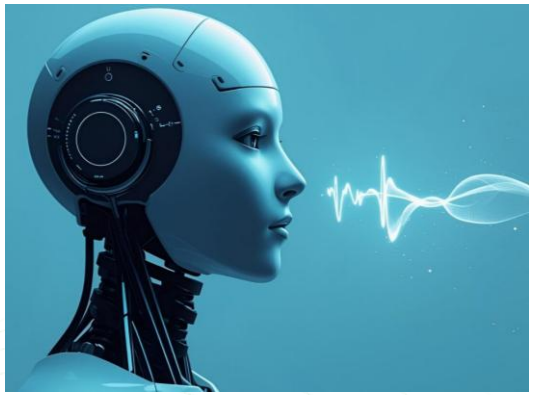


AI (Artificial Intelligence)

Jonathan Chen

AI 提供了一系列強大的技術，這些技術正在轉變產業並為企業帶來新的機會。

以下是可以利用來創新和擴展營運的關鍵 AI 功能。

- **影像生成**

  AI 在幾秒鐘內將簡單的文字描述轉換為高品質的逼真影片。例如，透過輸入「山上的日落」之類的提示，AI 可立即生成令人驚嘆的視覺效果。這項突破性的技術正在徹底變革行銷、娛樂和設計等創意產業，大幅加快內容創作過程。

- **文字生成**

  AI 可自動生成類似人類的文字，從電子郵件等簡短內容到複雜的報告。該技術廣泛運用於客戶支援、行銷和內容創作領域，透過簡化寫作程序來提高效率並節省寶貴的時間。

- **語音生成和辨識**

  AI 支援的語音生成可以創造自然、類似人類的語音，而語音識別使機器能夠理解和處理說出的字詞。這些技術是透過 Alexa 等虛擬助理提供順暢的語音啟用體驗以及增強客戶服務、智慧設備和可達性解決方案的關鍵因素。

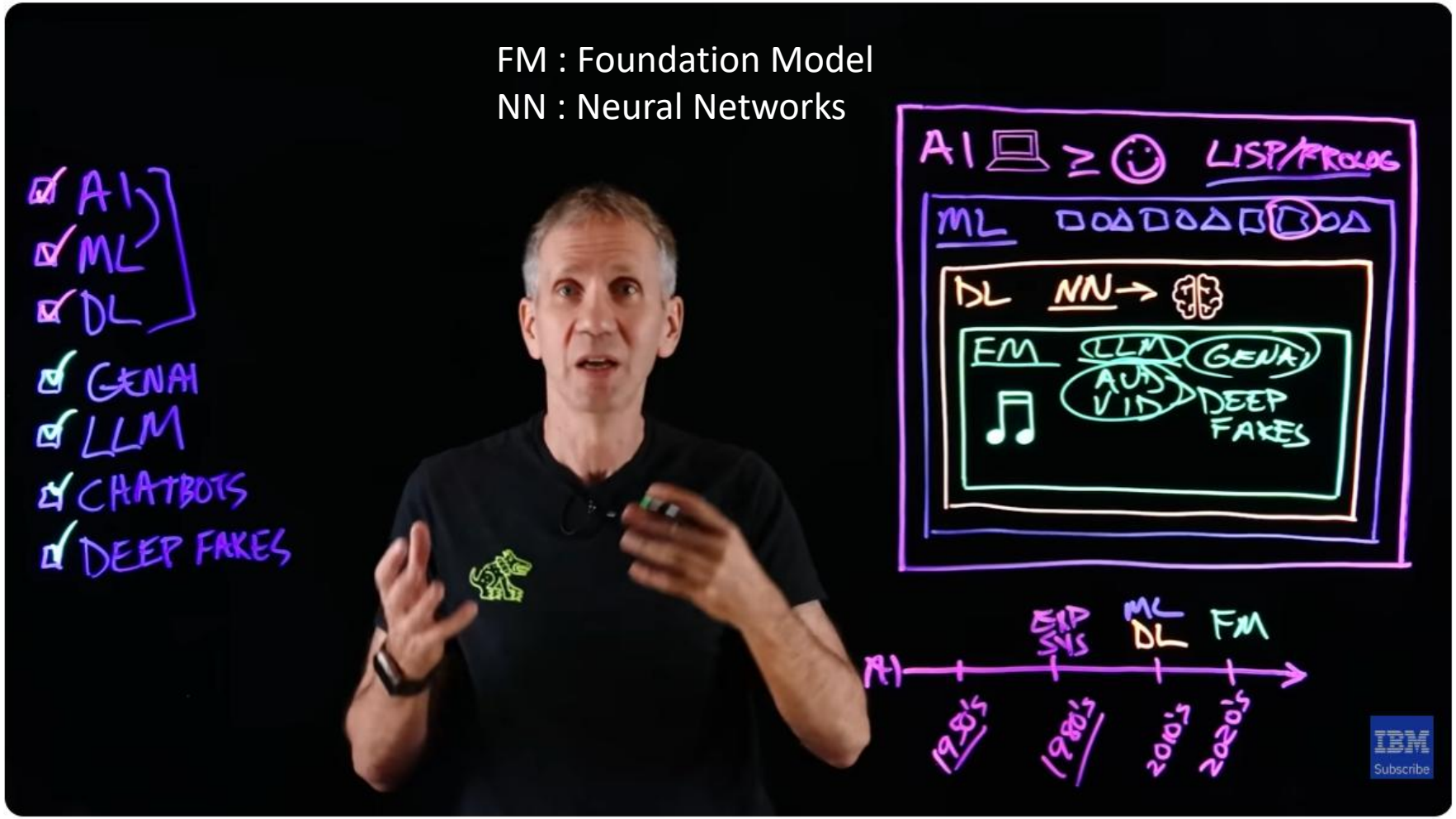
- **多模態 AI**

  多模態 AI 整合文字、影像和音訊資料，提供複雜內容的更全面理解。多模態 AI 透過一次識別物件、轉錄語音和解釋螢幕上的文字，即時提供進階洞見。對於利用 AI 進行影片分析、自動駕駛汽車等領域的產業，此功能至關重要，可實現更智慧、更快速的決策並挖掘新的創新可能性。

https://www.youtube.com/watch?v=UGdG4WpluJ8&t=16s

https://www.youtube.com/watch?v=qYNweeDHiyU

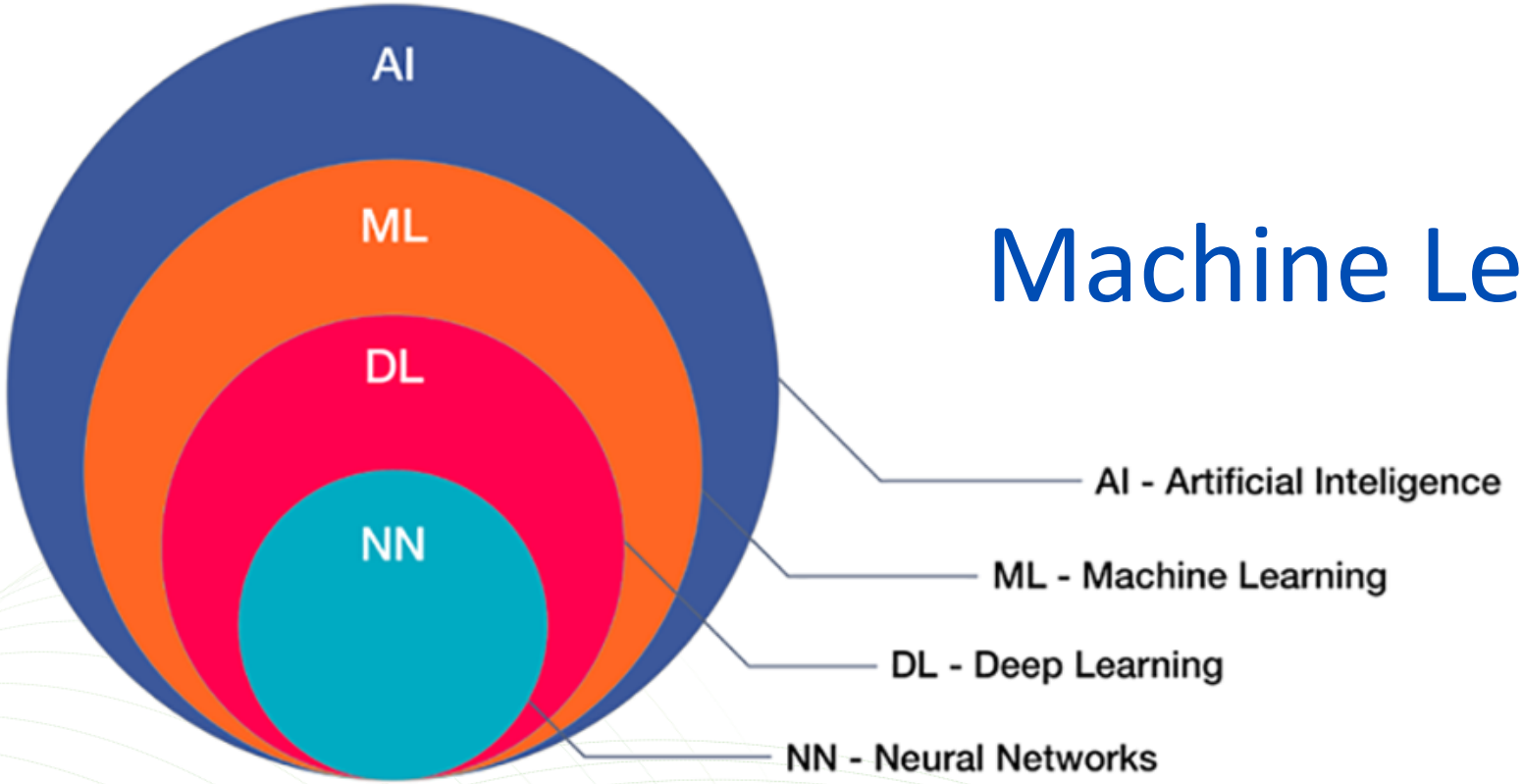
- AI - Artificial Intelligence
- ML - Machine Learning
- DL - Deep Learning
- Generative AI
- LLM
- CHATBOTS
- DEEP FAKES (Audio/Video)



AI, Machine Learning, Deep Learning and Generative AI Explained

FM: foundation model - LLM

Machine Learning



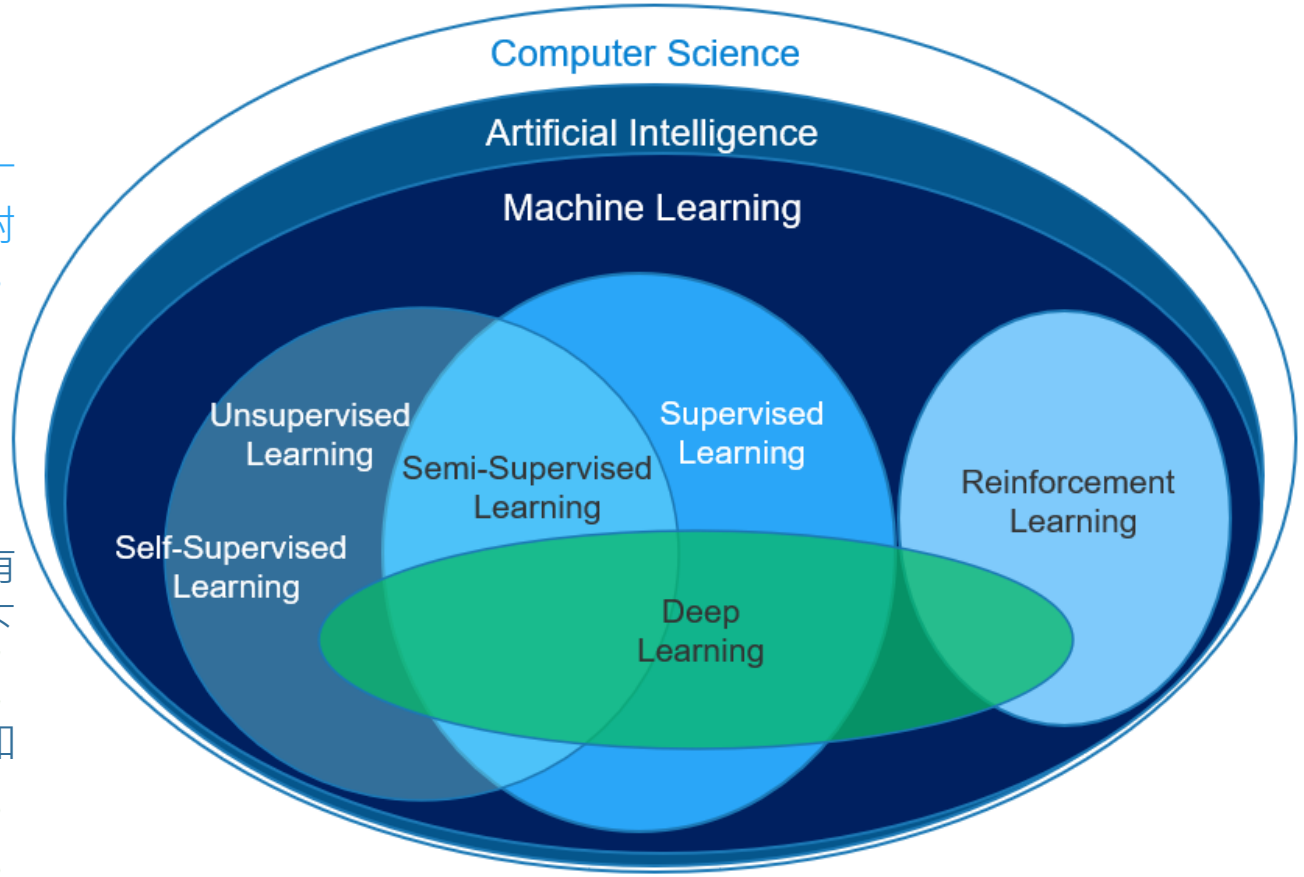
Supervised Learning、Unsupervised Learning、Reinforcement Learning

- **Supervised Learning**

在**監督式學習**中，可以使用一組輸入資料和一組對應的配對已標記輸出資料來訓練模型。通常是手動完成標記。

- **Unsupervised Learning**

非監督式機器學習是指在沒有任何已標記輸出資料的情況下提供演算法輸入資料。然後，演算法會自行識別資料內部和之間的模式與關係。



半監督式學習是指將監督式學習和非監督式學習技術同時套用於常見問題。它本身就是機器學習的另一種類別。

如果難以取得資料集的標籤，就可以套用半監督式學習。您可能擁有較少量的已標記資料，但有大量未標記的資料。與單獨使用已標記資料集相比，如果結合監督式和非監督式學習技術，則可以取得更高的準確性和效率。

可以使用監督式學習技術來解決已知結果以及已標記資料的問題。範例包括垃圾電子郵件分類、影像辨識，以及依據已知歷史資料預測股票價格。對於未標記資料且目標是發現模式、將類似執行個體分組或偵測異常的場景，可以使用非監督式學習。也可以將其用於缺少已標記資料的探索性任務。範例包括組織大型資料封存、建置推薦系統，以及依據客戶的購買行為分組客戶。

Classical Machine Learning

Task Driven

Supervised Learning (Pre-Categorized Data)

- Classification 分類 — Divide by color e.g. Identity Fraud Detection
- Regression 迴歸 — Divide by length e.g. Market Forecasting

- Predict a continuous numeric target variables

https://www.youtube.com/watch?v=E0Hmnixke2g

All Machine Learning algorithms explained in 17 min

Object:

Predication & Predictive Models

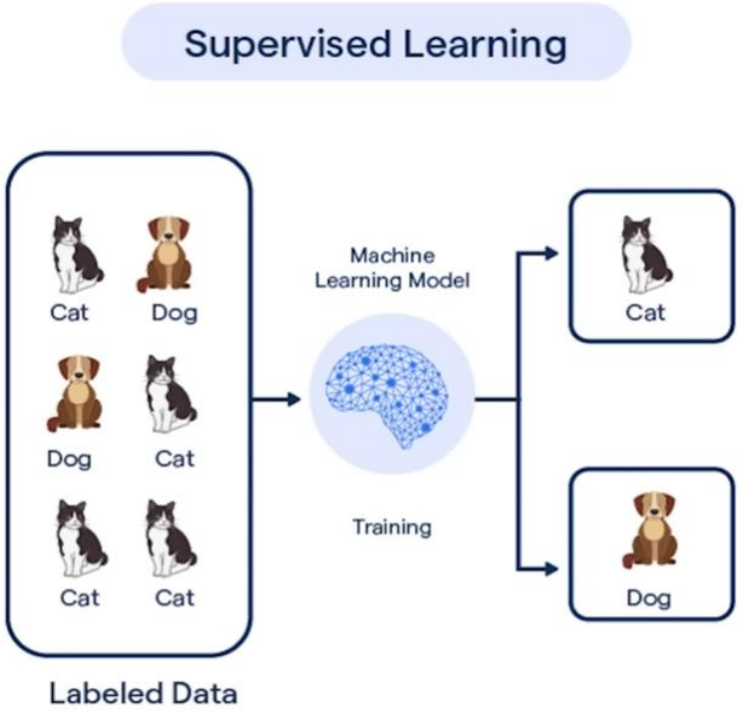
Data Driven

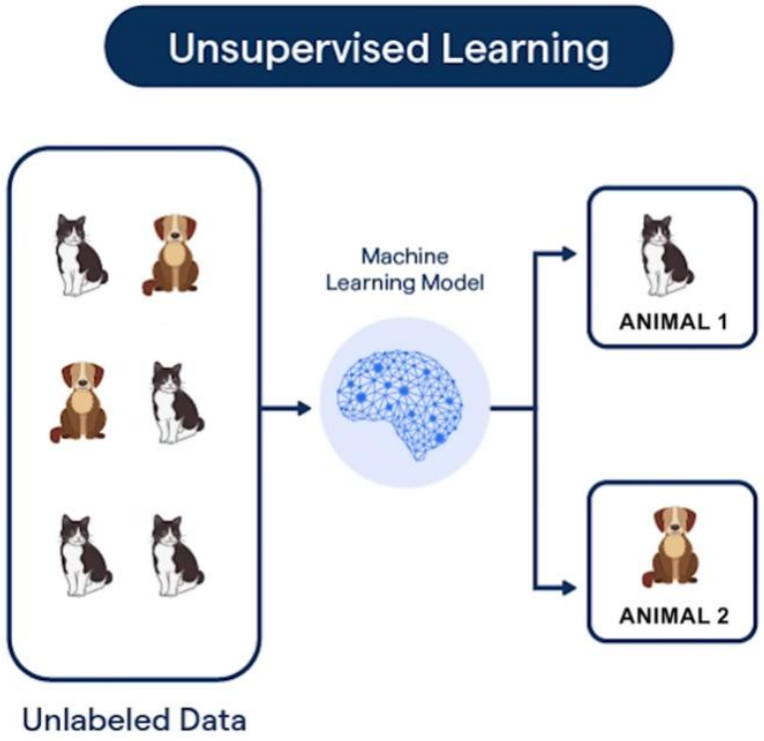
Unsupervised Learning (Unlabeled Data)

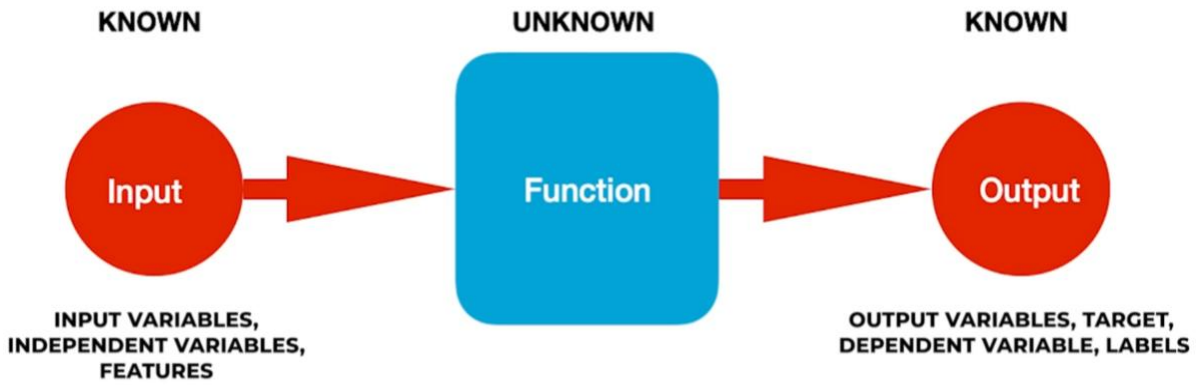
- Clustering 分群(簇) — Divide by similarity e.g. Targeted Marketing
- Association — Identify Sequences e.g. Customer Recommendation
- Dimension Reduction — Wider Dependencies e.g. Big Data Visualization

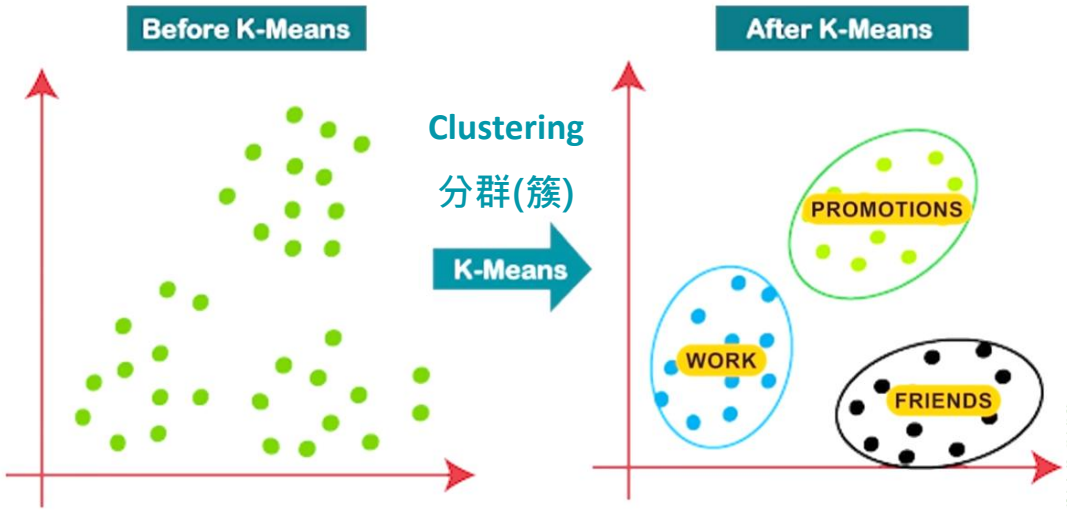
Object:

Pattern/Structure Recognition

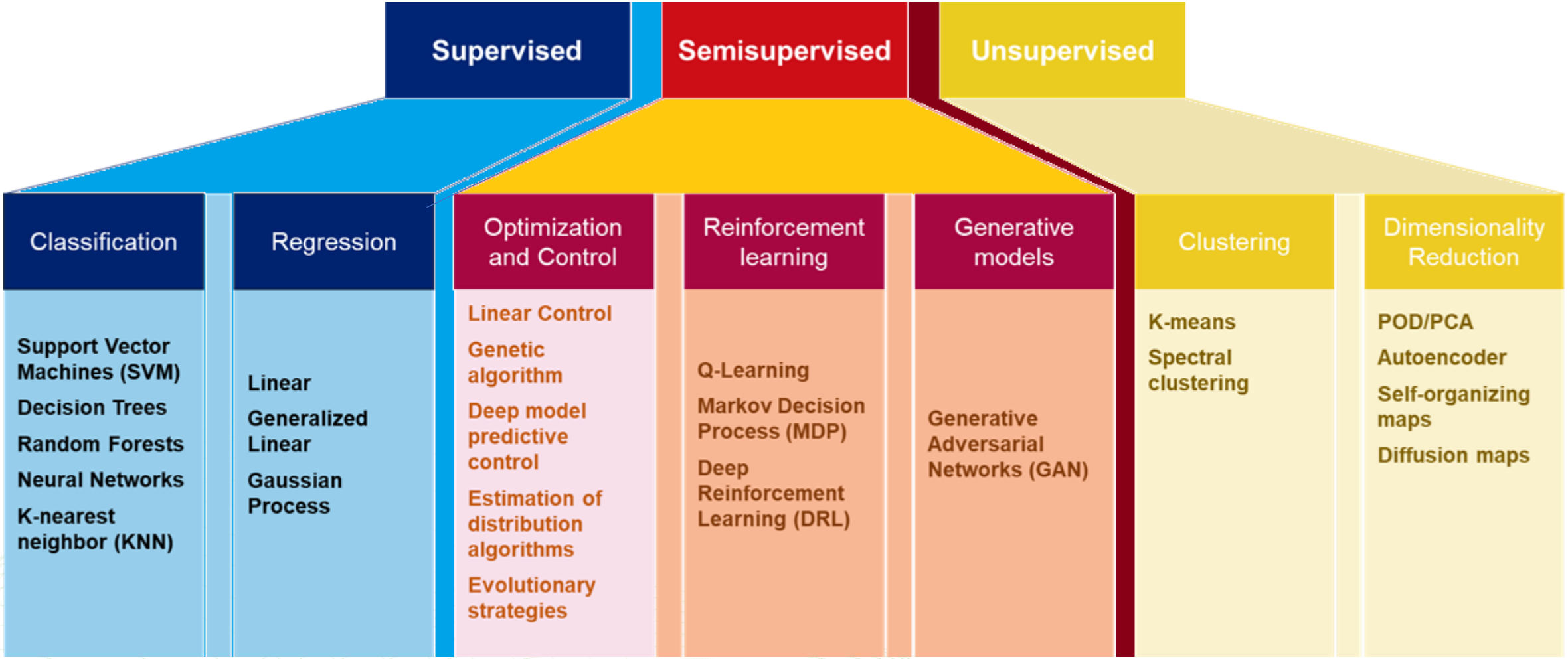

Supervised Learning
Cat
Dog
Dog
Cat
Cat
Cat
Machine Learning Model
Training
Cat
Dog
Labeled Data


Unsupervised Learning
Machine Learning Model
ANIMAL 1
ANIMAL 2
Unlabeled Data


KNOWN
UNKNOWN
KNOWN
Input
Function
Output
INPUT VARIABLES, INDEPENDENT VARIABLES, FEATURES
OUTPUT VARIABLES, TARGET, DEPENDENT VARIABLE, LABELS


Before K-Means
After K-Means
Clustering
分群(簇)
K-Means
PROMOTIONS
WORK
FRIENDS

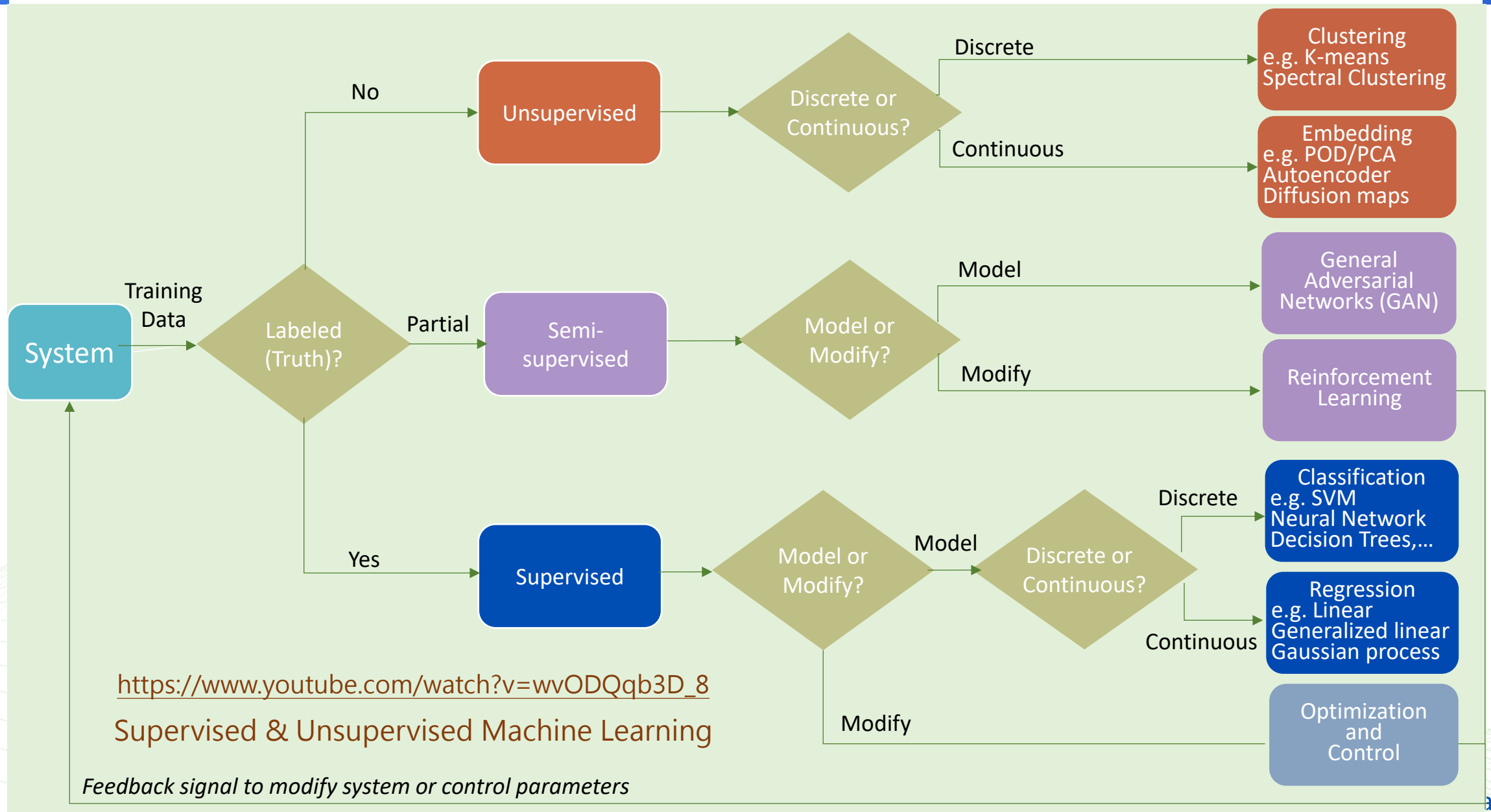
Types of Machine Learning



https://www.youtube.com/watch?v=wvODQqb3D_8

Supervised & Unsupervised Machine Learning

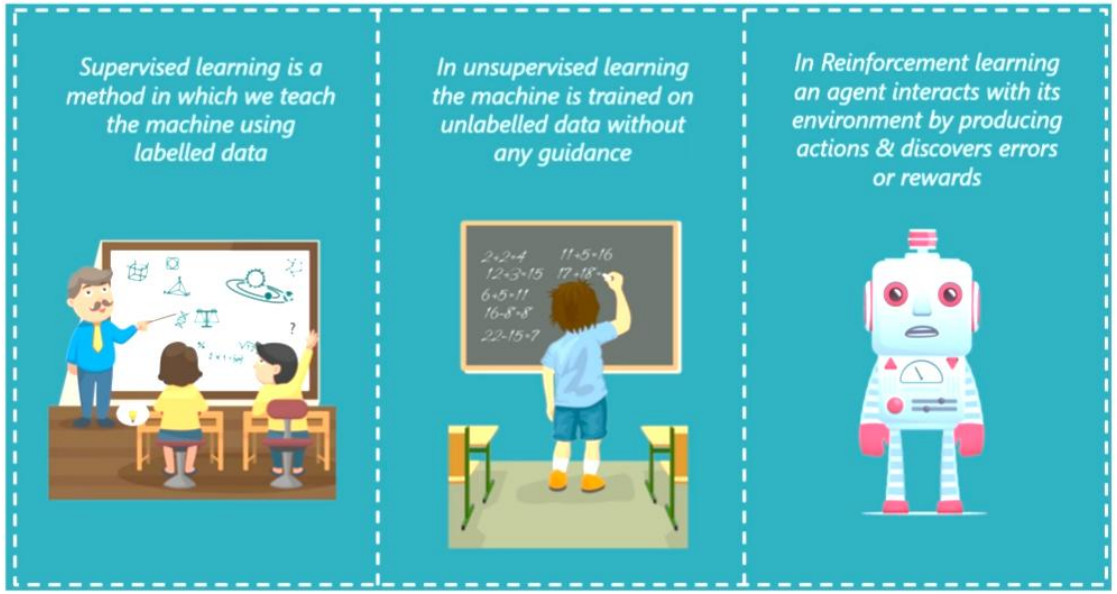
Types of Machine Learning

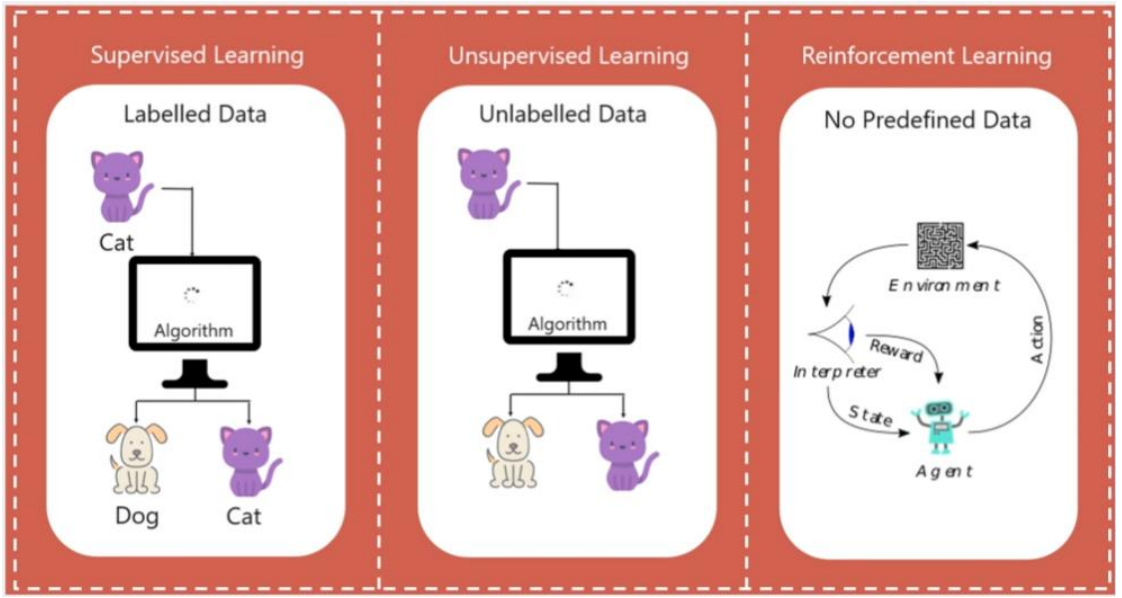


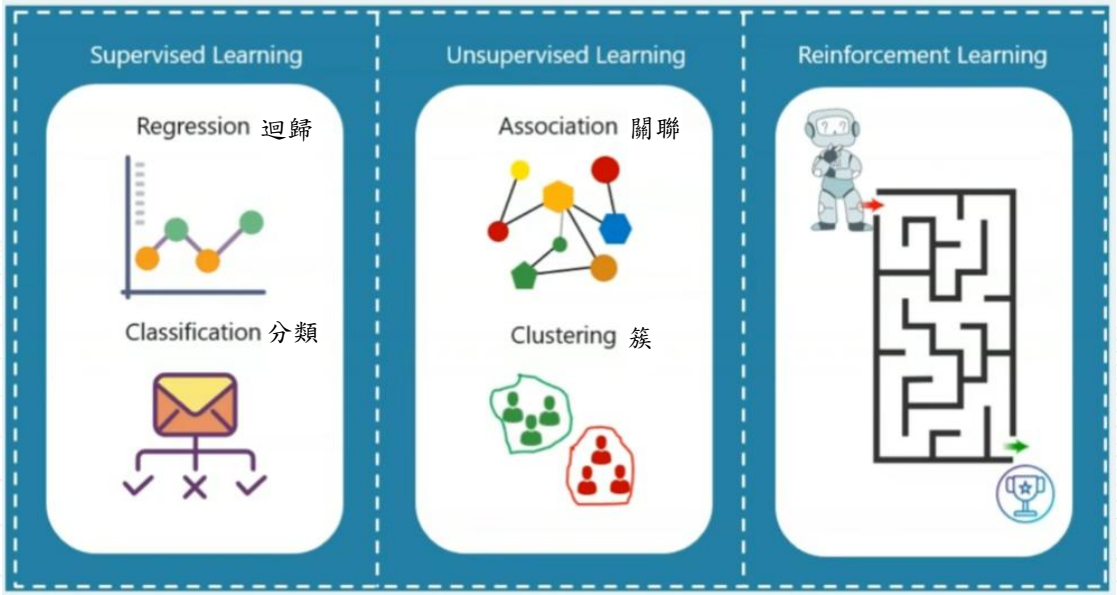
https://www.youtube.com/watch?v=wvODQqb3D_8

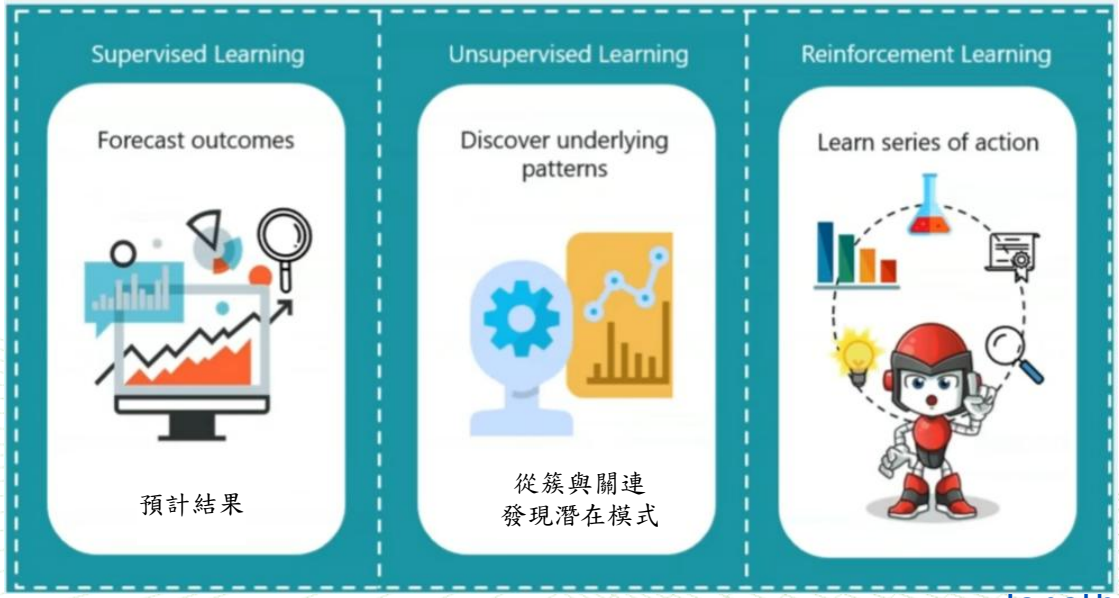
Supervised & Unsupervised Machine Learning

