

AI Algorithms Fundamentals & Applications

CA6001

Chapter 2

Dr Zhang Jiehuang

College of Computing and Data Science
Nanyang Technological University

email: *jiehuang.zhang@ntu.edu.sg*

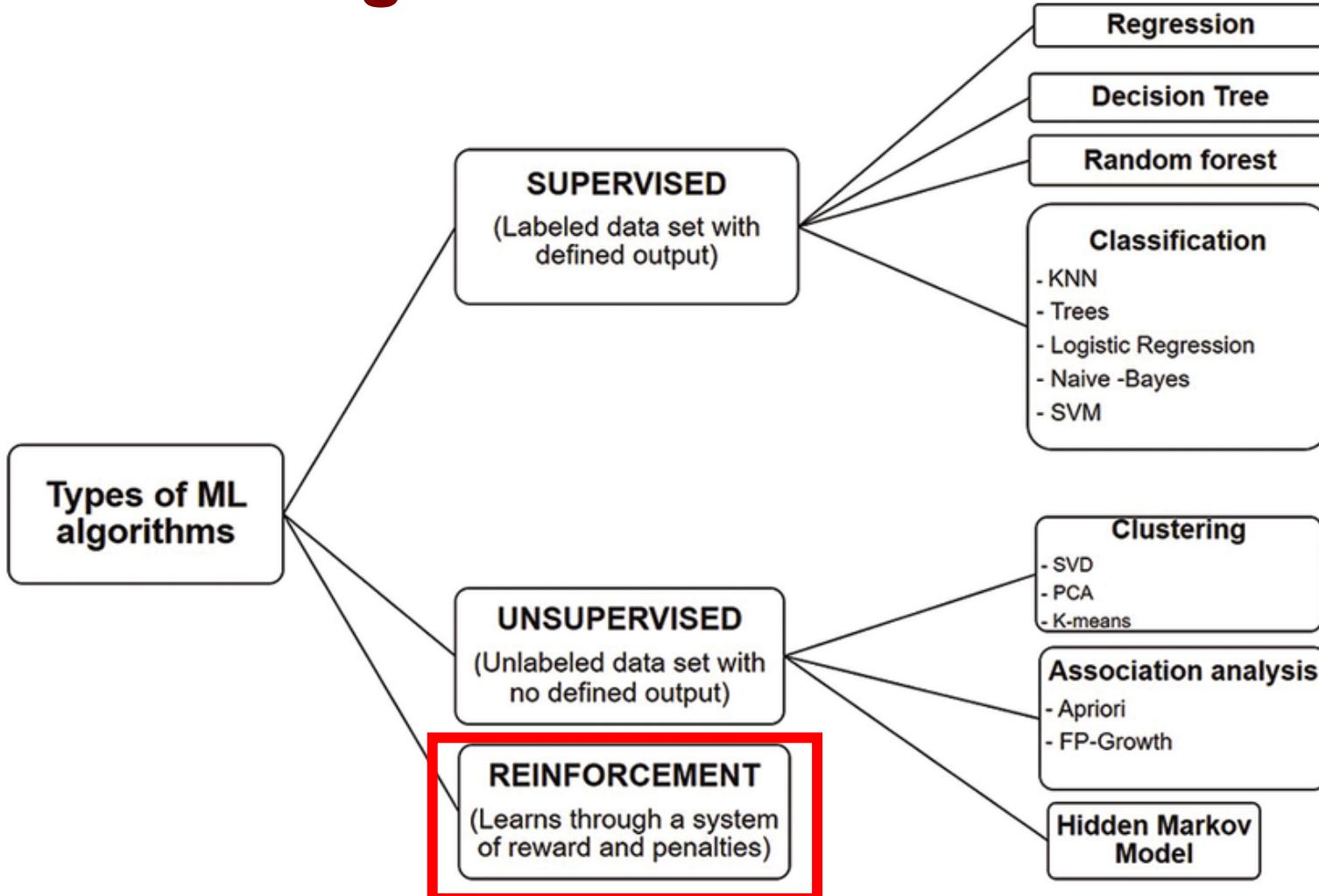


Chapter 2 – Reinforcement Learning

1. Types of ML algorithms
2. What Is Reinforcement Learning?
3. Difference between Supervised/Unsupervised Learning and RL
4. Reinforcement Learning Model
5. Use cases of Reinforcement Learning
6. Reinforcement Learning Implementations and Algorithms
7. Exploration Exploitation Strategies
8. Multi Armed Bandit Problem
9. Challenges and Future Directions



