從零開始認識生成式AI與圖書資訊領域的應用

黃乾綱, Chien-Kang Huang
ckhuang@ntu.edu.tw
臺灣大學工程科學及海洋工程學系

Outline

- 機器學習 Machine Learning 與深度學習 Deep Learning
- 人工智慧 AI 概述
- 機器學習的機制
- 大語言模型的基本原理
- AI 基礎模型(Foundation Models)
- AI 工具

機器學習與深度學習

機器學習 Machine Learning

- **機器學習**一詞是由Arthur Samuel於 1959 年創造的，他是IBM員工，也是電腦遊戲和人工智慧領域的先驅。
- **機器學習**是人工智慧的一個研究領域，涉及統計演算法的開發和研究，這些演算法可以從數據中學習並泛化到未見過的數據，從而在沒有明確指令的情況下執行任務。深度學習領域的進展使神經網路在性能上超越了許多先前的方法。
- 統計和數學最佳化（數學規劃）方法構成了機器學習的基礎。資料探勘是一個相關的研究領域，重點是透過無監督學習進行探索性資料分析(EDA)

https://en.wikipedia.org/wiki/Machine_learning

機器學習

- Tom M. Mitchell對**機器學習**領域研究的演算法提供了一個被廣泛引用的、更正式的定義
  - "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E.“
- 現代機器學習有兩個目標：
  - 根據已開發的模型對資料進行**分類**
  - 根據這些模型對未來的結果進行**預測**
  - 一種專門用於分類資料的假設演算法可以使用痣的電腦視覺以及監督學習來訓練它對癌性痣進行分類。股票交易的機器學習演算法可以告知交易者未來的潛在預測。

https://en.wikipedia.org/wiki/Machine_learning

機器學習和人工智慧的關係

- 機器學習源自於對人工智慧(AI) 的追求。在人工智慧作為一門學科的早期，一些研究人員對讓機器從數據中學習感興趣。他們嘗試用各種符號方法以及當時所謂的「神經網路」來解決這個問題。
- 對邏輯、基於知識的方法的日益重視導致了人工智慧和機器學習之間的裂痕。機率系統受到資料獲取和表示的理論和實際問題的困擾。
- 1980 年，專家系統已經主宰人工智慧，統計學不再受青睞。統計研究領域已經被淘汰。神經網路研究大約在同一時間被人工智慧和電腦科學放棄。
- John Hopfield, David Rumelhart, and Geoffrey Hinton 等其他學科的研究人員也在AI/CS 領域之外延續了這條路線，稱為「聯結主義」。他們的主要成功是在 20 世紀 80 年代中期，重新發明了反向傳播 (backpropagation)。
  - 2024 Nobel Prize in Physics

https://en.wikipedia.org/wiki/Machine_learning

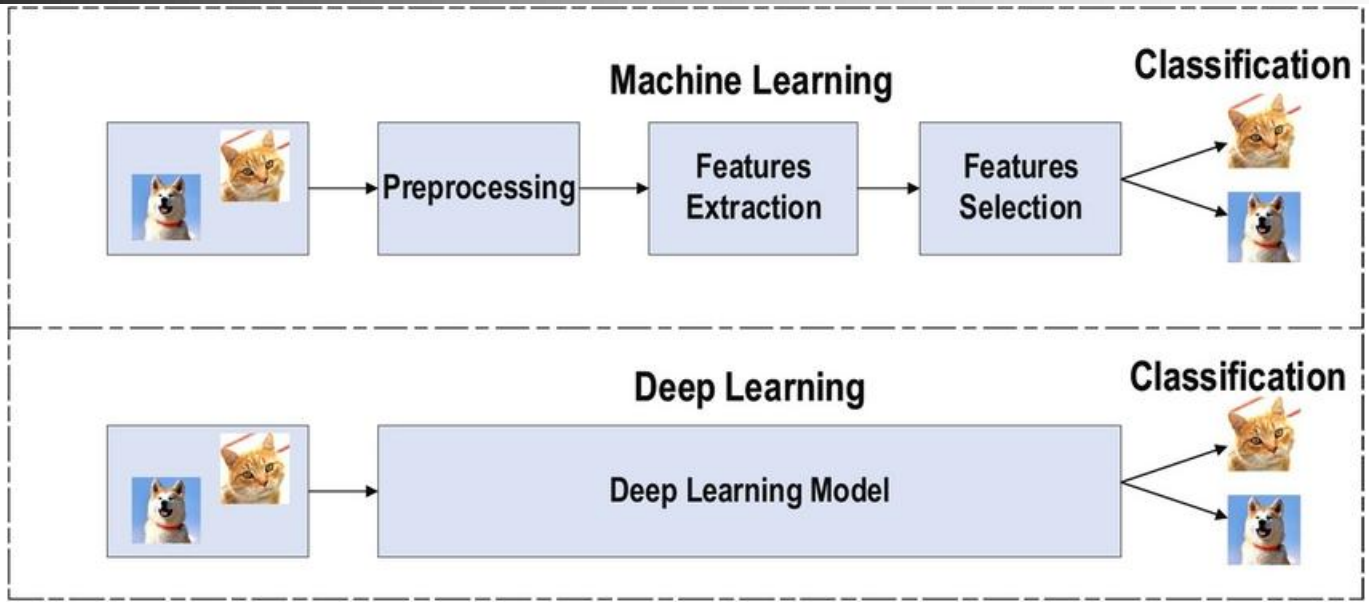
深度學習 Deep Learning

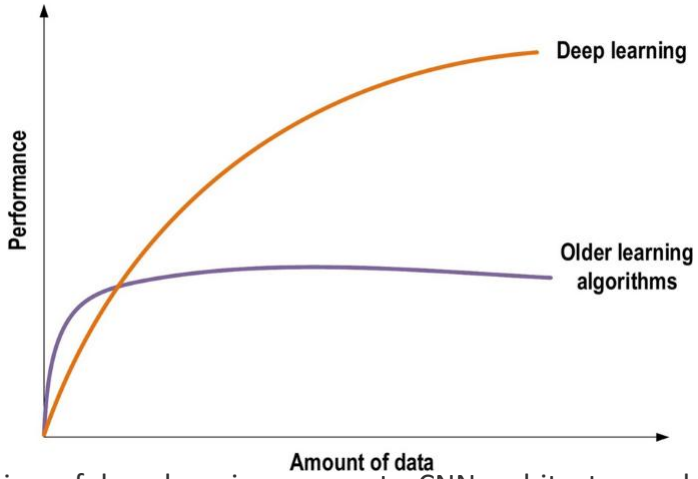
- **深度學習**是機器學習的子集，專注於利用神經網路執行分類、回歸和表示學習等任務。該領域的靈感來自於生物神經科學，其核心是將人工神經元堆疊成層並「訓練」它們來處理數據。形容詞「深」是指在網路中使用多層（從三層到幾百或幾千層）。
- 深度學習一詞於 1986 年由Rina Dechter引入機器學習界，13 並於 2000 年由 Igor Aizenberg 及其同事在布爾閾值神經元(Boolean threshold neurons)的背景下引入人工神經網絡。

常見的深度學習網路架構

- 全連接網路 FCN, fully connected networks
- 深度信念網路 DBN, deep belief networks
- 循環神經網路 RNN, recurrent neural networks
- 卷積神經網路 CNN, convolutional neural networks
- 生成對抗網路 GAN, generative adversarial networks
- 變換器 transformers
- 神經輻射場 NeRF, neural radiance fields
  - 使用數張局部 2D 影像重建複雜的 3D 場景

機器學習 和 深度學習 的差異 (1)





Alzubaidi, L., Zhang, J., Humaidi, A.J. *et al.* Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *J Big Data* **8**, 53 (2021). https://doi.org/10.1186/s40537-021-00444-8

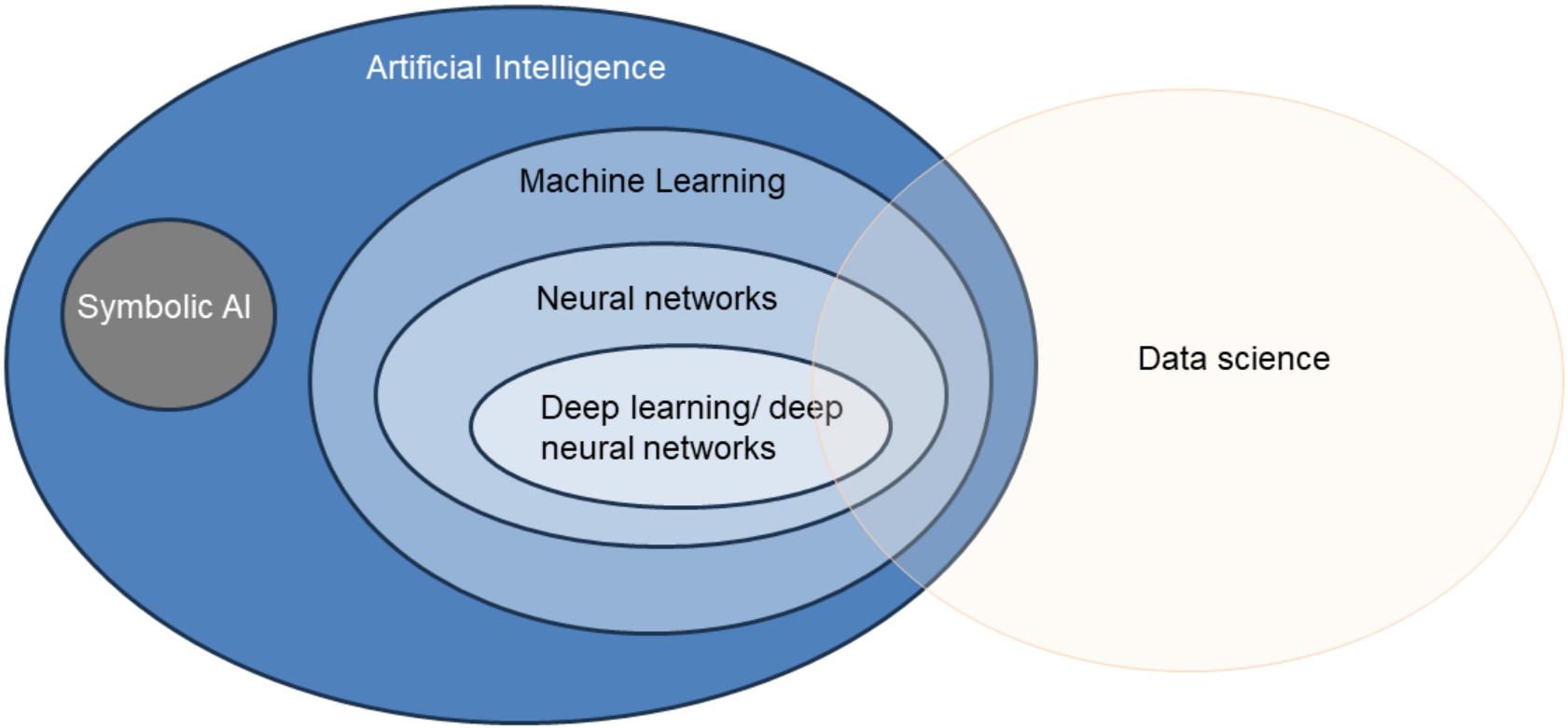
人工智慧概述

人工智慧 Artificial Intelligence

- 人工智慧，從廣泛的意義上來說，是機器（電腦系統）所表現出的智慧。它是電腦科學的一個研究領域，開發和研究方法和軟體，使機器能夠感知其環境，並利用學習和智慧採取行動，最大限度地提高實現既定目標的機會。
- 人工智慧的一些引人注目的應用包括
  - 高級網路搜尋引擎（例如， Google搜尋）
  - 推薦系統（ YouTube 、 Amazon和Netflix使用）
  - 透過人類語音進行互動（例如Google Assistant 、 Siri和Alexa ）
  - 自動駕駛汽車（例如Waymo ）
  - 生成和創意工具（例如ChatGPT和AI art ）
  - 以及戰略遊戲（例如國際象棋和圍棋）中超人的玩法和分析。
- 然而，許多人工智慧應用程式並不被視為人工智慧：「許多尖端人工智慧已經滲透到一般應用程式中，通常不會被稱為人工智慧，因為一旦某些東西變得足夠有用且足夠普遍，它就不再被貼上人工智慧的標籤。
- 在 2020 年代初人工智慧熱潮期間，許多公司使用該術語作為行銷流行語，經常，即使他們「實際上並沒有以物質方式使用人工智慧」。

https://en.wikipedia.org/wiki/Artificial_intelligence

AI 和 Data Science 的領域範疇



https://akfpartners.com/growth-blog/ceos-guide-to-artificial-intelligence-ai-and-machine-learning-ml

AI的研究目標和使用技術

- AI研究的各個子領域都圍繞著特定目標和特定工具的使用。人工智慧研究的傳統目標包括：
  推理、知識表示、規劃、學習、自然語言處理、感知和對機器人技術的支援。
- 為了實現這些目標，AI研究人員採用並整合了多種技術，包括：
  搜尋和數學優化、形式邏輯、人工神經網路以及基於統計學、運籌學和經濟學的方法。也借鑒了心理學、語言學、哲學、神經科學等學科的知識。

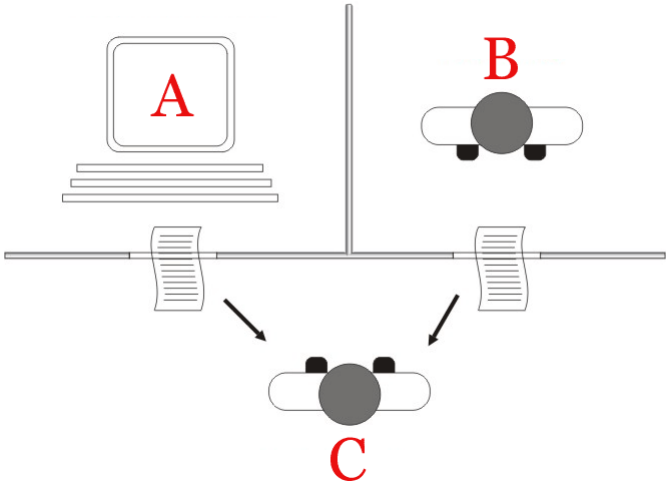
https://en.wikipedia.org/wiki/Artificial_intelligence

AI 的定義

- 1950, Alan Turing: “I propose to consider the question ‘can machines think’?”
- Turing 將問題從「機器是否思考」改為「機器是否可能展現出智能行為」。他設計了 Turing Test，用以衡量機器模擬人類對話的能力。由於我們只能觀察機器的行為，機器是否「真正」在思考或是否具有「心靈」並不重要。