Types of Machine Learning

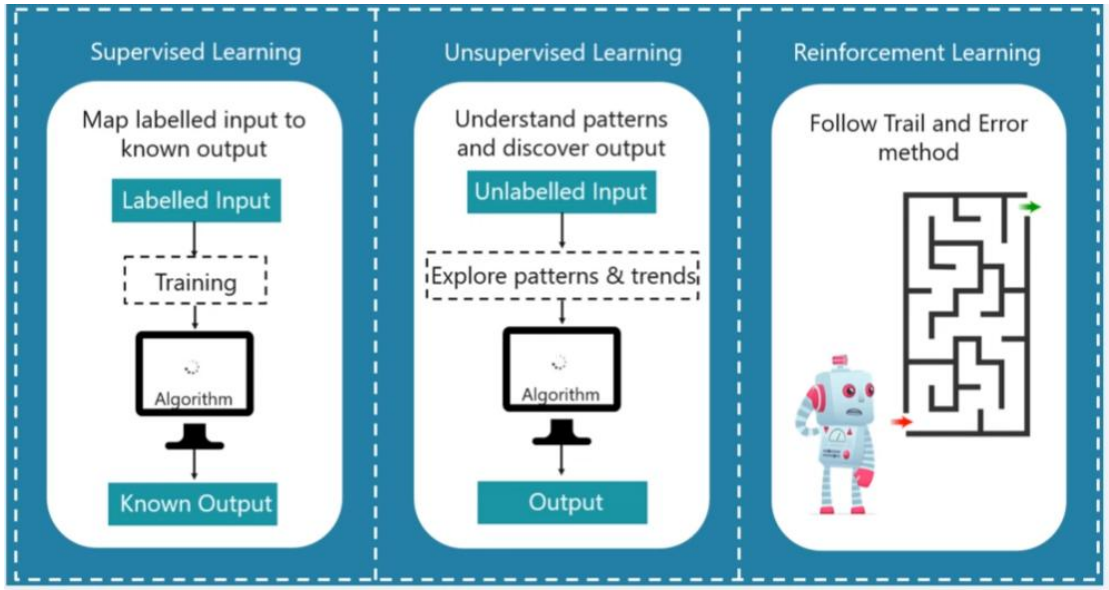


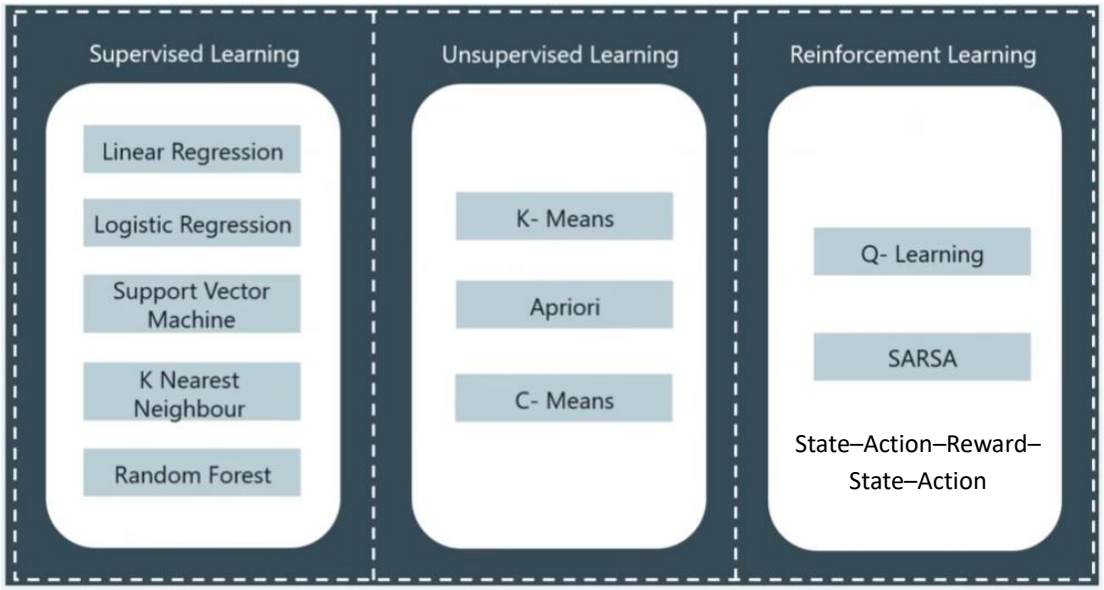


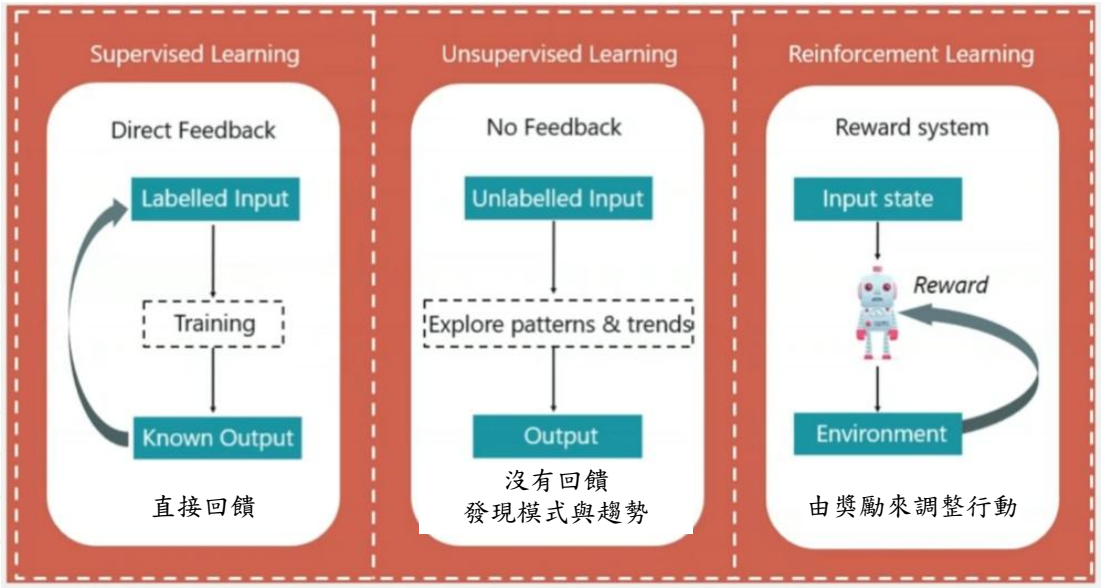


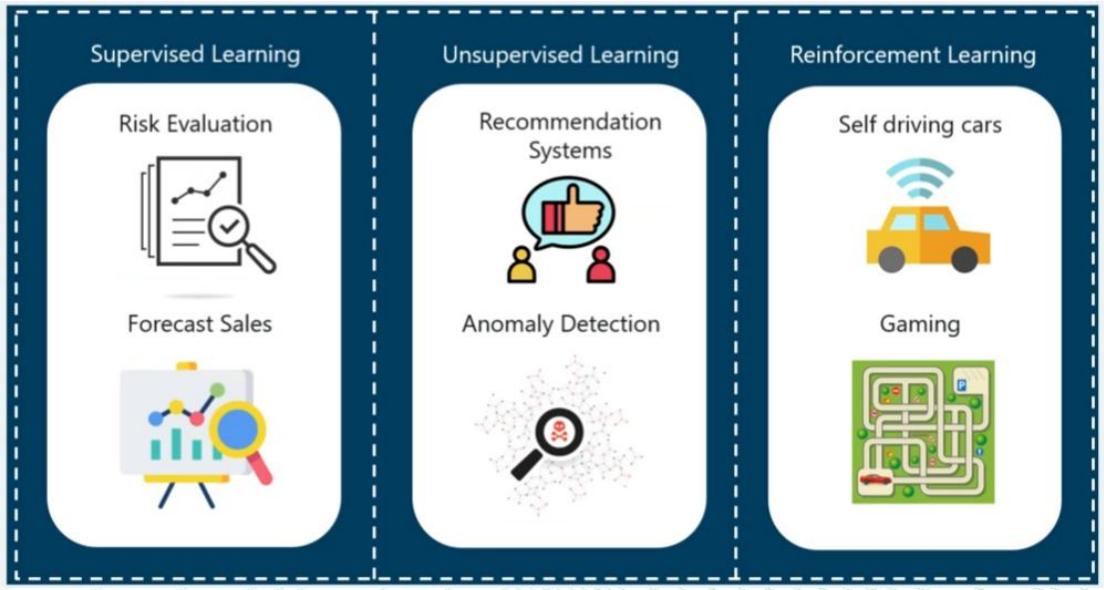


Types of Machine Learning



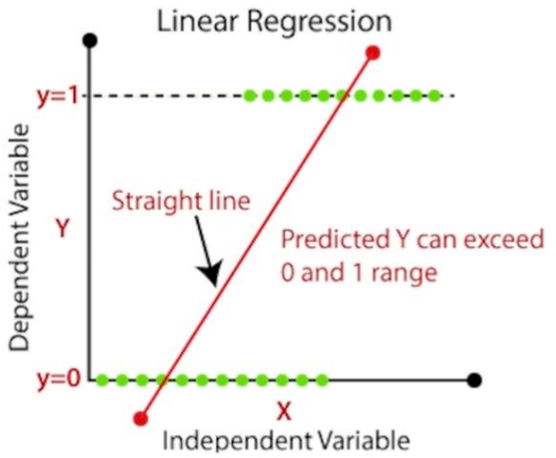


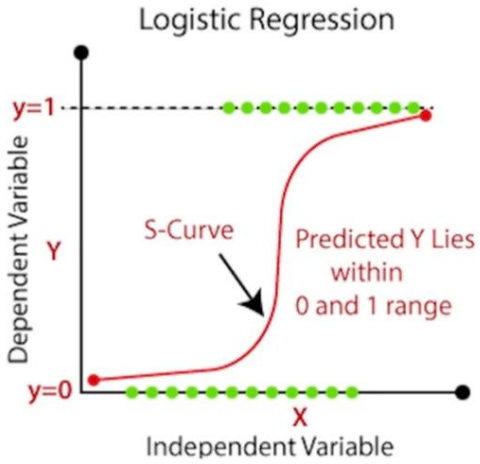




https://www.youtube.com/watch?v=xtOg44r6dsE Supervised vs Unsupervised vs Reinforcement Learning | Edureka Jonathan Chen

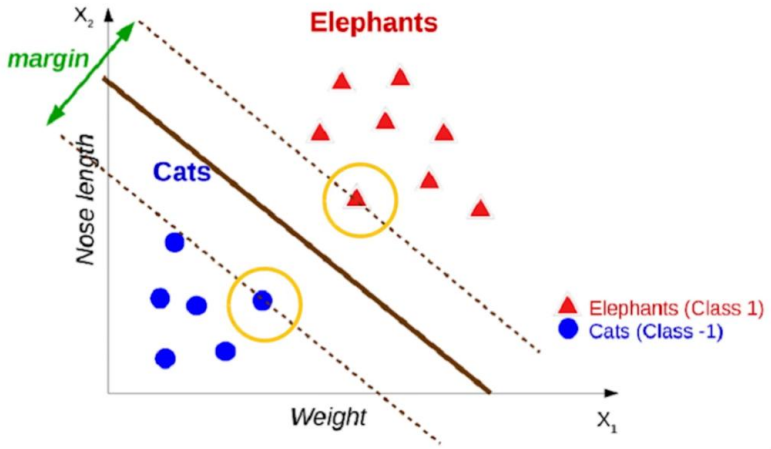
Algorithm for Supervised Learning - Classification and Regression



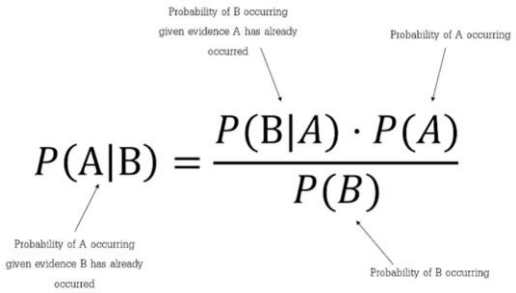


$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

SIGMOID FUNCTION

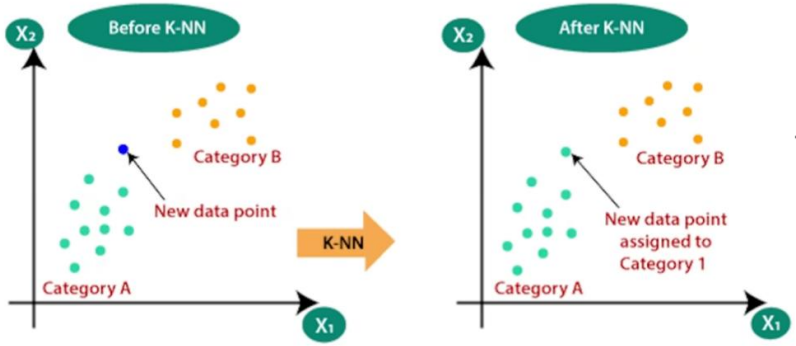


Support Vector Machine (SVM)

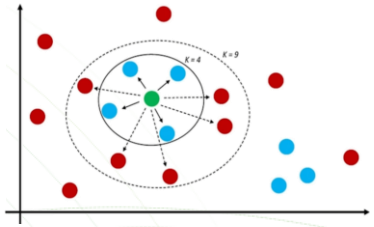


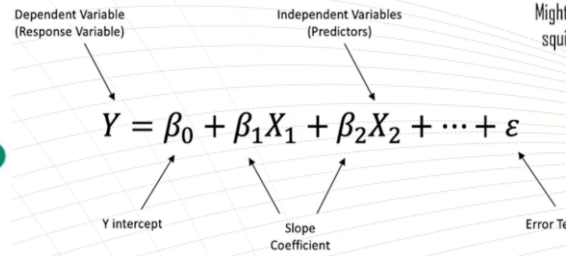
Naïve Bayes Classifier

K NEAREST NEIGHBORS (KNN)

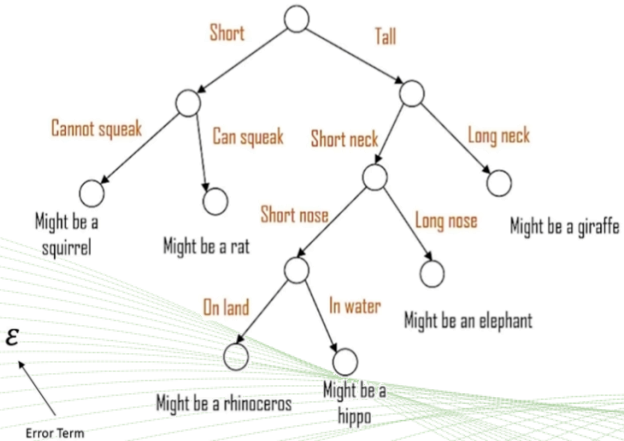


NON-PARAMETRIC ALGORITHM

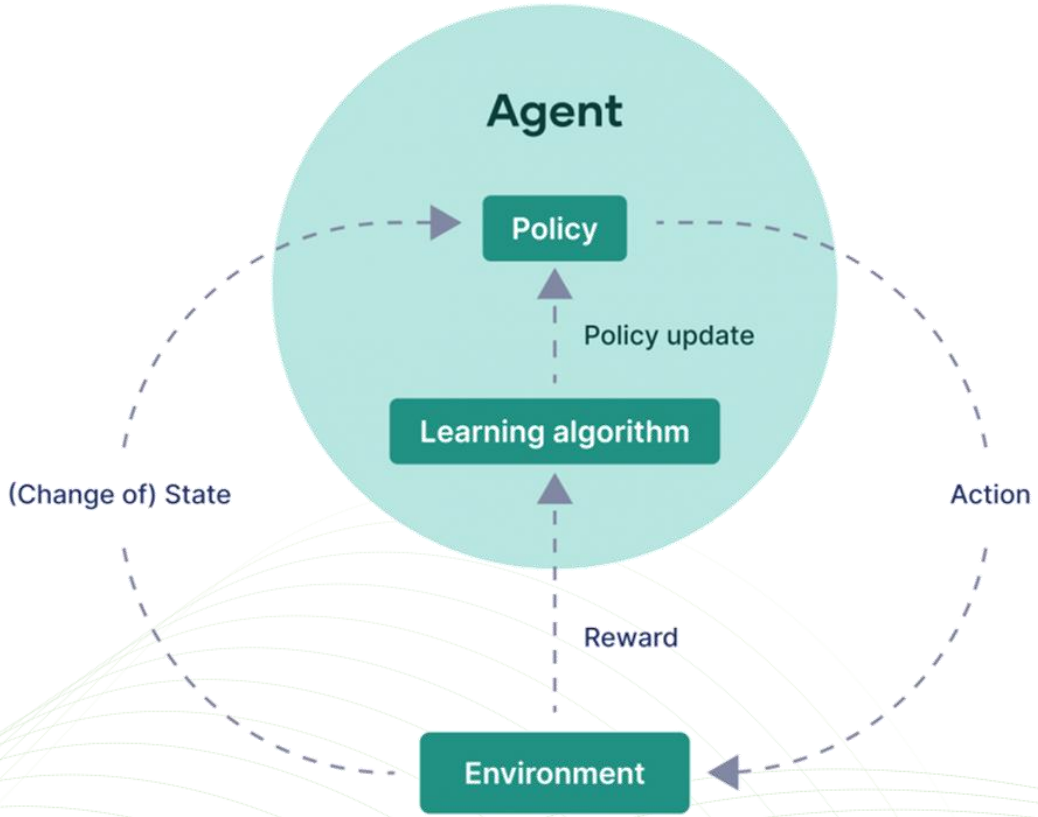




Decision Tree Classifier

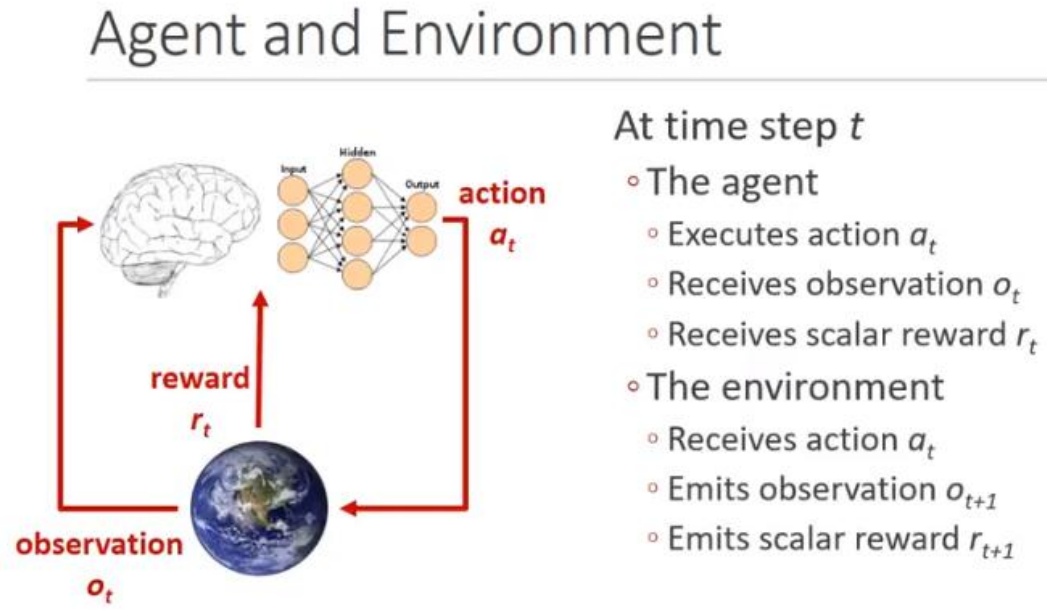


Random Forest Classifier



Reinforcement Learning

深度強化學習 (Reinforcement Learning) 概要圖



從上圖可以看出來，大腦代表agent，地球代表 environment (環境)，reward (獎勵)就是環境所提供的反饋，reward 由模型設計者定義，reward 的定義對強化學習來說是個很重要的一環。

MDP (Markov Decision Process) 馬可夫決策過程

MDP的一個重要觀念：” 未來只取決於當前”

Agent 部份(大腦)

訓練出一個agent(大腦)可以去適應environment(環境)

1. 會將environment環境每一個時間點的observation (觀察)的集合當作環境的狀態 (State)
2. 從環境的狀態 (State) 跟 reward(獎勵)再去選擇一個最好的 action(動作)，稱為 policy(策略)

Environment 部份(環境)

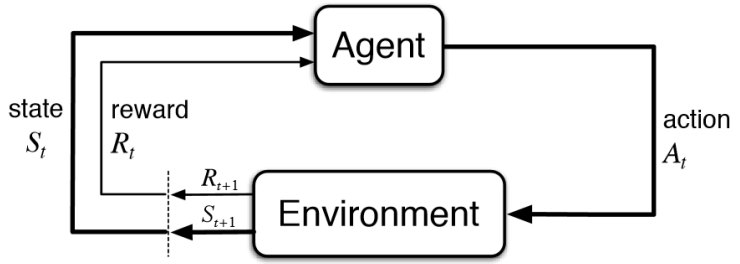
就是一個環境在不同的狀態下會有不同的情況，並將這些情況告訴大腦，讓大腦可以去學習。

1. 會接收 Agent執行的 action(動作)，並吐出 reward 跟 observation給 agent。

強化學習的目的就是找到一個最好的 Policy(策略)，可以讓 reward最多!!!

Jonathan Chen

瑪律可夫決策過程的工作原理　　MDP=(S,A,P,R,γ)



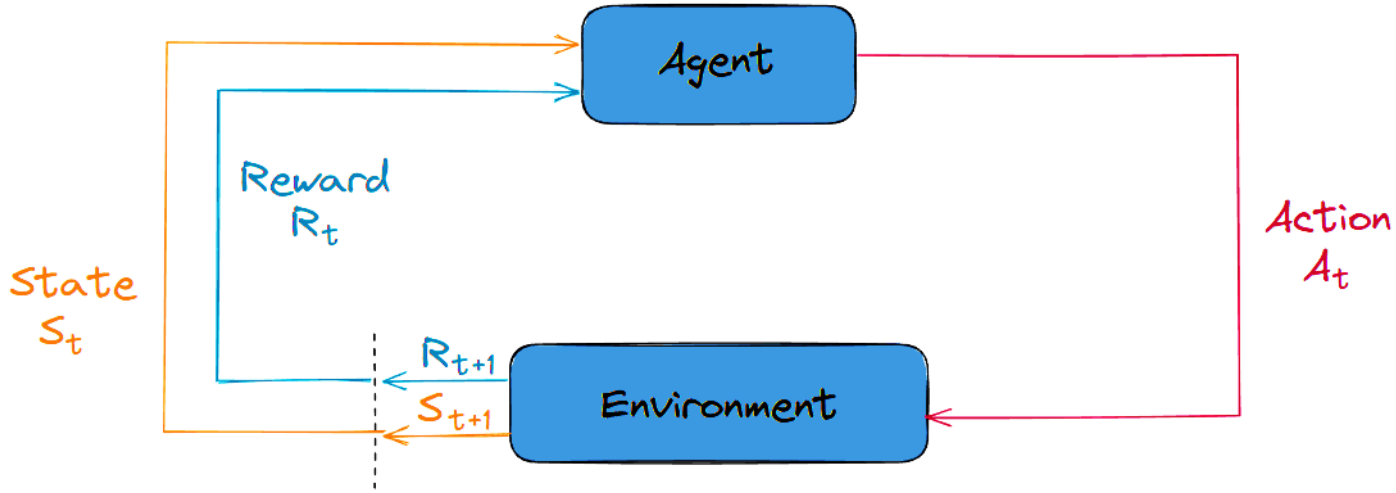
MDP 由幾個關鍵元件定義，這些元件描述了代理與其環境之間的交互：

- **狀態 (S)：** 智慧體可能處於的所有可能情況或配置的集合。例如，機器人在房間中的位置或產品的庫存水準。
- **動作 (A)：** 智慧體在每個狀態下可以採取的所有可能移動的集合。 對於機器人，這可以是向前、向左或向右移動。
- **轉移概率 (P)：**採取特定行動後，從當前狀態轉移到新狀態的概率。這反映了環境中的不確定性，例如機器人車輪打滑。
- **獎勵函數 (R)：**一種信號，指示轉換到新狀態的即時價值。獎勵可以是正面的或負面的，並引導代理朝著期望的結果前進。
- **策略 (π)：** 智慧體在每個狀態下選擇動作的策略。解決 MDP 的最終目標是找到一個最優策略，即在長期內最大化總預期獎勵的策略。

這個過程是迴圈的：智慧體觀察當前狀態，根據其策略選擇一個動作，獲得獎勵，然後轉移到新的狀態。這個迴圈不斷重複，使智慧體能夠從其經驗中學習。

Markov Decision Process

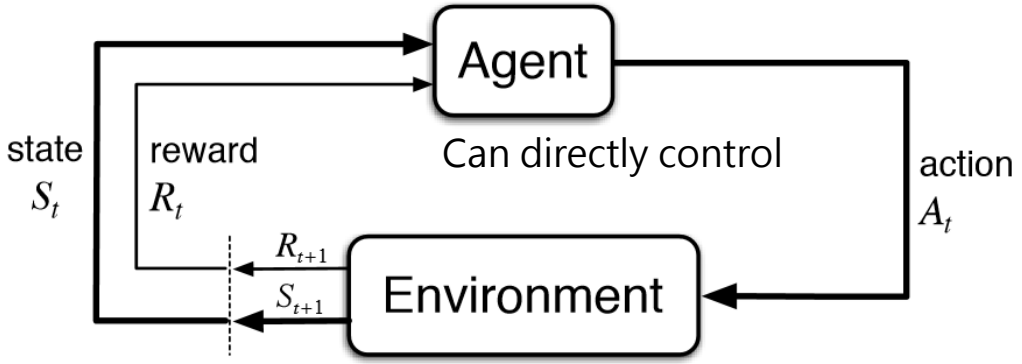
https://deeplizard.com/learn/video/eMxOGwbdqKY



Let's break down this diagram into steps.

1. At time t, the environment is in state S_t.
2. The agent observes the current state and selects action A_t.
3. The environment transitions to state S_{t+1} and grants the agent reward R_{t+1}.
4. This process then starts over for the next time step, $t+1$.
    - Note, $t+1$ is no longer in the future, but is now the present. When we cross the dotted line on the bottom left, the diagram shows $t+1$ transforming into the current time step t so that S_{t+1} and R_{t+1} are now S_t and R_t.

- The state and reward at time t depend on State-action pair for time (t-1).
- In an MDP, the Agent' s objective is to maximize the total reward.



Cannot directly control, interact with through agent.

Markov Decision Process (MDP)

$S_0, A_0, r_0, S_1, A_1, r_1, S_2, A_2, r_2, S_3, A_3, r_3, \circ\ \circ\ \circ$

Markov Property

Each state is only dependent on an immediately previous state.

