:

- ***Problem:*** The environment may change over time (e.g., user behavior in recommendation systems).
- ***Impact:*** Requires the agent to adapt continuously, complicating policy learning.

6. Partial Observability:

- ***Problem:*** The agent may not have full access to the environment’s state (e.g., noisy or missing data).
- ***Impact:*** Requires advanced techniques like recurrent models or belief-state tracking.

Challenges in Reinforcement Learning

Here are some key challenges in Reinforcement Learning (RL) summarized:

7. Reward Engineering:

- ***Problem:*** Designing appropriate reward functions is complex and critical to successful learning.
- ***Impact:*** Poorly designed rewards can lead to unintended behaviors or failed training.

8. Scalability:

- ***Problem:*** RL algorithms struggle to scale efficiently in large or multi-agent environments.
- ***Impact:*** Limits applicability in domains like traffic control or multi-robot systems.

Challenges in Reinforcement Learning

Here are some key challenges in Reinforcement Learning (RL) summarized:

9. Safety and Ethical Concerns:

- ***Problem:*** Ensuring safe exploration and decision-making in critical systems (e.g., healthcare, autonomous vehicles).
- ***Impact:*** Mistakes during learning can have catastrophic consequences.

10. Generalization:

- ***Problem:*** RL agents often overfit to the training environment and fail to generalize to new, unseen scenarios.
- ***Impact:*** Reduces the robustness and reliability of RL systems in real-world deployments.

Challenges in Reinforcement Learning

Here are some key challenges in Reinforcement Learning (RL) summarized:

11. Computation Costs:

- ***Problem:*** Training deep RL models demands substantial computational resources.
- ***Impact:*** Limits accessibility to individuals or organizations without advanced hardware.

12. Multi-Agent Coordination:

- ***Problem:*** Multi-agent RL requires agents to collaborate or compete effectively.
- ***Impact:*** Increases complexity due to dynamic and potentially adversarial interactions.

សាកលវិទ្យាល័យន័រតុន
NORTON UNIVERSITY

• *Example:* FAQs AI Automator

```python
import os
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.pipeline import make_pipeline
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Function to load FAQs from the CSV file
def load_faqs(csv_path):
    try:
        data = pd.read_csv(csv_path)
        if 'Question' not in data.columns or 'Response' not in data.columns:
            raise ValueError("CSV file must contain 'Question' and 'Response' columns.")
        return data
    except Exception as e:
        print(f"Error loading FAQs: {e}")
        return pd.DataFrame(columns=['Question', 'Response'])

# Function to display and interact with FAQs
def show_faqs(data):
    if data.empty:
        print("No FAQs available.")
        return
    print("\nFrequently Asked Questions:")
    for i, row in data.iterrows():
        print(f"{i+1}. {row['Question']}")
    print("\nEnter the number of the question you'd like to know more about or type 'back' to return.")
    while True:
        user_input = input("Your choice: ").strip()
        if user_input.lower() == 'back':
            break
        try:
            choice = int(user_input) - 1
            if 0 <= choice < len(data):
                print(f"\n{data.iloc[choice]['Question']}\nBot: {data.iloc[choice]['Response']}\n")
            else:
                print("Invalid choice. Please try again.")
        except ValueError:
            print("Invalid input. Please enter a number.")
```

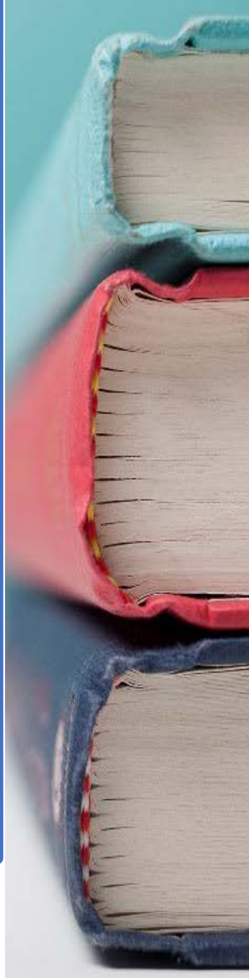
- **data.iloc[choice]:** Accesses the specified row in the DataFrame by its integer index (choice). Index-based selection method (**iloc**)
- **['Question'] and ['Response']:** Retrieve the question and response text from the selected row.

• *Example:* FAQs AI Automator

```python
# Function to evaluate the model
def evaluate_model(model, X_test, y_test):
    try:
        y_pred = model.predict(X_test)
        accuracy = accuracy_score(y_test, y_pred)
        print(f"Model accuracy: {accuracy * 100:.2f}%")
        return accuracy
    except Exception as e:
        print(f"Error during evaluation: {e}")
        return 0.0

# Function for interacting with the chatbot
def interact_with_chatbot(model, faqs_data):
    print("\nChatbot is ready! Type 'exit' to quit or 'faq' to check FAQs.")
    while True:
        user_input = input("You: ")
        if user_input.strip().lower() == 'exit':
            print("Goodbye!")
            break
        elif user_input.strip().lower() == 'faq':
            show_faqs(faqs_data)
        else:
            try:
                response = model.predict([user_input])[0]
                print(f"Bot: {response}")
            except Exception as e:
                print(f"Error generating response: {e}")

```



• *Example:* FAQs AI Automator

```
# Main function
if __name__ == "__main__":
    # Path to the CSV file
    csv_path = "cleaned_data.csv"  # Replace with your CSV file path

    # Load FAQs and training data
    if not os.path.exists(csv_path):
        print(f"Error: CSV file not found at '{csv_path}'.")
        exit()

    data = load_faqs(csv_path)
    if data.empty:
        print("No valid data found in the CSV file.")
        exit()

    X = data['Question']
    y = data['Response']

    # Train-test split
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    # Train the model dynamically
    print("Training the model...")
    model = make_pipeline(
        CountVectorizer(stop_words='english'),
        LogisticRegression(max_iter=1000, solver='liblinear')
    )
    model.fit(X_train, y_train)

    # Evaluate the model
    accuracy = evaluate_model(model, X_test, y_test)
    if accuracy < 0.8:
        print("Warning: Model accuracy is below 80%. Consider improving the dataset.")

    # Interact with the chatbot
    interact_with_chatbot(model, data)
```

4. Homework

- What is reinforcement learning, and how does it differ from supervised and unsupervised learning?
- What are the key characteristics that define reinforcement learning systems?
- What are the two primary types of reinforcement learning, and how do they differ in their approaches?
- What are the main steps involved in a reinforcement learning process?
- What are some of the major challenges faced in reinforcement learning, and how can they be mitigated?

សាកលវិទ្យាល័យន័រតុន
NORTON UNIVERSITY

Thank you