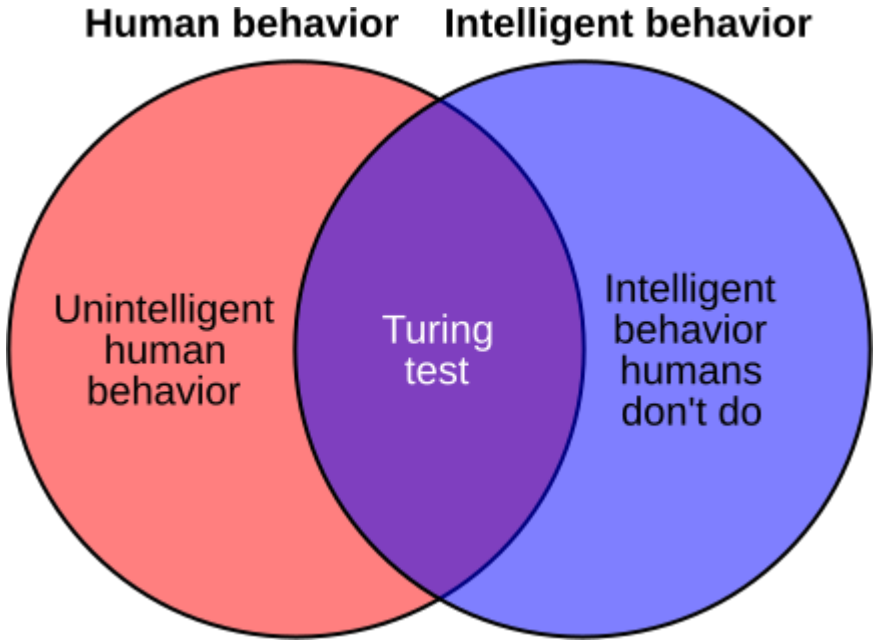
圖靈測試 Turing Test

- 英國電腦科學家艾倫·圖靈於1950年提出的思想實驗，稱為「模仿遊戲」（imitation game），



圖靈測試的「標準解釋」是指玩家 C（審查者）的任務是嘗試判斷玩家 A 和 B 中誰是電腦，誰是人類。審查者只能依據書面問題的回答來做出判斷。

https://en.wikipedia.org/wiki/Turing_test



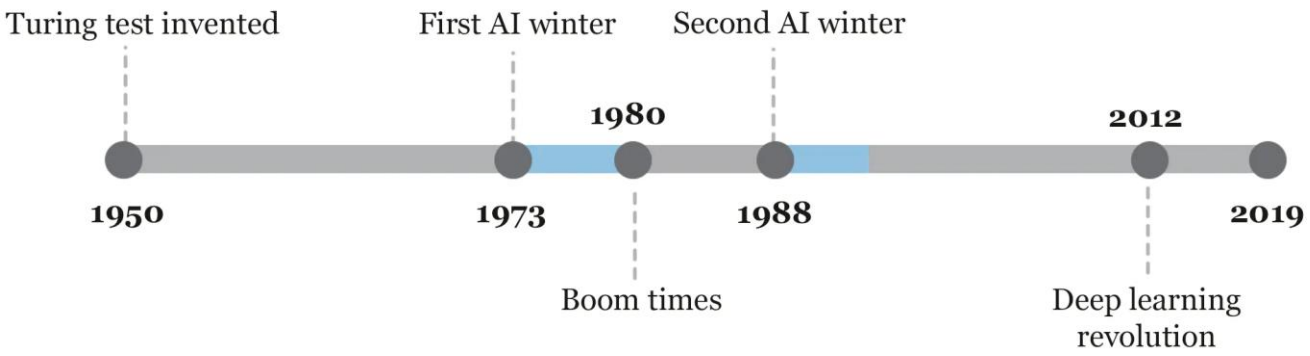
圖靈測試可以提供一些智能的證據，但它會懲罰非人類的智能行為。

https://en.wikipedia.org/wiki/Artificial_intelligence

AI 寒冬之後的熱潮

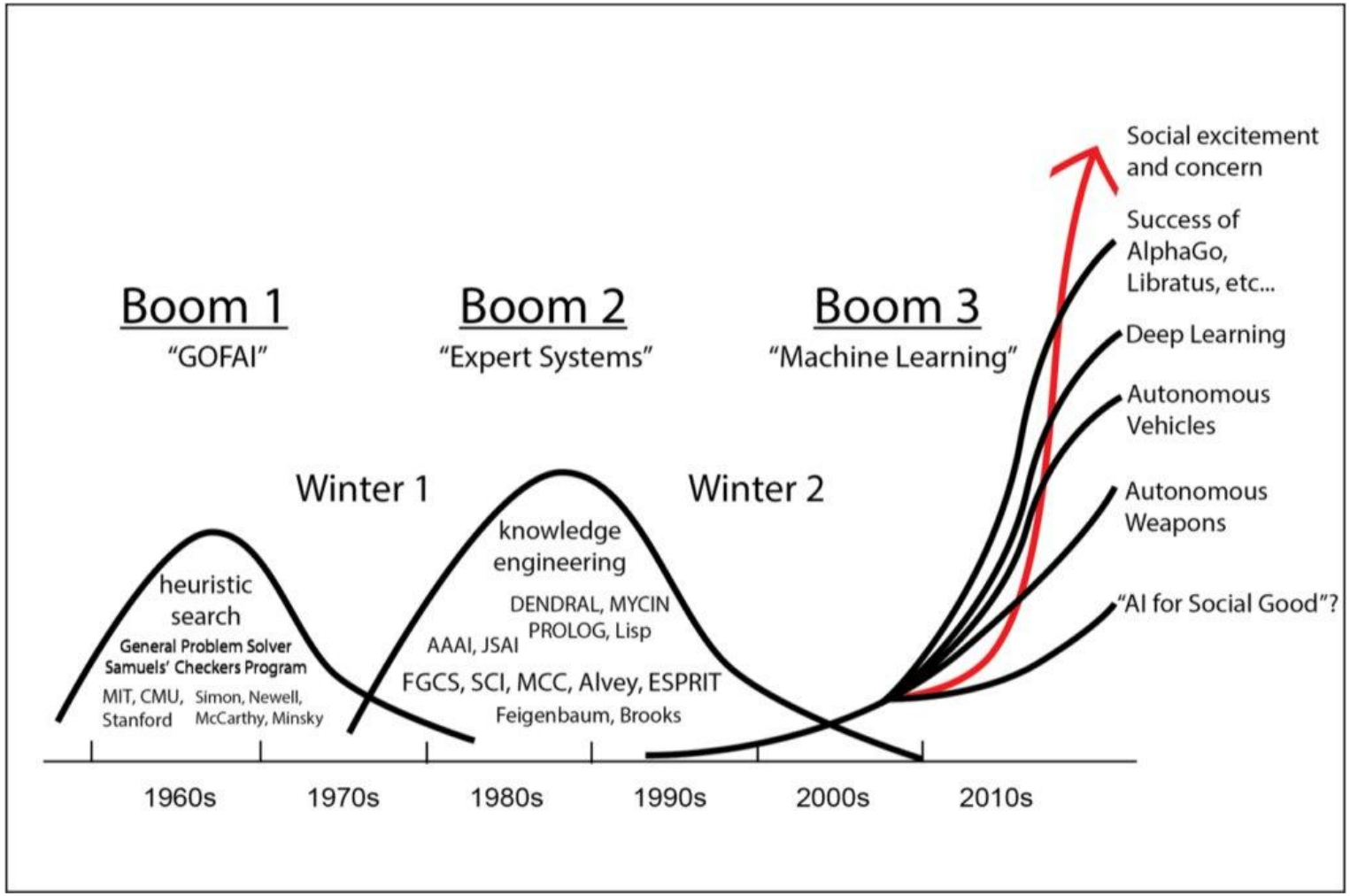
- 人工智慧（AI）於 1956 年被確立為一門學術領域，之後經歷了多次樂觀的發展週期，隨後又陷入失望和資金短缺的時期，這些時期被稱為「AI寒冬」。
- 自 2012 年以來，隨著深度學習在性能上超越了早期的 AI 技術，AI 的資金投入和關注度大幅提升。
- 這種增長在 2017 年之後因為變換器（Transformer）架構的出現進一步加速，到 2020 年代初，AI 領域的投資已達數千億美元，被稱為「AI熱潮」。

https://en.wikipedia.org/wiki/Artificial_intelligence



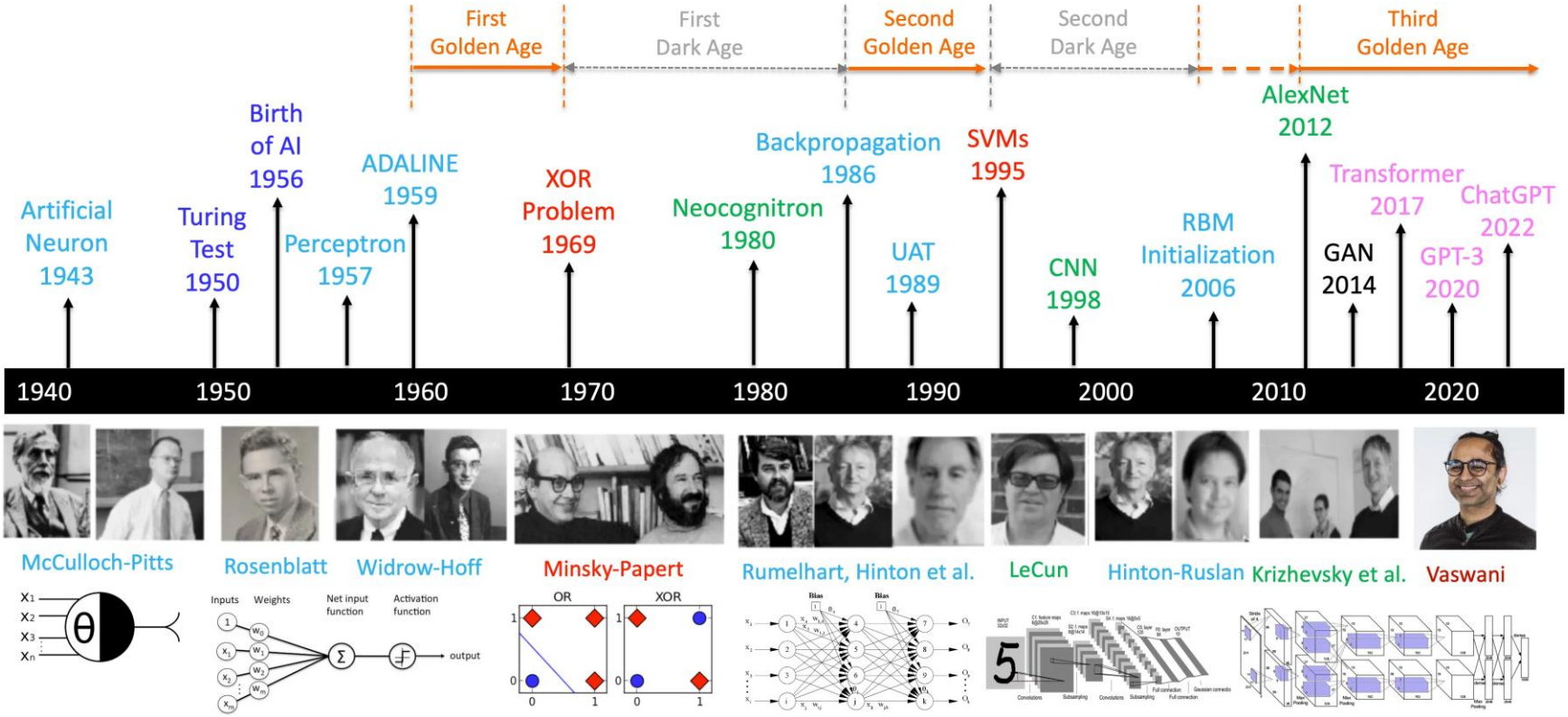
https://www.perplexity.ai/page/a-historical-overview-of-ai-wi-A8daV1D9Qr2STQ6tgLEOtg

AI Booms and AI Winters



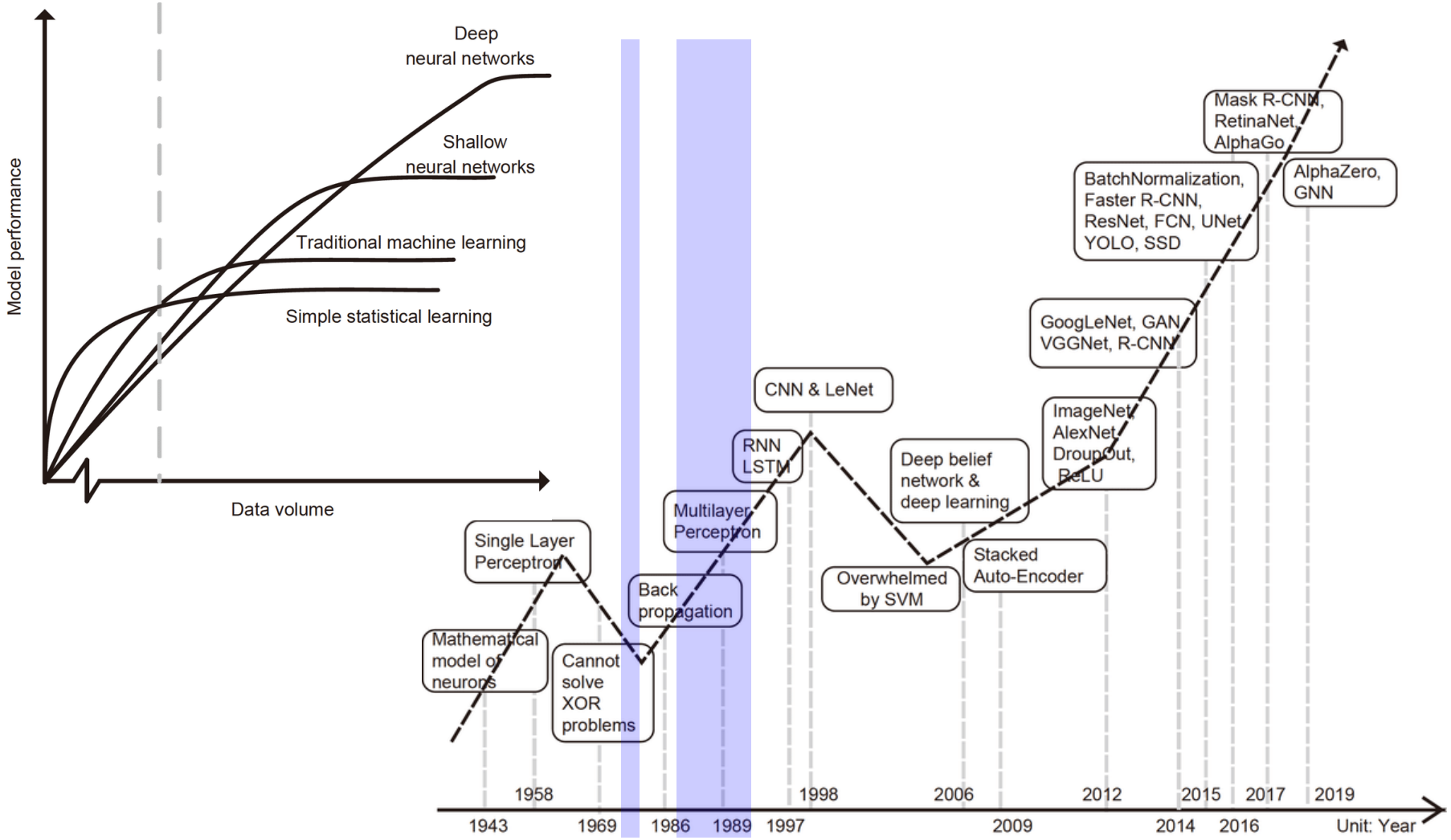
https://www.perplexity.ai/page/a-historical-overview-of-ai-wi-A8daV1D9Qr2STQ6tgLEOtg

A Brief History of AI with Deep Learning



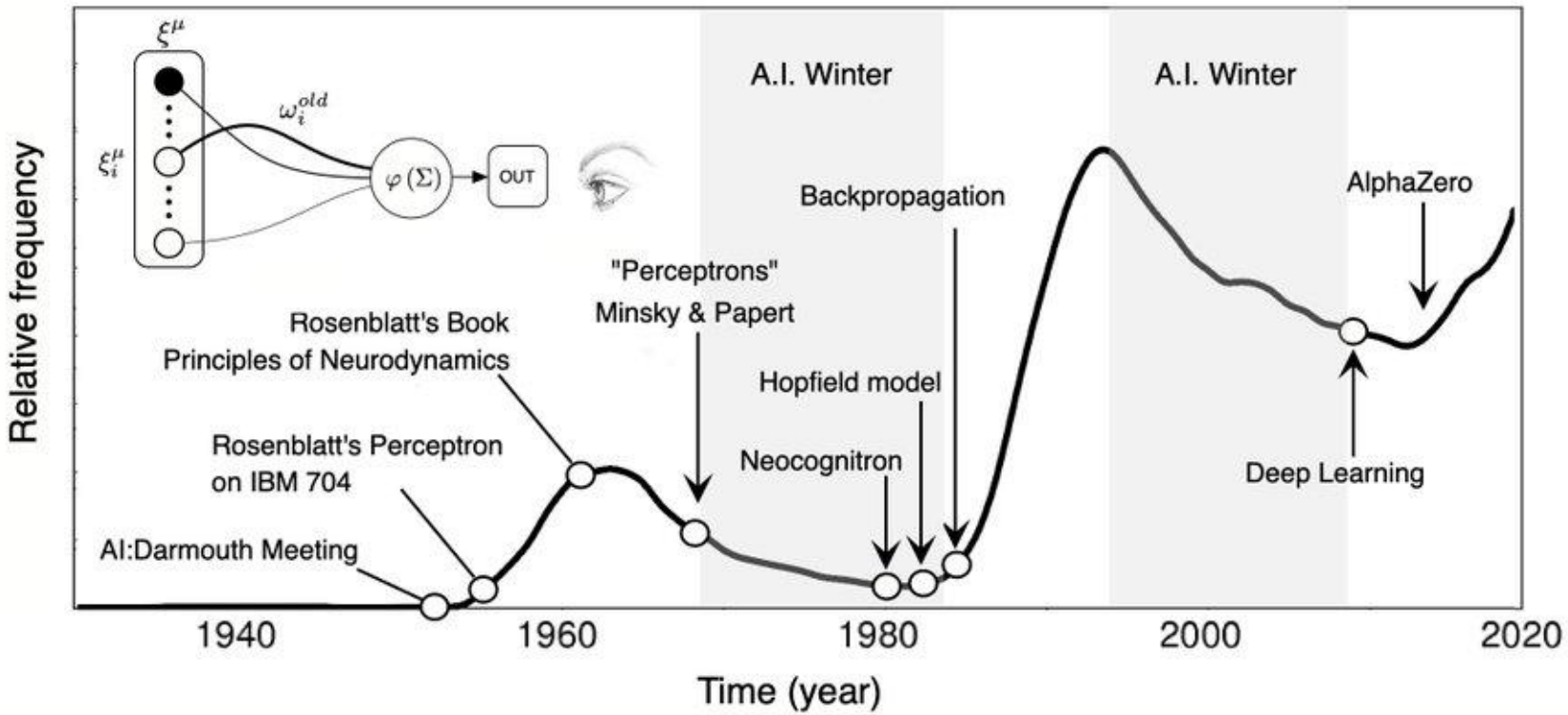
https://medium.com/@lmpo/a-brief-history-of-ai-with-deep-learning-26f7948bc87b

AI 之冬 和 Deep Learning
主要有1974-80及1987-93兩個主要的低谷時期



Guo, Q., Jin, S., Li, M. *et al.* Application of deep learning in ecological resource research: Theories, methods, and challenges. *Sci. China Earth Sci.* **63**, 1457–1474 (2020). https://doi.org/10.1007/s11430-019-9584-9

"Perceptron" (the first supervised neural network) in Google Ngrams



https://x.com/ricard_sole/status/1497306702181548039

https://books.google.com/ngrams/graph?content=perceptron&year_start=1930&year_end=2019&corpus=26&smoothing=3

生成式人工智慧 Generative AI

- 生成式人工智慧 (生成式 AI 、 GenAI 、或GAI) 是人工智慧的一個子集，它使用生成模型來產生文字、圖像、影片或其他形式的資料。和結構並使用它們根據輸入產生新資料，這些資料通常以自然語言提示的形式出現。
- 基於Transformer的深度神經網絡，特別是大型語言模型(LLMs) 的改進，促成了 2020 年代初生成式 AI 系統的AI 熱潮。
  - 其中包括ChatGPT 、 Copilot 、 Gemini和LLaMA等聊天機器人；
  - 文字轉圖像人工智慧圖像生成系統，如Stable Diffusion 、 Midjourney和DALL-E
  - 以及文字到影片的AI生成器，例如Sora。

機器學習的機制
(一個簡單的案例)

Chatgpt: “利用一維資料, 及 logistic function 舉例說明，如何用 machine learning的方式作 binary classification”

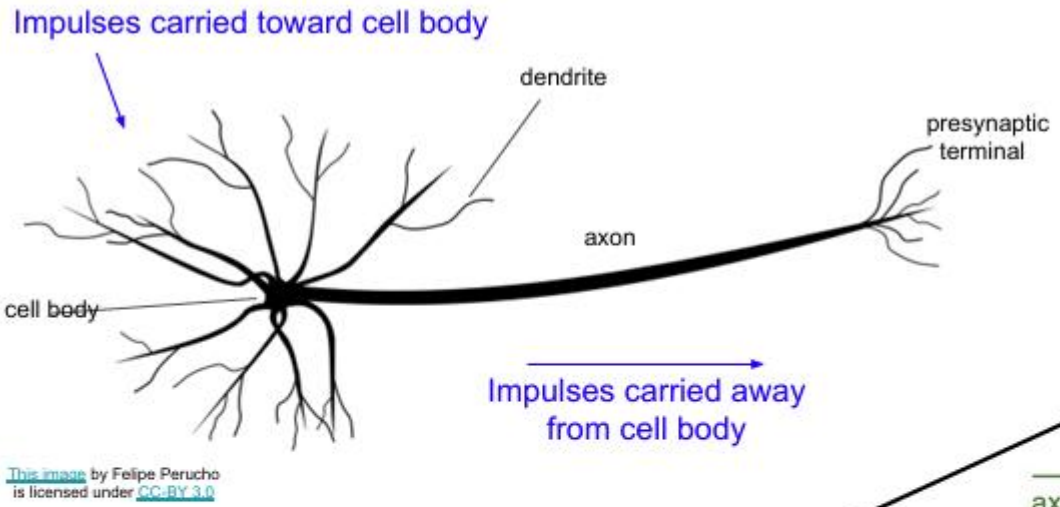
問題背景

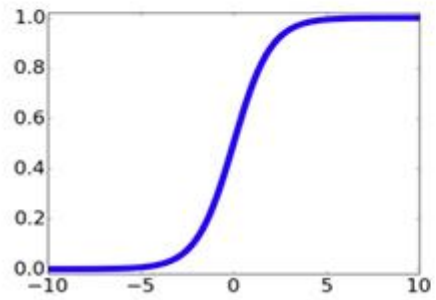
- 假設我們有一維資料，以及對應的二元標籤 $y \in \{0,1\}$ 。例如， x 可能代表某種測量值，而 y 表示該測量值是否超過某個分類標準（例如病患是否患病）。
- 數據準備
  - 假設有以下數據：

$$x = [1,2,3,4,5]$$
$$y = [0,0,0,1,1]$$

  - 這裡 y 表示 $x < 4$ 時為 0， $x \geq 4$ 時為 1。
- 目標是利用 Logistic Regression，學習一個函數來預測 $P(y = 1|x)$，並根據該概率進行分類。