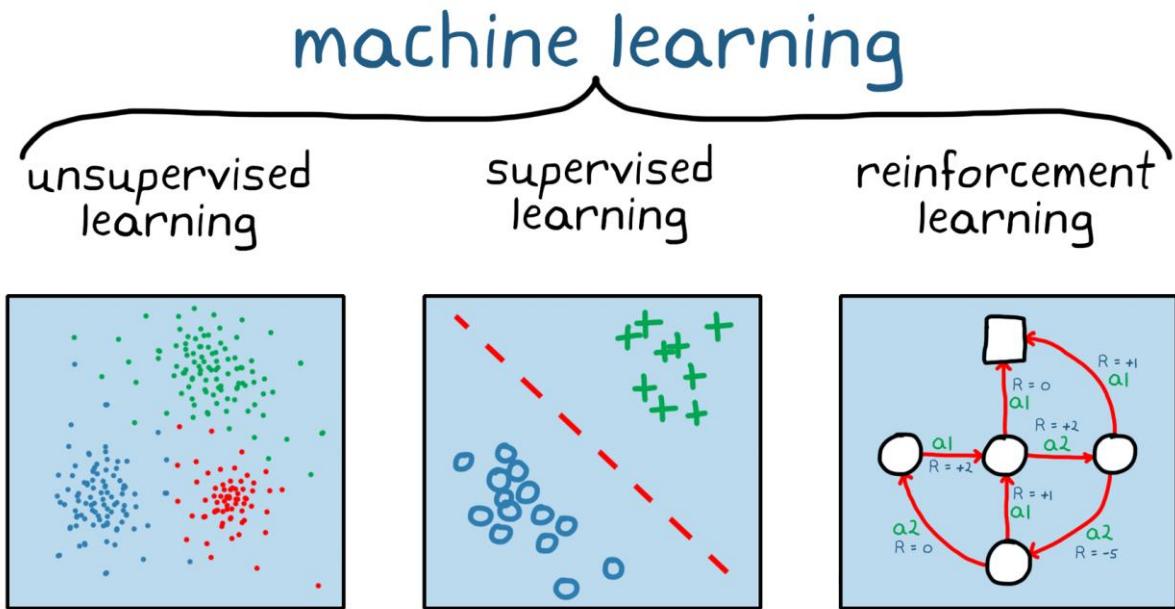
What Types of ML Algorithms are there?



Abbreviations: KNN: k-nearest neighbour; SVM: Support Vector Machine; SVD: Singular Value Decomposition; PCA: Principal Component Analysis; FP: Frequent pattern



What is Reinforcement Learning?



1. Reinforcement (RL) Learning is the 3rd type of Machine Learning Technique, besides SL,UL
2. RL agent aims to learn the optimal policy to maximize the cumulative reward for an agent in an environment
3. RL is a trial-and-error process of learning how to navigate an environment
4. There is no data labels in RL, the only feedback is reward or no reward. We can guide agent to reward (RL with human feedback)



What is the Difference: RL VS SL/UL

Three Types of Machine Learning

Supervised Learning

Has outcome information (“labels”)

Finds patterns that relate to those outcomes

Uses patterns to predict outcomes not yet known

Unsupervised Learning

No outcome information available

Analyzes or identifies groups without labels or human instruction

Offers insights into characteristics that define groups

Reinforcement Learning

Makes decisions based on trial and error

Decision-making algorithm is constantly refined based on “rewards”

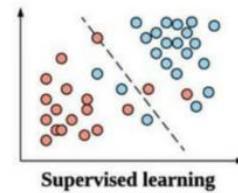
Excels in complex situations



What is the Difference: RL VS SL/UL

Supervised Learning

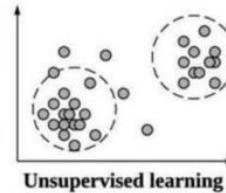
Learning a function that maps input data to output data, based on **labeled** examples.



Supervised learning

Unsupervised Learning

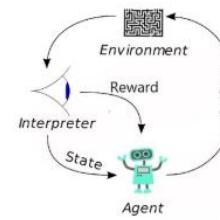
Finding hidden patterns or intrinsic structures in **unlabeled** data.



Unsupervised learning

Reinforcement Learning

learning how to act or behave when given occasional **reward** or punishment signals



SL: Learning from a textbook with the correct answer in the back

UL : Finding people that share your interest and hobbies

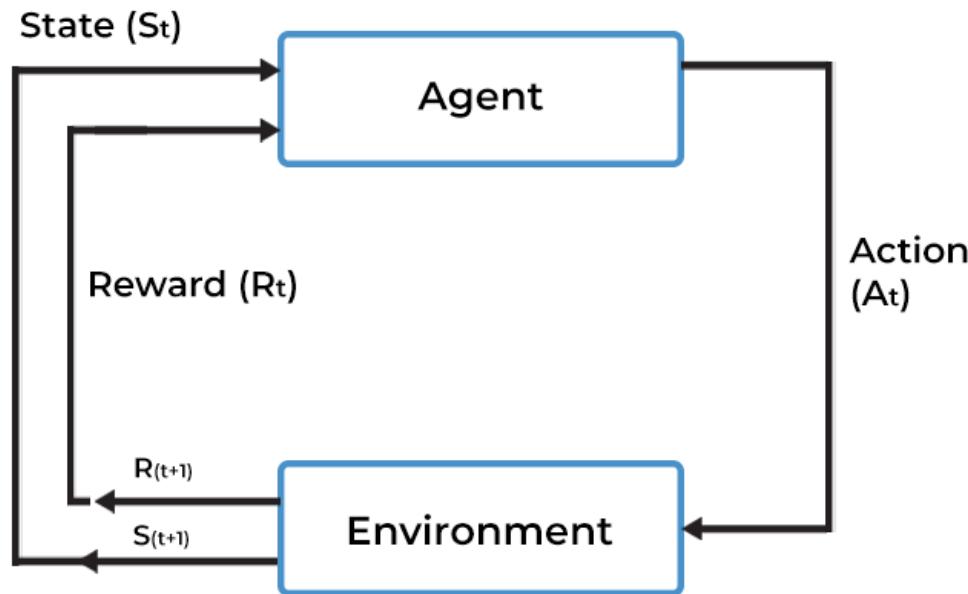
RL : Learning how to ride a bicycle by practicing and improving with each attempt