Policy π: S -> a

π = (a | S) = probability

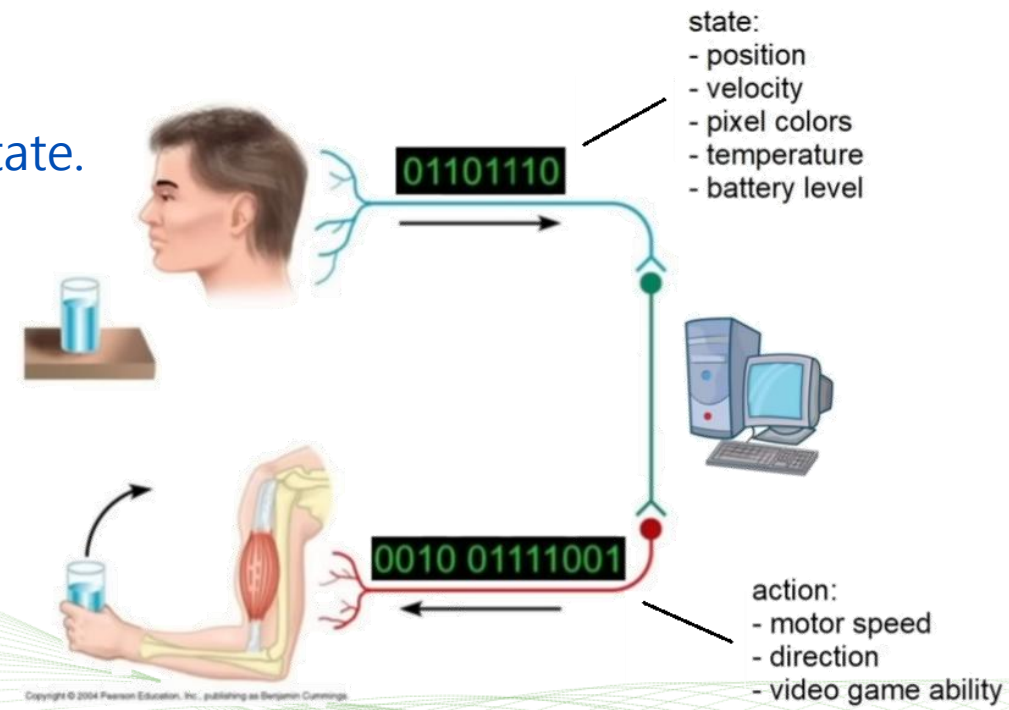
Return G_t

$G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} \circ\ \circ\ \circ$

γ discount factor $0 \leqq \gamma \leqq 1$

Goal

Find policy which maximizes return.

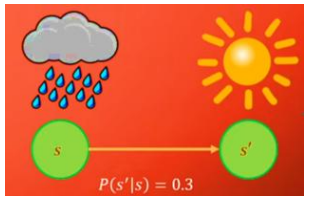


Markov Chain

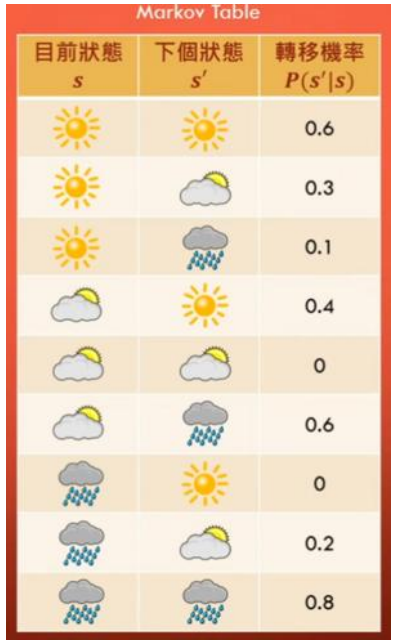
Transition 轉移 : 從一個狀態 (s) 移動到另一個狀態 (s’)

Transition Probability 轉移概率 :

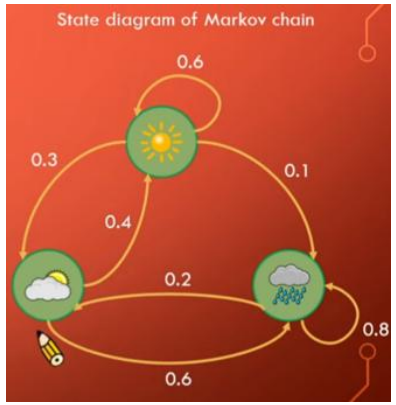
- 從一個狀態 (s) 移動到另一個狀態 (s’)的機率
- P(s’ | s)



Table



Matrix



Markov Reward Process (MRP)

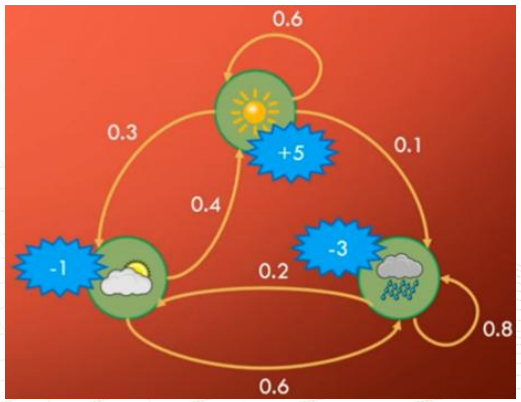
State 狀態 : (s)

Transition Probability (轉移概率) :

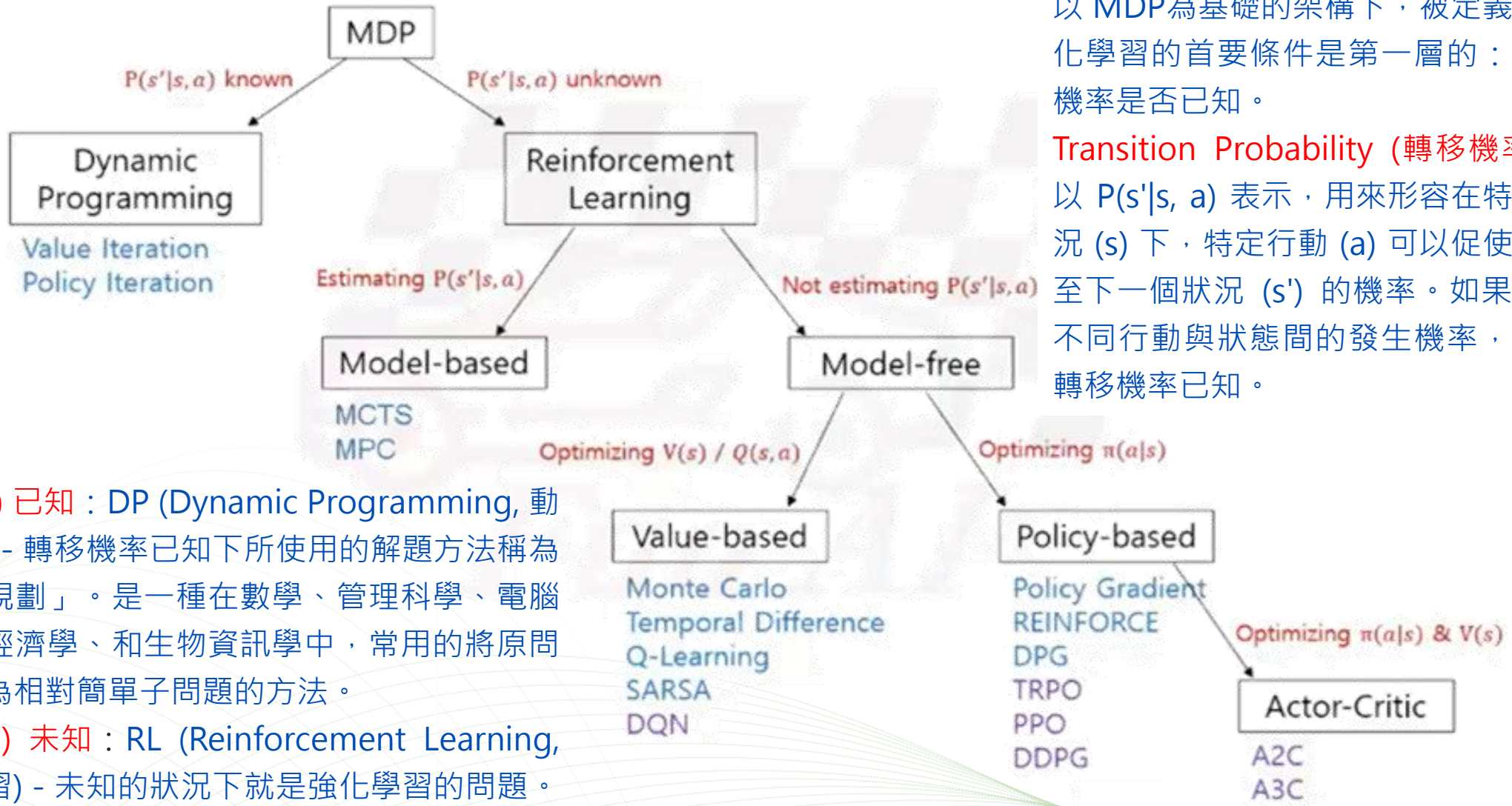
- P(s’ | s)

Reward function (獎勵函數) : r

- R(s)



MDP (Markov Decision Process, 馬可夫決策過程)



以 MDP為基礎的架構下，被定義為強化學習的首要條件是第一層的：轉移機率是否已知。

Transition Probability (轉移機率)，以 P(s'|s, a) 表示，用來形容在特定狀況 (s) 下，特定行動 (a) 可以促使轉移至下一個狀況 (s') 的機率。如果知道不同行動與狀態間的發生機率，則稱轉移機率已知。

P(s'|s, a) 已知：DP (Dynamic Programming, 動態規劃) - 轉移機率已知下所使用的解題方法稱為「動態規劃」。是一種在數學、管理科學、電腦科學、經濟學、和生物資訊學中，常用的將原問題分解為相對簡單子問題的方法。

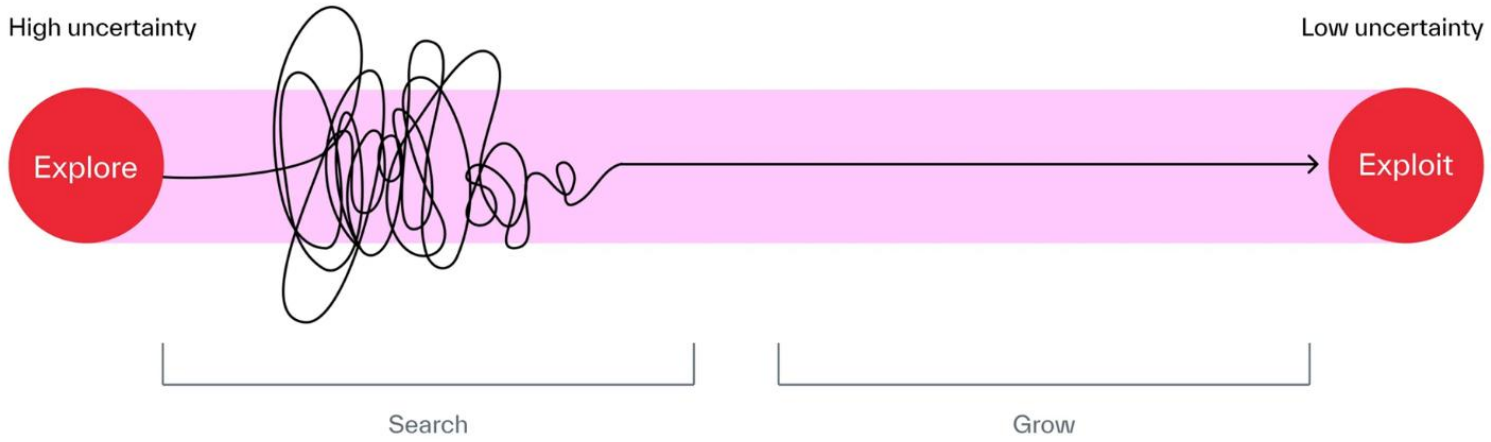
P(s'|s, a) 未知：RL (Reinforcement Learning, 強化學習) - 未知的狀況下就是強化學習的問題。必須以 sample 的方式，模擬環境資訊作為 training data 做訓練。

MDP 的目標就是做出一系列能夠達到目標的決策，也就是在每一個「時步/時間點 (time step)」做出一個「決策/行為 (action)」

Jonathan Chen

Exploration and Exploitation (探索和利用)

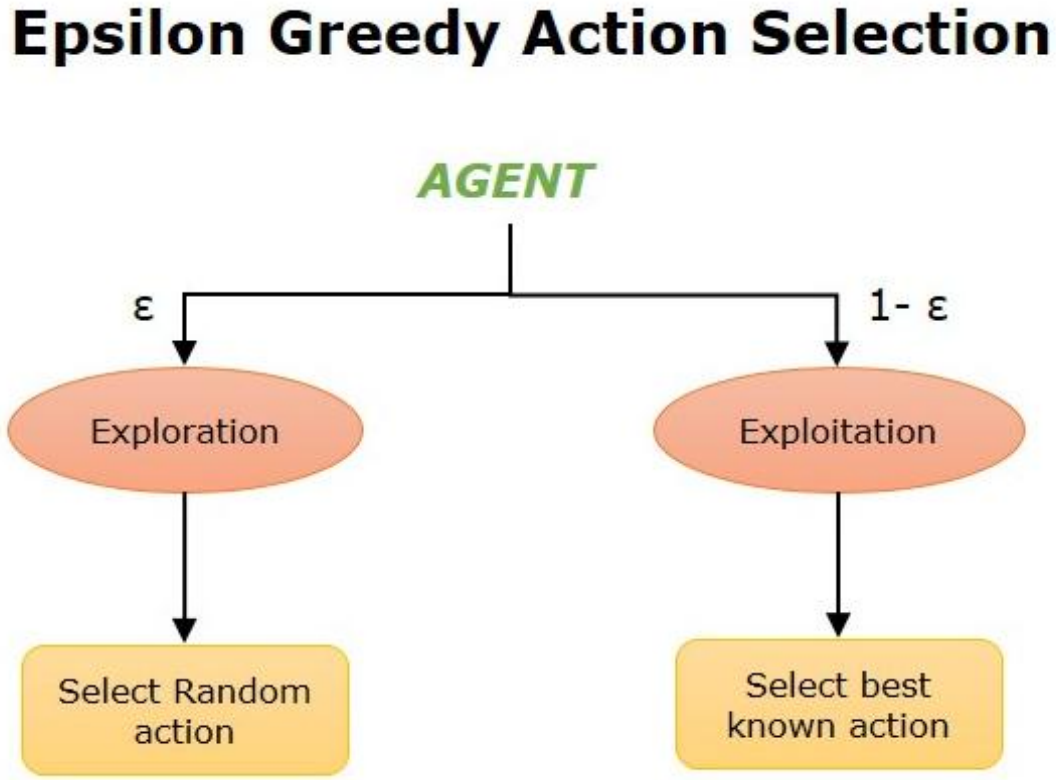
Exploration is the act of searching for new opportunities, knowledge, or actions, while **Exploitation** is the act of utilizing existing knowledge to maximize immediate rewards



探索(**Exploration**) - 利用(**Exploitation**)困境，也稱為探索(**Exploration**) - 利用(**Exploitation**)權衡，是許多領域中出現的決策的基本概念。它被描述為兩種相反策略之間的平衡行為。利用涉及根據系統的當前知識選擇最佳選項，而探索涉及嘗試新的選項，這些選項可能會在未來帶來更好的結果，但會犧牲利用機會。在許多目標是實現長期利益最大化的決策問題中，找到這兩種策略之間的最佳平衡是至關重要的挑戰。

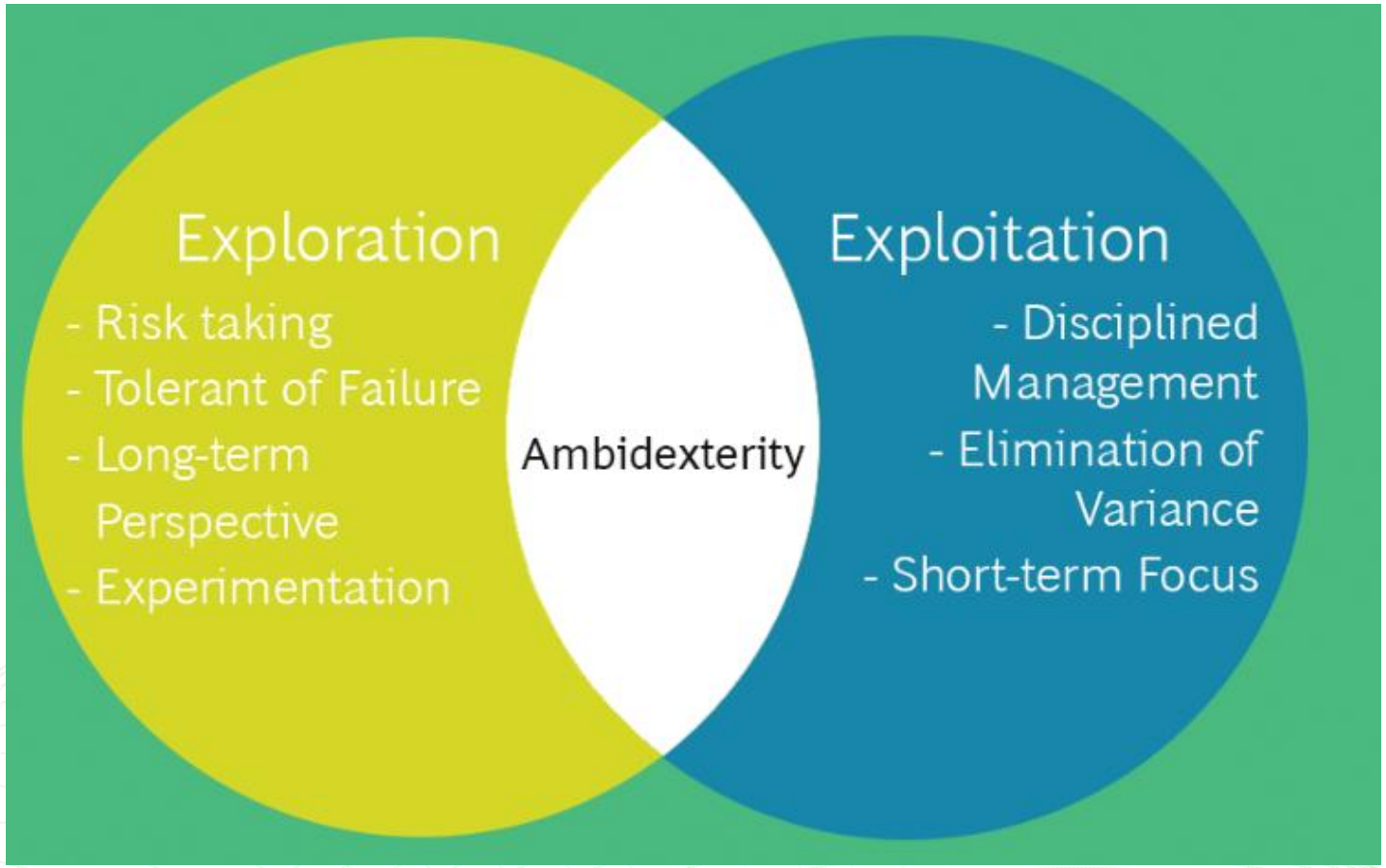
Exploration and Exploitation (探索和利用)

HIGH	Uncertainty	LOW
Search and breakthrough	Focus	Efficiency and growth
Many small bets, expecting few outsized winners	Financial philosophy	Safe haven with **steady returns** and dividends
Iterative experimentation, embracing speed, failure, learning and rapid adaption	Culture and process	**Linear execution**, embracing planning, predictability, and minimal failure
Explorers who excel in uncertainty	People and skills	**Managers who are strong at organizing and planning**



Epsilon-greedy is a reinforcement learning strategy that balances exploitation (choosing the best-known action) with exploration (choosing a random action to discover new, potentially better options). The "epsilon" (ϵ) value determines the probability of exploration: the agent chooses the best action with probability $1 - \epsilon$ and a random action with probability ϵ. For example, if ϵ=0.1 , there is a 90% chance it will exploit and a 10% chance it will explore. This balance is crucial for long-term success, as agents can get stuck with a suboptimal choice if they only exploit.

Exploration and Exploitation (探索和利用)



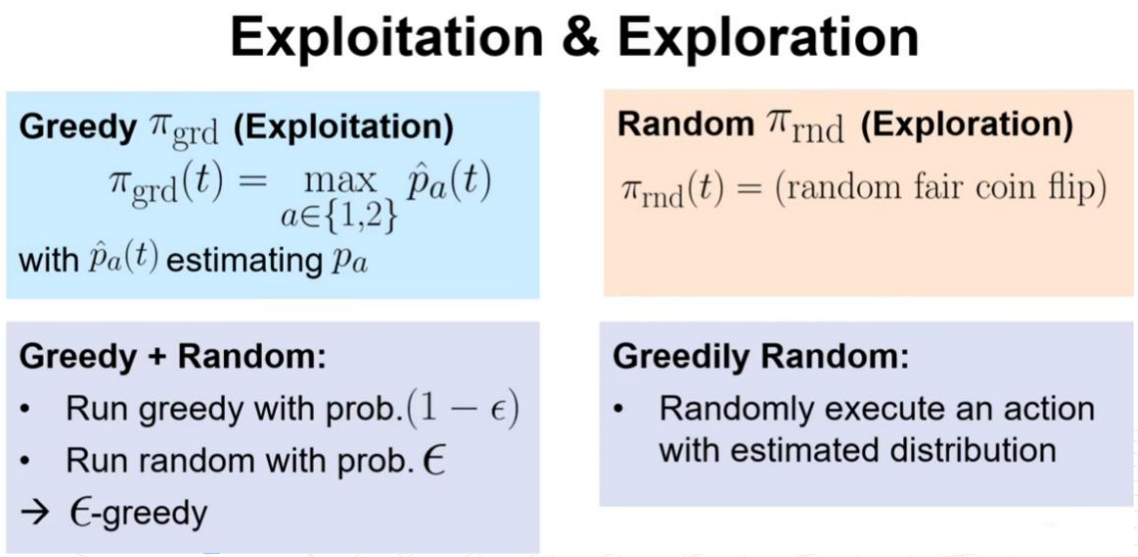
How epsilon-greedy works

- **The probability:** The algorithm uses a value called epsilon (ϵ) to control the balance. A higher ϵ value means more exploration, while a lower ϵ value means more exploitation.
- **The decision:** $0 < \varepsilon < 1$: The agent balances exploration and exploitation.
  - With a probability of ϵ, the agent selects a random action.
  - With a probability of $1-\epsilon$, the agent selects the action it currently believes is the best (the "greedy" action).

Exploration and Exploitation (探索和利用)

The Epsilon-Greedy Strategy

One of the simplest and most widely used approaches to balance exploration and exploitation is the **epsilon-greedy strategy**.