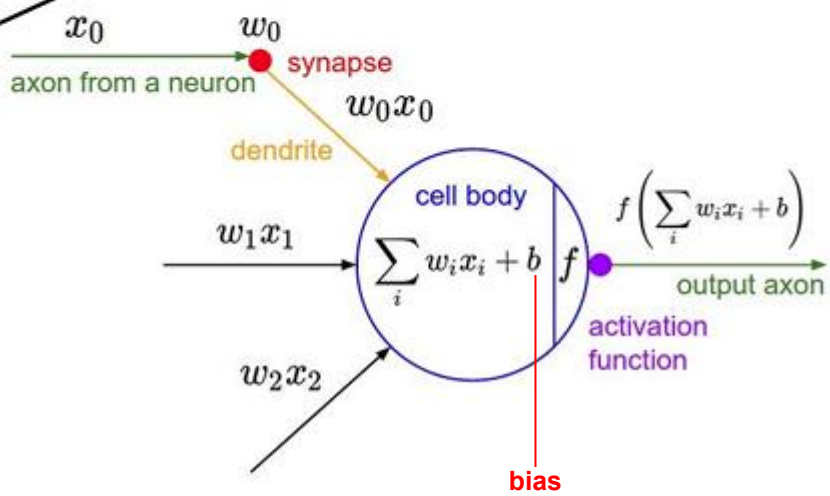
神經元 與 類神經元





sigmoid activation function

$$\frac{1}{1+e^{-x}}$$



圖片來自 https://cs231n.stanford.edu/ 課程投影片

Logistic Regression 原理

- Logistic Regression 的核心是使用 **Logistic Function**，其形式為：

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

- 其中，z 通常表示模型的線性組合：

$$z = w \cdot x + b$$

- w 是權重(weight)，b 是偏置項(bias)。
- Logistic Regression 的預測概率為：

$$P(y = 1|x) = \sigma(w \cdot x + b)$$

訓練過程

- 數據準備
  - 假設有以下數據：

$$x = [1,2,3,4,5]$$
$$y = [0,0,0,1,1]$$

  - 這裡 y 表示 $x < 4$ 時為 0， $x \geq 4$ 時為 1。
- 模型構建
  - 初始化權重 w 和偏置 b，例如初始值為 0。
- 損失函數
  - 使用 **交叉熵損失函數** (Cross-Entropy Loss)：

$$L = -\frac{1}{N}\sum_{i=1}^{N}[y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i)]$$

  - 其中

$$\hat{y}_i = P(y = 1|x_i)$$

- 梯度下降
  - 使用梯度下降更新參數 w 和 b ，直到損失函數收斂。

最大似然估計
Maximum Likelihood Estimation, MLE

- 條件概率

$$P(y=1|x) = \sigma(w \cdot x + b)$$
$$P(y=0|x) = 1 - \sigma(w \cdot x + b)$$
$$P(y|x) = \sigma(w \cdot x + b)^{y}\big(1 - \sigma(w \cdot x + b)\big)^{1-y}$$

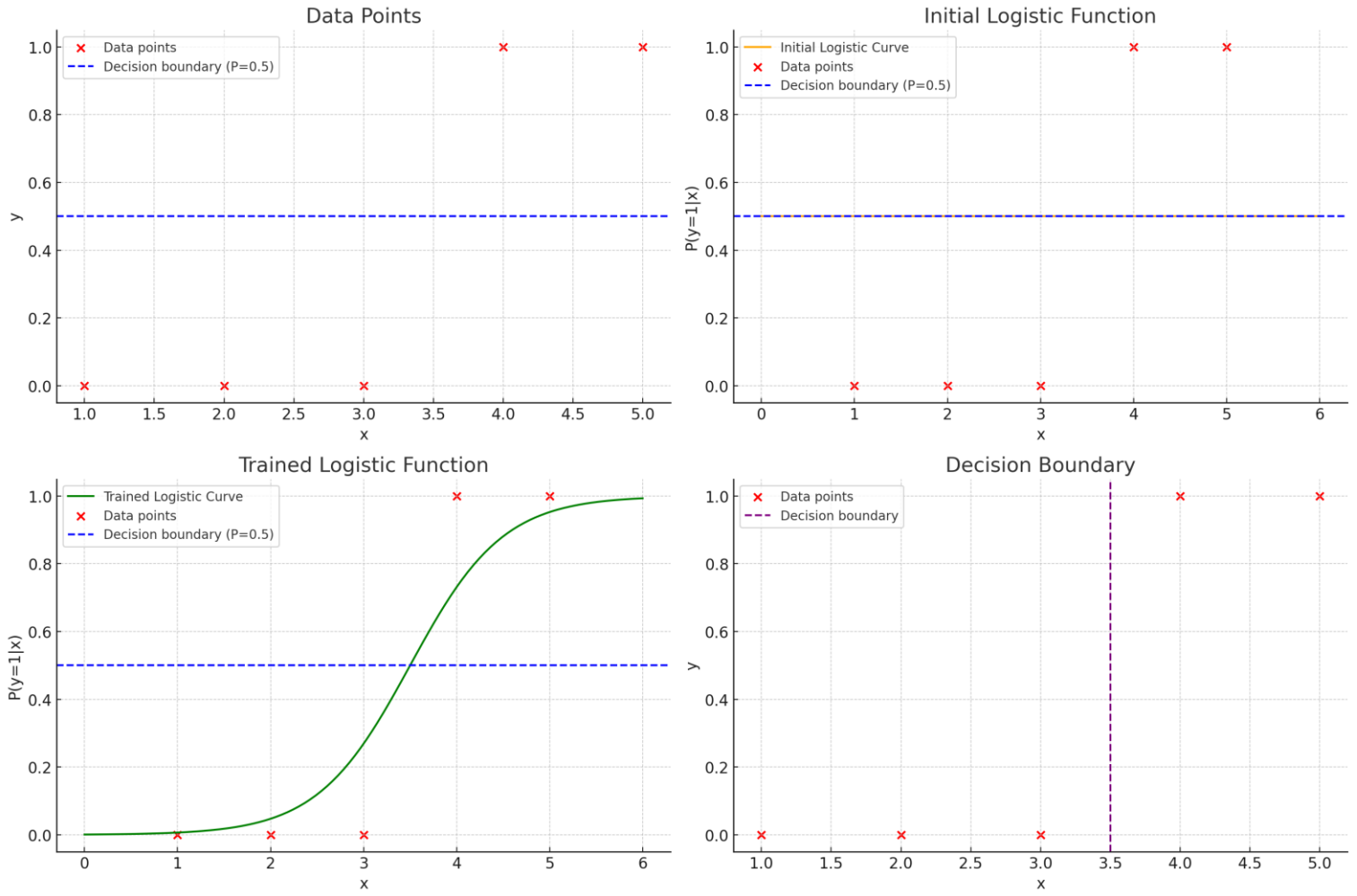
- 似然函數 (Maximum Likelihood)

$$L(w,b) = \prod_{i=1}^{N} P(y_i|x_i)$$
$$= \prod_{i=1}^{N} \sigma(w \cdot x_i + b)^{y_i}\big(1 - \sigma(w \cdot x_i + b)\big)^{1-y_i}$$

- 對$L(w,b)$取 log 就可以轉化為**交叉熵損失函數** (Cross-Entropy Loss)

其他相關資料: https://arbu00.blogspot.com/2017/02/5-logistic-regressionoverfitting-and.html

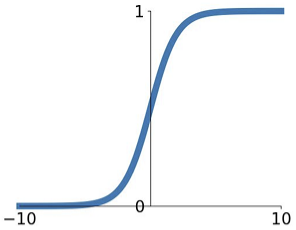
圖示步驟


Data Points
Data points
Decision boundary (P=0.5)
y
x
Initial Logistic Function
Initial Logistic Curve
Data points
Decision boundary (P=0.5)
P(y=1|x)
x
Trained Logistic Function
Trained Logistic Curve
Data points
Decision boundary (P=0.5)
P(y=1|x)
x
Decision Boundary
Data points
Decision boundary
y
x

Activation Function

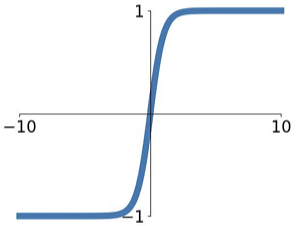
Sigmoid

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



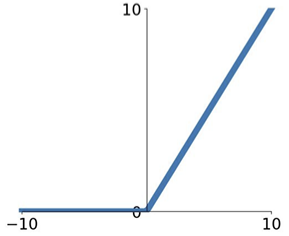
tanh

$$\tanh(x)$$



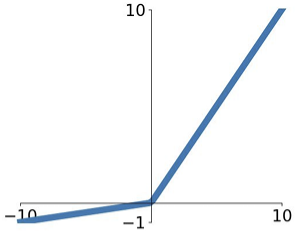
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

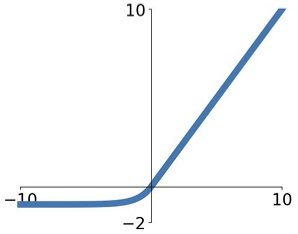


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

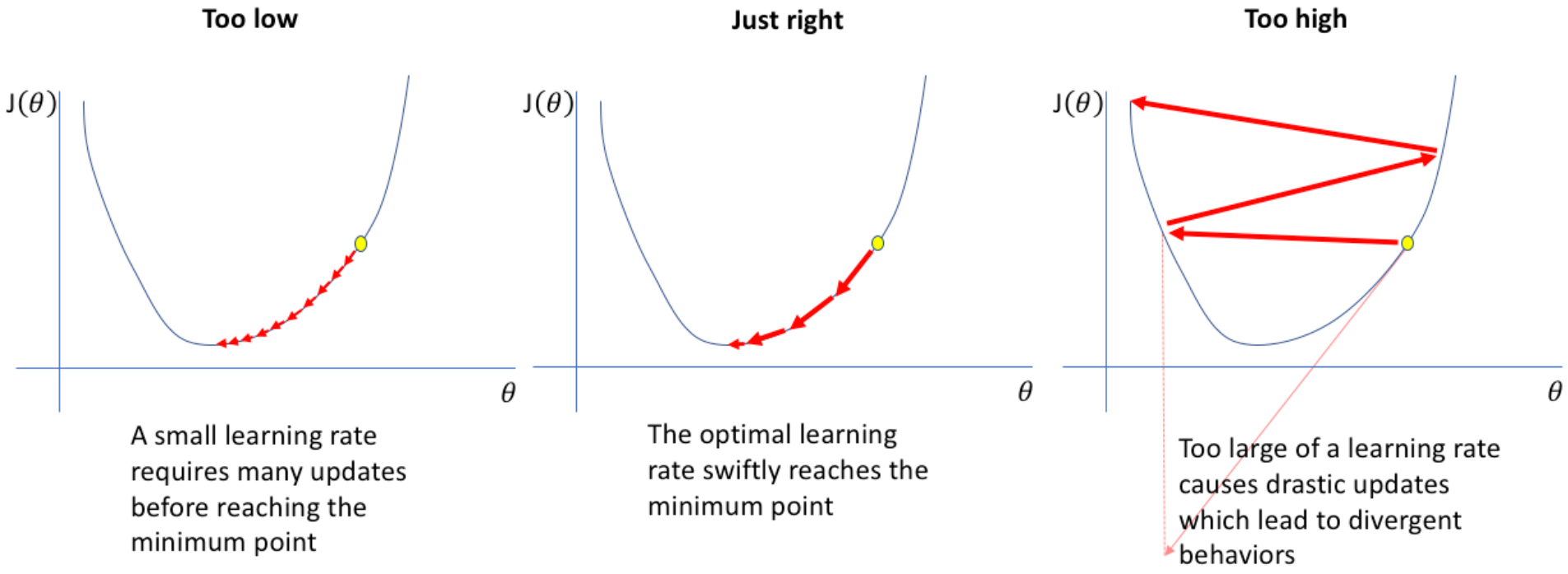
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



圖片來自 https://cs231n.stanford.edu/ 課程投影片

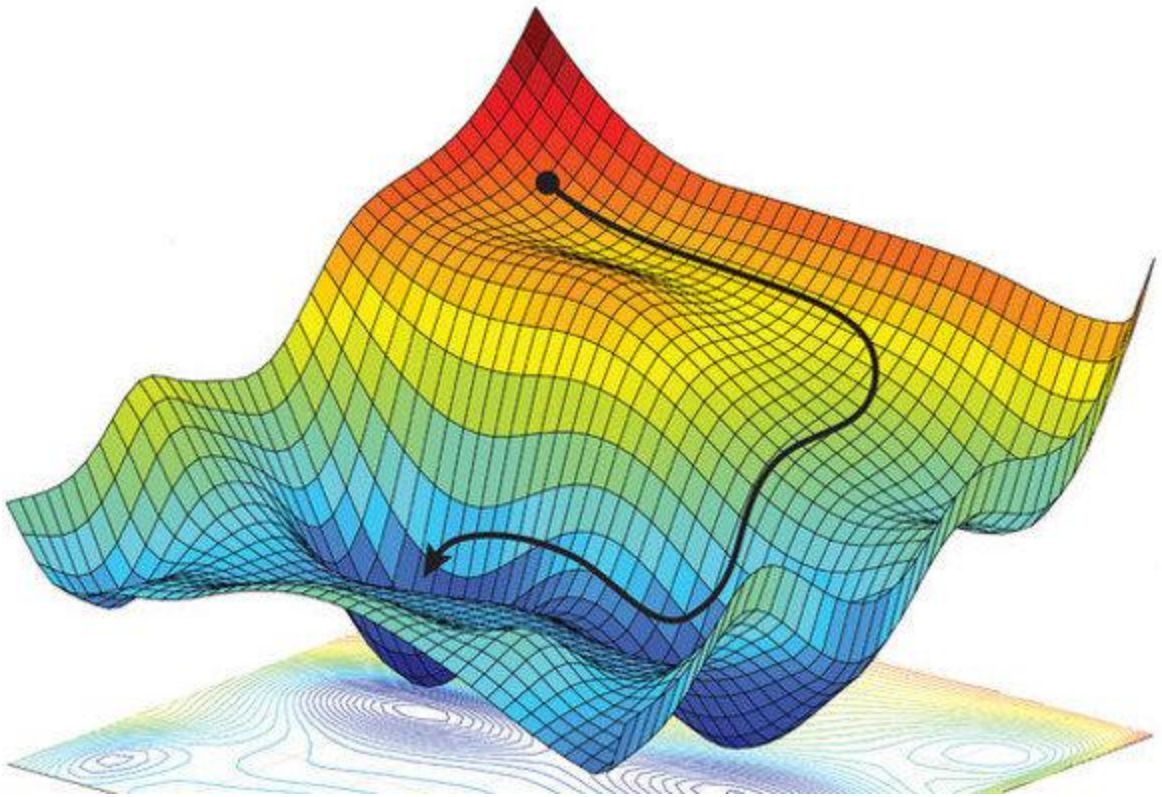
Minimum Value and Gradient (Slope)

- Learning steps for finding the min value.



https://pub.aimind.so/gradient-descent-best-optimization-technique-bfcb392b95e9

Hyperplane of Loss Function



https://discuss.pytorch.org/t/why-is-my-loss-function-going-down-and-jumping/170508

Find Gradient with Backpropogation

Backpropagation: a simple example

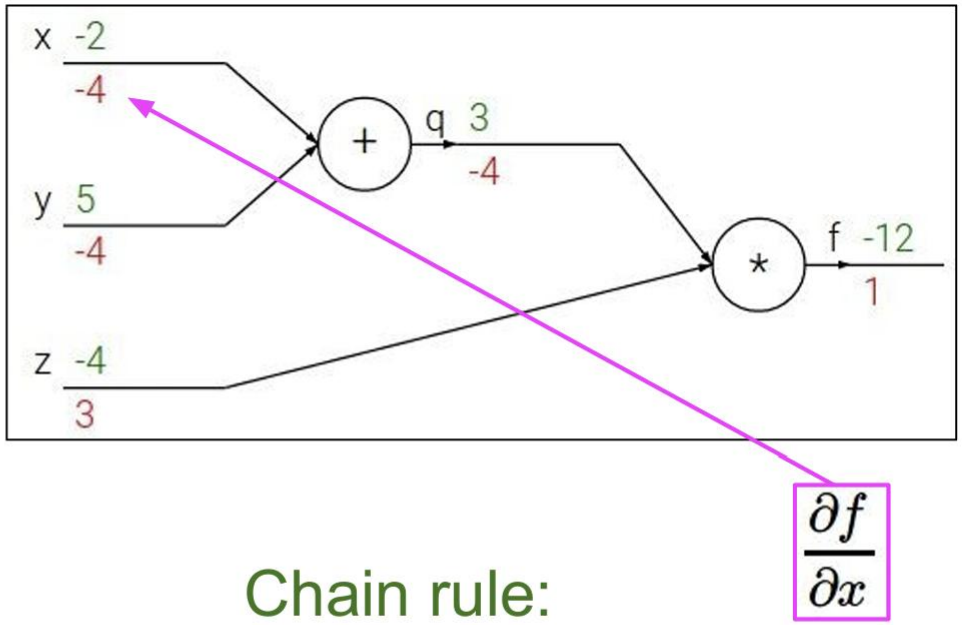
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Chain rule:

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial x}$$

Upstream gradient

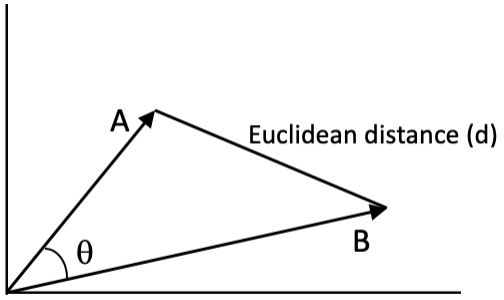
Local gradient

圖片來自 https://cs231n.stanford.edu/ 課程投影片

大語言模型的基本原理

語言理解的背景說明

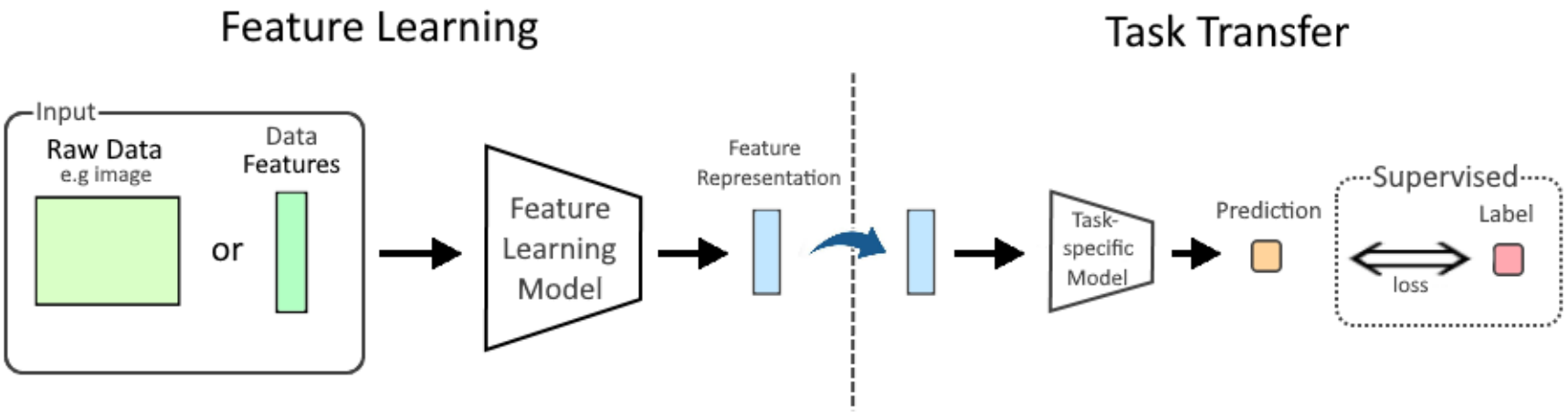
- 如何學習語言，如何理解語言
  - 字義 ➔ 句型句義 ➔ 段落及文章結構
- 上下文 (context)
  - 我們用字的組合來決定句子
  - 但我們也用前後的字來決定字的意義
- 如何用計算表達語言理解
  - 字義 ➔ 字的相關性相似度
  - 句型句義 ➔ 字的短距離前後連接關係
  - 段落及文章結構 ➔ 字的長距離連接關係
- 轉換成數學模型 (向量空間, 機率)
  - 點向量代表字
  - 點向量可以算距離，用向量內積與夾角餘弦值(cosine) 來代表相似度
  - 用n-gram語言模型來表示上下文的連接關係



$$similarity(A,B) = \cos(\theta) = \frac{A \cdot B}{\|A\| \times \|B\|}$$

表徵學習 Feature learning

- 在機器學習(ML) 中，表徵學習 Feature learning / Representation learning 是一組技術，可讓系統從原始資料中自動發現特徵偵測或分類所需的表示。這取代了手動特徵工程，並允許機器學習特徵並使用它們來執行特定任務。



https://en.wikipedia.org/wiki/Feature_learning

詞嵌入 Word Embedding (word2vec)