Reinforcement Learning Model



REINFORCEMENT LEARNING MODEL



1. In the above figure, a computer may represent an agent in a particular state (S_t).
2. It takes action (A_t) in an environment to achieve a specific goal.
3. As a result of the performed task, the agent receives feedback as a reward or punishment (R).
4. Agent aims to maximise rewards over a cumulative time period



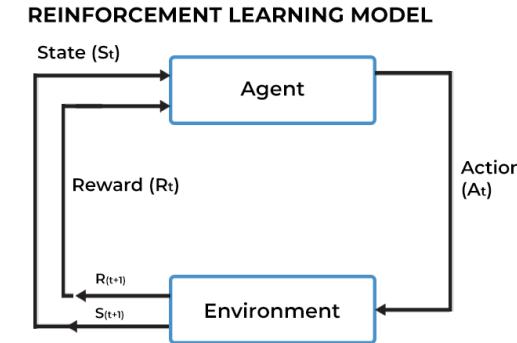
Reinforcement Learning Workflow

Step 1: Define/Create the Environment

Agent starts in an environment, physical or simulated.

Step 2: Specify the Reward

Rewards acts as a performance metric for the agent and allows agent to evaluate the task quality against its goals. Offering appropriate rewards to the agent may require a few iterations to finalise the right course of action



Step 3: Define the Agent and Type of Training Algorithm

Once environment and rewards are finalised, then we can define the agent and choose the suitable type of training algorithm

Step 4: Training and Validation of Agent

Training to finetune the policy and allow agent to explore, can be time and resource intensive

Step 5: Implement the Policy

Training is complete and optimal policy is confirmed. Model/Policy can then be deployed.



Motivation: RL can Find New Solutions



“Move 37” in Lee Sedol AlphaGo match:
reinforcement learning discovers new move that no
human has made



These AI generated images looks like something a
human might draw, and it can draw things out of a
human's imagination



RL Learns to Play Atari Games



Figure 1: Screen shots from five Atari 2600 Games: (*Left-to-right*) Pong, Breakout, Space Invaders, Seaquest, Beam Rider

Input is raw pixels from an Atari game console, fed into a Convolutional Neural Network

Training is done with Q-learning, where output is a value function estimating future rewards

Final model outperforms human expert players



RL Learns to Drive a Car



RL + Computer Vision is used to train the self driving car

Initially, the car steers wildly from left to right

Slowly, after every training iteration, the car learns to drive smoother and keep inside the lane

When car makes a mistake, the safety driver takes over and corrects

By the 11th training iteration, the car is able to drive smoothly without human intervention

No rules were hard coded into the car, RL allows the car to learn from scratch how to drive



RL with a Chicken

- <https://www.youtube.com/watch?v=spfpBrBjntg>



Cartpole from OpenAI

OpenAI's Cartpole is considered the hello world of reinforcement learning, a classic problem

A pole is attached to the cart, moving along a frictionless track, the pole stays upright and the goal is to prevent the pole from falling over by moving the cart



The environment gives 4 observations: position and velocity of the cart, angle and angular velocity of pole