Here's how it works:

- With probability **ε** (epsilon), the agent **explores** by selecting a random action.
- With probability **1-ε**, the agent **exploits** by selecting the best action according to its current knowledge.

Epsilon (ε) is a value between 0 and 1 that controls the exploration rate:

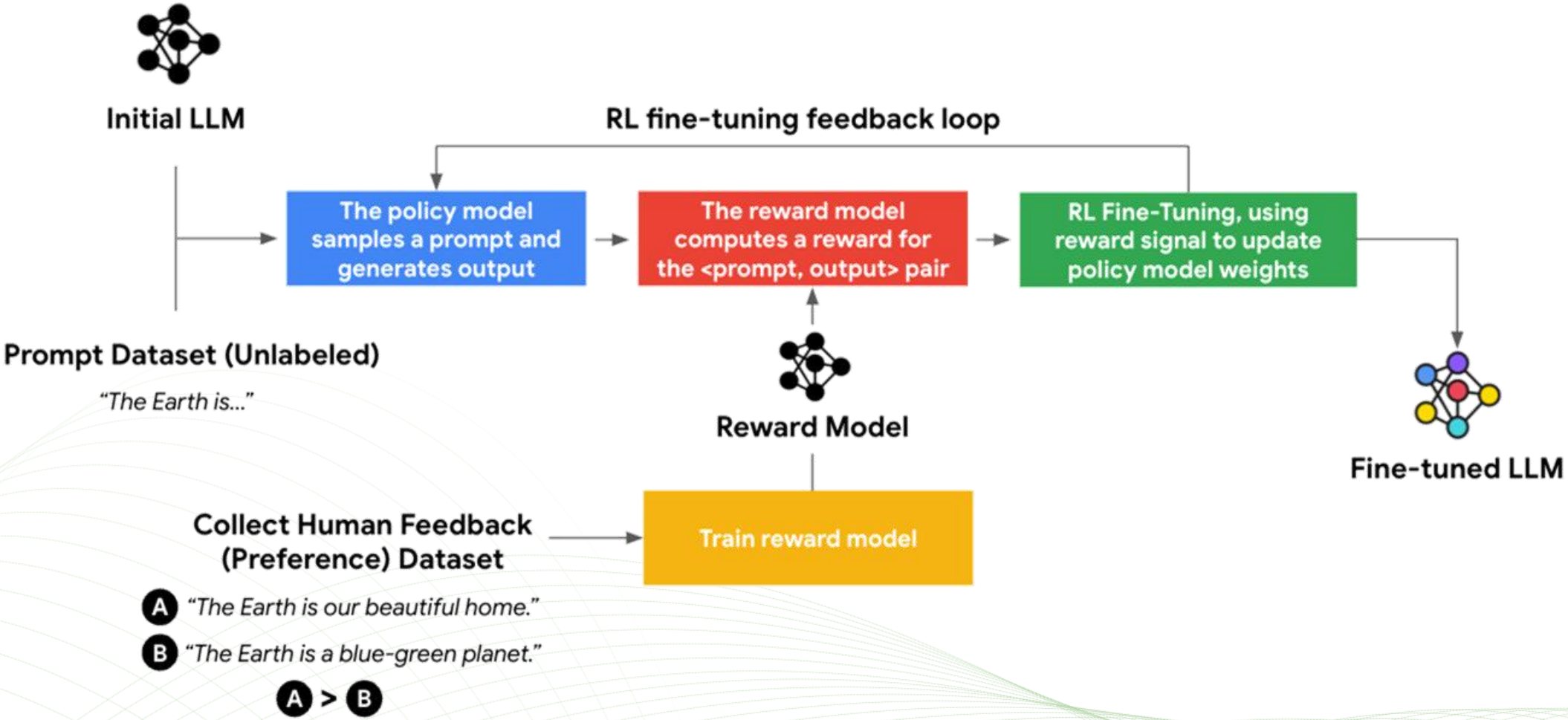
- **ε = 0**: The agent is purely greedy, always exploiting its current knowledge; this is how the agent was behaving in the previous lesson's code.
- **ε = 1**: The agent always explores, selecting random actions.
- **0 < ε < 1**: The agent balances exploration and exploitation.

RLHF
Reinforcement Learning from Human Feedback

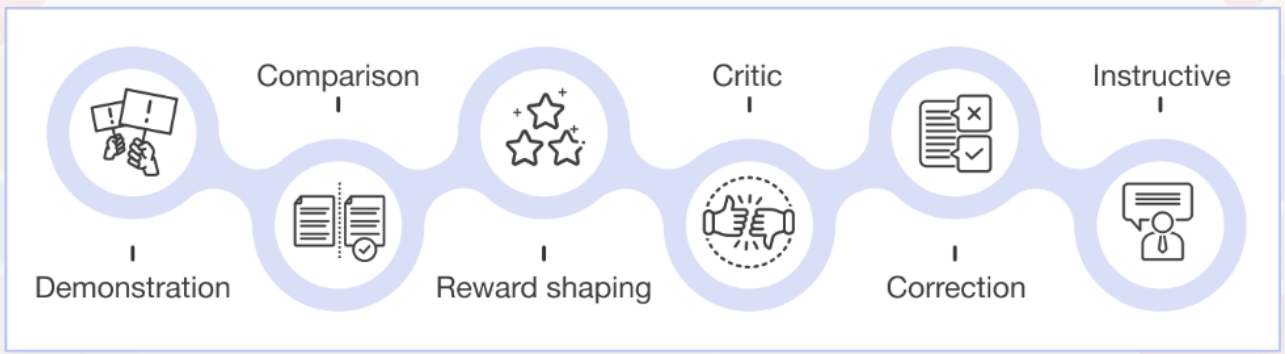
Jonathan Chen

Reinforcement Learning from Human Feedback

RLHF tuning consists of two phases: **reward modeling** and **reinforcement learning**.



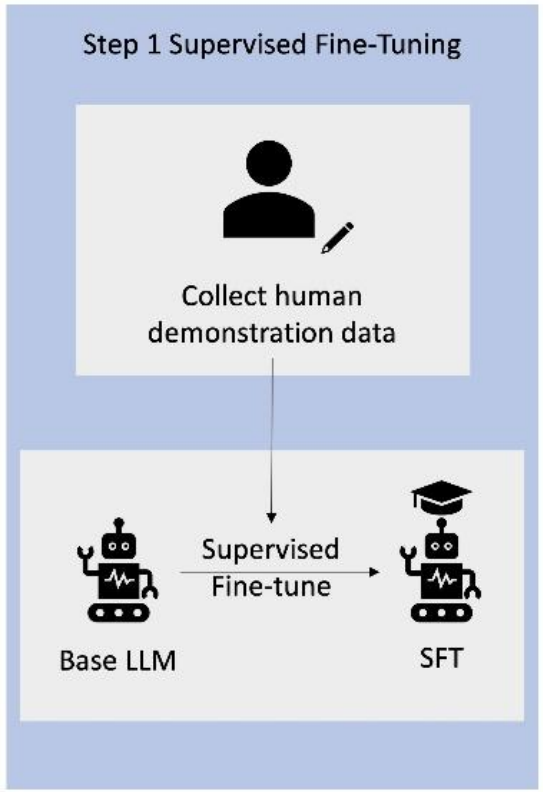
Types of human feedback in RLHF

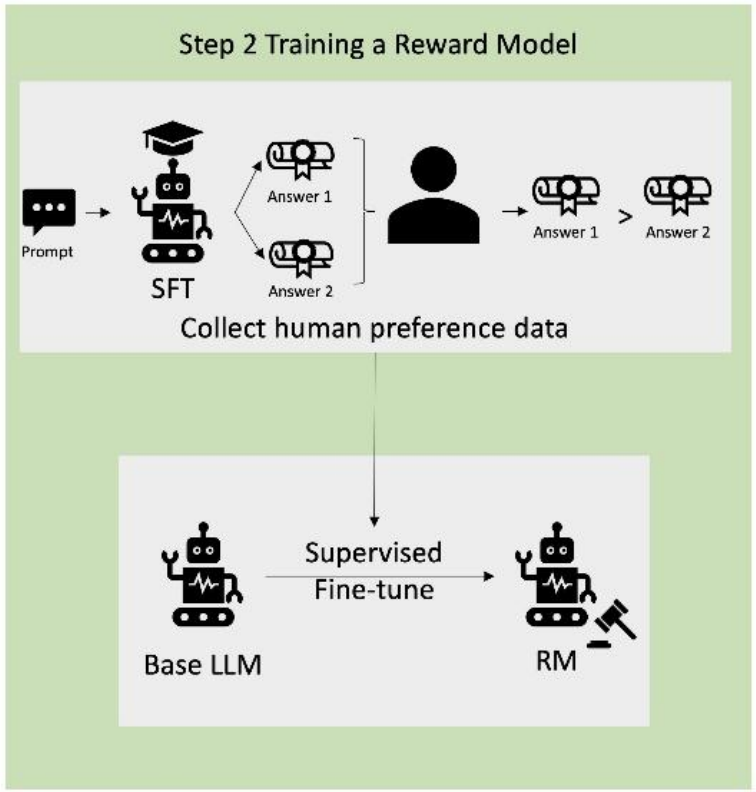


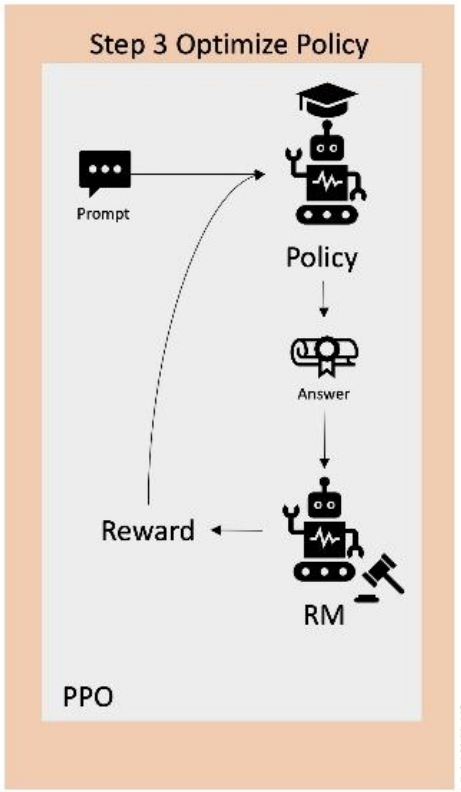
SFT: Supervised Fine-Tuning

RM: Reward Modeling

PPO: Proximal Policy Optimization

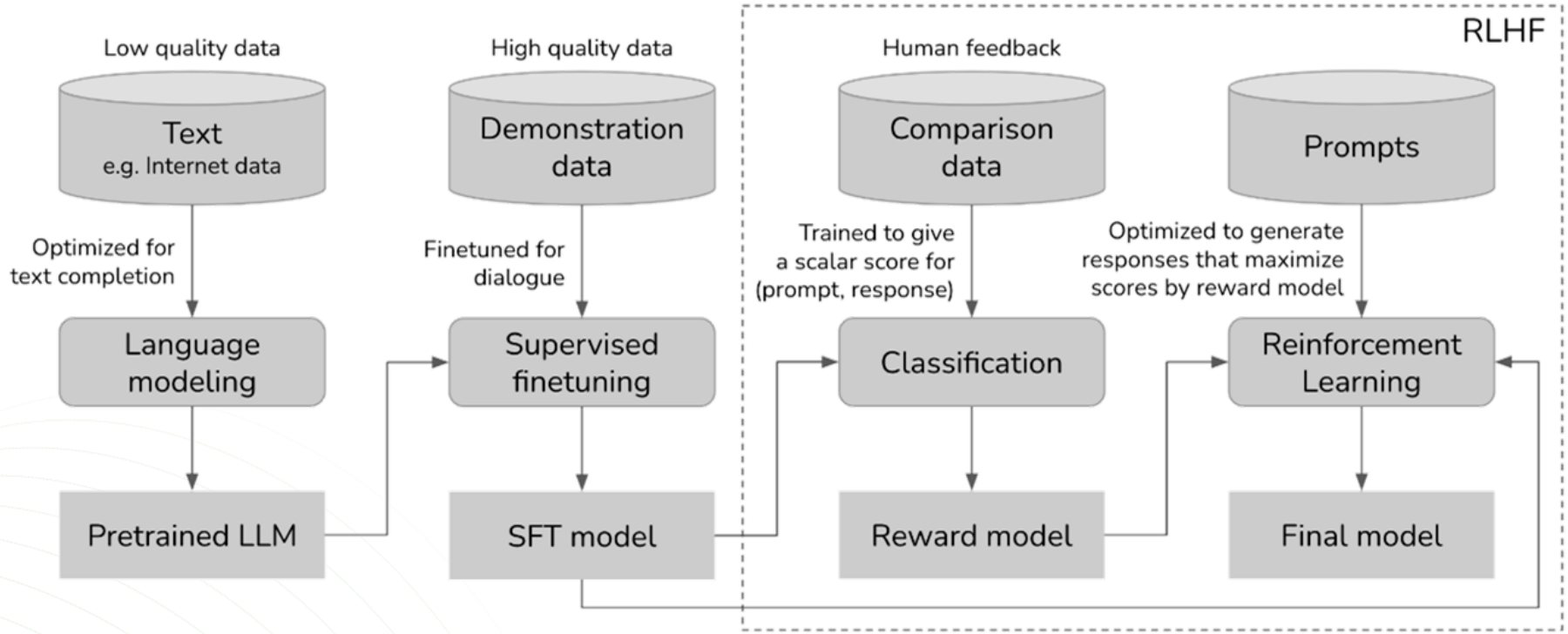






Reinforcement Learning from Human Feedback

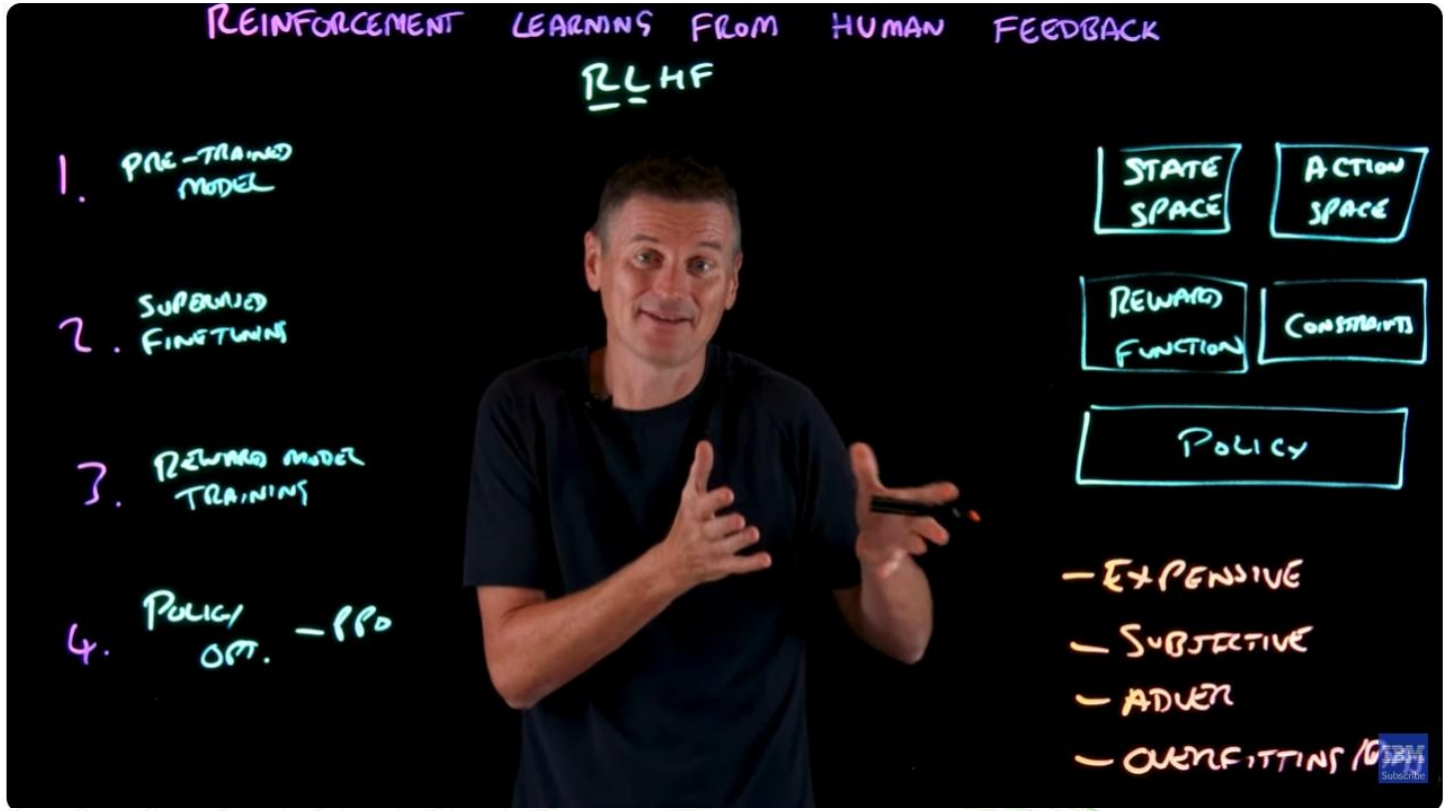
Pre-Trained Model | Supervised Fine-Tuning | Reward Model | Policy Optimization/PPO



Scale May '23	**>1 trillion** tokens	**10K - 100K** (prompt, response)	**100K - 1M** comparisons (prompt, winning_response, losing_response)	**10K - 100K** prompts
Examples **Bolded**: open sourced	GPT-x, Gopher, **Falcon**, LLaMa, **Pythia**, **Bloom**, **StableLM**	**Dolly-v2, Falcon-Instruct**		InstructGPT, ChatGPT, Claude, **StableVicuna**

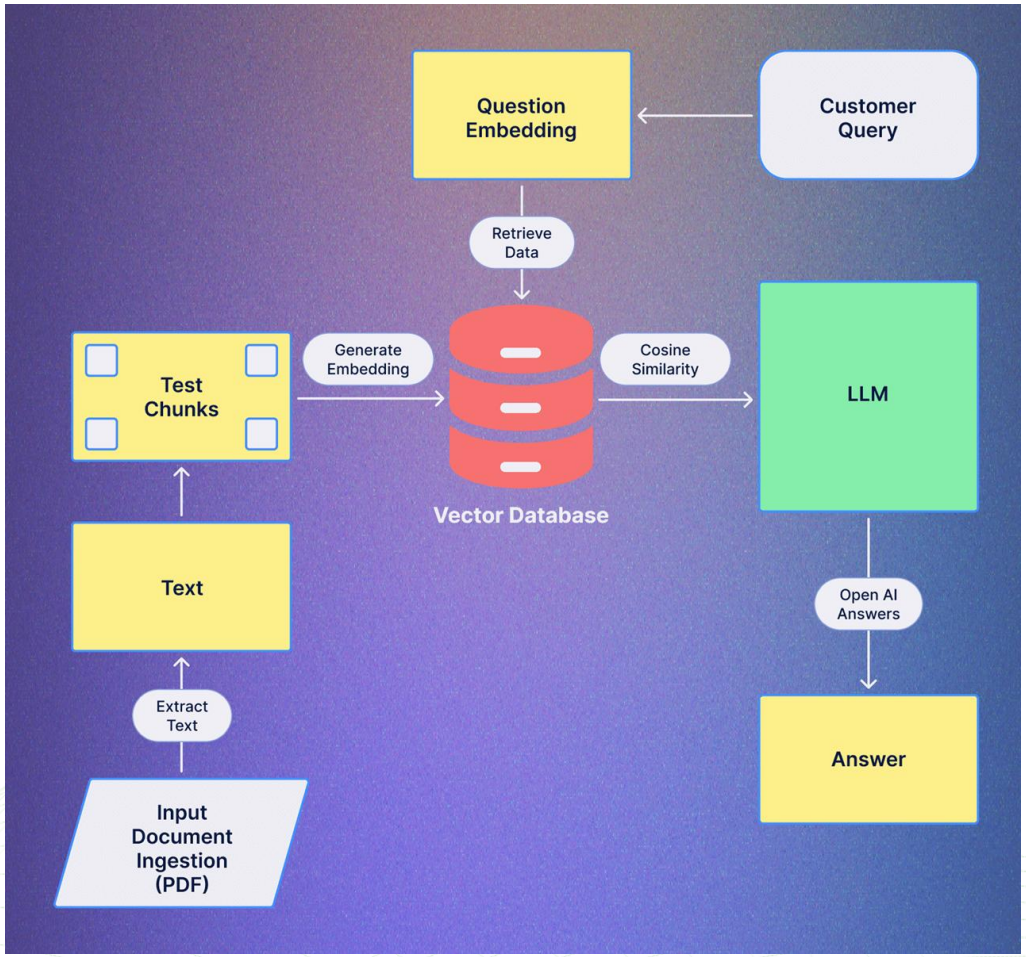
RLHF 的應用超越 LLM 界限，拓展到其他類型的生成式 AI。以下是一些範例：