…government debt problems turning into ***banking*** *crises as happened in 2009…*

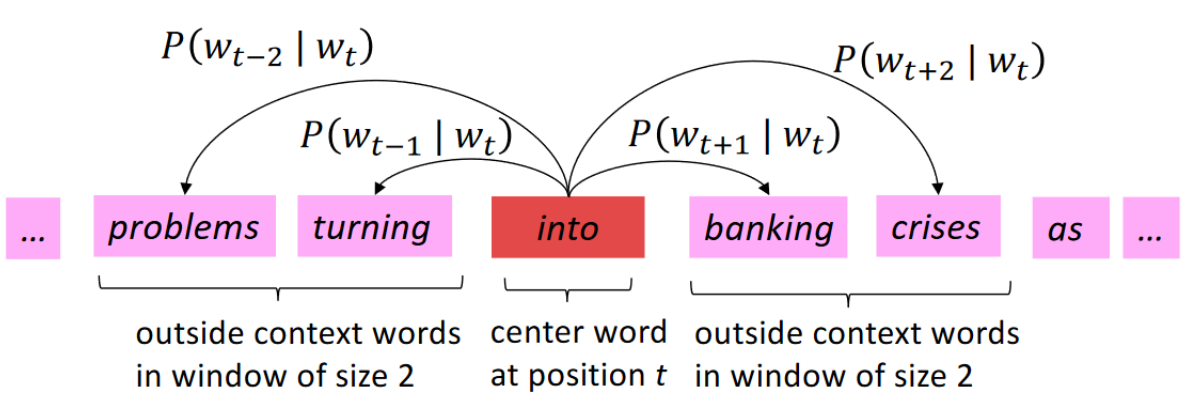
…saying that Europe needs unified ***banking*** *regulation to replace the hodgepodge…*

…India has just given its ***banking*** *system a shot in the arm…*

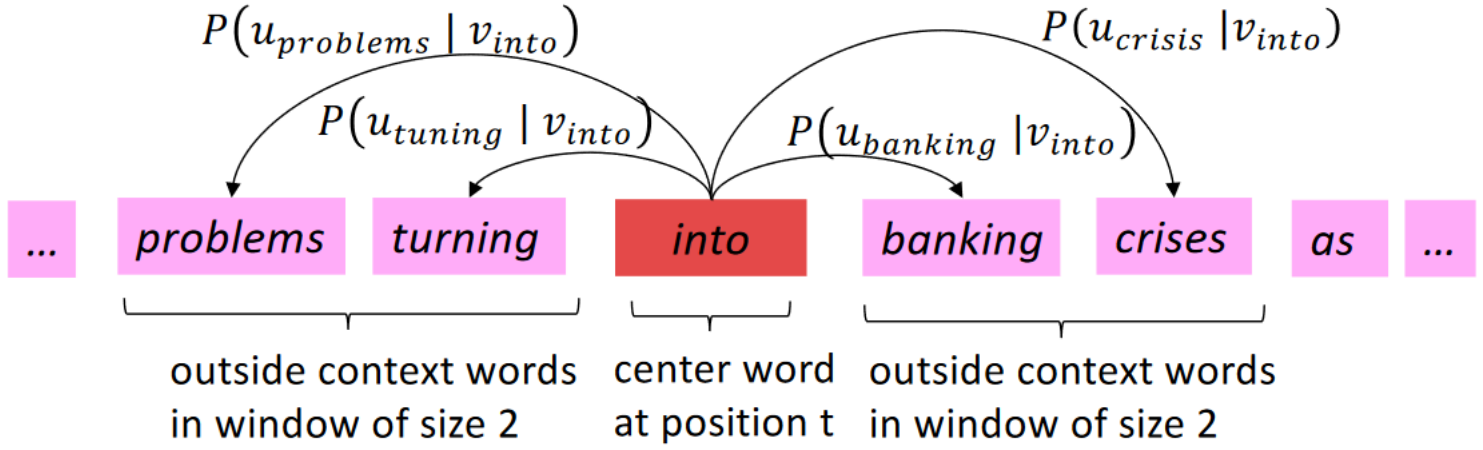
These context words will represent ***banking***

banking =

$$\begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \end{pmatrix}$$



圖片來自 https://web.stanford.edu/class/cs224n/ 課程投影片



② Exponentiation makes anything positive

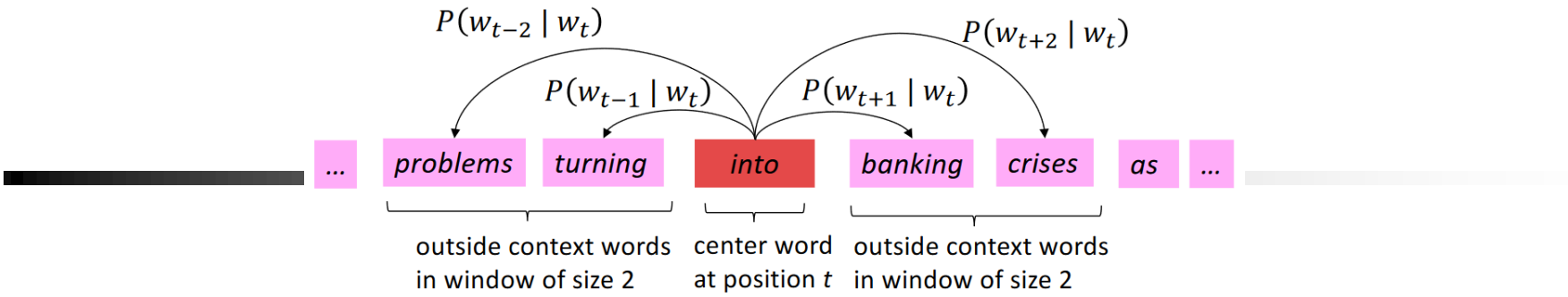
① Dot product compares similarity of *o* and *c*.
$u^T v = u.v = \sum_{i=1}^{n} u_i v_i$
Larger dot product = larger probability

$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

③ Normalize over entire vocabulary to give probability distribution

- v_w when *w* is a center word
- u_w when *w* is a context word

圖片來自 https://web.stanford.edu/class/cs224n/ 課程投影片



For each position $t = 1, \ldots, T$, predict context words within a window of fixed size m, given center word w_j. Data likelihood:

$$\text{Likelihood} = L(\theta) = \prod_{t=1}^{T} \prod_{\substack{-m \le j \le m \\ j \neq 0}} P(w_{t+j} \mid w_t; \theta)$$

θ is all variables to be optimized

sometimes called a *cost* or *loss* function

The objective function $J(\theta)$ is the (average) negative log likelihood:

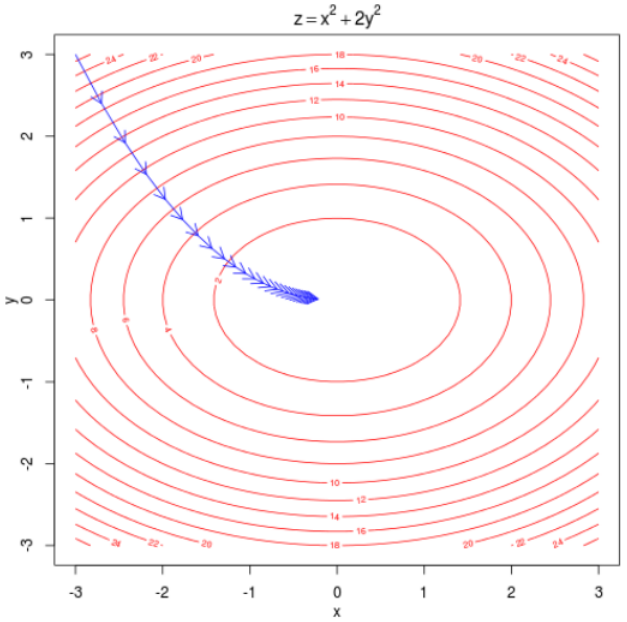
$$J(\theta) = -\frac{1}{T} \log L(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{\substack{-m \le j \le m \\ j \neq 0}} \log P(w_{t+j} \mid w_t; \theta)$$

Minimizing objective function ⟺ Maximizing predictive accuracy

圖片來自 https://web.stanford.edu/class/cs224n/ 課程投影片

圖片來自 https://web.stanford.edu/class/cs224n/ 課程投影片

$$\theta = \begin{bmatrix} v_{aardvark} \\ v_a \\ \vdots \\ v_{zebra} \\ u_{aardvark} \\ u_a \\ \vdots \\ u_{zebra} \end{bmatrix} \in \mathbb{R}^{2dV}$$



- Update equation (in matrix notation):

$$\theta^{new} = \theta^{old} - \alpha \nabla_\theta J(\theta)$$

α = *step size* or *learning rate*

- Update equation (for single parameter):

$$\theta_j^{new} = \theta_j^{old} - \alpha \frac{\partial}{\partial \theta_j^{old}} J(\theta)$$

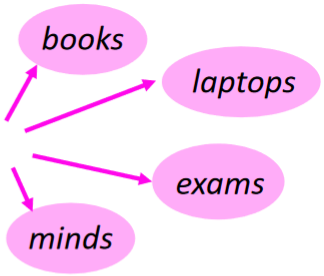
- Algorithm:

```
while True:
    theta_grad = evaluate_gradient(J,corpus,theta)
    theta = theta - alpha * theta_grad
```

語言模型 Language Model

- **Language Modeling** is the task of predicting what word comes next

the students opened their _____



- More formally: given a sequence of words $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(t)}$, compute the probability distribution of the next word $\boldsymbol{x}^{(t+1)}$:

$$P(\boldsymbol{x}^{(t+1)} | \ \boldsymbol{x}^{(t)}, \ldots, \boldsymbol{x}^{(1)})$$

where $\boldsymbol{x}^{(t+1)}$ can be any word in the vocabulary $V = \{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{|V|}\}$

- A system that does this is called a **Language Model**

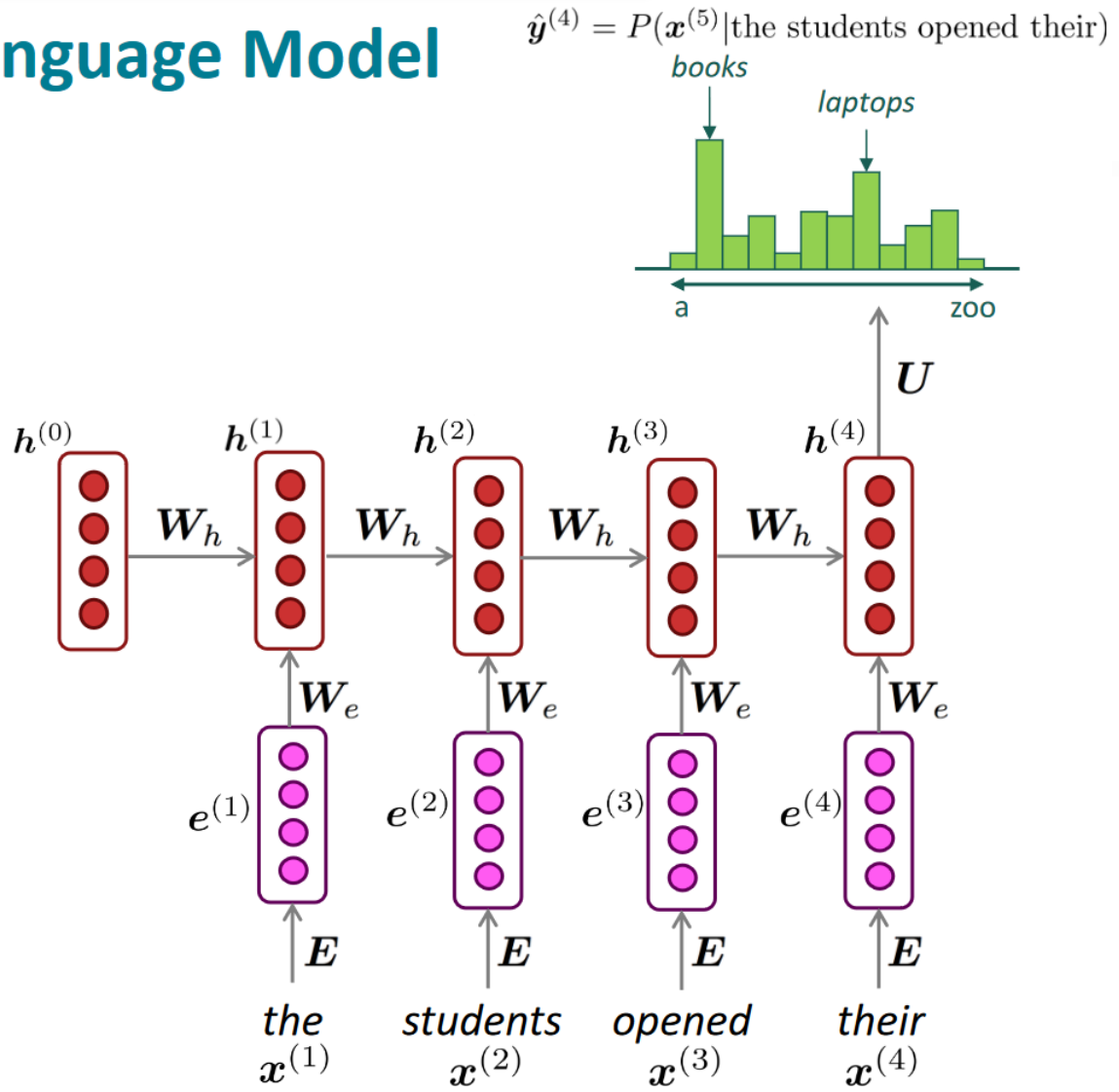
$$P(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(T)}) = P(\boldsymbol{x}^{(1)}) \times P(\boldsymbol{x}^{(2)} | \ \boldsymbol{x}^{(1)}) \times \cdots \times P(\boldsymbol{x}^{(T)} | \ \boldsymbol{x}^{(T-1)}, \ldots, \boldsymbol{x}^{(1)})$$

$$= \prod_{t=1}^{T} P(\boldsymbol{x}^{(t)} | \ \boldsymbol{x}^{(t-1)}, \ldots, \boldsymbol{x}^{(1)})$$

This is what our LM provides

圖片來自 https://web.stanford.edu/class/cs224n/ 課程投影片

A Simple RNN Language Model



output distribution

$$\hat{y}^{(t)} = \text{softmax}\left(Uh^{(t)} + b_2\right) \in \mathbb{R}^{|V|}$$