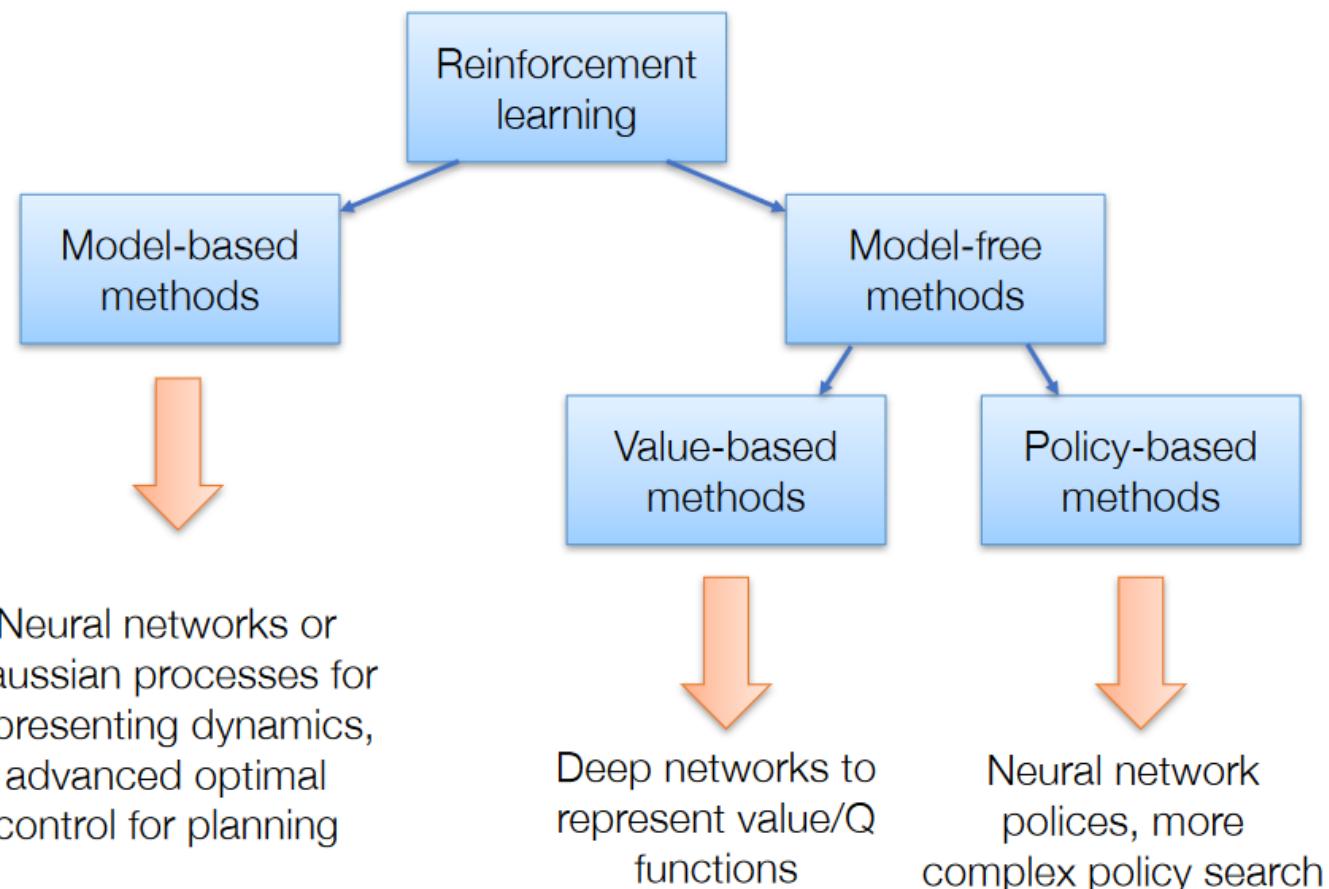
Agent controls the cartpole by moving left or right

Rewards is given for every timestep that pole remains upright

Episode ends when pole tips over or cart moves outside boundaries, or 200 timesteps pass



Reinforcement Learning Implementations



Policy-based: This RL approach aims to maximize the system reward by employing deterministic policies, strategies, and techniques.

Value-based: The agent uses value functions (e.g., Q-tables) to estimate the potential rewards of actions in different states, choosing the best action.

Model-based: The agent builds an internal model of the environment to predict future states and rewards, optimizing decisions accordingly.



Reinforcement Learning Algorithms



Off-policy, model-free algorithm that learns from random actions (greedy policy).

Q refers to quality of activities that maximize rewards generated

State-Action-Reward-State-Action (SARSA) algorithm.

On-policy method that learns from current state and actions for implementing RL process

Deep refers to use of neural network as in deep learning

2D arrays are replaced by neural networks for efficient calculation of state values



Exploration Exploitation Strategy

Reinforcement Learning addresses the problem of making decisions that strikes a balance between Exploration and Exploitation.

Exploration: Finding new actions to learn more about the environment and knowledge of action-reward pairs, as well as reducing uncertainty

Epsilon-Greedy Exploration: Epsilon-greedy algorithms manage to unify those two characteristics (exploitation and exploration) by sometimes choosing completely random actions with probability epsilon while continuing to use the current best-known action with probability (1 - epsilon).

Exploitation: using the accumulated knowledge to make decisions that maximize the expected reward based on the present information

Greedy Algorithms: Greedy algorithms tend to choose the locally optimal solutions at each step without consideration of the potential impact on the overall solution. They are often efficient in terms of computation time; however, this approach may be suboptimal when sacrifices are required to achieve the best global solution



Discount Factor γ

$$Q_{st,at} = Q_{st,at} + \alpha * (r_t + \gamma * \max Q(st+1, a) - Q_{st,at})$$

The diagram shows the Q-learning update rule:
$$Q_{st,at} = Q_{st,at} + \alpha * (r_t + \gamma * \max Q(st+1, a) - Q_{st,at})$$
. The terms are labeled as follows:

- New value: $Q_{st,at}$
- Current value: $Q_{st,at}$ (before the update)
- Reward: r_t
- Discount factor: γ
- Future value estimate: $\max Q(st+1, a)$
- Learning rate: α

In RL, we have to balance between prioritising immediate rewards versus future rewards, example of instant vs delayed gratification, short term versus long term growth

This parameter named the discount factor, γ , is a value between 0 and 1 that determines how much importance an agent places on future rewards compared to immediate rewards

The discount factor discounts the value of rewards received further in the future,

Higher γ value means the agent prioritises future rewards more significantly

Lower γ value means the agent prioritises immediate reward more significantly

Multi-Armed Bandit Problem

A gambler must choose which of several slot machines (or "one-armed bandits") to play, each with a different, unknown payout rate.

Goal: The gambler wants to maximize their winnings over a series of plays.

