

# **神經與行為模型建構 (Neural & Behavioral Modeling)**

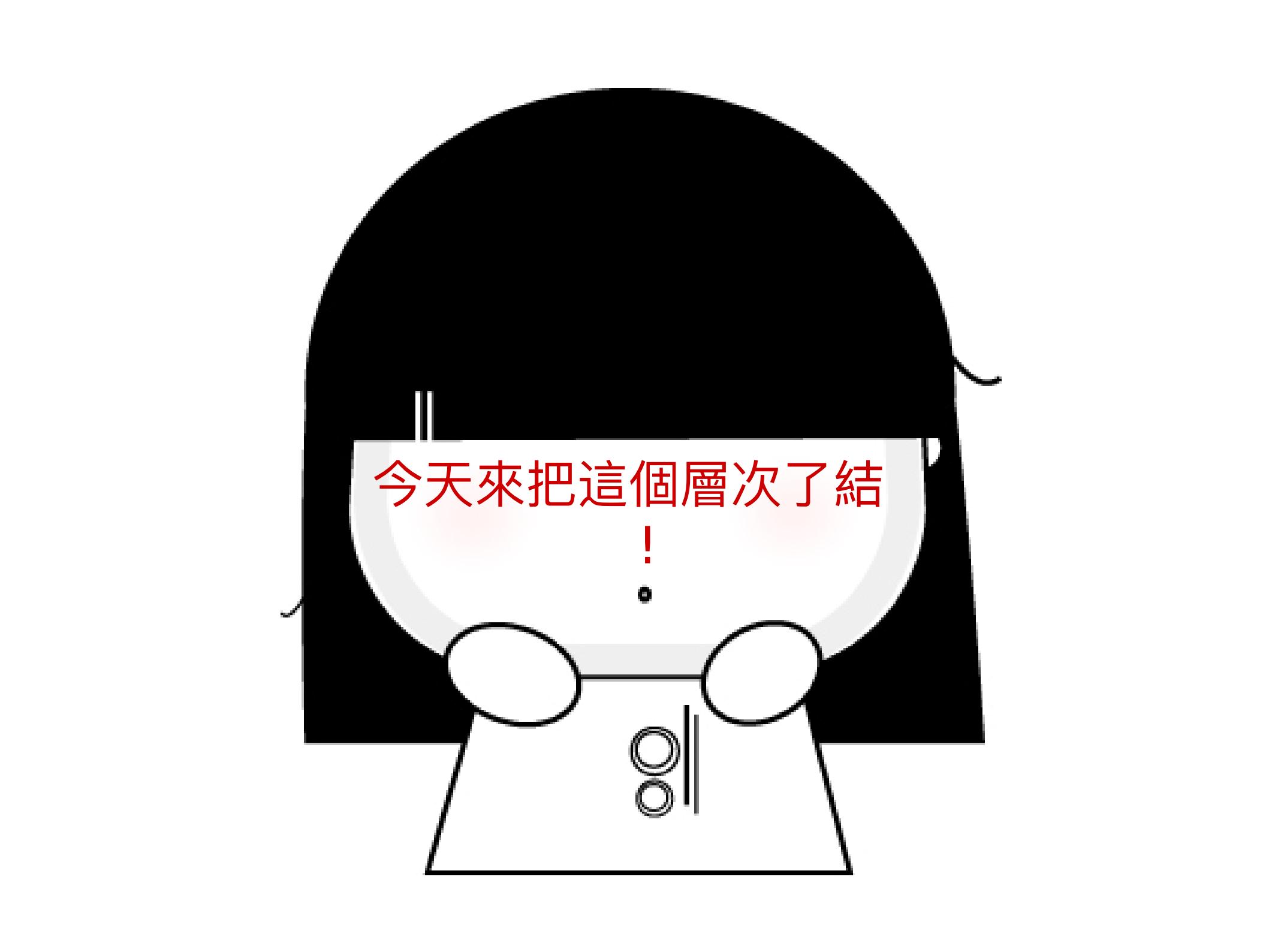
課號：Psy5352

識別碼：227U2810

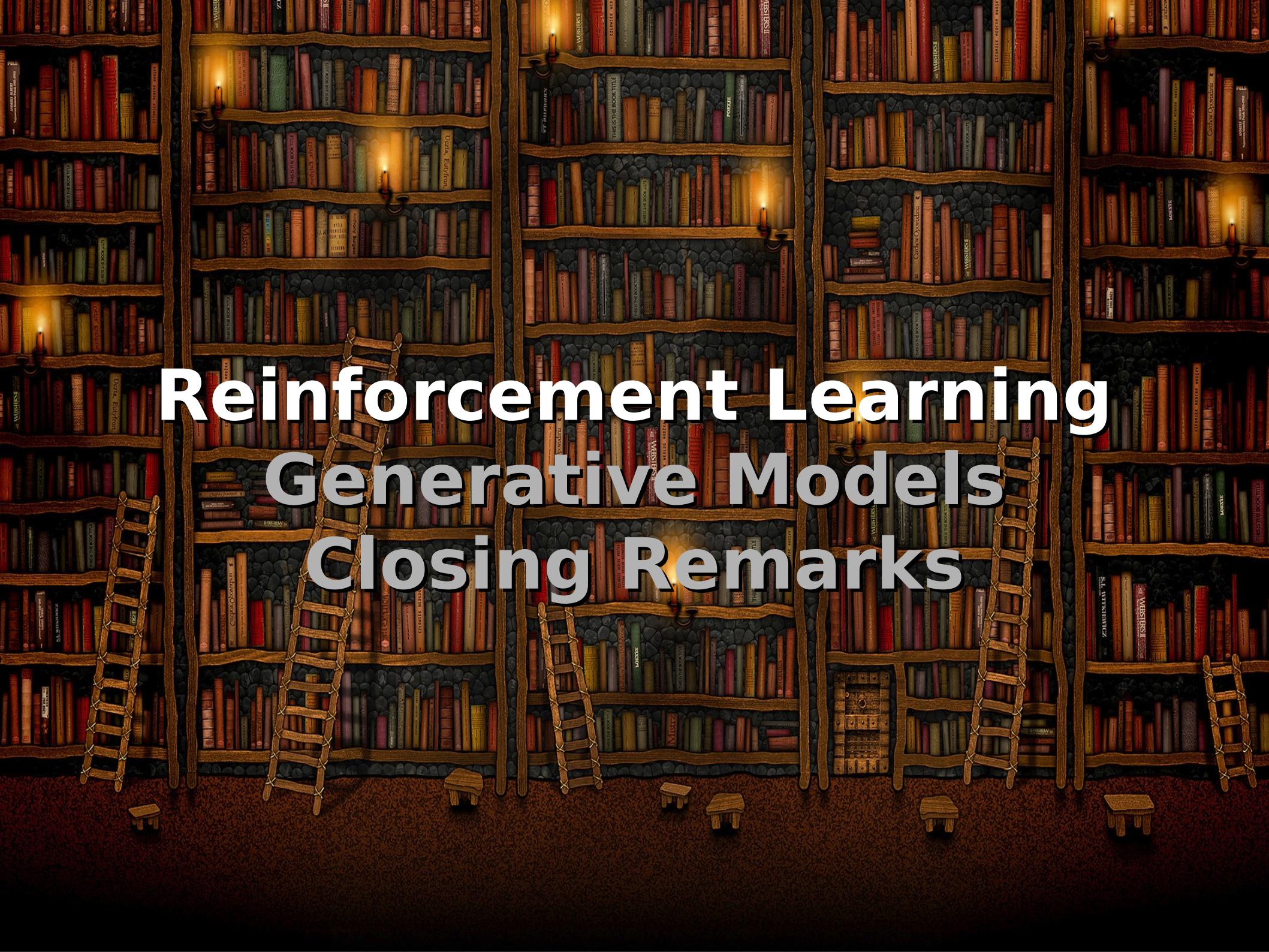
教室：普 101

時間：— 234





今天來把這個層次了結  
！