hidden states

$$h^{(t)} = \sigma\left(W_h h^{(t-1)} + W_e e^{(t)} + b_1\right)$$

$h^{(0)}$ is the initial hidden state

word embeddings

$$e^{(t)} = Ex^{(t)}$$

words / one-hot vectors

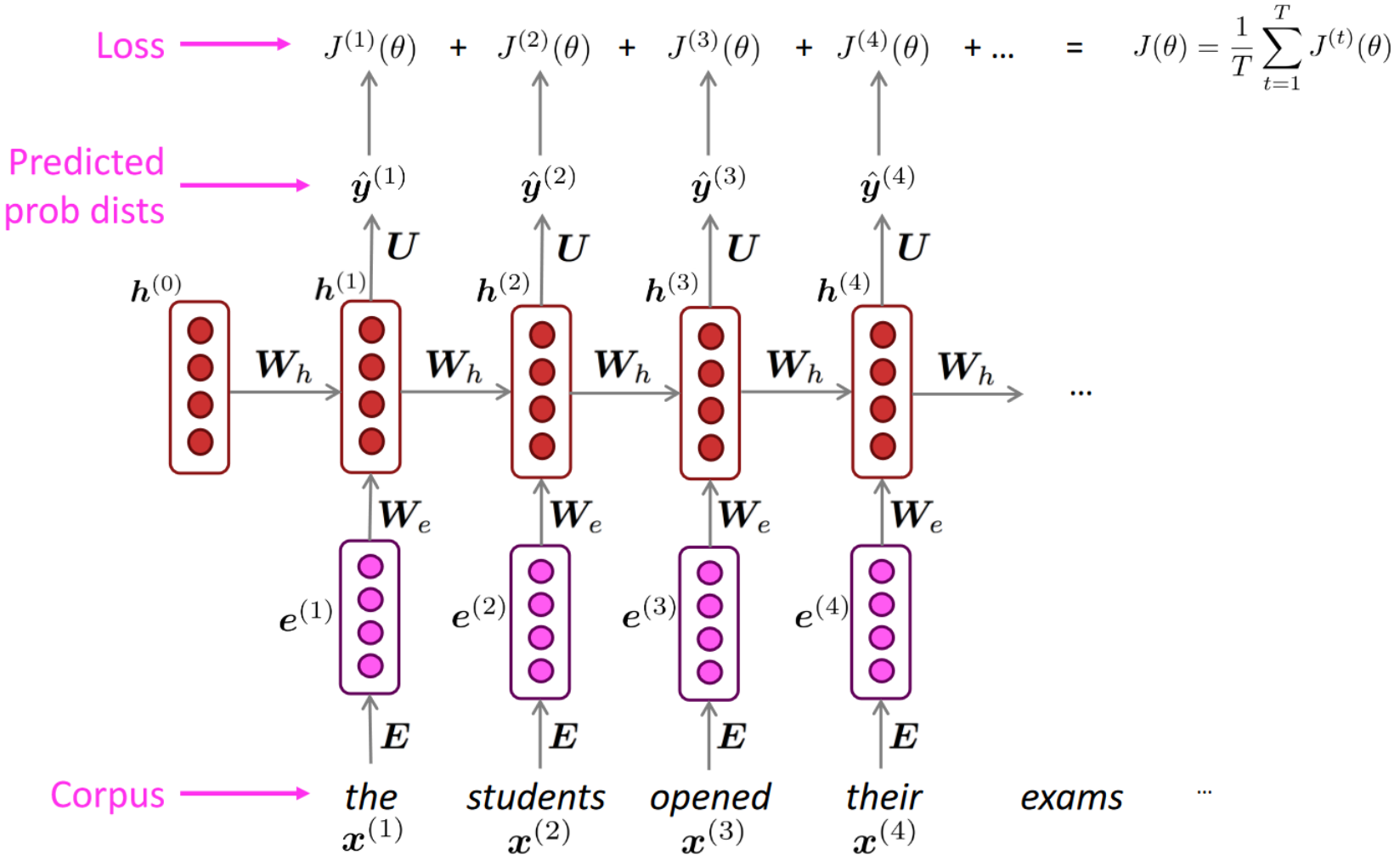
$$x^{(t)} \in \mathbb{R}^{|V|}$$

Note: *this input sequence could be much longer now!*

圖片來自 https://web.stanford.edu/class/cs224n/ 課程投影片

Training an RNN Language Model

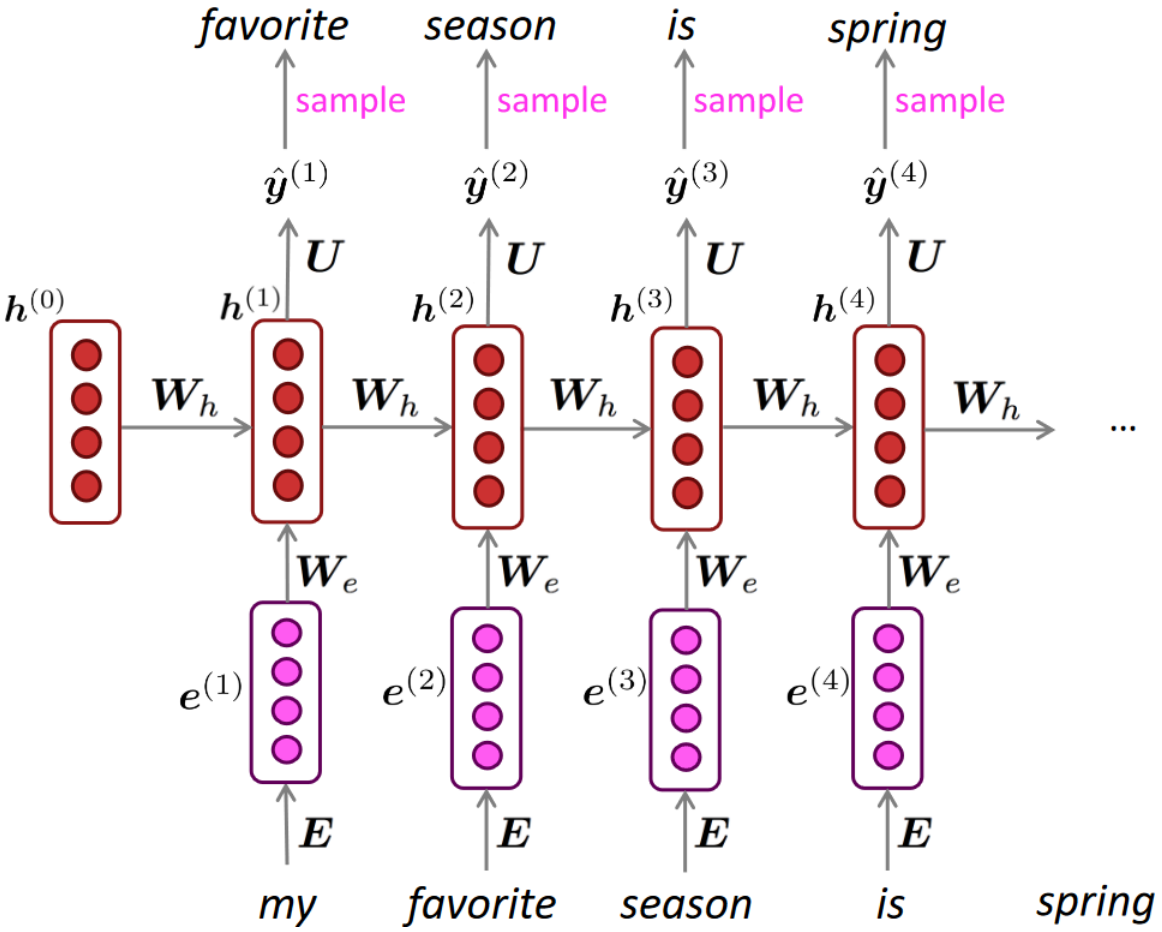
"Teacher forcing"



圖片來自 https://web.stanford.edu/class/cs224n/ 課程投影片

Generating text with a RNN Language Model

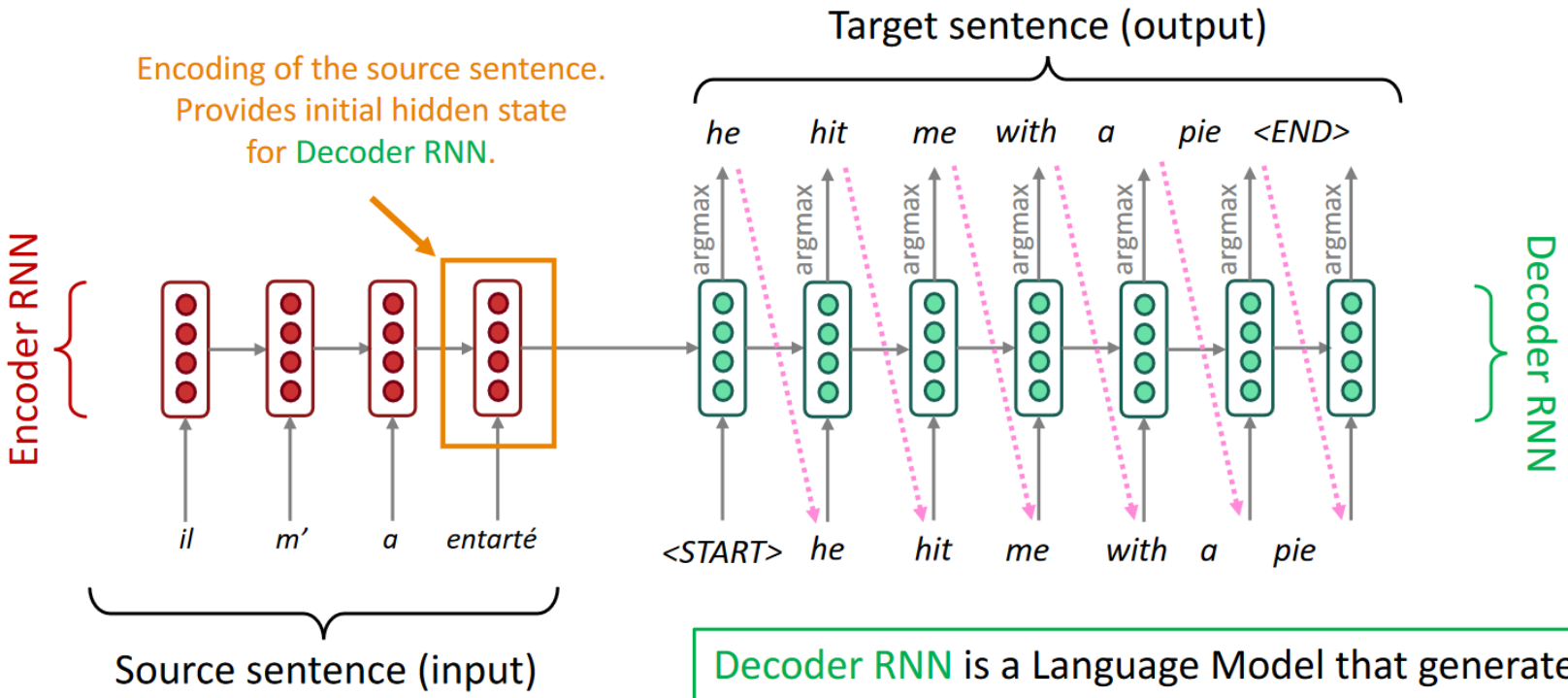
Just like a n-gram Language Model, you can use a RNN Language Model to generate text by repeated sampling. Sampled output becomes next step's input.



圖片來自 https://web.stanford.edu/class/cs224n/ 課程投影片

Neural Machine Translation (NMT)

The sequence-to-sequence model



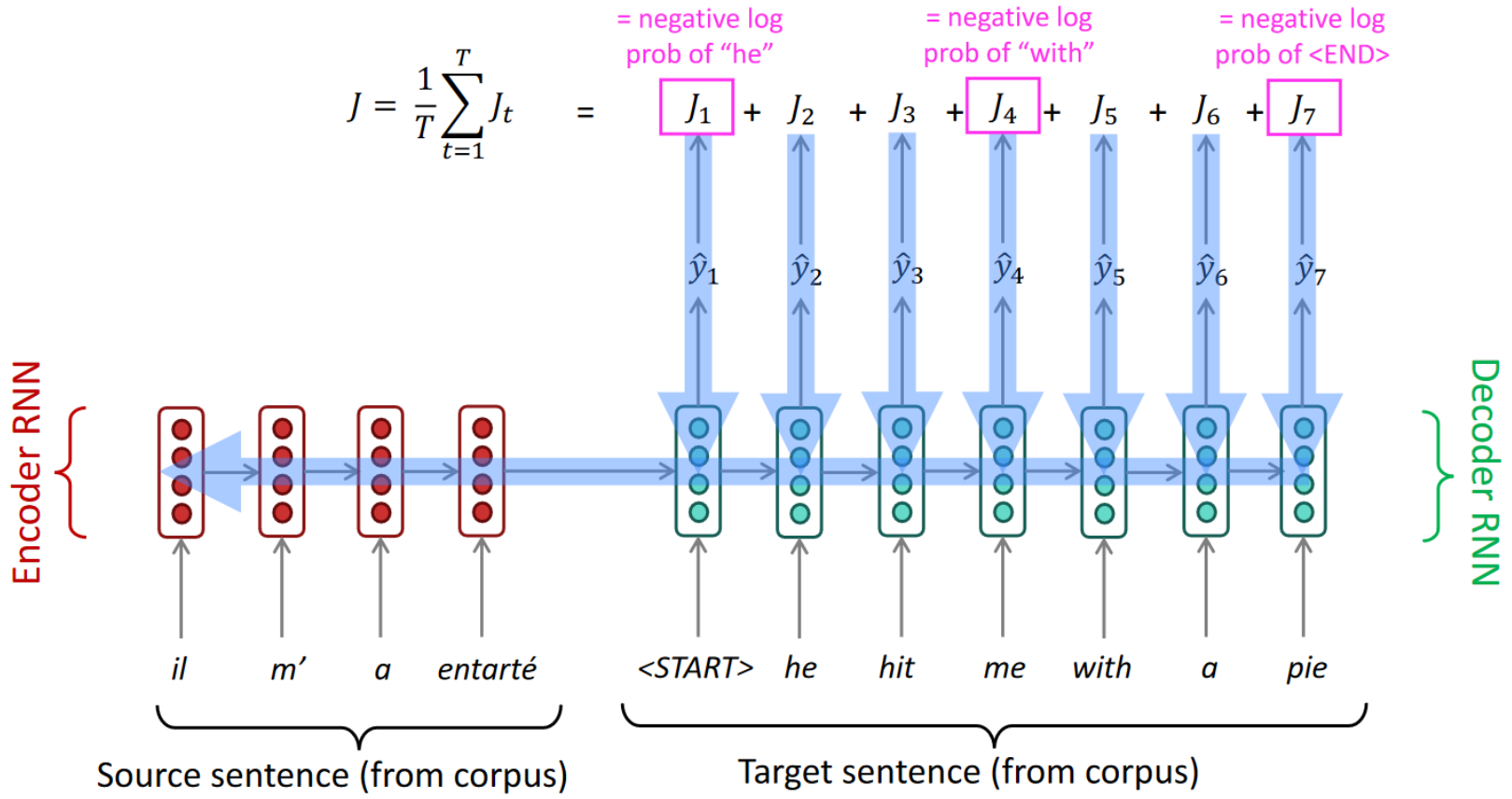
Decoder RNN is a Language Model that generates target sentence, *conditioned on encoding*.

Encoder RNN produces an encoding of the source sentence.

Note: This diagram shows **test time** behavior: decoder output is fed in ⋯→ as next step's input

圖片來自 https://web.stanford.edu/class/cs224n/ 課程投影片

Training a Neural Machine Translation system



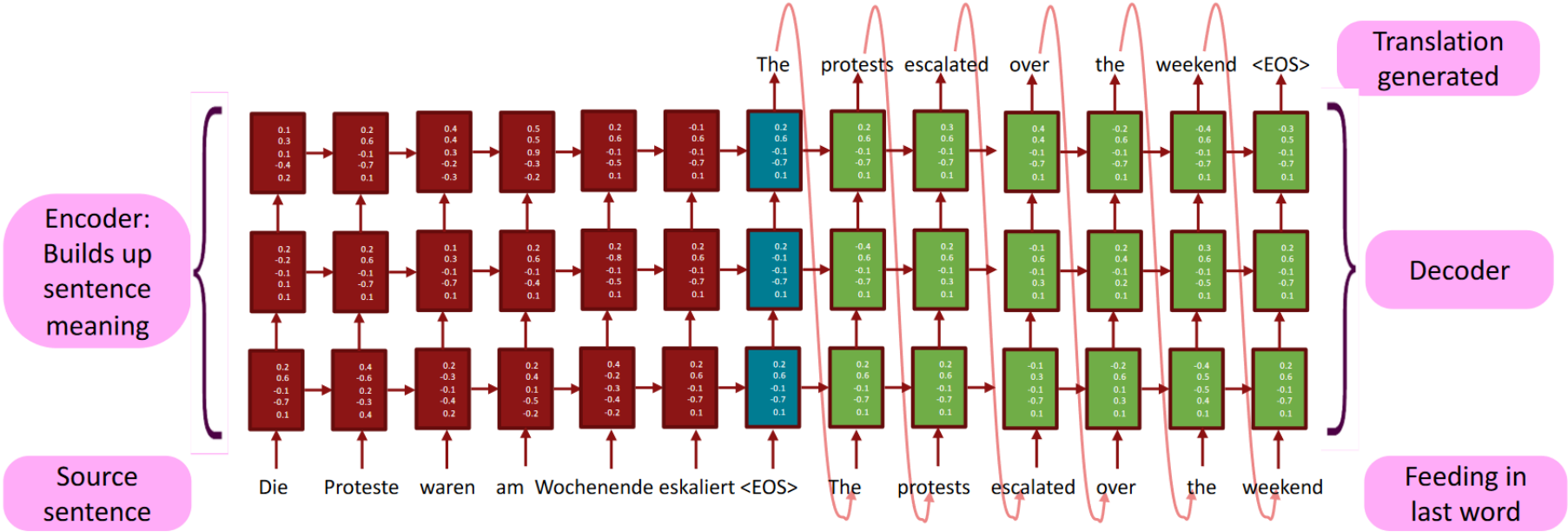
Seq2seq is optimized as a **single system.** Backpropagation operates "*end-to-end*".

圖片來自 https://web.stanford.edu/class/cs224n/ 課程投影片

Multi-layer deep encoder-decoder machine translation net

[Sutskever et al. 2014; Luong et al. 2015]

The hidden states from RNN layer *i* are the inputs to RNN layer *i*+1

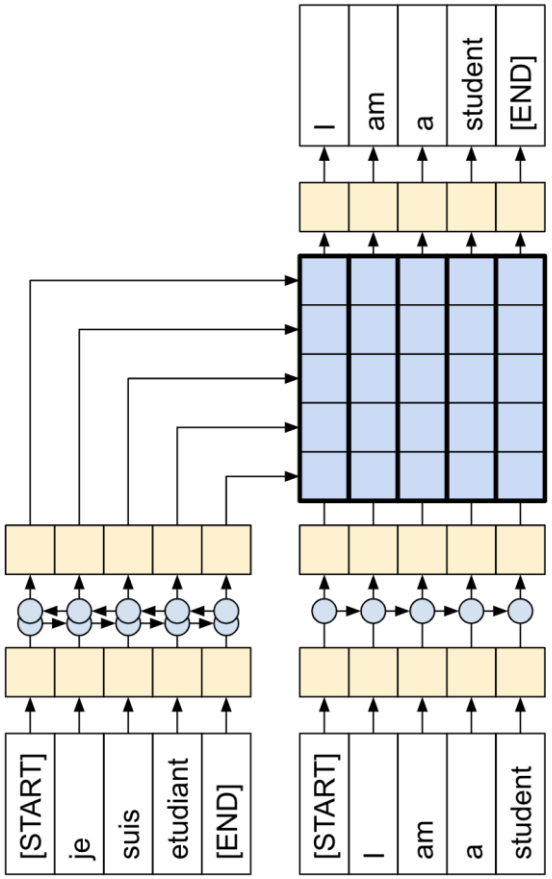


Conditioning = Bottleneck

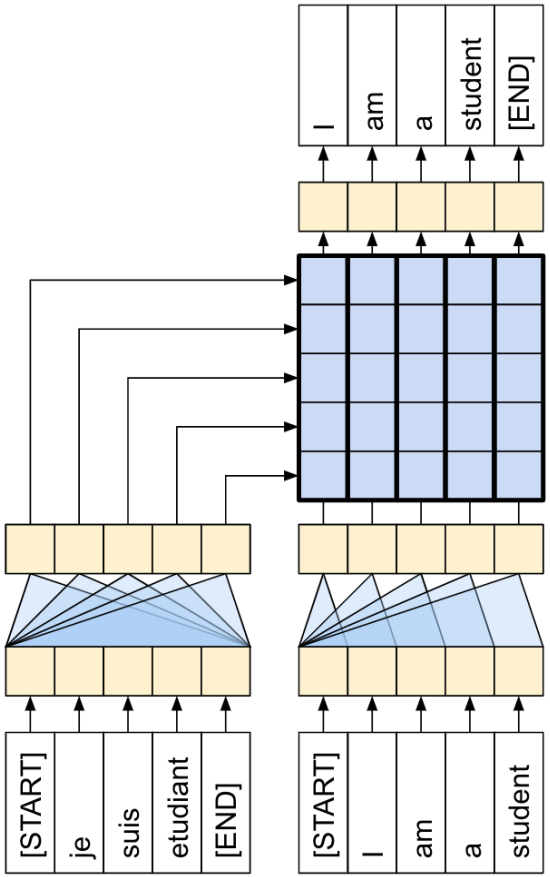
圖片來自 https://web.stanford.edu/class/cs224n/ 課程投影片