Epsilon-Greedy Strategy

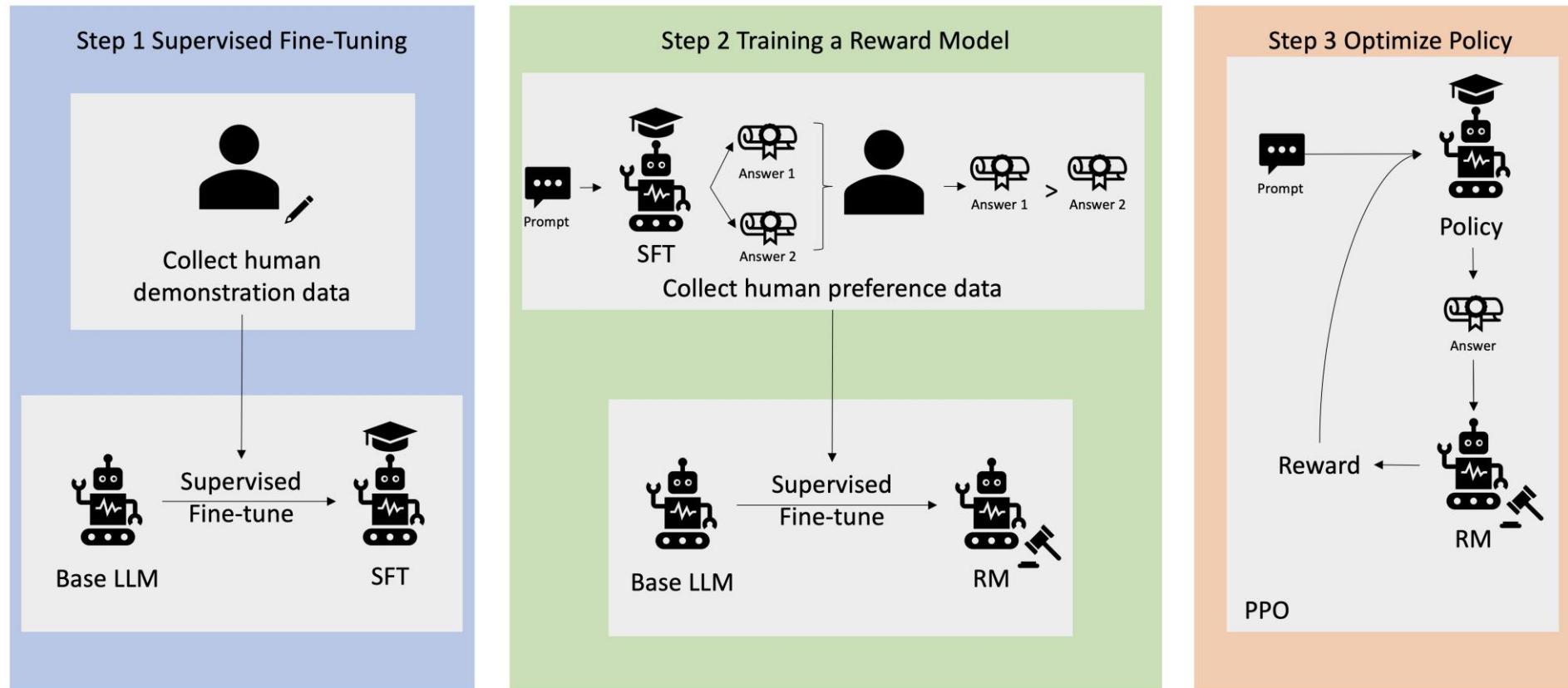
In this strategy, with probability ϵ (a small value, say 0.1), the gambler randomly chooses a slot machine to play (exploration). With probability $1-\epsilon$ the gambler chooses the machine that has the highest estimated payout based on past outcomes (exploitation).

Let $Q(a)$ be the estimated value (average reward) of action a , and $N(a)$ be the number of times action a has been chosen. After each play, update the estimated value of the chosen machine a using:

$$Q(a) = Q(a) + \frac{1}{N(a)} (R - Q(a)), R \text{ is the reward received from machine } a.$$

The epsilon-greedy strategy ensures that no machine is left unexplored, as ϵ allows for random selection irrespective of the estimated values. This prevents missing out on potentially better machines due to lack of initial exploration.

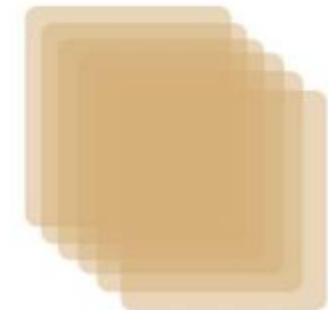
Reinforcement Learning with Human Feedback



RLHF is a machine learning technique that uses human feedback to optimize AI models to self learn efficiently. Human feedback is incorporated into the reward function, so that the model performs tasks aligned with human goals. RLHF is used in training GenAI models, including LLMs.

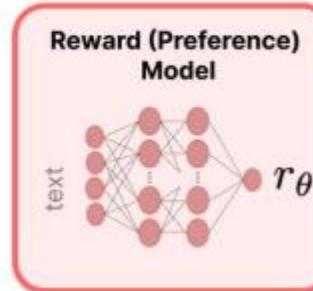
Reinforcement Learning with Human Feedback

Prompts Dataset



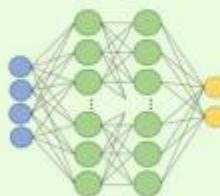
Sample many prompts

Train on
{sample, reward} pairs



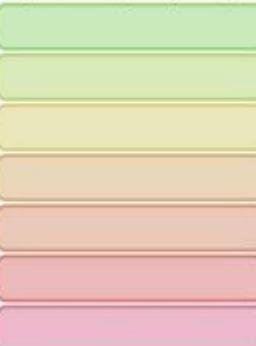
Outputs are ranked
(relative, ELO, etc.)