- RLHF 可用於 AI 圖像生成：例如判斷藝術作品的逼真度、技巧或基調。
- 音樂生成方面，RLHF 可協助創作符合特定活動氣氛和配樂的音樂
- RLHF 可用於語音助理，引導聲音聽起來更友善、表現出好奇且值得信賴。



https://www.youtube.com/watch?v=T_X4XFwKX8k

Reinforcement Learning from Human Feedback (RLHF) Explained



LLM

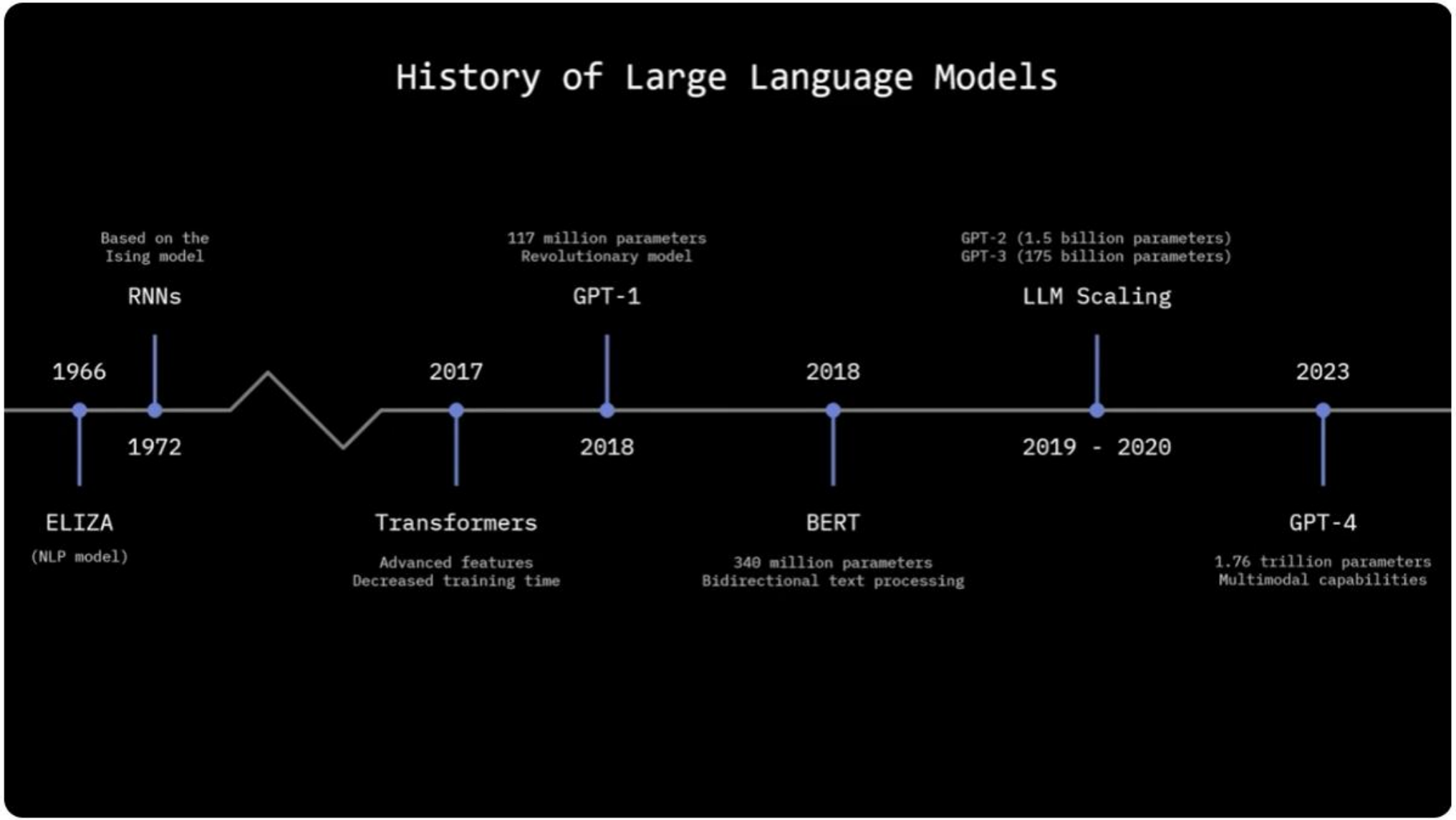
(Large Language Model)

&

RAG

(Retrieval-Augmented Generation)

Jonathan Chen


History of Large Language Models
1966
ELIZA
(NLP model)
1972
RNNs
Based on the
Ising model
2017
Transformers
Advanced features
Decreased training time
2018
GPT-1
117 million parameters
Revolutionary model
2018
BERT
340 million parameters
Bidirectional text processing
2019 - 2020
LLM Scaling
GPT-2 (1.5 billion parameters)
GPT-3 (175 billion parameters)
2023
GPT-4
1.76 trillion parameters
Multimodal capabilities

LLM (Large Language Model) & RAG (Retrieval-Augmented Generation)

LLM 是一種關鍵人工智慧 (AI) 技術，為智慧聊天機器人和其他自然語言處理 (NLP) 應用程式提供支援。目標是建立能夠透過交叉參照權威知識來源，在各種情境下回答使用者問題的機器人。不幸的是，LLM 技術的性質導致 LLM 回應中引入不可預測性。此外，LLM 訓練資料是靜態的，並為其所擁有的知識引入了截止日期。

LLM 的已知挑戰包括：

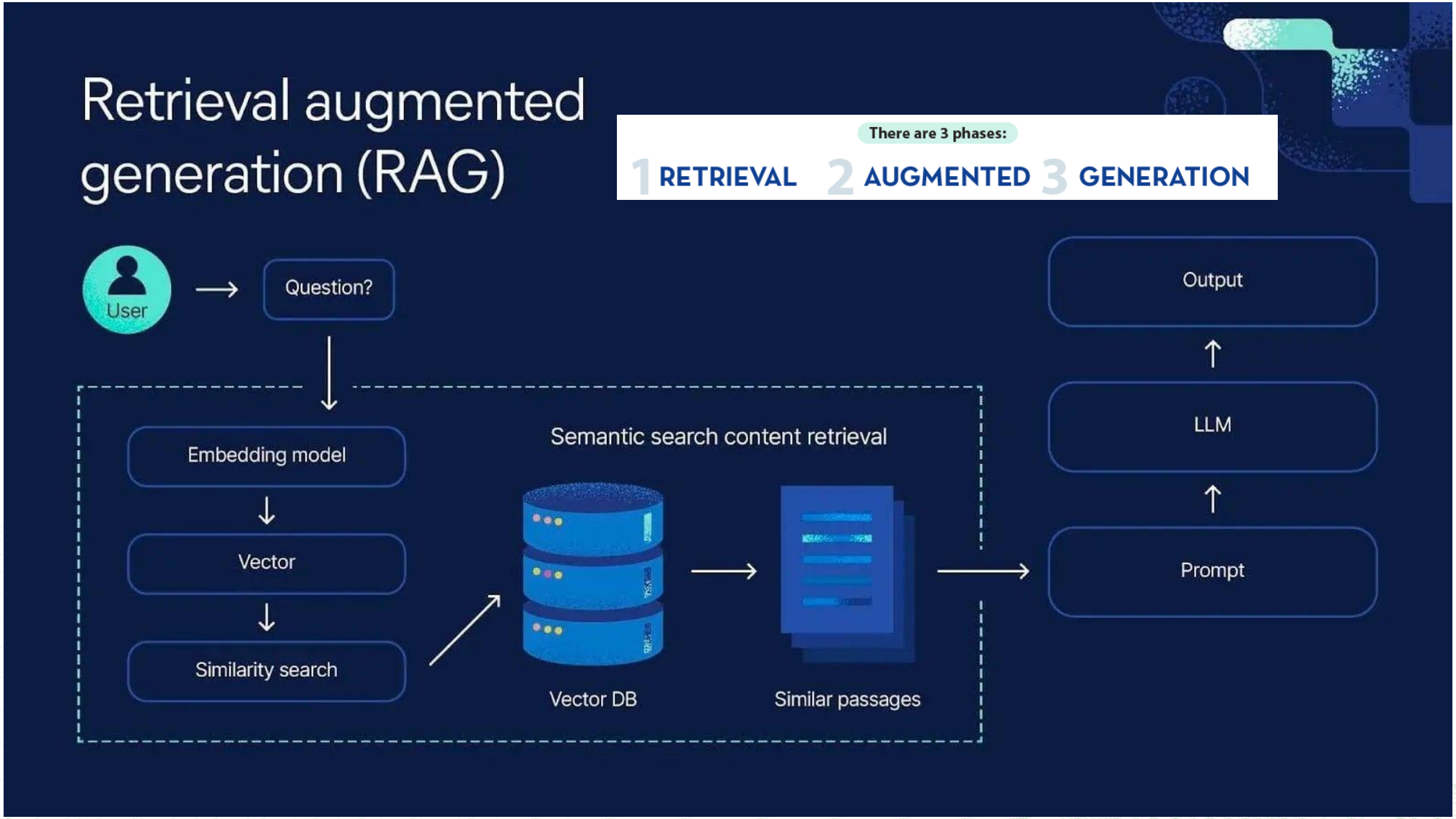
- 當沒有答案時顯示虛假資訊。
- 當使用者期待特定的最新回應時，顯示過期或一般資訊。
- 從非授權來源建立回應。
- 由於術語混亂而建立不正確的回應，其中不同的訓練來源使用相同的術語來談論不同的事情。

RAG 是解決這些挑戰的其中一種方法。它重新導向 LLM，從權威、預先確定的知識來源中擷取相關資訊。組織對產生的文字輸出有更大的控制權，並且使用者深入了解 LLM 如何產生回應。

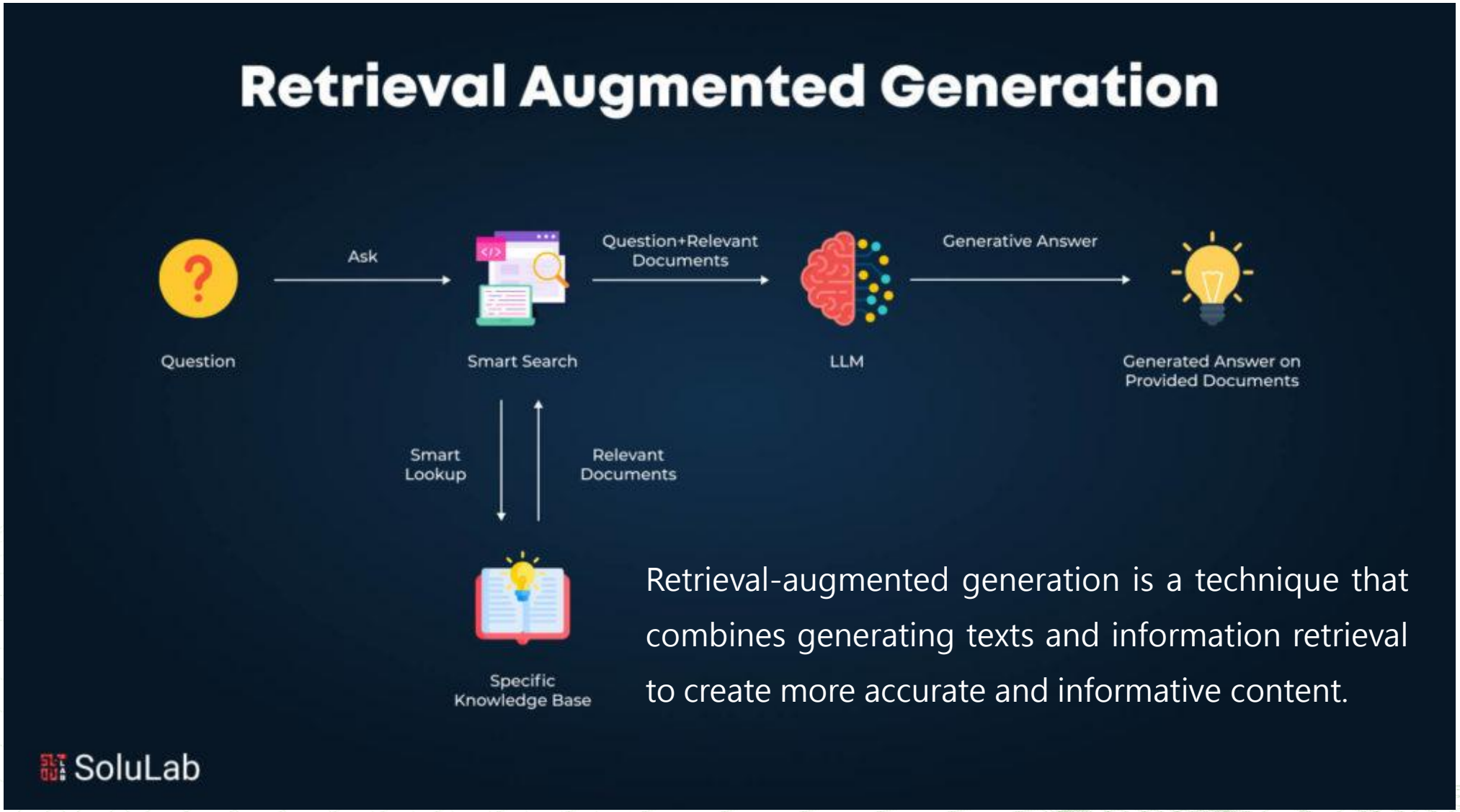
https://www.youtube.com/watch?v=T-D1OfcDW1M

What is Retrieval-Augmented Generation (RAG)?

RAG (Retrieval-Augmented Generation 檢索增強生成)



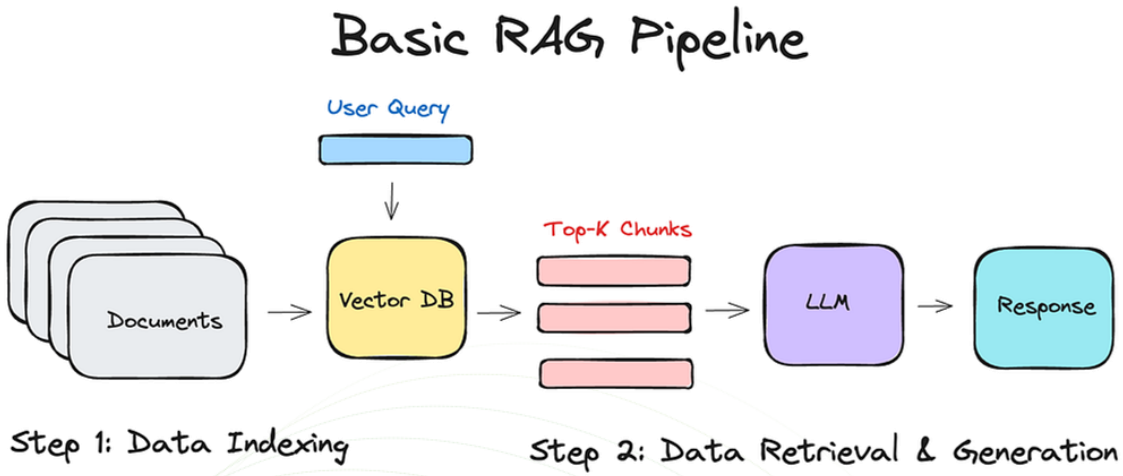
RAG (Retrieval-Augmented Generation 檢索增強生成)



Retrieval-augmented generation is a technique that combines generating texts and information retrieval to create more accurate and informative content.

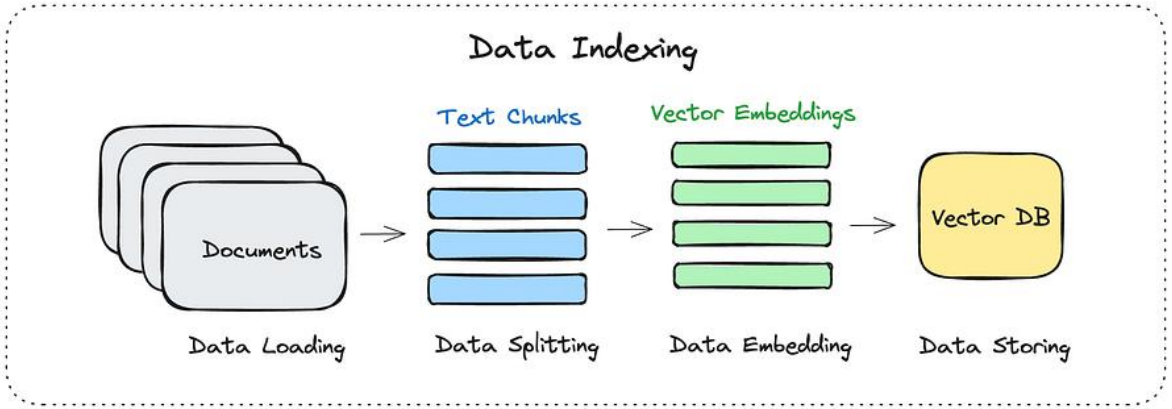
RAG (Retrieval-Augmented Generation 檢索增強生成)

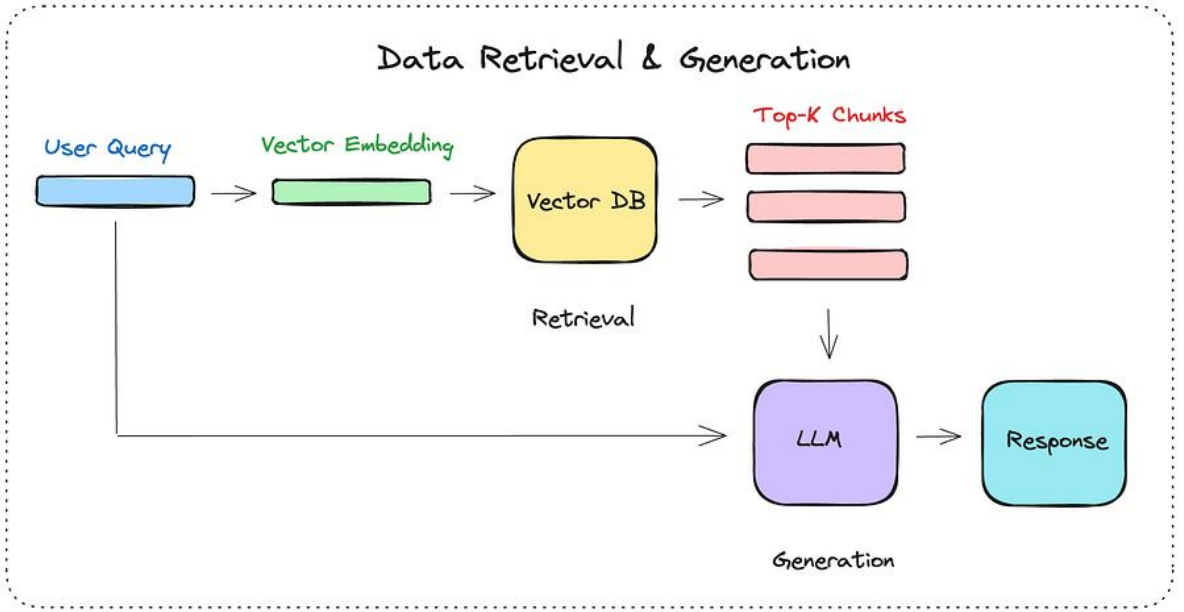
User Query → [Retriever] → Top-k Relevant Documents → [LLM Generator] → Final Answer



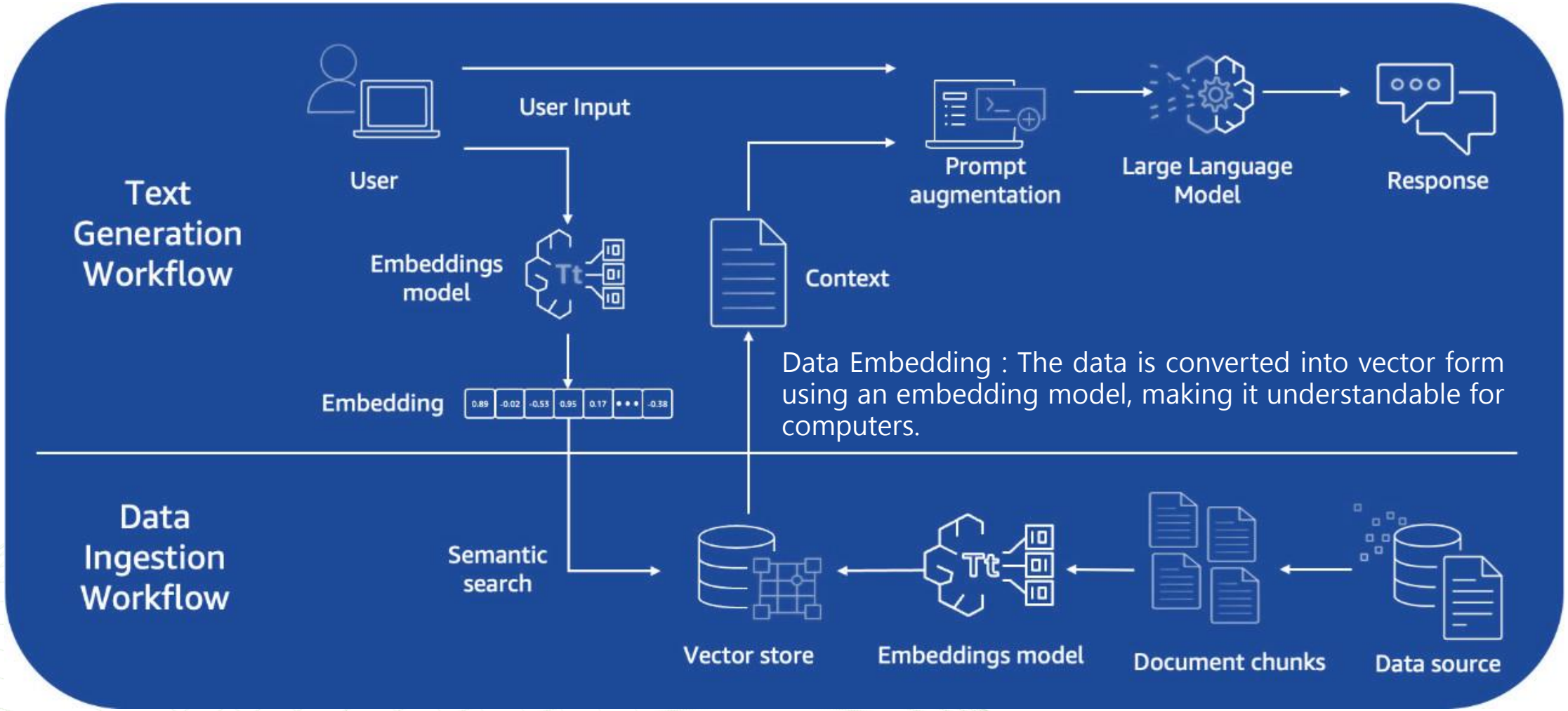
Basic RAG Pipeline consists of 2 parts:

Data Indexing and Data Retrieval & Generation



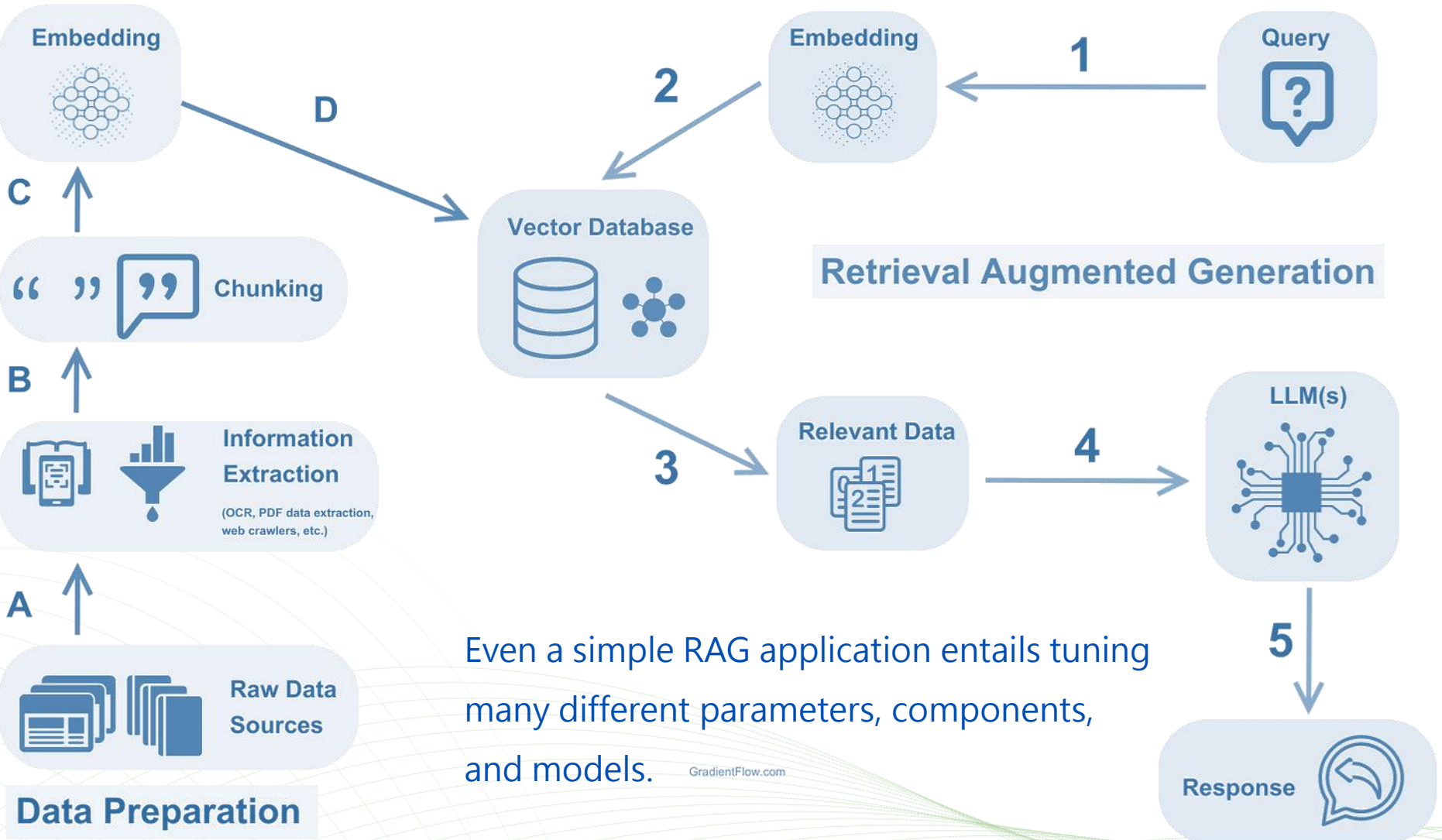


RAG (Retrieval-Augmented Generation 檢索增強生成)



- An extensive compilation of texts, datasets, documents or other informational sources needs to be provided.

RAG (Retrieval-Augmented Generation 檢索增強生成)



Even a simple RAG application entails tuning many different parameters, components, and models.

GradientFlow.com

RAG (Retrieval-Augmented Generation 檢索增強生成)

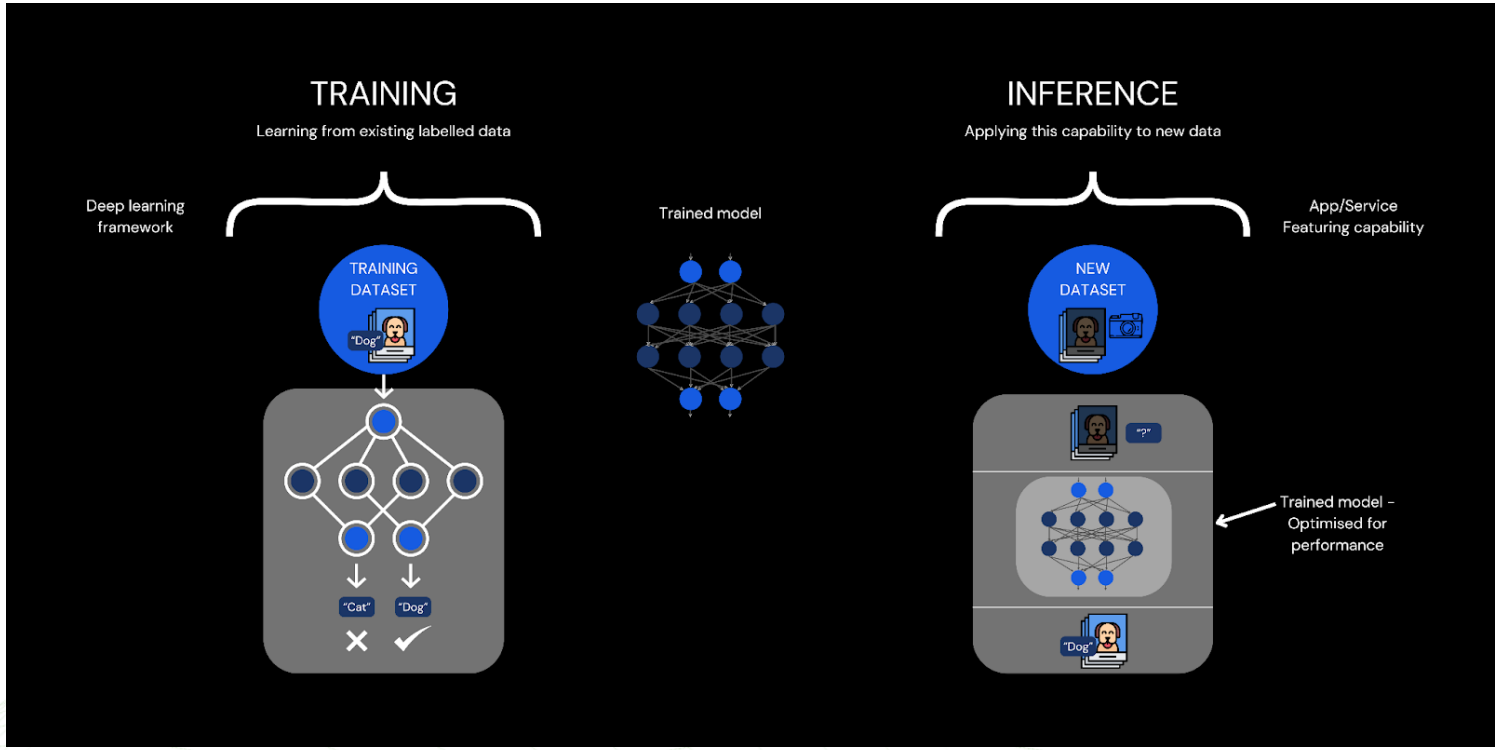
- RAG 是一種較新的人工智慧技術，透過讓大型語言模型 (LLM) 在不經過再訓練的情況下利用更多資料資源，改善生成式 AI 的品質。
- RAG 模型會根據組織自己的資料建立知識儲存庫，並且可以持續更新儲存庫，以協助生成式 AI 提供及時、符合情境的答案。
- 使用自然語言處理的聊天機器人和其他對話式系統，可以從 RAG 和生成式 AI 獲益良多。
- 導入 RAG 需要向量資料庫等技術，以快速對新資料進行編碼，然後搜尋該資料以輸入 LLM 中。

RAG 技術可用於改善生成式 AI 系統對提示的回應品質，超越 LLM 單獨提供的程度。

優點如下：

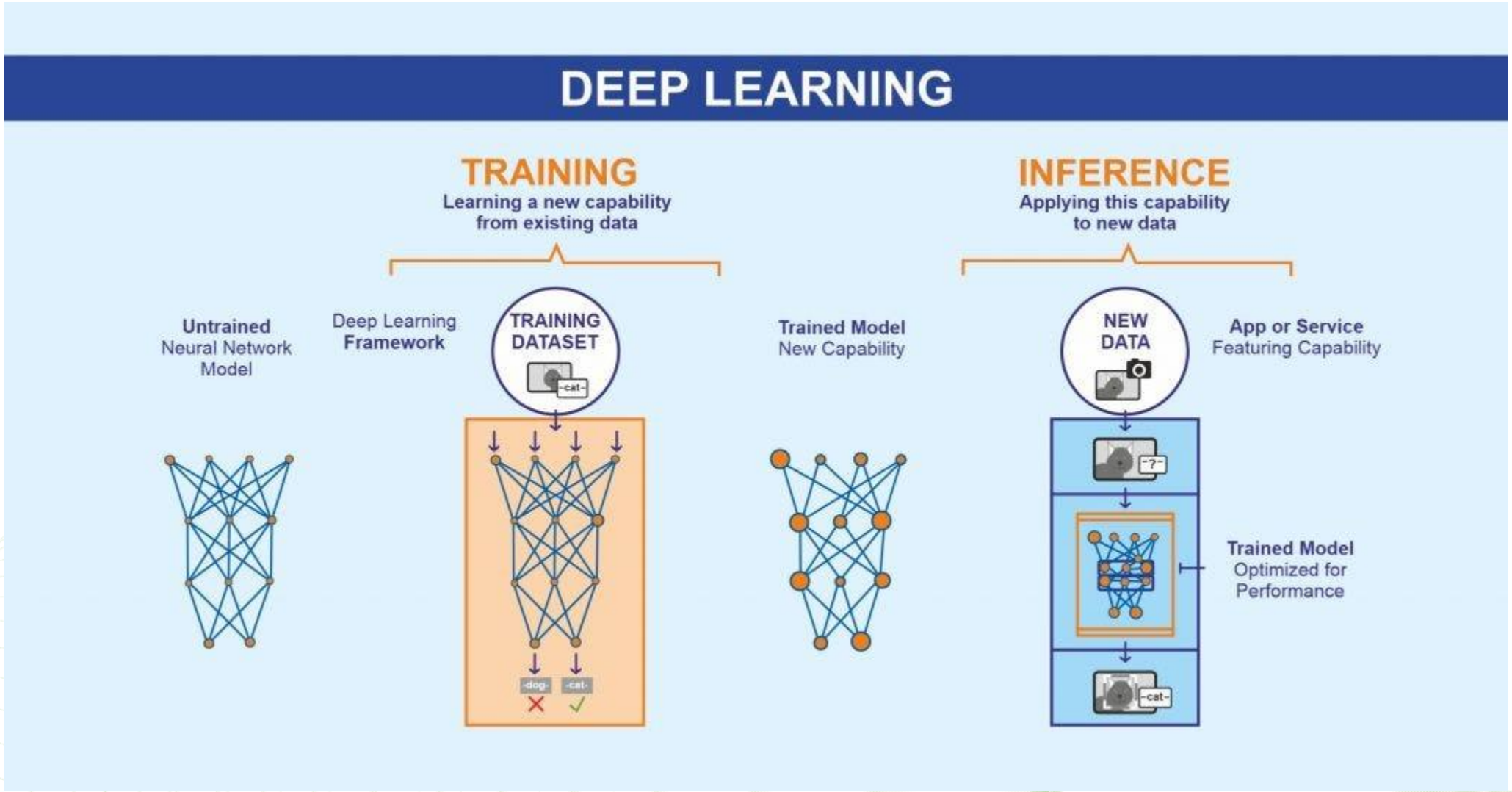
- RAG 能夠存取的資訊，可能比用來訓練 LLM 的資料還要新。
- RAG 知識儲存庫中的資料可以持續更新，而不會產生高昂成本。
- RAG 知識儲存庫中的資料，可能比一般 LLM 中的資料更符合情境。
- 可以找到 RAG 向量資料庫中的資訊來源。而且因為知道資料來源，所以可以更正或刪除 RAG 中不正確的資訊。

簡而言之，RAG 可協助 LLM 提供更好的答案。

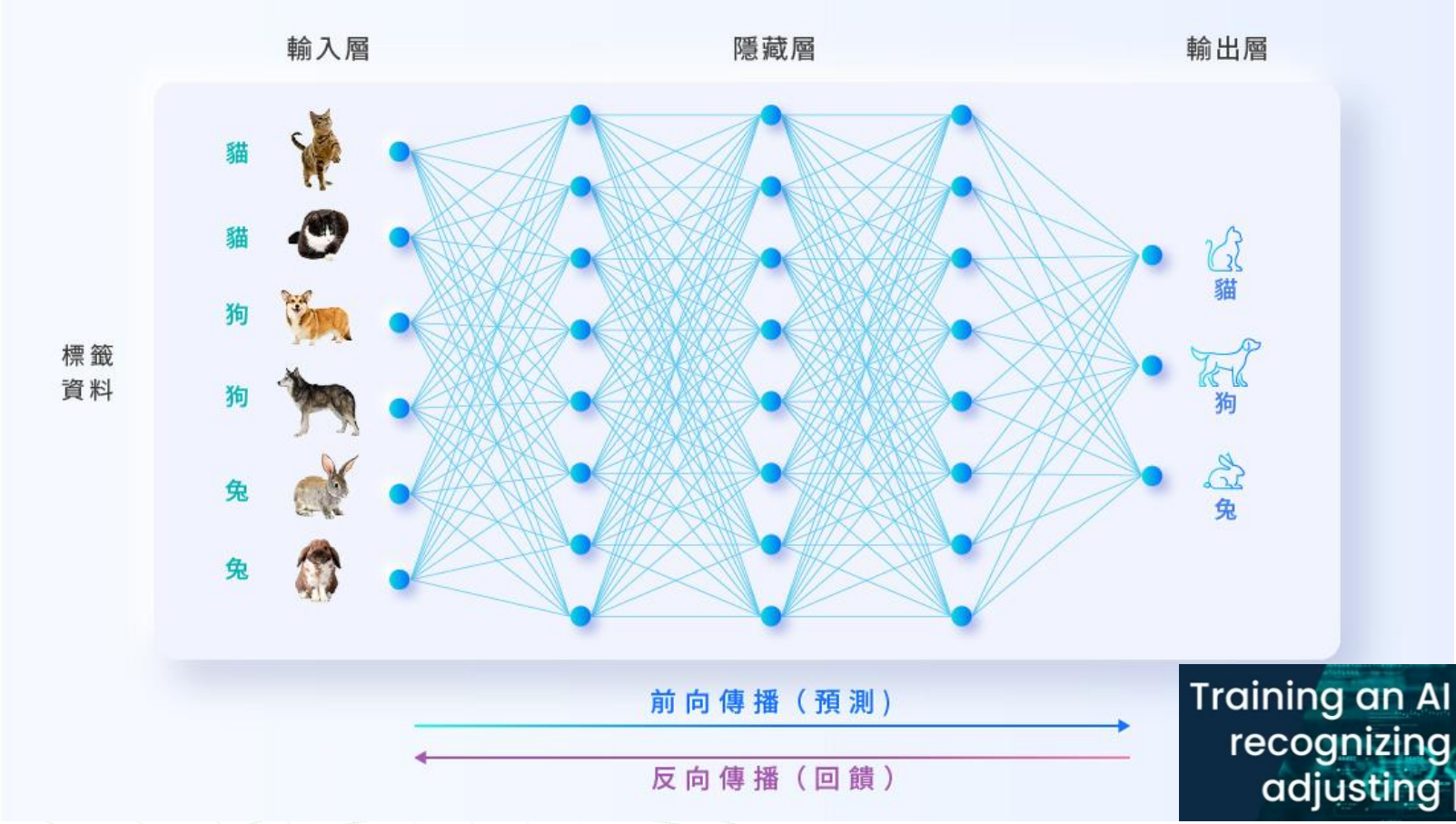


AI Inference

AI Training vs. AI Inference

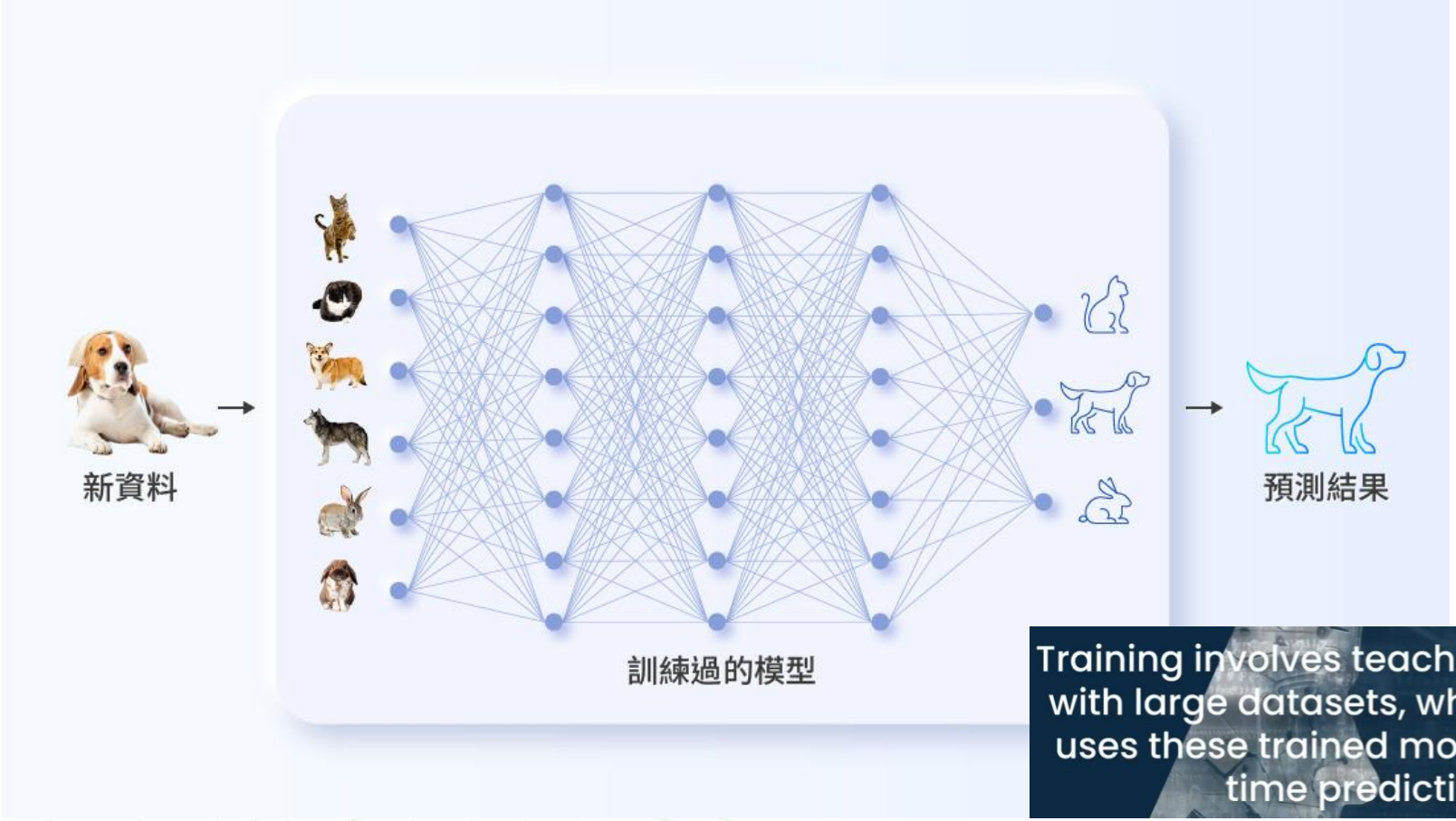


AI Training



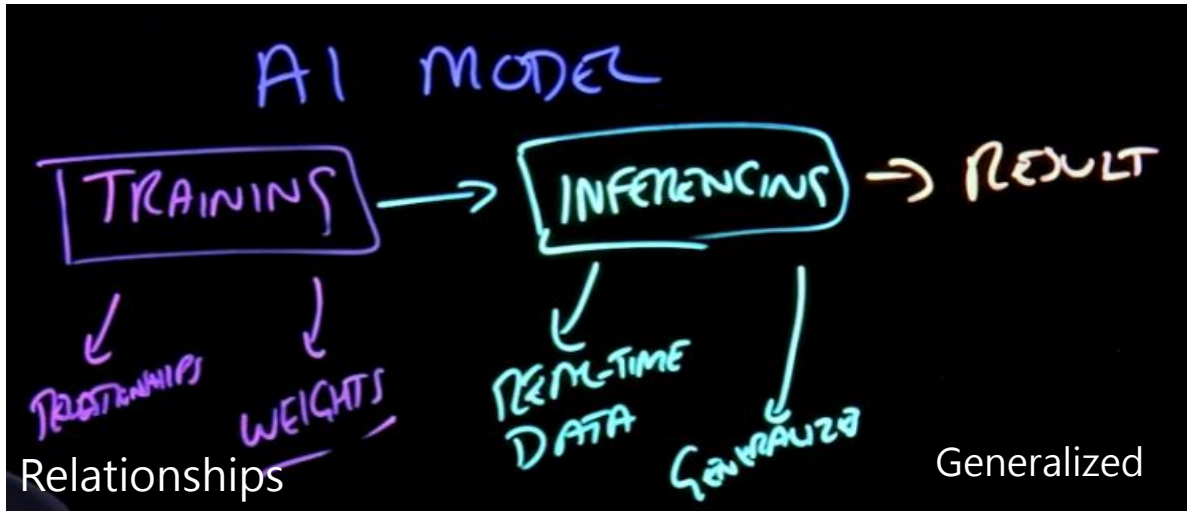
當輸入值(input)從一層節點傳輸到下一層節點，人工智慧會依據輸出值(output)的正確性，給予參數一個正數或負數的權重(weight)，等於是在幫決策過程打分數。透過「前向傳播」（或稱預測）和「反向傳播」（或稱回饋）的反覆循環，參數的權重變得非常精確，AI幾乎每次都會挑選正確的輸出值。

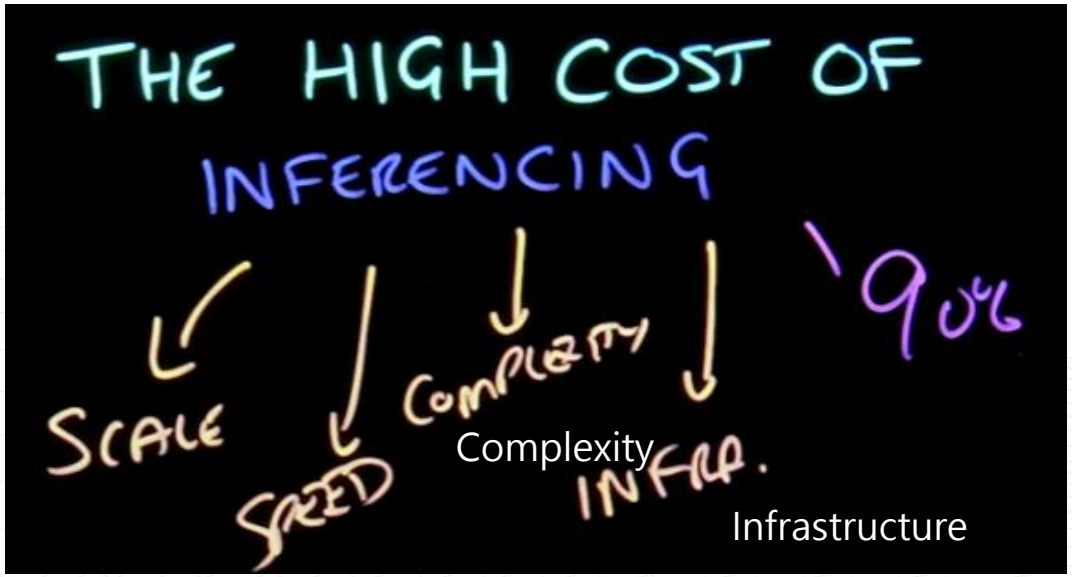
What is AI Inference?



