Reinforcement Learning 101

Learn the essentials of Reinforcement Learning!

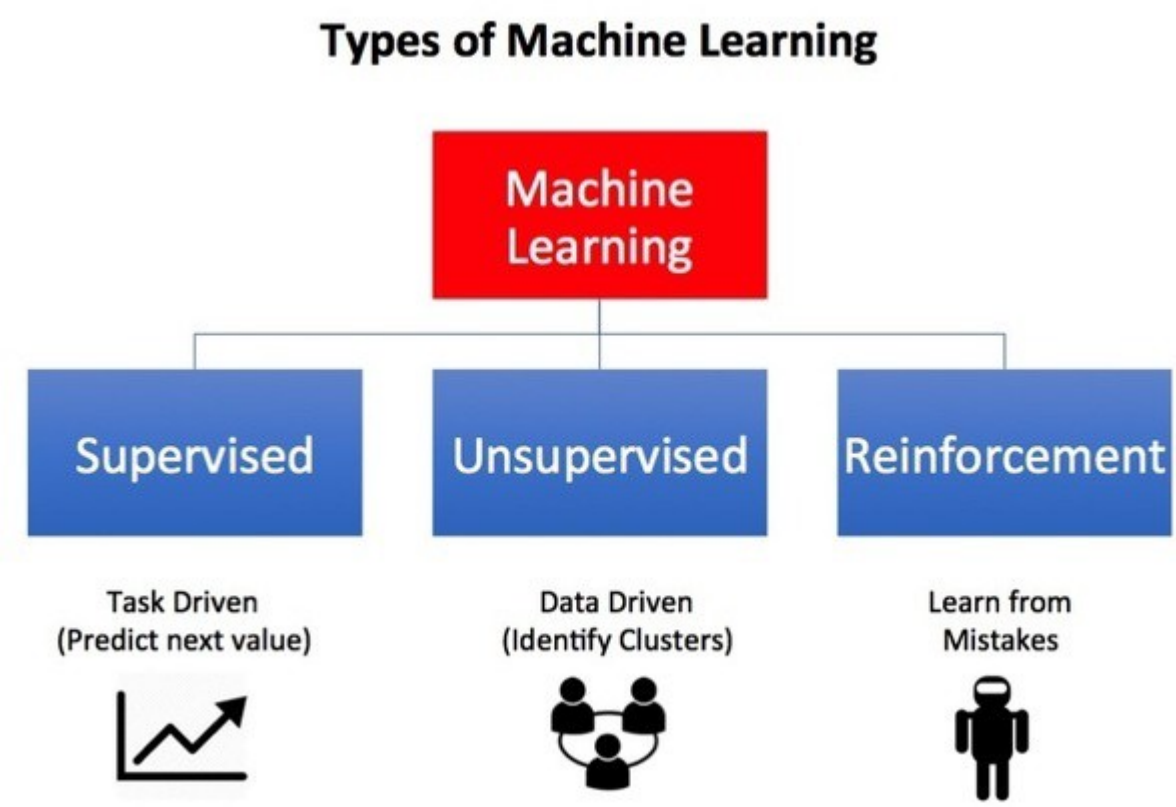
Shweta Bhatt Mar 19, 2018 · 6 min read

Photo by Daniel Cheung on Unsplash

Reinforcement Learning(RL) is one of the hottest research topics in the field of modern Artificial Intelligence and its popularity is only growing. Let's look at 5 useful things one needs to know to get started with RL.