Initial Language Model



Generated text
Lorem ipsum dolor
sit amet, consecetur
adipiscing elit. Aen
Donec quam felis
vulputate eget, arc
Nam quam nunc
eros faucibus tincid
luctus pulvinar, her

Human Scoring



LE

Can you explain to me how RL with human feedback works for language models like I am five?



Sure! So, imagine you have a robot named Rufus who wants to learn how to talk like a human. Rufus has a language model that helps him understand words and sentences.

First, Rufus will say something using his language model. For example, he might say "I am a robot."

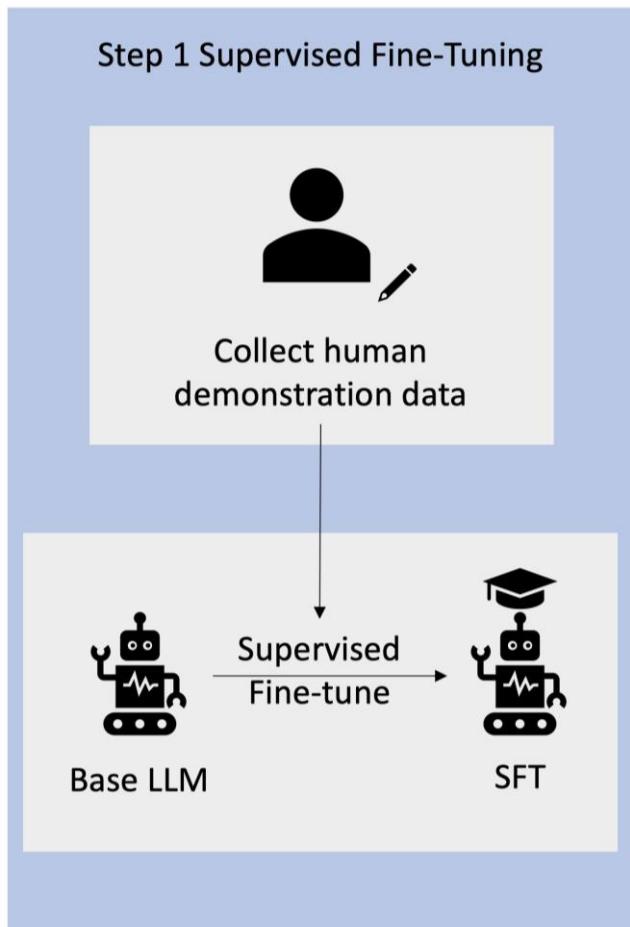
Then, a human will listen to what Rufus said and give him feedback on whether it sounded like a natural sentence a human would say. The human might say, "That's not quite right, Rufus. Humans don't usually say 'I am a robot.' They might say 'I'm a robot' or 'I am a machine.'"

Rufus will take this feedback and use it to update his language model. He will try to say the sentence again, using the new information he received from the human. This time, he might say "I'm a robot."

The human will listen again and give Rufus more feedback. This process will continue until Rufus can say sentences that sound natural to a human.

Over time, Rufus will learn how to talk like a human thanks to the feedback he receives from humans. This is how language models can be improved using RL with human feedback.

RLHF: Data Collection



First step in RLHF is to collect a set of human generated prompts and response pairs for the training data

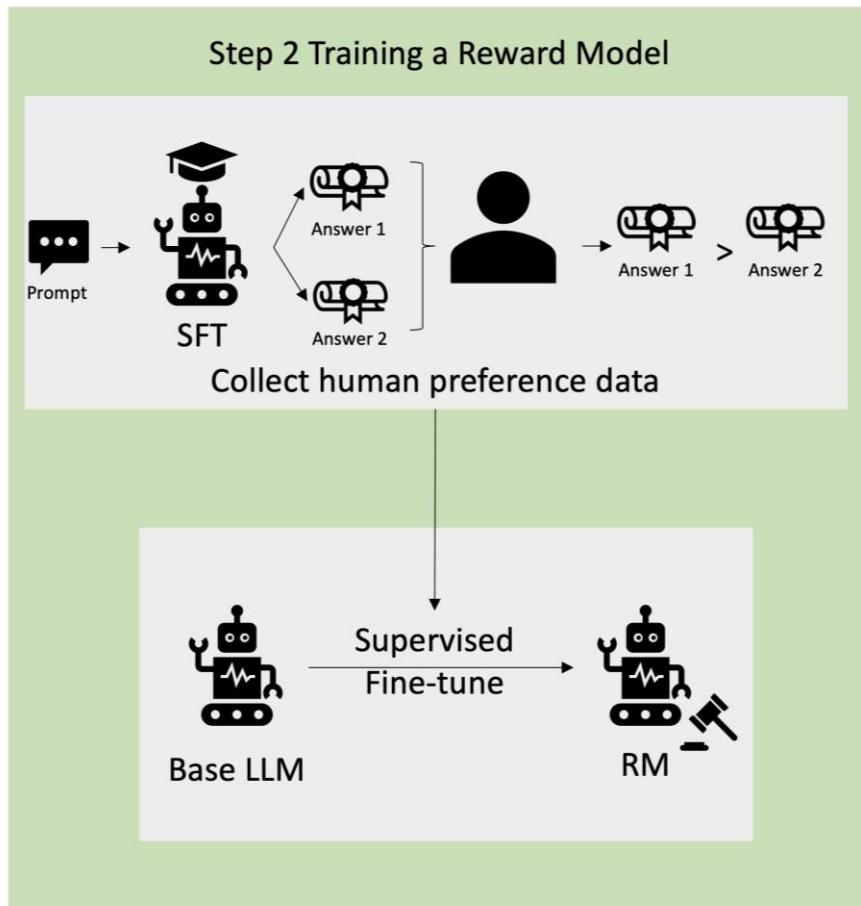
For example the prompts might be:

- *“Where is the location of the HR department in NTU?”*
- *“What is the approval process for social media posts?”*
- *“What does the Q1 report indicate about performance compared to previous quarterly reports?”*

A human expert then answers these questions with accurate, natural responses

These prompts, responses pairs form the training data to train the LLM

RLHF: Supervised Finetuning of LLM



A commercial pretrained model can serve as the base model for RLHF, such openAI o1, meta's LLAMA, google gemini etc

Additional techniques such as Retrieval Augmented Generation (RAG) can be used to finetune to the organization's internal knowledge base

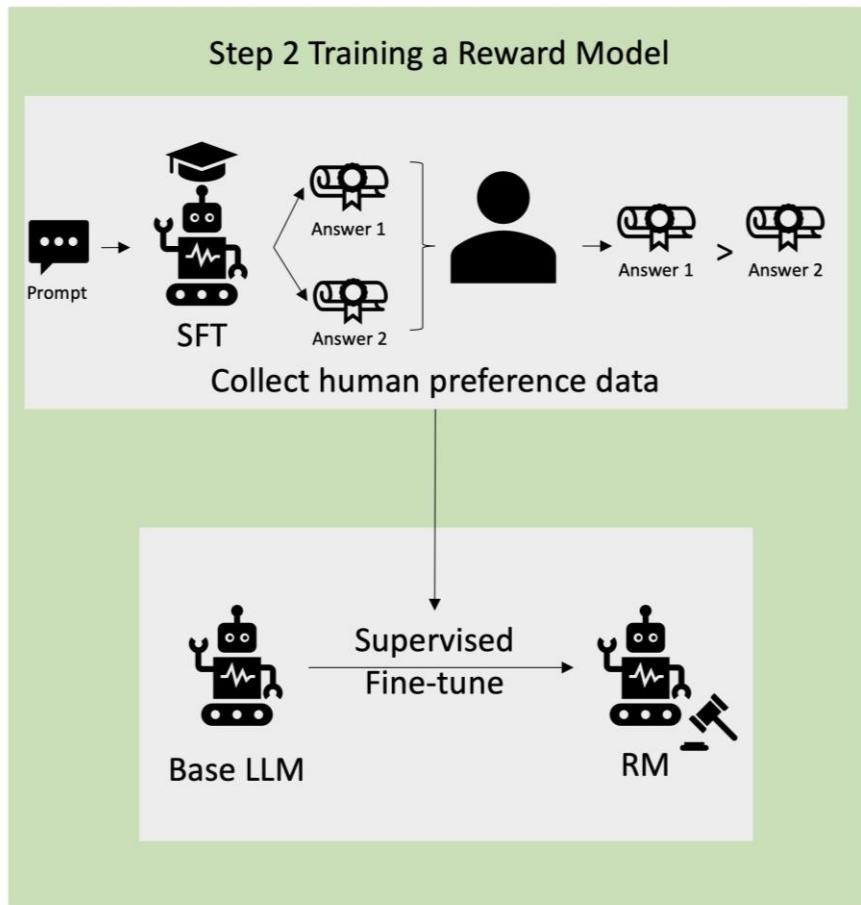
When finetuning is completed, we compare its response to the human responses collected in previous step. Mathematical techniques determine the degree of similarity between the two

With the degree of similarity, the model now has a policy that is designed to form responses that score closer to human responses