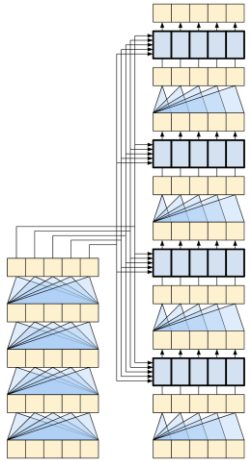
Attention 取代 RNN

The RNN+Attention model



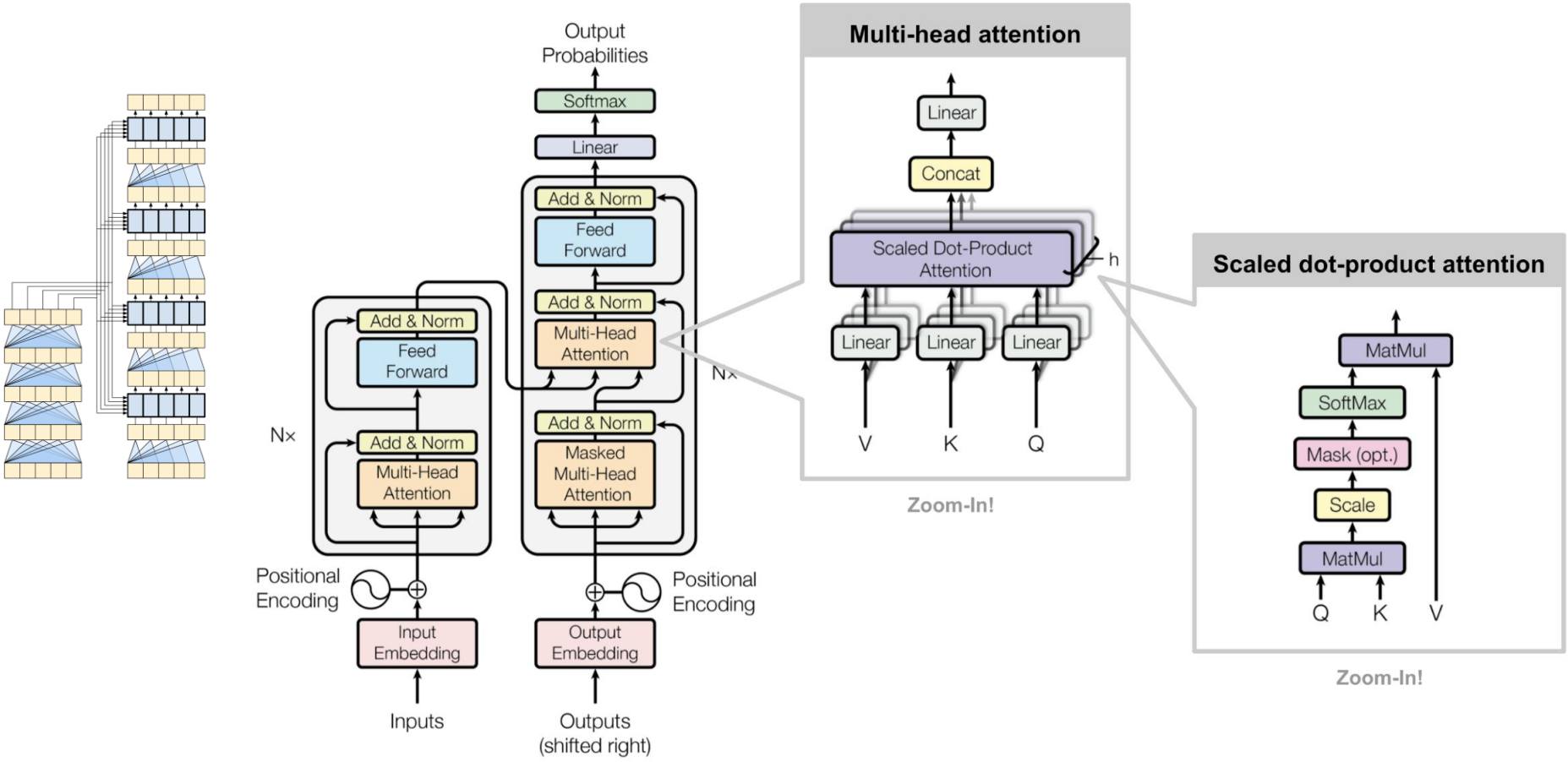
A 1-layer transformer





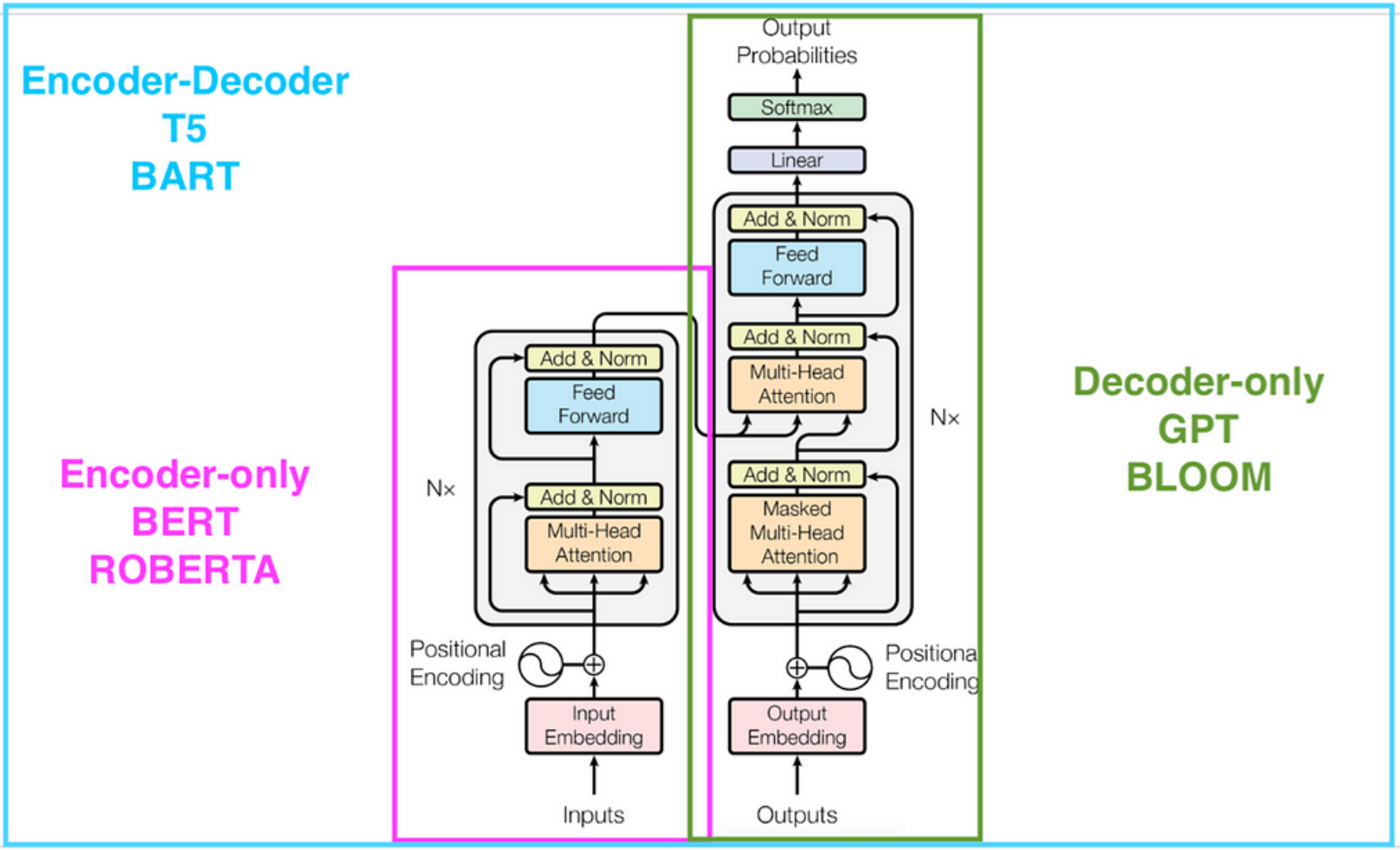
https://www.tensorflow.org/text/tutorials/transformer

Transformer



Vaswani, Ashish; Shazeer, Noam; Parmar, Niki; Uszkoreit, Jakob; Jones, Llion; Gomez, Aidan N; Kaiser, Łukasz; Polosukhin, Illia (2017). **"Attention is All you Need"**. Advances in Neural Information Processing Systems. 30. Curran Associates, Inc.

Encoder and Decoder



https://medium.com/@reyhaneh.esmailbeigi/bert-gpt-and-bart-a-short-comparison-5d6a57175fca

Transformers

ENCODER ONLY

aka

auto-encoding models

TASKS

- Sentence classification
- Named entity recognition
- Extractive question-answering
- Masked language modeling

EXAMPLES

BERT, RoBERTa, distilBERT

DECODER ONLY

aka

auto-regressive models

TASKS

- Text generation
- Causal language modeling

EXAMPLES

GPT-2, GPT Neo, GPT-3

ENCODER-DECODER

aka

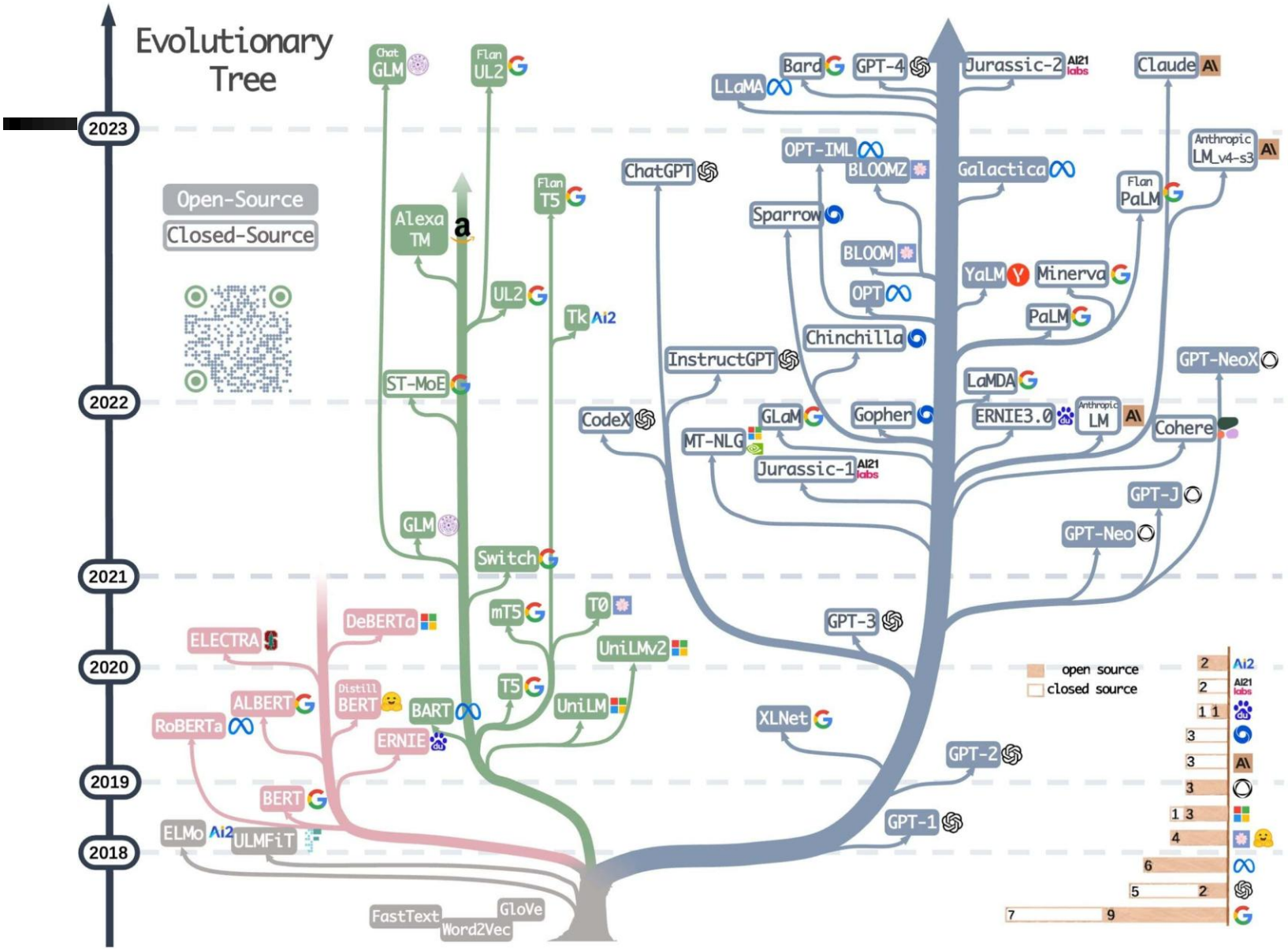
sequence-to-sequence models

TASKS

- Translation
- Summarization
- Generative question-answering

EXAMPLES

BART, T5, Marian



https://www.baeldung.com/cs/bert-vs-gpt-3-architecture

AI 基礎模型
Foundation Models

History of Machine Learning

Model Size (#parameters)	None (Hand-designed rules)	Very few	Few	Large	Very large
Features	Hand-designed	Hand-designed	Learnt (Simple features)	Learnt (Hierarchy of features)	Learnt (Hierarchy of features)
Learning	None	Supervised	Supervised	Supervised	Self-supervised + in-context learning
Data	None	Very few labeled-data	Few labeled-data	Large labeled-data	Very large unlabeled data (+ small labeled data)
Adaptability	None	None	Little	Medium	Large

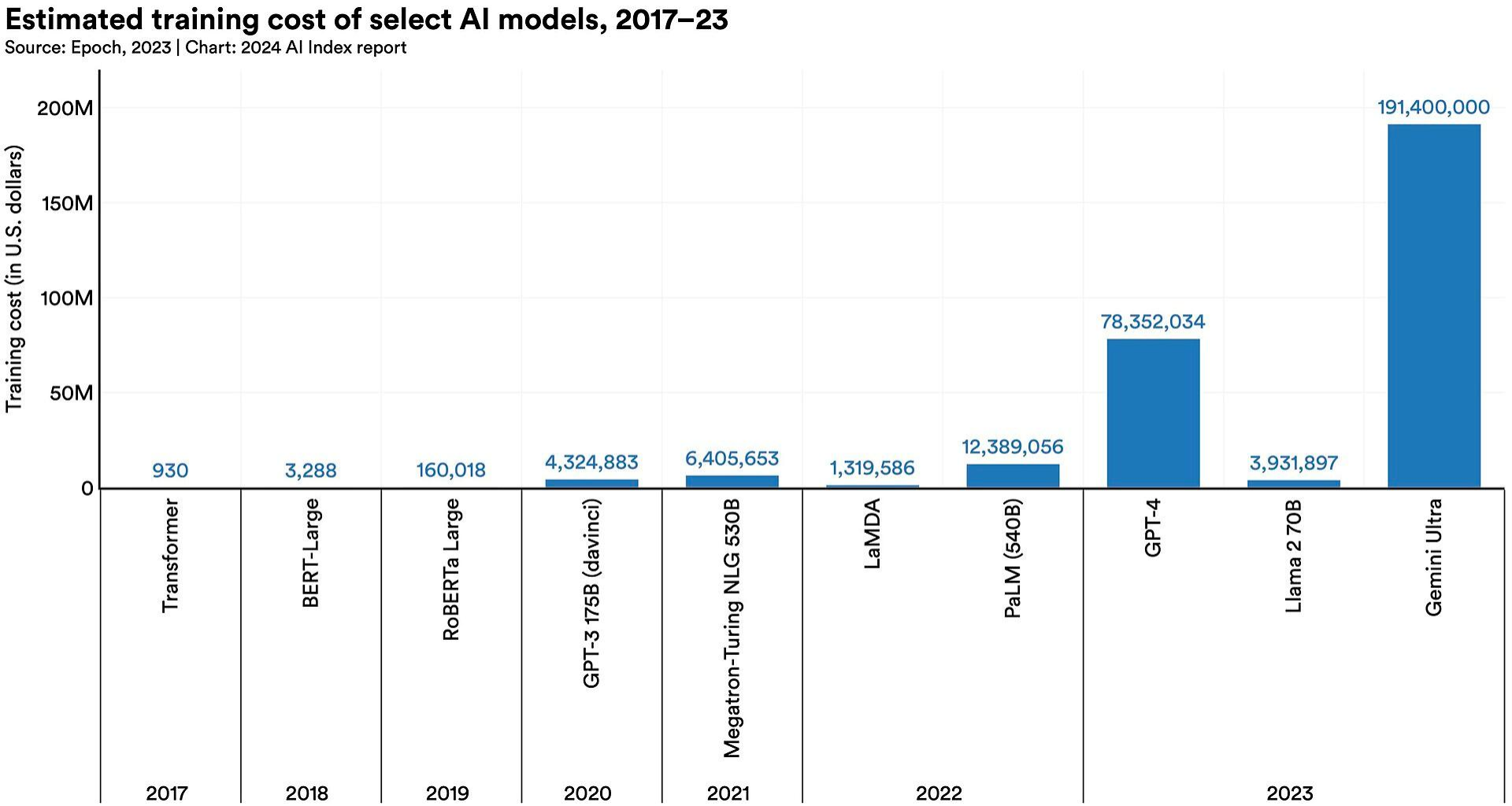
Foundation Models – A New Paradigm for Artificial Intelligence, https://link.springer.com/article/10.1007/s12599-024-00851-0

AI 基礎模型

- **基礎模型**（亦稱大型人工智慧模型）是一種機器學習或深度學習模型，透過在龐大的數據集上進行訓練，使其能夠應用於廣泛的使用場景。例如，大型語言模型等生成式人工智慧應用通常是基礎模型的典型案例。
- **建立**基礎模型通常需要極高的資源投入，最昂貴的模型可能耗費數億美元，用於支付所需的基礎數據和計算資源。相較之下，將現有的基礎模型調整以**適應**特定任務或直接使用，成本要低得多。

https://en.wikipedia.org/wiki/Foundation_model

Investment in computing capabilities to train larger AI models has rapidly increased

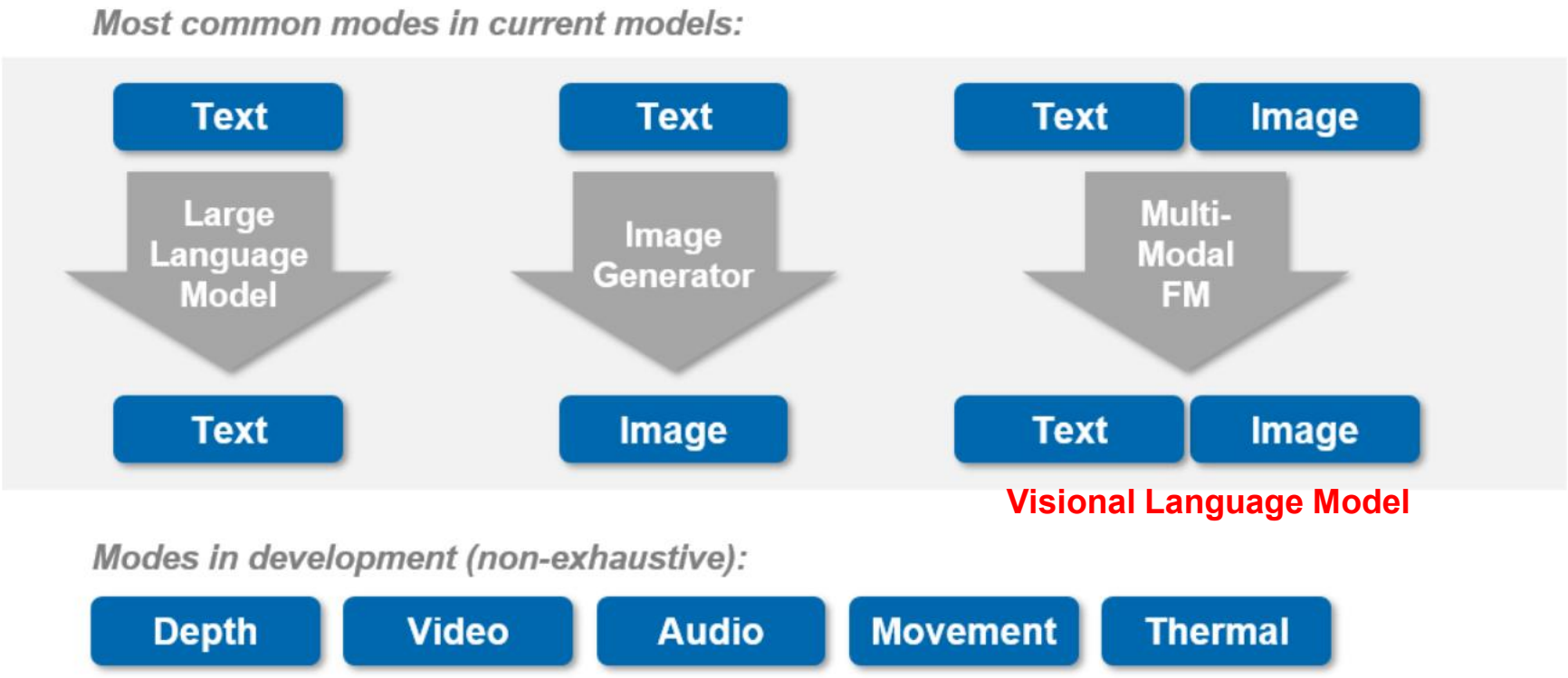


https://aiindex.stanford.edu/wp-content/uploads/2024/04/HAI_AI-Index-Report-2024_Chapter1.pdf

AI 基礎模型 (cont.)