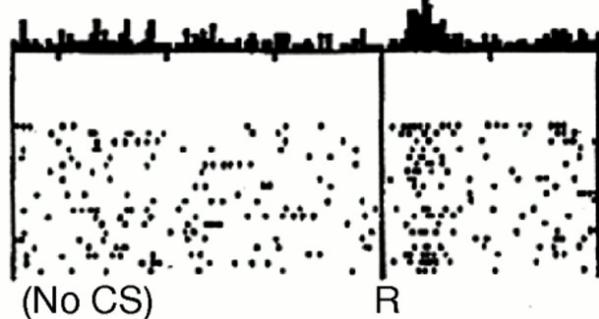


# Reinforcement Learning Generative Models Closing Remarks

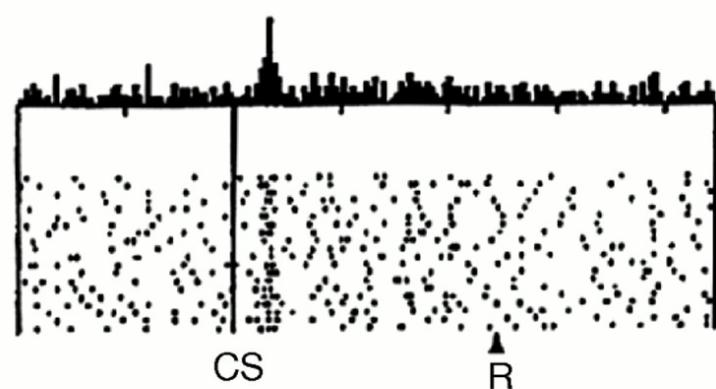
# 獎賞與多巴胺 (Dopamine)

Do dopamine neurons report an error  
in the prediction of reward?