1. What is Reinforcement Learning? How does it compare with other ML techniques?

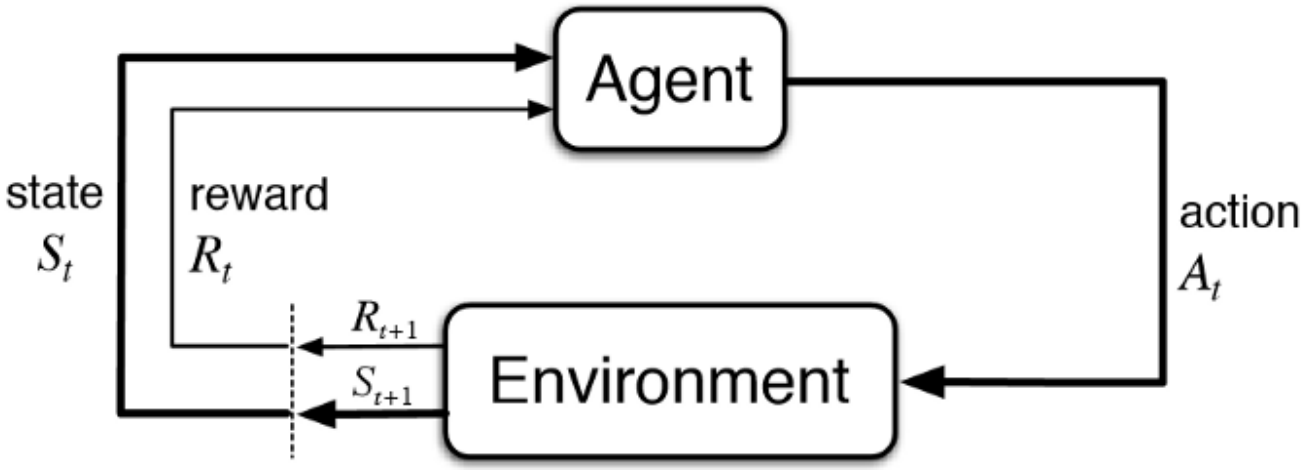
Reinforcement Learning(RL) is a type of machine learning technique that enables an agent to learn in an interactive environment by trial and error using feedback from its own actions and experiences.



Though both supervised and reinforcement learning use mapping between input and output, unlike supervised learning where the feedback provided to the agent is **correct set of actions** for performing a task, reinforcement learning uses **rewards and punishments** as signals for positive and negative behavior.

As compared to unsupervised learning, reinforcement learning is different in terms of goals. While the goal in unsupervised learning is to find similarities and differences between data points, in the case of reinforcement learning the goal is to find a suitable

action model that would maximize the **total cumulative reward** of the agent. The figure below illustrates the **action-reward feedback loop** of a generic RL model.

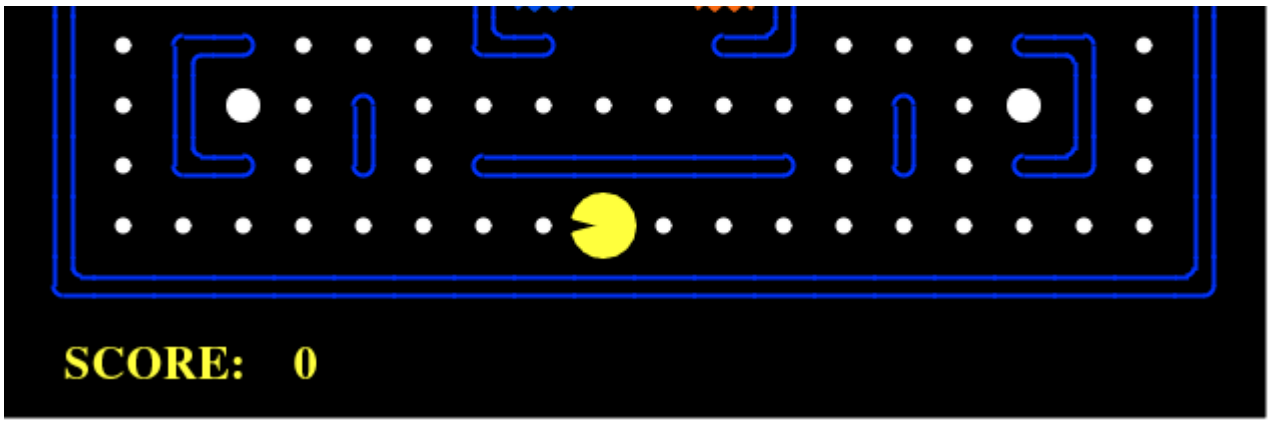


2. How to formulate a basic Reinforcement Learning problem?

Some key terms that describe the basic elements of an RL problem are:

1. **Environment —** Physical world in which the agent operates
2. **State —** Current situation of the agent
3. **Reward —** Feedback from the environment
4. **Policy —** Method to map agent's state to actions
5. **Value —** Future reward that an agent would receive by taking an action in a particular state

An RL problem can be best explained through games. Let's take the game of **PacMan** where the goal of the agent(PacMan) is to eat the food in the grid while avoiding the ghosts on its way. In this case, the grid world is the interactive environment for the agent where it acts. Agent receives a reward for eating food and punishment if it gets killed by the ghost (loses the game). The states are the location of the agent in the grid world and the total cumulative reward is the agent winning the game.



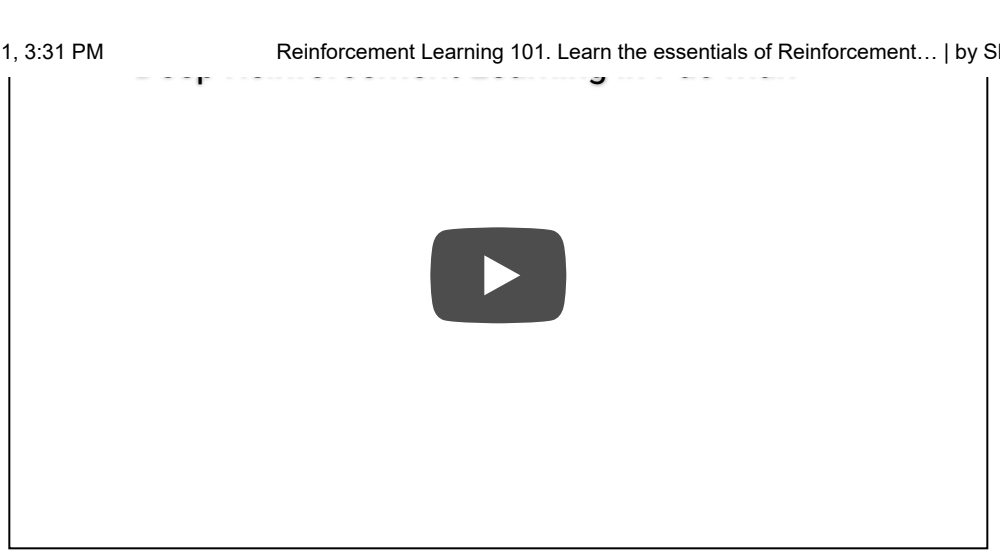
In order to build an optimal policy, the agent faces the dilemma of exploring new states while maximizing its overall reward at the same time. This is called **Exploration vs Exploitation** trade-off. To balance both, the best overall strategy may involve short term sacrifices. Therefore, the agent should collect enough information to make the best overall decision in the future.

Markov Decision Processes(MDPs) are mathematical frameworks to describe an environment in RL and almost all RL problems can be formulated using MDPs. An MDP consists of a set of finite environment states S, a set of possible actions A(s) in each state, a real valued reward function R(s) and a transition model P(s', s | a). However, real world environments are more likely to lack any prior knowledge of environment dynamics. Model-free RL methods come handy in such cases.

Q-learning is a commonly used model-free approach which can be used for building a self-playing PacMan agent. It revolves around the notion of updating Q values which denotes value of performing action *a* in state *s*. The following value update rule is the core of the Q-learning algorithm.

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

Here's a video demonstration of a PacMan Agent that uses Deep Reinforcement Learning.