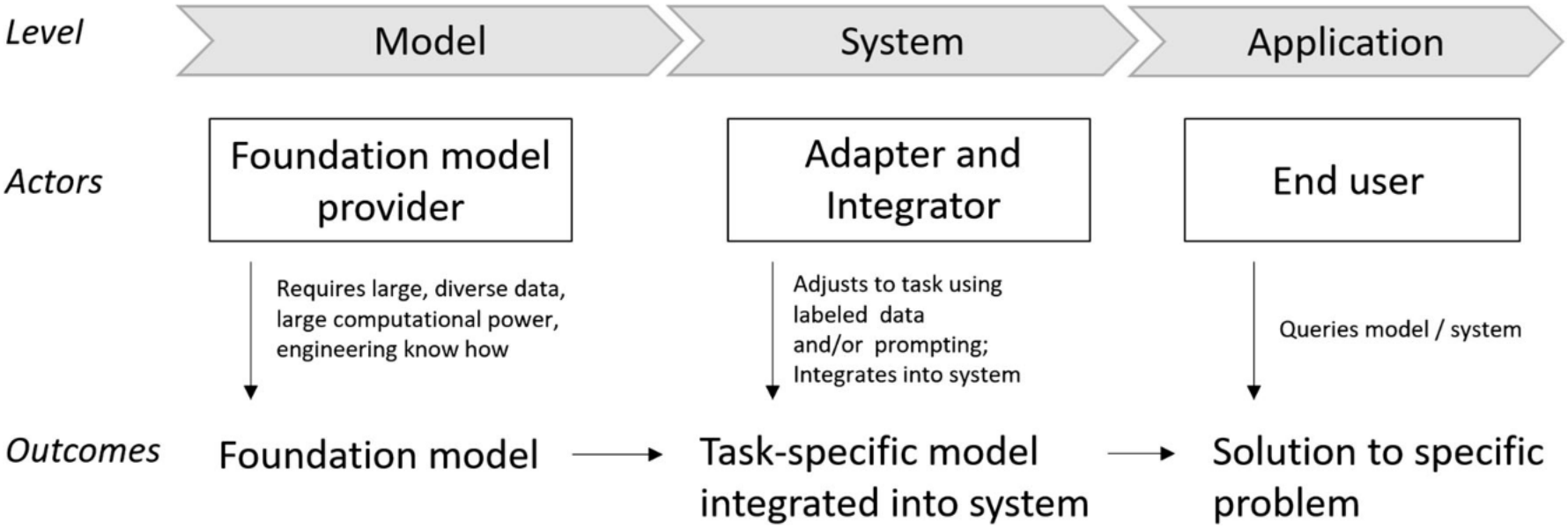
- 基礎模型的早期範例
  - Language models (LMs) : OpenAI's **GPT** series and Google's **BERT**
  - Images : **DALL-E** and Flamingo
  - Music : MusicGen for music
  - Robotic control : RT-2
- 基礎模型也正在被開發應用於天文學、放射學、基因組學、音樂、程式設計、時間序列預測、數學以及化學等領域。

Different Modalities of Foundation Model



AI Foundation Models: Initial Report, https://assets.publishing.service.gov.uk/media/65081d3aa41cc300145612c0/Full_report_.pdf

AI development involving foundation models



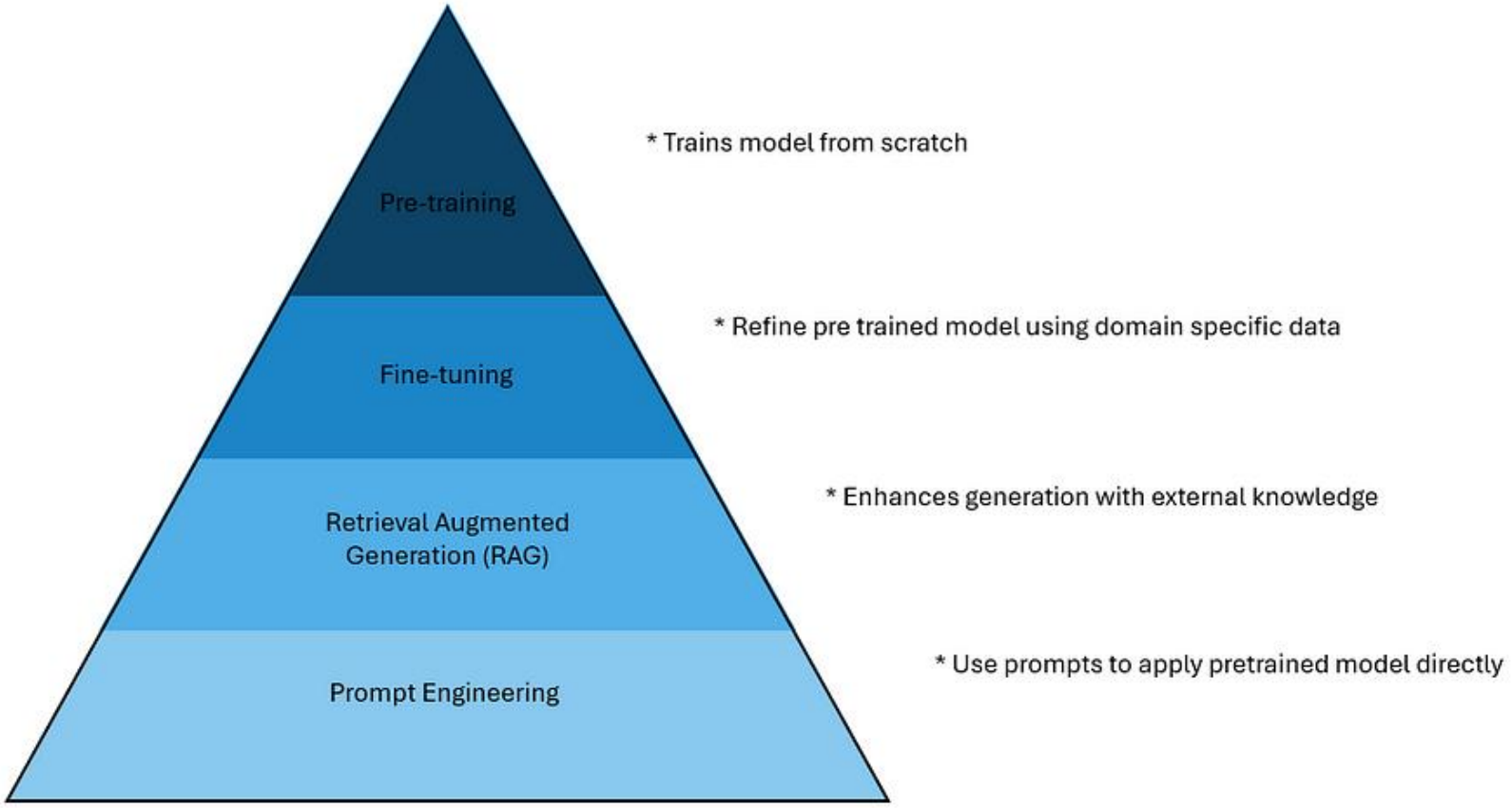
Foundation Models – A New Paradigm for Artificial Intelligence, https://link.springer.com/article/10.1007/s12599-024-00851-0

如何應用 Generative AI model 於特定任務

- 遷移學習（Transfer Learning）
  - 一種機器學習（ML）技術，將從一個任務中學到的知識重新用於相關任務，以提升其效能。
- 檢索增強生成（Retrieval Augmented Generation）
  - 賦予生成式人工智慧模型信息檢索能力的技術。它修改與大型語言模型（LLM）的互動方式，使模型能參考特定文檔集來回應用戶查詢，藉此補充來自模型自身龐大靜態訓練數據的資訊。
- 提示工程（Prompt Engineering）
  - 設計指令結構的過程，使生成式人工智慧（AI）模型能夠解讀並理解該指令。

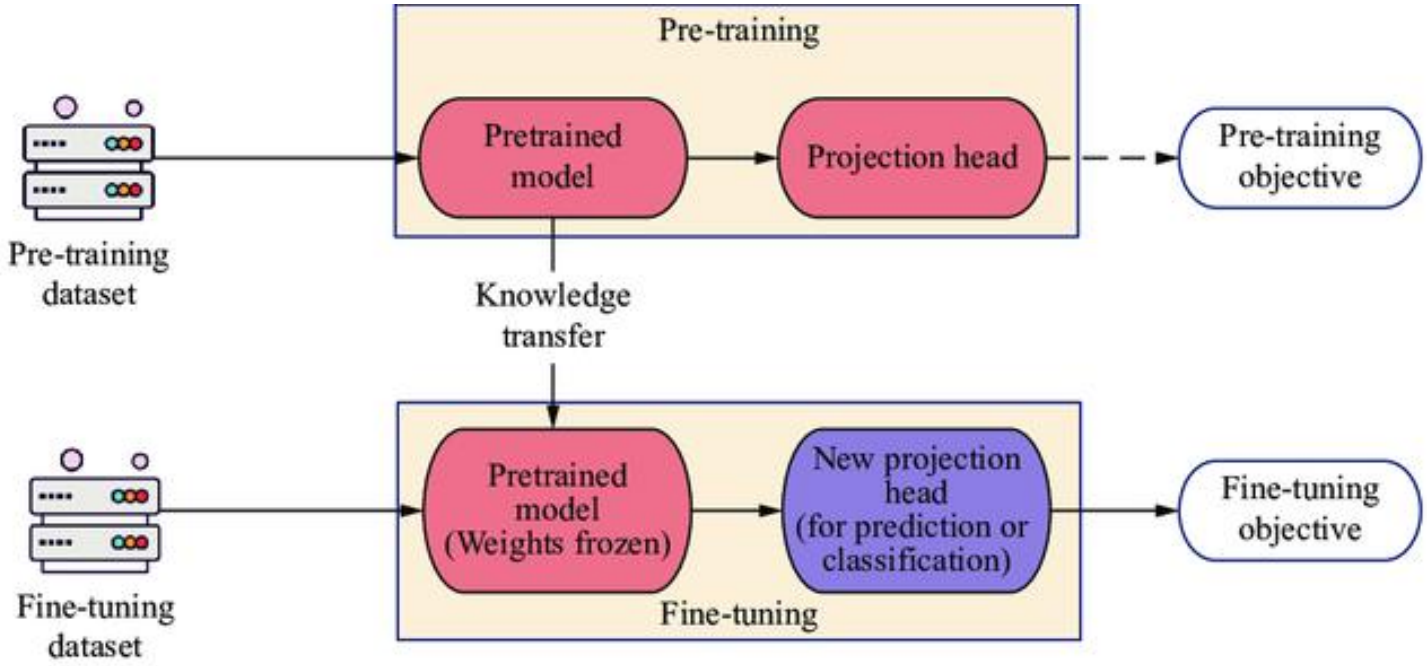
https://en.wikipedia.org/wiki/Transfer_learning
https://en.wikipedia.org/wiki/Retrieval-augmented_generation
https://en.wikipedia.org/wiki/Prompt_engineering

Generative AI 應用技巧位階



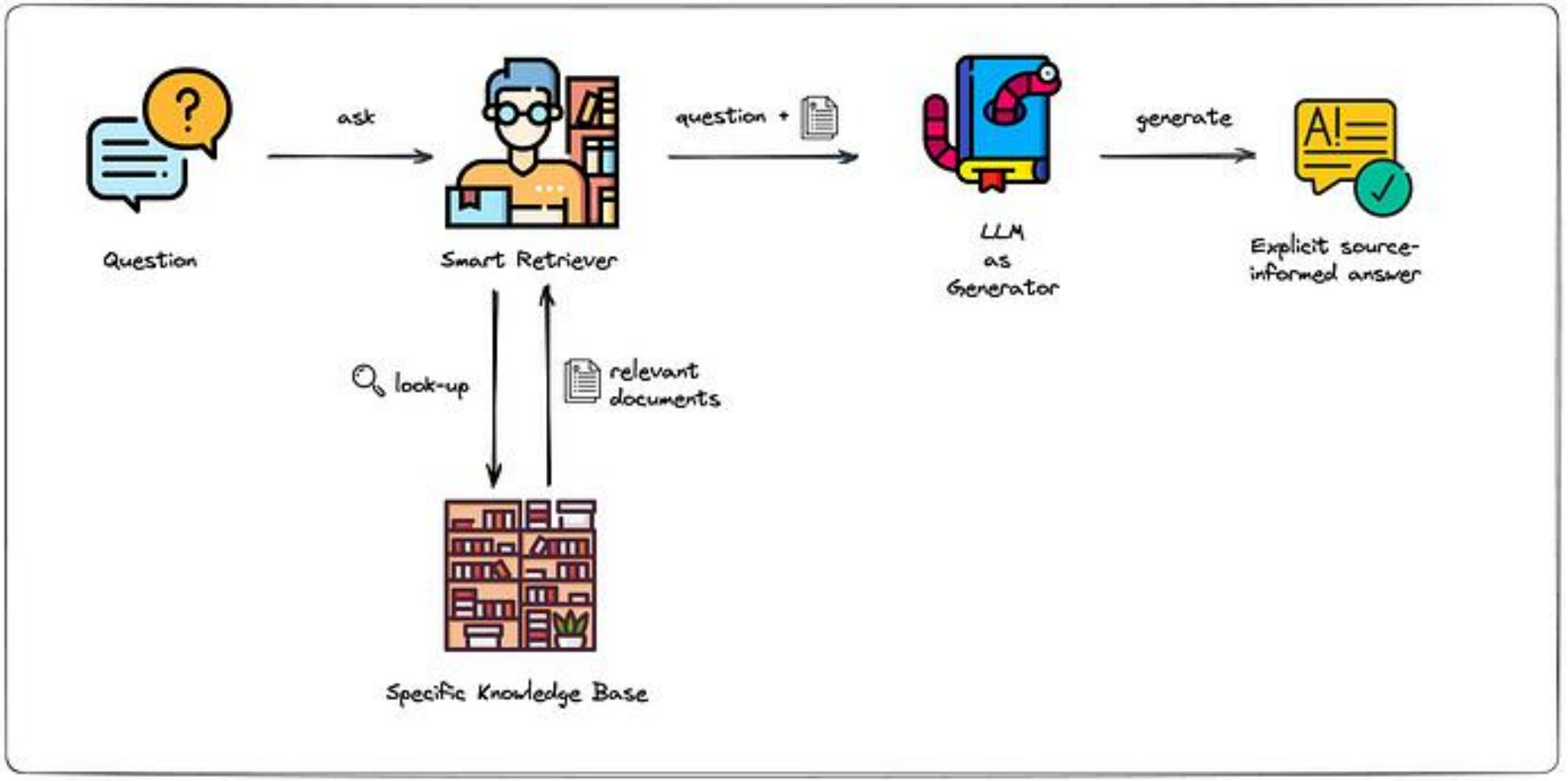
https://medium.com/@jainpalak9509/use-llms-pre-training-fine-tuning-rag-and-prompt-engineering-564d5670f44d

Transfer Learning



https://www.researchgate.net/figure/Illustration-of-pre-training-Pre-training-is-a-part-of-transfer-learning-If-the_fig1_368698963

RAG (Retrieval Augmented Generation)



https://www.ml6.eu/blogpost/leveraging-llms-on-your-domain-specific-knowledge-base

Prompt Engineering Guide (OpenAI, 6 strategies)

1. 撰寫清晰的指令
   - 在查詢中包含細節以獲得更相關的答案
   - 要求模型扮演特定角色
   - 使用分隔符號清楚地區分輸入的不同部分
   - 明確指定完成任務所需的步驟
   - 提供範例
   - 明確說明輸出內容的期望長度
2. 提供參考文本
   - 指示模型根據參考文本回答
   - 要求模型從參考文本中引用資料
3. 將複雜任務分解為簡單子任務
   - 使用意圖分類來識別用戶查詢最相關的指令
   - 對於需要進行超長對話的應用，總結或篩選先前的對話內容
   - 將長文件分段總結，並以遞歸方式構建完整摘要

https://platform.openai.com/docs/guides/prompt-engineering

(OpenAI, 6 strategies)

4. 給模型時間「思考」(使用 Chain of Thought)
   - 指示模型先整理出解決方案，再給出結論
   - 使用內部獨白或一系列查詢隱藏模型的推理過程
   - 問模型是否遺漏了之前答案中的任何內容
5. 使用外部工具 (使用 RAG)
   - 使用內嵌的搜索實現高效的知識檢索
   - 執行程式以進行更準確的計算，或調用外部 API
   - 讓 GPT 利用特定功能
6. 系統化測試變更
   - 根據標準答案評估模型輸出

- https://platform.openai.com/docs/guides/prompt-engineering

提示工程的複雜技巧

- Chain of Thought 思維鏈
  - 根據 Google 的說法，Chain-of-thought（CoT）提示被認為是一種技術，可讓大型語言模型（LLMs）在給出最終答案之前，將問題拆解為一系列的中間步驟。2022 年，Google 還聲稱，使用思路鏈提示可以透過引導模型模仿人類思維的推理過程來回答多步驟的問題，從而提升推理能力。據 Google 和 Amazon 的公告，思路鏈技術假設能幫助大型語言模型克服某些需要邏輯思考與多步驟解決的推理任務的困難，例如算術問題或常識推理問題。

- 提供複雜提示工程技巧的網路資訊
  - https://en.wikipedia.org/wiki/Prompt_engineering
  - https://www.promptingguide.ai/
  - https://learnprompting.org/docs/introduction

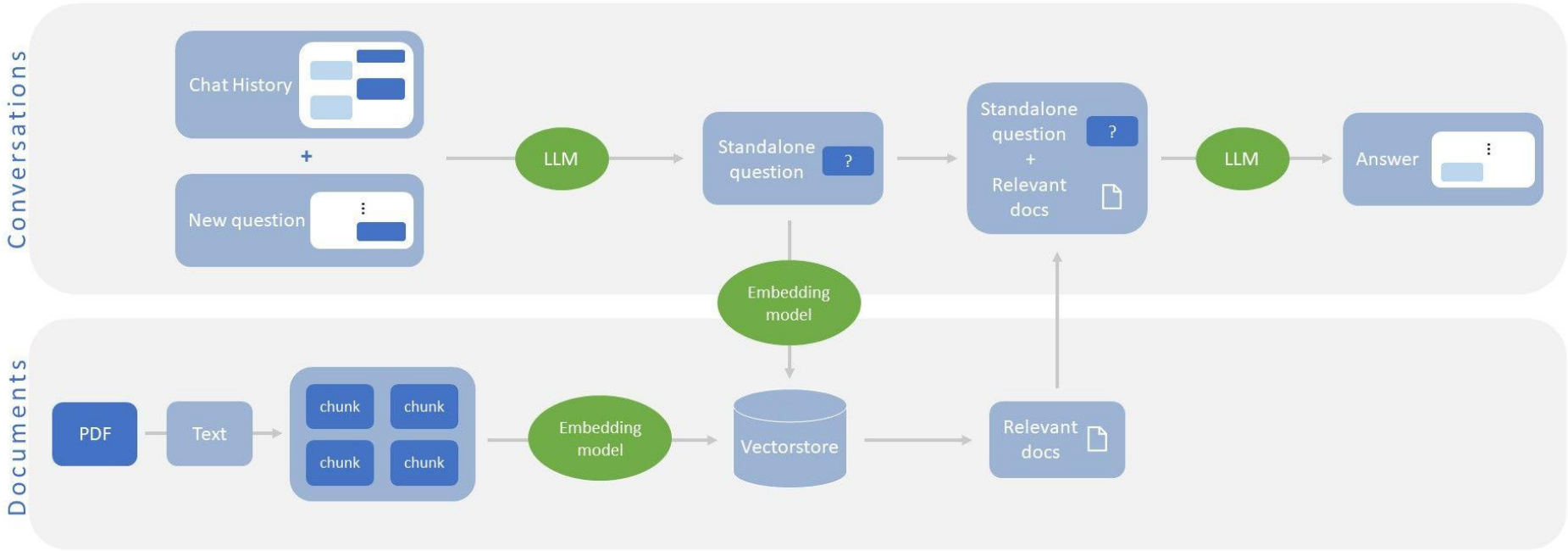
構建和管理與大語言模型(LLMs)互動的應用

- **LangChain**
  - 2022 年 10 月 24 日首次發布。
  - 由 Harrison Chase 創立，最初在其創業公司 Robust Intelligence 工作時開發。
- **MCP** (Model Context Protocol)
  - 2024 年 11 月 25 日由 Anthropic 發布。
  - 由 Anthropic 開發，並已獲得多家 AI 公司（如 OpenAI、Google DeepMind）的支持。
- Google **A2A** (Agent-to-Agent)
  - 2025 年 4 月 9 日由 Google 發布。
  - 由 Google 開發，並已獲得超過 100 家技術公司的支持。
- 其他工具

LangChain

- **用途與功能**：LangChain 是一個開源框架，旨在幫助開發者構建基於大語言模型（如 GPT）的應用。它的核心目的是將不同的語言模型和外部數據源（如API、資料庫、文件系統等）整合在一起，並支持構建多步推理和自定義的應用程式。
- **特點**：
  - **多數據源集成**：LangChain 支援與不同的數據源互動，從簡單的 API 請求到複雜的檢索-增強生成（RAG）功能。
  - **工作流和鏈接管理**：可以將不同的處理步驟鏈接在一起，支持多輪對話、複雜的推理和狀態管理。
  - **自定義組件**：LangChain 提供了很多自定義組件和工具，幫助用戶根據需求設計、擴展和運行應用。
- **與其他兩者的關係**：LangChain 通常用於實現複雜的應用，並且可以與 MCP 和 A2A 結合使用，特別是當涉及到跨多個系統和數據源的集成時。marization, chatbots, and code analysis.