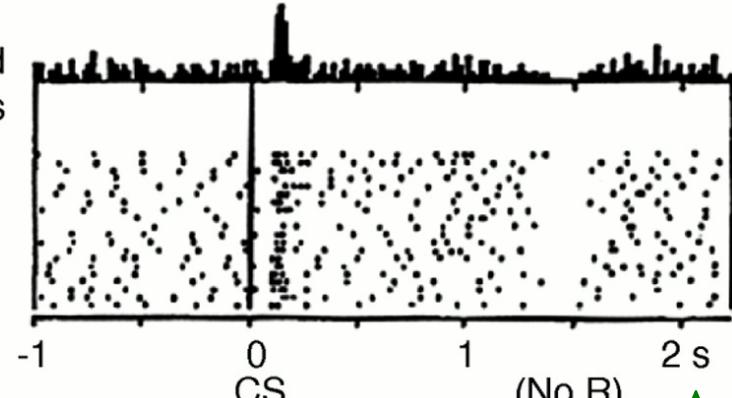
No prediction  
Reward occurs



Reward predicted  
Reward occurs



Reward predicted  
No reward occurs



古典制約學習  
可看成是學 S-S 連結  
或是刺激替代

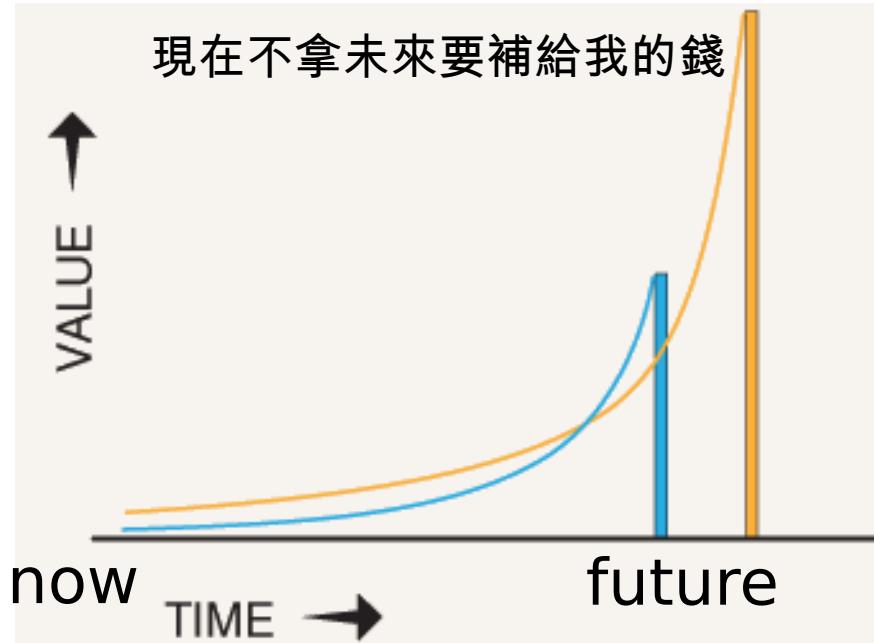
注意左圖神經元反應的  
是獎賞的預測錯誤  
而非獎賞本身

DA dip =  $R - \hat{R}$   
而  $r$  的預測可透過  
temporal  $\delta$  rule 學習

$$R^{\hat{t}+1} = \sum_i W_i^t S_i^t$$
$$\Delta W_i = S_i^t \delta^{t+1} = S_i^t (R^{t+1} - \hat{R}^{t+1})$$

# 更精確的獎賞估計應考慮長遠的未來

沒帶來立即獎賞的 CS 其實預測之後的各種獎賞



獎賞的時間離當下  
愈遠價值愈低  $V^{t+1} = R^{t+1} + \gamma (R^{t+2} + \gamma R^{t+3} + \dots) = R^{t+1} + \gamma V^{t+2}$   