More rewards are given for response closer to human ones

This policy forms the basis of all future decision making for the mdoel

RLHF: Separate Reward Model from Human Ratings



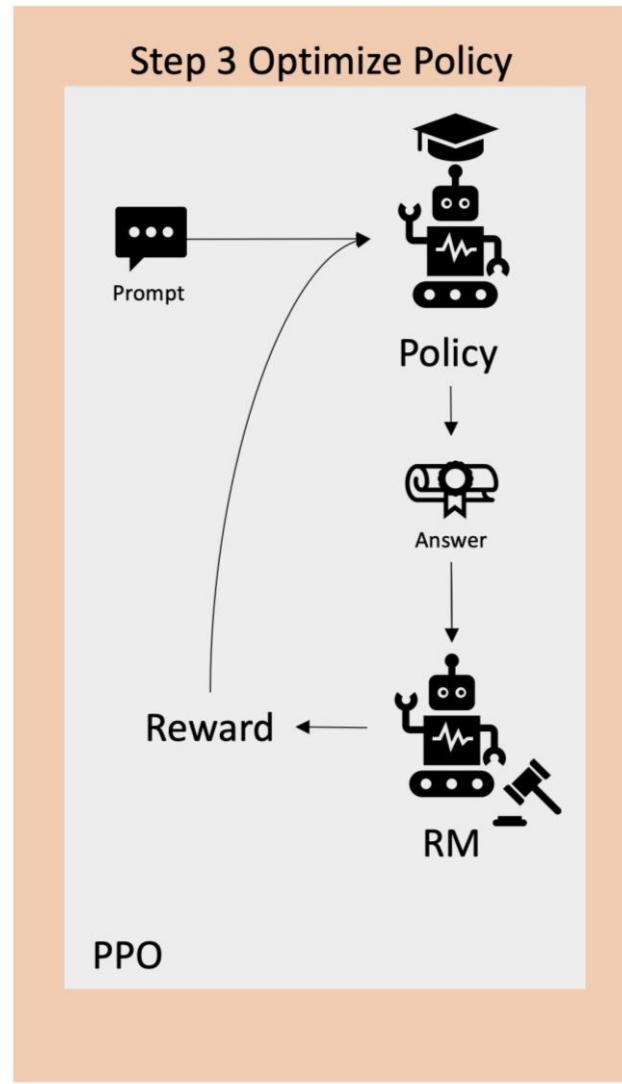
The core of RLHF is to train a separate AI reward model based on human feedback, then using this model as a reward function to optimize policy

Given a set of multiple responses from the model answering the same prompt, humans can indicate their preference regarding the quality of each response

Then we can use these response-rating preferences to build the reward model that automatically estimates the score a human will give any prompt response generated by the model

This the second reward model used to finetuning the LLM, separate from previous step

RLHF: Separate Reward Model from Human Ratings



The LLM then uses the reward model to automatically refine its policy before responding to the prompts

Using the reward model, the LLM internally evaluates a series of responses and chooses the response that is most likely to result in the greatest reward

This means that it meets human preferences in a more optimized manner

RLHF is recognized as the industry standard for ensuring that LLMs produce content that is harmless, truthful and aligned with human values

Challenges and Future Direction of RL

Challenges:

1. **Sample Efficiency:** RL need a large number of interactions with the environment to learn effective policies
2. **Computation Cost:** RL models need intensive resource to train
3. **Stability and Convergence:** Some algorithms, especially neural networks, are unstable and difficult to achieve convergence
4. **Ethical Concerns:** Unintended consequences and side effects especially in sensitive domains like healthcare and finance

Future Directions:

1. **Deep Reinforcement Learning:** RL with neural networks can lead to more independent and flexible systems if agents can perceive the extremely detailed nuances in data
2. **Transfer Learning Integration:** Enables agents to leverage previous skills and apply to new domains
3. **Multi Agent Reinforcement Learning:** Agents interact with each other, opening up many possibilities
4. **Robustness and Generalization:** Ability to learn many diverse skills greatly enhances RL adoption

DeepSeek R1: Group Relative Policy Optimization (GRPO)

$$J_{\text{GRPO}}(\theta) = \left[\frac{1}{G} \sum_{i=1}^G \min \left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)} A_i, \text{clip} \left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}, 1 - \epsilon, 1 + \epsilon \right) A_i \right) - \beta D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}) \right]$$

The main thing we want to maximize

If probability ratio is too far away from 1, clip it

Regularization term

G : Number of samples in a group

ϵ : A small positive number < 1

β : Weight of KL-div term

q : An input query/question

o_i : i^{th} output sample from previous policy (i.e. model)

A_i : Normalized reward for i^{th} output sample (at group level)

$\pi_\theta(o_i|q)$: Current model's probability of generating o_i in response to q

$\pi_{\theta_{old}}(o_i|q)$: Previous model's probability of generating o_i in response to q

$\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}$: New model's likelihood of o_i relative to old one
We want this to be big when reward (A_i) is big

References for Chapter 2 – Reinforcement Learning

- https://www.researchgate.net/publication/351021675_Artificial_intelligence_in_cancer_diagnostics_and_therapy_Current_perspectives
- <https://pecan.ai/blog/3-types-of-machine-learning/>
- <https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-reinforcement-learning/>
- <https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-reinforcement-learning/>
- <https://rail.eecs.berkeley.edu/deeprlcourse/deeprlcourse/static/slides/lec-1.pdf>
- <https://www.cs.toronto.edu/~vmnih/docs/dqn.pdf>
- <https://wayve.ai/thinking/learning-to-drive-in-a-day/>
- <https://aws.amazon.com/what-is/reinforcement-learning-from-human-feedback/>
- <https://huggingface.co/blog/rlhf>
- <https://www.analyticsvidhya.com/blog/2021/02/introduction-to-reinforcement-learning-for-beginners/>
- <https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-reinforcement-learning/>
- <https://www.geeksforgeeks.org/a-beginners-guide-to-deep-reinforcement-learning/>
- <https://www.geeksforgeeks.org/multi-armed-bandit-problem-in-reinforcement-learning/>
- <https://www.artiba.org/blog/the-future-of-reinforcement-learning-trends-and-directions>
- <https://discuss.boardinfinity.com/t/what-is-q-learning-in-reinforced-machine-learning/17456>

Chapter 2 – Reinforcement Learning

**The End
Questions?**