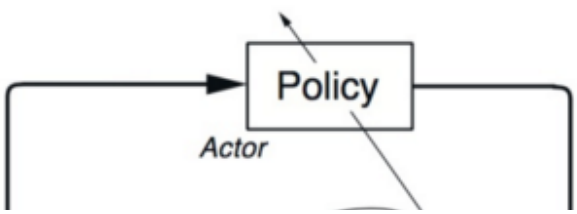


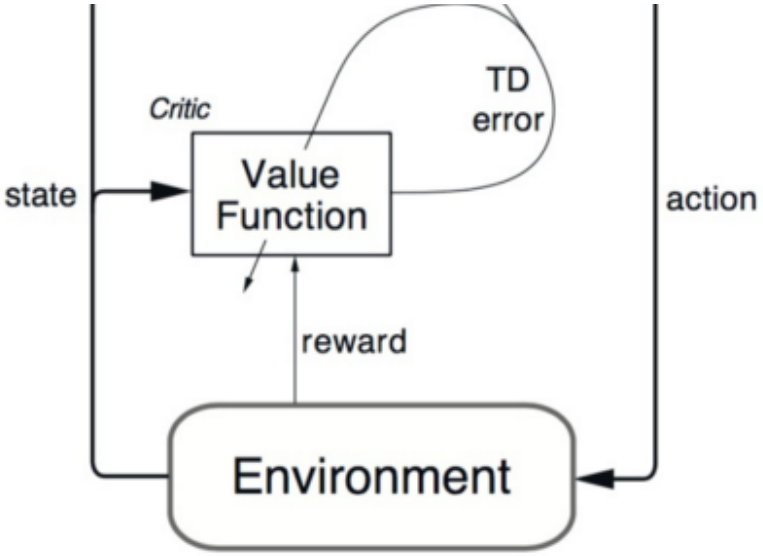
3. What are some of the most used Reinforcement Learning algorithms?

Q-learning and **SARSA** (State-Action-Reward-State-Action) are two commonly used model-free RL algorithms. They differ in terms of their exploration strategies while their exploitation strategies are similar. While Q-learning is an off-policy method in which the agent learns the value based on action a* derived from the another policy, SARSA is an on-policy method where it learns the value based on its current action *a* derived from its current policy. These two methods are simple to implement but lack generality as they do not have the ability to estimates values for unseen states.

This can be overcome by more advanced algorithms such as **Deep Q-Networks(DQNs)** which use Neural Networks to estimate Q-values. But DQNs can only handle discrete, low-dimensional action spaces.

Deep Deterministic Policy Gradient(DDPG) is a model-free, off-policy, actor-critic algorithm that tackles this problem by learning policies in high dimensional, continuous action spaces. The figure below is a representation of **actor-critic** architecture.





4. What are the practical applications of Reinforcement Learning?

Since, RL requires a lot of data, therefore it is most applicable in domains where simulated data is readily available like gameplay, robotics.

1. RL is quite widely used in building AI for playing computer games. **AlphaGo Zero** is the first computer program to defeat a world champion in the ancient Chinese game of Go. Others include ATARI games, Backgammon ,etc

2. In robotics and industrial automation, RL is used to enable the robot to create an efficient adaptive control system for itself which learns from its own experience and behavior. DeepMind's work on **Deep Reinforcement Learning for Robotic Manipulation with Asynchronous Policy updates** is a good example of the same. Watch this interesting demonstration video.

Other applications of RL include abstractive text summarization engines, dialog agents(text, speech) which can learn from user interactions and improve with time, learning optimal treatment policies in healthcare and RL based agents for online stock trading.

5. How can I get started with Reinforcement Learning?

For understanding the basic concepts of RL, one can refer to the following resources.

1. **Reinforcement Learning-An Introduction**, a book by the father of Reinforcement Learning- **Richard Sutton** and his doctoral advisor **Andrew Barto**. An online draft of the book is available here.
2. **Teaching material** from **David Silver** including video lectures is a great introductory course on RL.
3. Here's another **technical tutorial** on RL by **Pieter Abbeel** and **John Schulman** (Open AI/ Berkeley AI Research Lab).

For getting started with building and testing RL agents, the following resources can be helpful.

1. **This blog** on how to train a Neural Network ATARI Pong agent with Policy Gradients from raw pixels by **Andrej Karpathy** will help you get your first Deep Reinforcement Learning agent up and running in just 130 lines of Python code.
2. **DeepMind Lab** is an open source 3D game-like platform created for agent-based AI research with rich simulated environments.
3. **Project Malmo** is another AI experimentation platform for supporting fundamental research in AI.
4. **OpenAI gym** is a toolkit for building and comparing reinforcement learning algorithms.