(temporal discounting)

學習過程：讓  $\delta V \equiv R^{t+1} + \gamma \hat{V}^{t+2}(S^{t+1}) - \hat{V}^{t+1}(S^t) \rightarrow 0$

$$\hat{V}^{t+1} = \sum_i W_i^t S_i^t$$

$$\Delta W_i = S_i^t (V^{t+1} - \hat{V}^{t+1}) = S_i^t \delta V^{t+1}$$

但  $V$  要考慮所有時間的  $r$ :

$$V^{t+1} = R^{t+1} + \gamma R^{t+2} + \gamma^2 R^{t+3} \dots$$

若  $\gamma=0$  回到上頁公式  
且上式可改寫成遞迴形式：

$$V^{t+1} = R^{t+1} + \gamma (R^{t+2} + \gamma R^{t+3} + \dots) = R^{t+1} + \gamma V^{t+2}$$

預測  $V$  的學習目標：等式成立

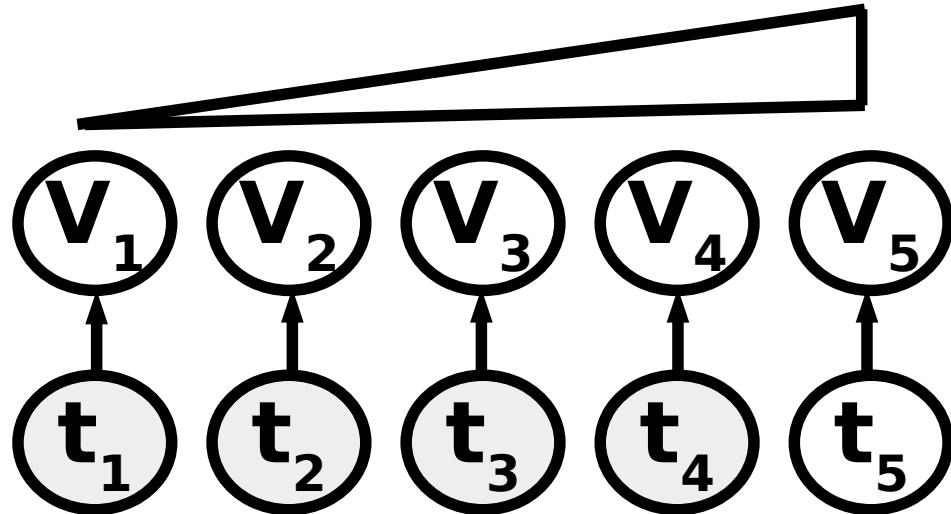
$$\hat{V}^{t+1}(S^t) \leftarrow R^{t+1} + \gamma \hat{V}^{t+2}(S^{t+1})$$