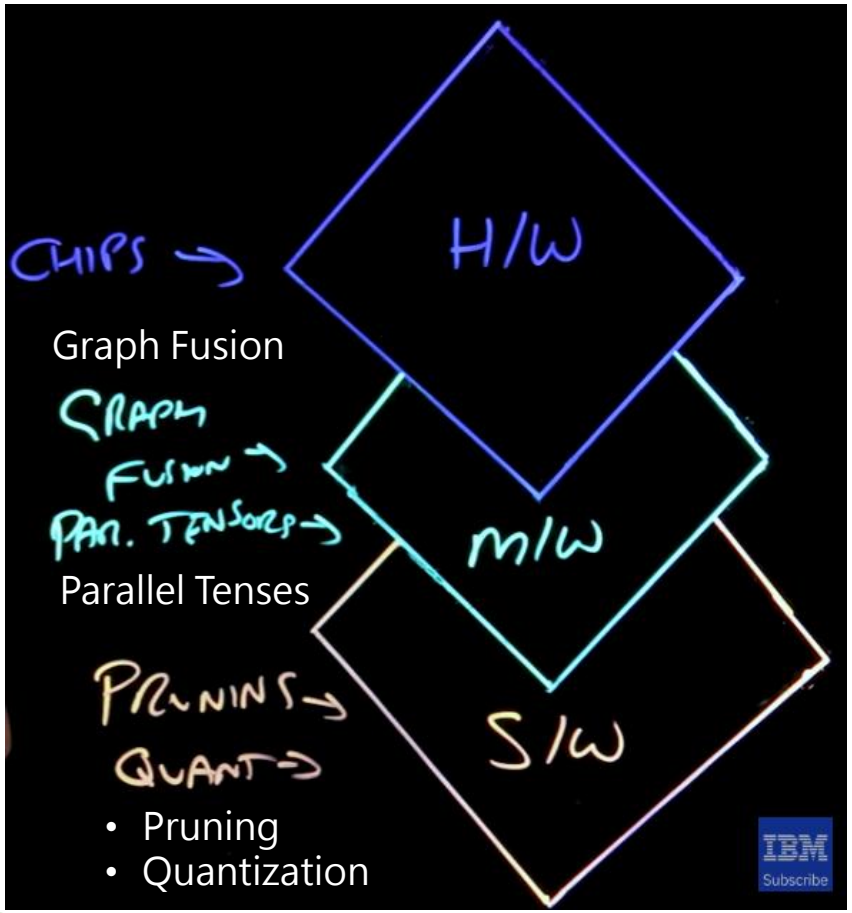
人工智慧進行「**推論**」的方式，就是拿訓練階段累積的參數比對新的輸入值，「**預測**」使用者期望接收的輸出值。AI模型也會記錄人類的反應，留意哪些答覆讓使用者感到滿意、哪些被送回要求修改，這些經驗值有助於推進下一輪的AI訓練。由此可見，人工智慧也是「從錯誤中學習」，因此生成式AI變得愈來愈進步、愈人性化。

AI「訓練」(training) 與「推論」(inference)





https://www.youtube.com/watch?v=XtT5i0ZeHHE



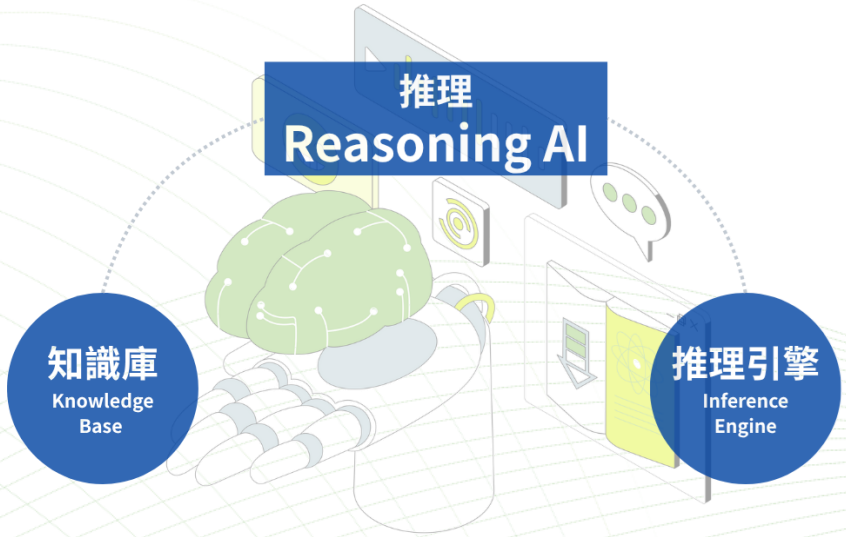
Reduce the number/precision of model weights

Jonathan Chen

https://www.gss.com.tw/e-forum/user-tell/4376-reasoning-ai-future-trends-2

在過去，許多AI模型，特別是大型語言模型（LLMs），主要依賴於「訓練時計算」（training-time computing），即根據它們已經見過的大量數據來預測下一個詞或生成內容。這種「快速思考」的方式對於簡單的任務非常有效。然而，對於複雜且高風險的問題，僅僅依賴直覺式的快速反應是遠遠不夠的。

Reasoning AI 的出現，標誌著 AI 從「快速思考」向「慢思考」的轉變。當一個模型「停下來思考」時，它不再只是重複模式或從過去的數據中提取預測。它會**權衡不同的情境，思考可能的結果，並基於邏輯做出決策**。這種深思熟慮的過程，通常被稱為「推理時計算」（inference-time computing），雖然需要更多的努力，但能帶來更具意義的結果。



Reasoning AI 的核心概念，包括「**知識庫**」和「**推理引擎**」兩個主要組成部分

- ✓ 知識庫為 AI 提供了「理解世界」的基礎資訊。
- ✓ 推理引擎是 AI 推理系統的「大腦」。

Jonathan Chen

AI「訓練」(training) 與「推論」(inference)

AI models and AI inference are fundamental concepts in machine learning and artificial intelligence. Models are trained to learn patterns and relationships in data, while inference involves applying the trained model to new data to make predictions or generate outputs.

An AI model is the result of training a machine learning algorithm on a dataset, enabling it to learn patterns and relationships within the data. AI inference involves using the trained model to make predictions or generate outputs on new, unseen data. Both components are crucial in the lifecycle of an AI system, with training enabling the model to learn and inference enabling it to apply that learning to real-world scenarios.

Model Training:

- **Purpose**: Model training aims to teach the AI model to recognize patterns and relationships within the training data.
- **Process**: During training, the model iteratively adjusts its internal parameters to minimize the difference between its predictions and the actual labels in the training data. This process involves optimizing a loss function.
- **Data**: Training requires a labeled dataset with known inputs and corresponding outputs.
- **Computational Intensity**: Model training is computationally intensive and time-consuming, often requiring specialized hardware.

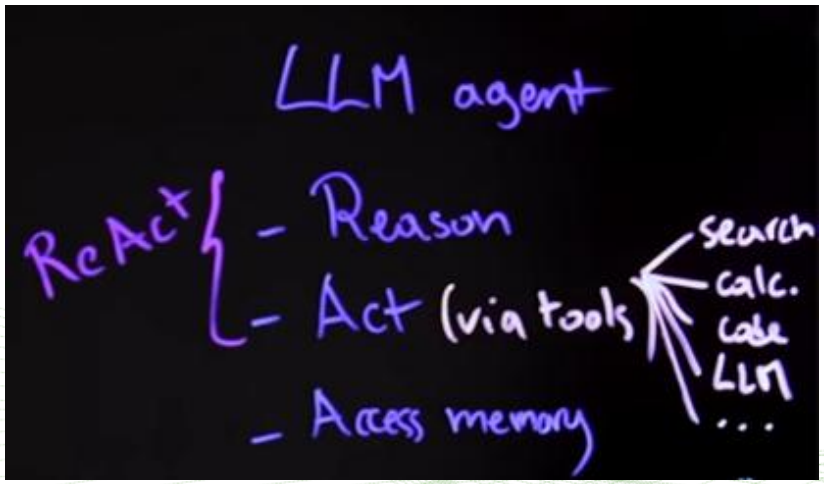
Inference:

- **Purpose**: Inference applies the trained model to new, unseen data to generate predictions or outputs.
- **Process**: Inference involves passing an input through the trained model and obtaining an output based on the learned patterns.
- **Data**: Inference can be performed on a single input at a time or in batches.
- **Computational Intensity**: Inference is generally less computationally intensive than training, making it suitable for real-time or near-real-time applications.

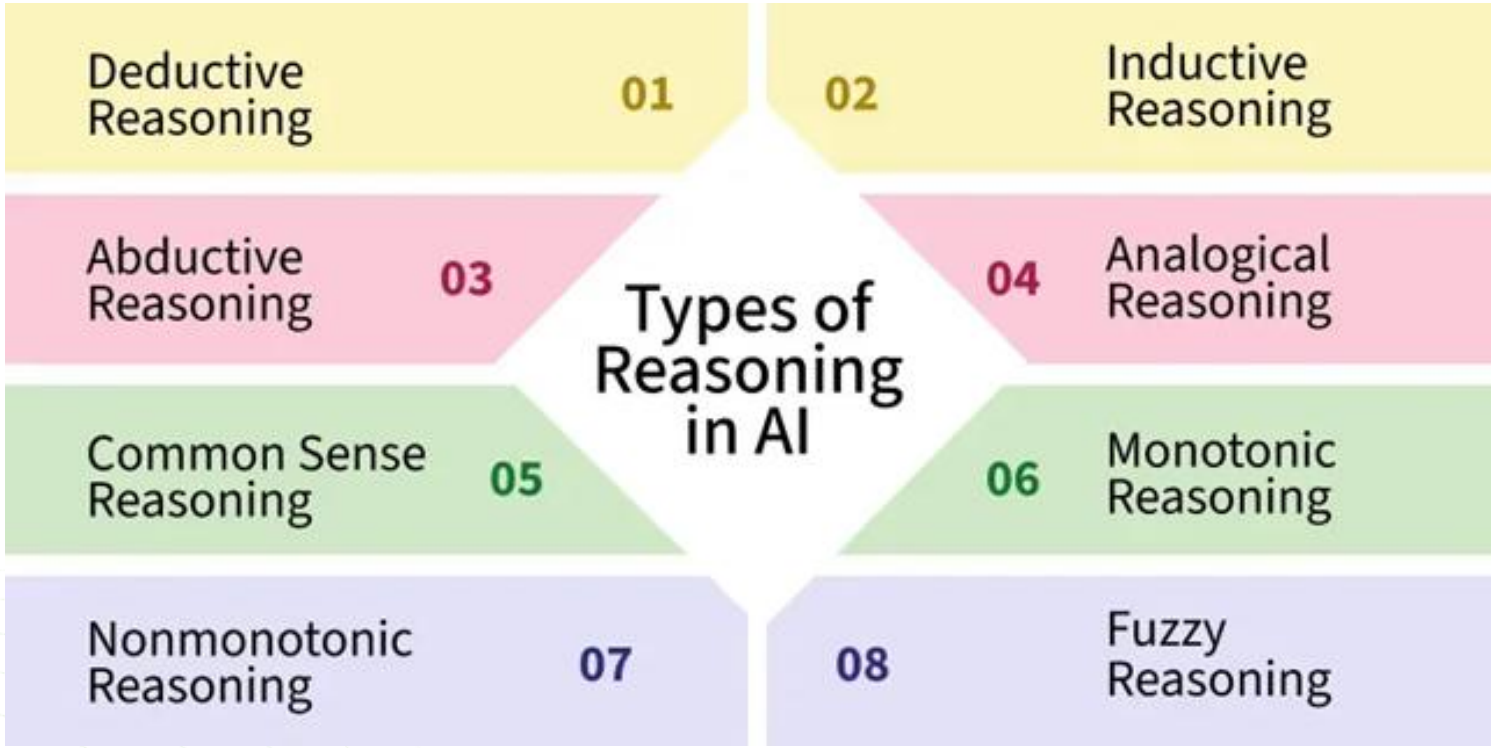
Reasoning AI 的主要類型

- 溯因推理 (Abductive Reasoning)
- 代理推理 (Agentic Reasoning) - 兩種常見的代理 AI 推理範式包括 ReAct（Reasoning and Action）和 ReWOO（Reasoning WithOut Observation）。ReAct 採用「**思考-行動-觀察**」策略逐步解決問題並迭代改進回應，而 ReWOO 則在形成回應之前進行預先規劃。

  Query ➔ Plan-Think ➔ Act ➔ Observe ➔ Answer
- 類比推理 (Analogical Reasoning)
- 常識推理 (Commonsense Reasoning) - 常識推理利用關於世界的普遍知識和關於日常生活的實用知識來做出決策。
- 演繹推理 (Deductive Reasoning) - 演繹推理是從一般事實或更廣泛的假設中得出具體結論的「由上而下」方法。
- 模糊推理 (Fuzzy Reasoning) - 處理不確定性和不精確性，允許真理具有程度，而不是絕對的真或假二元結果。
- 歸納推理 (Inductive Reasoning)
- 神經符號推理 (Neuro-symbolic Reasoning)
- 機率推理 (Probabilistic Reasoning)
- 空間推理 (Spatial Reasoning)
- 時間推理 (Temporal Reasoning)
- 單調推理 (Monotonic Reasoning)
- 非單調推理 (Nonmonotonic Reasoning)



Types of Reasoning in Artificial Intelligence



演繹推理
(Deductive Reasoning)

歸納推理
(Inductive Reasoning)

溯因推理
(Abductive Reasoning)

類比推理
(Analogical Reasoning)

常識推理
(Commonsense Reasoning)

單調推理
(monotonic Reasoning)

非單調推理
(Nonmonotonic Reasoning)

模糊推理
(Fuzzy Reasoning)

In Artificial Intelligence, reasoning is the process by which machines simulate human-like decision-making, problem-solving and predictions. Just as humans use logic to navigate the world, AI systems rely on different types of reasoning to draw conclusions, interpret data and adapt to new situations.

Jonathan Chen

Reasoning AI 應用場景

自動駕駛與
機器人技術

智慧製造與
供應鏈管理

行銷推廣與
情感分析

金融服務與
詐欺檢測

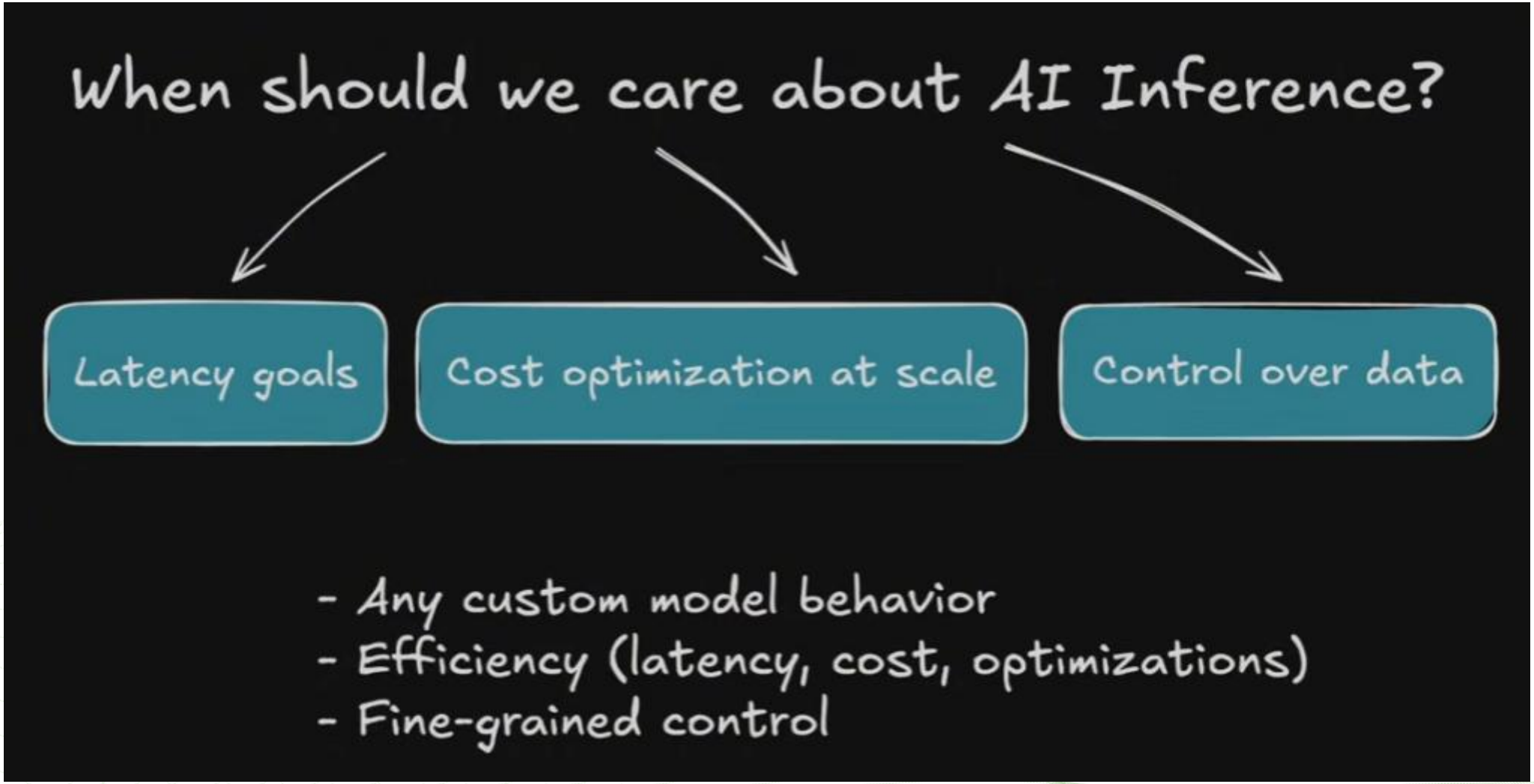
Reasoning AI 在不同應用場景中的實際應用

Reasoning AI 賦予了機器類似人類的「**思考**」能力，使其能夠理解複雜問題、評估不同情境並得出有意義的結論。從醫療診斷到自動駕駛，從自然語言處理到金融詐欺檢測，Reasoning AI 的應用潛力無限，預示著一個更加智能、高效的未來。

Jonathan Chen

AI Models That Reason - Leading models include:

- **NVIDIA Cosmos Reason:** Open, customizable, 7-billion-parameter reasoning VLM for physical AI and robotics — lets robots and vision AI agents reason like humans, using prior knowledge, physics understanding and common sense to understand and act in the real world.
- **NVIDIA Llama Nemotron Models**: Built on Meta' s Llama models, the Llama Nemotron family of models includes Nano, Super, and Ultra variants, tailored for edge devices and data centers. It features toggled reasoning capabilities and excels in multistep tasks like tool utilization, math, and instruction adherence.
- **DeepSeek-R1**: Known for its affordability and robust performance, DeepSeek-R1 excels in mathematical reasoning, coding, and scientific problem-solving. It employs reinforcement learning and multi-stage training, allowing users to observe its step-by-step thought process for greater trust and explainability.
- **OpenAI o1 and o3-mini**: These models, available on ChatGPT, focus on simulated reasoning, enabling them to pause and reflect on their internal thought processes before responding. OpenAI o3-mini improves upon o1 by offering faster responses, reduced costs, and enhanced accuracy in STEM domains.


When should we care about AI Inference?
Latency goals
Cost optimization at scale
Control over data
- Any custom model behavior
- Efficiency (latency, cost, optimizations)
- Fine-grained control

The difference between AI model Training and AI Inference

- Training an AI involves teaching it to recognize patterns using large amounts of labeled data.
- During Training, the AI adjusts its internal settings based on examples, which requires significant computer resources.
- Training typically involves showing the AI millions of images or data points to help it learn distinctions, such as between cats and dogs.

- The goal of training is to give AI the knowledge needed to perform specific tasks later on.
- Inference is the phase where the AI applies what it has learned to new, unseen data.
- Inference happens when you ask a voice assistant a question or receive product recommendations online.

- Inference is faster and uses less computing power compared to training because it relies on existing knowledge.
- Training is usually done offline with large datasets, while inference runs continuously in real-time applications.

- You might periodically update or fine-tune a pre-trained model to improve its performance for specific tasks.
- Building effective AI system involves understanding when to train models and when to perform inference for optimal results.
- Once trained, models are deployed to perform inference quickly, making applications more responsive.
- Proper management of training and inference processes helps create resource-efficient and effective AI-powered features.

Jonathan Chen