Build a ChatPDF with LangChain

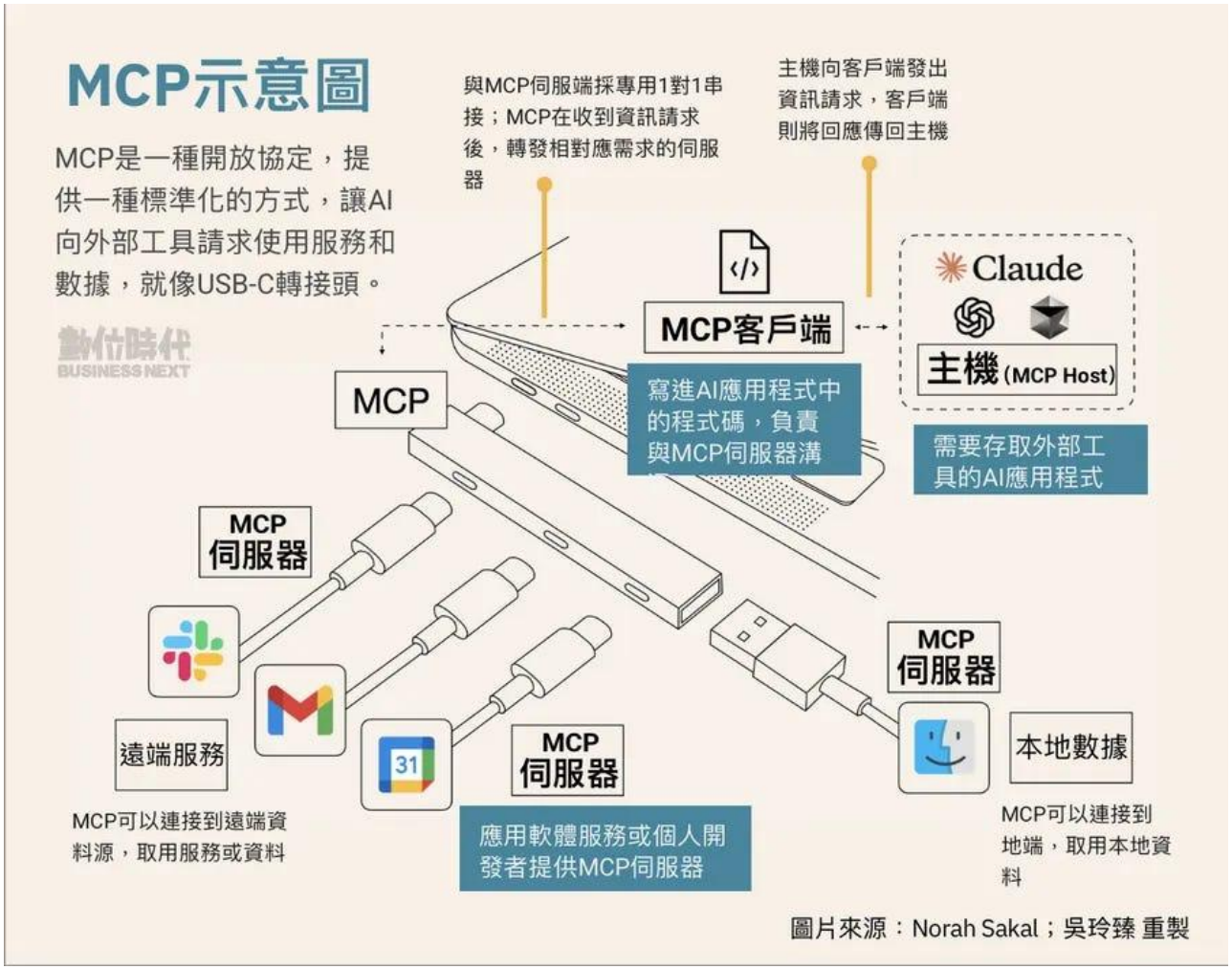


https://edge.aif.tw/express-langchain-chatpdf/

Model Context Protocol (MCP)

- **用途與功能**：MCP 是一種設計規範，旨在標準化模型的上下文管理。它專注於如何有效地傳遞、處理和管理模型的上下文信息，從而提高模型在多輪對話或複雜任務中的一致性和精確性。
- **特點**：
  - 上下文管理：MCP 主要集中在管理模型的上下文 (例如，對話歷史、任務狀態等) ，以便在長期的交互中保持一致性。
  - 可擴展性和靈活性：MCP 可以作為一個通用協議，與不同的模型和框架進行集成，確保它們能夠在多輪對話中維持有效的上下文。
  - 支持跨模型操作：MCP 可用於處理來自多個模型或系統的上下文，並協調它們的交互。
- **與其他兩者的關係**：MCP 主要是針對上下文管理的規範，LangChain 和 Google A2A 可以利用 MCP 來管理多輪對話中的上下文，尤其是當涉及到跨平台和多模型的交互時。

Model Context Protocol



Google A2A (Agent-to-Agent)

- **用途與功能**：Google A2A 是一種專為多個智能代理（agents）之間的協作設計的系統。它旨在促進不同代理之間的有效通信和協作，尤其是在複雜任務和多步驟流程中。A2A 可以將不同的代理系統（例如，語言模型、檢索引擎等）整合在一起，進行協同工作。
- **特點**：
  - 多代理協作：A2A 允許不同的代理彼此協作，傳遞信息、協同推理，並完成複雜任務。
  - 跨系統交互：支持跨多個系統和服務進行交互，使得不同的服務和工具可以共享任務的上下文和狀態。
  - 高效的任務拆解：能夠將大任務拆解為多個子任務，並由不同代理分別處理這些子任務，最終將結果匯總。
- **與其他兩者的關係**：Google A2A 和 LangChain 有很多相似之處，特別是在處理多步推理和多代理協作方面。A2A 可以視為一種更專注於多代理交互的架構，而 LangChain 更強調語言模型的應用和數據源的整合。MCP 在這些系統中通常作為一個上下文管理工具，確保跨代理和跨模型的協同一致性。

差異與範圍對比

工具/功能	**Function Calling**	**LangChain**	**MCP**	**Google A2A**
定義	語言模型調用外部函數或服務	開發框架，將語言模型與外部數據源及服務進行集成	標準化語言模型的上下文管理和傳遞	促進多個智能代理之間的協作與信息交換
範圍	調用單一外部函數或服務	多數據源集成，支持多步推理工作流	保證多輪對話中的一致性，跨系統和多模型協同工作	跨代理協作，將任務拆解並分配給不同代理處理
主要功能	動態調用外部功能，如API查詢、計算等	集成多個數據源、設計複雜的應用和工作流	高效處理上下文，保證多輪對話和跨系統協作的一致性	跨代理任務協作，拆解複雜任務並協同解決
應用場景	資料查詢、外部計算或動作執行	構建多輪對話應用，複雜的資料檢索和生成任務	長期對話、複雜推理和跨系統協作	多代理合作解決複雜任務，協同處理多步推理和任務分配

其他工具 (一)

- **Haystack (由Deepset開發)**
  - **用途**：Haystack 是一個開源框架，主要用於構建從多種數據源（如文檔、資料庫等）中提取和檢索信息的系統。它與多種語言模型（例如BERT, GPT等）集成，支持問答系統、文檔檢索、推理等功能。
  - **特點**：提供強大的檢索-增強生成（RAG）功能，允許用戶基於大量文檔生成回答，並且支持多種檢索後處理和數據源管理功能。
- **OpenAI API + Custom GPTs**
  - **用途**：OpenAI 的 API 讓開發者可以直接構建自己的應用並與 GPT 模型進行交互。Custom GPTs 允許開發者創建定制化的 GPT-3 和 GPT-4 模型，以特定任務為基礎進行調整，並集成外部數據源來提高其能力。
  - **特點**：高度定制化，支持多輪對話管理，並且能夠與外部API進行集成，適合開發個性化的應用。
- **GPT Index (LlamaIndex)**
  - **用途**：LlamaIndex（之前叫GPT Index）是一個專為大型語言模型（如 GPT）設計的開源框架，幫助處理和管理大量文檔資料。它特別適合處理知識庫或文檔密集型應用，可以在這些數據源上建立索引並使用生成模型回答問題。
  - **特點**：可以與多個數據源（例如SQL、CSV、PDF）無縫集成，支持即時查詢和生成答案。

其他工具 (二)

- **Semantic Kernel (由Microsoft開發)**
  - **用途**：Semantic Kernel 是一個由 Microsoft 開發的開源框架，允許開發者輕鬆創建和組織基於語言模型的工作流。它旨在幫助開發者構建更高效的自動化應用，並集成外部工具和數據源。
  - **特點**：支持將 AI、知識庫、API 以及其他外部工具（例如 SQL、Power Automate 等）整合在一起。
- **BotPress**
  - **用途**：BotPress 是一個開源的聊天機器人框架，支持集成大語言模型（如 GPT 系列），並提供管理對話流、語意理解和上下文跟蹤的功能。
  - **特點**：強大的圖形界面設計，易於構建多輪對話和處理複雜的用戶查詢，適用於聊天機器人和虛擬助手的構建。
- **Rasa**
  - **用途**：Rasa 是一個開源框架，專為構建智能對話體驗而設計。它支持自然語言理解（NLU）和對話管理，並可以與外部數據源和API進行集成。
  - **特點**：Rasa 的核心特點是其可擴展性和靈活性，適用於複雜的對話流設計，並支持自定義的知識庫集成。
- **Weaviate**
  - **用途**：Weaviate 是一個開源的向量數據庫，用於構建 AI 驅動的應用。它專門支持基於語言模型的檢索和生成任務，將文本和多媒體數據轉換為向量進行高效檢索。
  - **特點**：可以輕鬆集成大語言模型和向量檢索，適用於需要處理和檢索大量文本的場景。

AI 工具

The best AI productivity tools in 2025

- Chatbots (ChatGPT, Claude, Meta AI, Zapier Central)
- Search engines (Perplexity, Google AI Overviews, Arc Search)
- Content creation (Jasper, Anyword, Writer)
- Grammar checkers and rewording tools (Grammarly, Wordtune, ProWritingAid)
- Video creation and editing (Runway, Descript, Wondershare Filmora)
- Image generation (DALL·E 3, Midjourney, Ideogram)
- Social media management (FeedHive, Vista Social, Buffer)
- Voice and music generation (ElevenLabs, Suno, AIVA)
- Knowledge management and AI grounding (Mem, Notion AI Q&A, Personal AI)
- Task and project management (Asana, Any.do, BeeDone)
- Transcription and meeting assistants (Fireflies, Avoma, tl;dv)
- Scheduling (Reclaim, Clockwise, Motion)
- Email (Shortwave, Microsoft Copilot Pro for Outlook, Gemini for Gmail)
- Slide decks and presentations (Tome, Beautiful.ai, Slidesgo)
- Resume builders (Teal, Enhancv, Kickresume)
- Automation (Zapier)
- Other AI productivity tools

https://zapier.com/blog/best-ai-productivity-tools/

The 40 Best AI Tools for 2025 (Tried and Tested)

The Best AI Tools by Category

- Chatbots: ChatGPT, Claude
- Video Generation and Editing: Synthesia, Runway
- Writing: Rytr, Sudowrite
- Grammar and Writing Improvement: Grammarly, Wordtune
- Search Engines: Perplexity, ChatGPT search
- Social Media Management: Vista Social, FeedHive
- Image Generation: Midjourney, DALL·E 3
- Graphic Design: Canva Magic Studio, Looka
- App Builders: Bubble, Bolt
- Project Management: Asana, ClickUp
- Transcription and Meeting Assistants: tl;dv, Nyota
- Scheduling: Reclaim, Clockwise
- Customer Service: Tidio AI, Hiver
- Recruitment: Textio, CVViZ
- Knowledge Management: Notion AI Q&A, Guru
- Email: SaneBox, Shortwave
- Presentations: Gamma, Presentations.ai
- Resume Builders: Teal, Kickresume
- Voice Generation: ElevenLabs, Murf
- Music Generation: Suno, Udio

https://www.synthesia.io/post/ai-tools

Other AI Tools

- Data analyze
  - https://chatgpt.com/g/g-HMNcP6w7d-data-analyst
  - https://julius.ai/
  - https://chatgpt.com/g/g-lq0OG5y0A-big-data-insight-engine
- Coding
  - https://codesubmit.io/blog/ai-code-tools/
  - https://www.geeksforgeeks.org/ai-coding-assistant-tools/
  - https://blog.fabrichq.ai/latest-ai-code-generators-a-comprehensive-list-b7ce1a461fac
  - https://research.aimultiple.com/ai-coding-benchmark/

OpenAI ChatGPT 的文件

- Introduction
  - https://openai.com/index/chatgpt/
- Documentation
  - https://platform.openai.com/docs/overview
- Prompt Examples
  - https://platform.openai.com/docs/examples
  - https://team-gpt.com/blog/chatgpt-examples/

LLM 效能排名

- https://livebench.ai/#
- https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard
- https://www.vellum.ai/llm-leaderboard
- https://klu.ai/llm-leaderboard

生成式 AI 的應用層次

- 生成式 AI 在應用中扮演的各種角色，對比於從簡易到深入的難易層次，我們可以將這些角色劃分為以下幾個層次：
  1. **基礎層次** 主要包括簡單的文本生成、翻譯和自動化審查等任務。
  2. **中級層次** 涉及多輪對話、情感分析和較為創意的生成。
  3. **進階層次** 包括專業領域應用、高級創意生成、編程協助和個性化推薦等。
  4. **高級層次** 涉及跨模態生成、創新解決方案、推理決策等複雜的生成任務。
  5. **專家層次** 則關乎模型微調、自我學習和特定領域的高級定制應用。

1. 基礎層次：簡單生成與內容處理

- **內容生成 (Content Generation)**：
  - 簡單的文本創作（如生成簡短文章、標題、社交媒體貼文等）。
  - 基本的圖片生成或音樂創作，對於新手來說，這些生成可以基於簡單的提示生成常規內容。
- **語言翻譯與轉換 (Language Translation & Transformation)**：
  - 基本的語言翻譯，如將短句或單一語言內容轉換為另一種語言。
  - 基本的語調或風格轉換（如將正式文本轉為非正式文本）。
- **自動化內容審查 (Content Moderation)**：
  - 自動檢測不當內容或敏感詞，並進行過濾，這是相對簡單的自動化任務。

2. 中級層次：情境理解與多輪對話

- **智能對話 (Intelligent Dialogue)**：
  - 基於上下文進行簡單的多輪對話，適用於基本的客服機器人或虛擬助手。
- **知識提取與組織 (Knowledge Extraction & Organization)**：
  - 從文本中提取具體的資訊（如日期、地點、人名等），並對其進行基本的分類和整理。
- **創意生成 (Innovation & Creativity)**：
  - 根據提供的提示生成具創意的簡單內容，適用於非專業創意工作者或初學者。
- **情感分析 (Sentiment Analysis)**：
  - 檢測文本中的情感或情緒，例如分析社交媒體的情感趨勢。

3. 進階層次：專業應用與深度創作

- **自動化內容審查 (Content Moderation)**：
  - 更高級的內容審查，分析複雜的上下文來判定是否存在敏感、違規內容，這通常需要更精確的情境理解。
- **語調與風格轉換 (Tone & Style Transfer)**：
  - 根據指定的語氣或風格生成文本，這不僅是語言轉換，還要求創建符合特定風格的內容。
- **創意生成 (Creative Generation)**：
  - 高級的創意寫作或藝術創作，如小說創作、劇本寫作、創意廣告文案生成等，這涉及較為複雜的創作過程。
- **自動化編程與代碼生成 (Automated Programming & Code Generation)**：
  - 生成程式碼或協助進行代碼優化，這需要對代碼結構和語言有更深入的理解。
- **個性化推薦系統 (Personalized Recommendation Systems)**：
  - 根據用戶的需求和行為生成動態推薦，這需要更高級的數據分析能力。

4. 高級層次：跨模態應用與深度學習

- **跨模態生成 (Cross-modal Generation)**：
  - 從文本生成圖像、音樂或其他類型的多媒體內容，或者反向地，從圖像或音頻生成描述性文本，這涉及到多個模態的深度整合。
- **創新與創造 (Innovation & Problem Solving)**：
  - 根據特定問題提出創新的解決方案，這不僅是生成創意內容，還涉及到問題解析和創新思維的應用。
- **自動化決策與推理 (Automated Decision-Making & Reasoning)**：
  - 基於複雜的資料進行邏輯推理和決策支持，這是深度推理和高級推理任務，通常需要高級模型和精確的輸入數據。

5. 專家層次：模型微調與定制化應用

- **專業應用定制 (Specialized Application Customization)**：
  - 對生成式 AI 模型進行微調，使其專門適應某一領域（如醫療、法律、金融等）的需求，並生成專業領域的高質量內容。
- **自我學習與增強學習 (Self-learning & Reinforcement Learning)**：
  - 模型能夠根據反饋進行自我學習與優化，並根據環境變化自我調整，這需要極高的技術層次。

LLM/生成式 AI 使用上應注意的極限

1. 準確性與事實性問題
   - 錯誤生成（幻覺）“hallucination”
   - 無法辨識真偽
2. 上下文理解的局限
   - 短期記憶與長距依賴困難
   - 無法深入推理或理解隱含意圖
3. 創意與創新的局限
   - 缺乏真正的創造性
   - 結果重複，缺乏多樣性
4. 倫理與偏見的問題
   - 數據偏見與偏向性
   - 道德與倫理挑戰
5. 語境適應的局限
   - 對特定領域知識的缺乏
   - 無法處理多模態數據
6. 生成內容的可控性與方向性
   - 難以完全控制生成結果
   - 不符合用戶期望
7. 語言理解的普遍性與局限
   - 文化和語境差異
   - 理解深度的限制
8. 資源消耗與效率
   - 計算成本高昂
   - 快速反應有困難

圖書館相關產業的AI風潮

- Clarivate AI Product
- AI 元數據助理
  - https://clarivate.com/zh/news/ai-metadata-assistant-preview-available-for-all-alma-customers/
- Web of Science Research Assistant
  - https://clarivate.com/academia-government/zh/blog/wosra/

感謝您的聆聽！