# 古典制約的 TD Learning

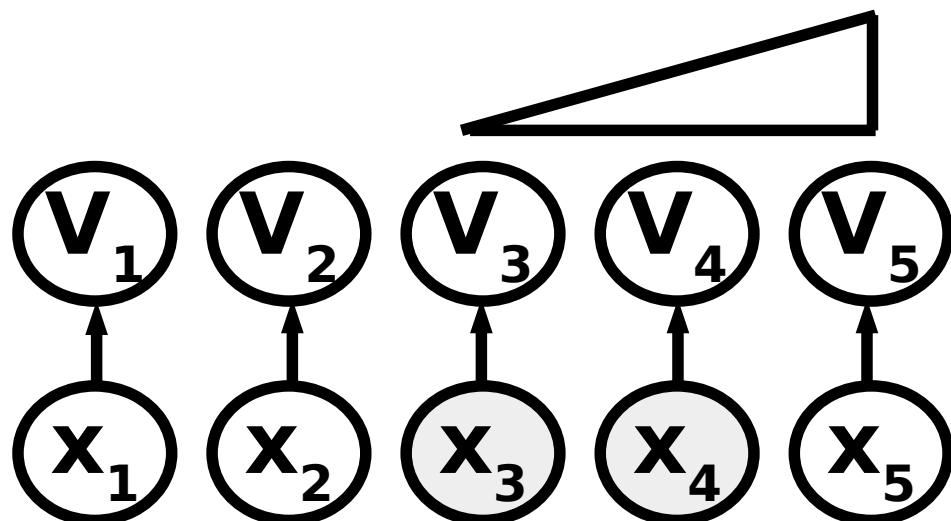
用現在的  $R_t$  來調上一個時間點的  $V(S_{t-1})$

$$\text{TD Learning} \quad V(S_t) \leftarrow V(S_t) + \alpha [R_{t+1} + \gamma V(S_{t+1}) - V(S_t)] \quad S_t$$

————— Previous estimate ————— Reward t+1 ————— Discounted value on the next step ————— TD Target



若  $S=t$  則  $V(t)$  變為無邊界的時間折價函數  
(如無刺激出現但固定 5 秒後出現酬賞)



若  $S=x(t)$  則  $V(x)$  變為有邊界的時間折價函數  
(如有刺激出現，且之後固定 2 秒後出現酬賞)

# 操作制約的 Q-Learning (1/2)

Q(s,a) 是狀態 s 與行動 a 的函數

$$Q_{t+1}(s_t, a_t) = \underbrace{Q_t(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha_t(s_t, a_t)}_{\text{learning rate}} \cdot \left( \underbrace{R_{t+1} + \gamma \max_a Q_t(s_{t+1}, a)}_{\substack{\text{learned value} \\ \text{reward} \\ \text{discount factor} \\ \text{estimate of optimal future value}}} - \underbrace{Q_t(s_t, a_t)}_{\text{old value}} \right)$$

在 t 時要選什麼動作變成下一個狀態要根據既定 policy  
計算上常離散化 s 與 a 來做成一個 Q(s,a) 的查找表

Game Board:



Current state (s):

0 0 0  
0 1 0

Q Table:

$\gamma = 0.95$

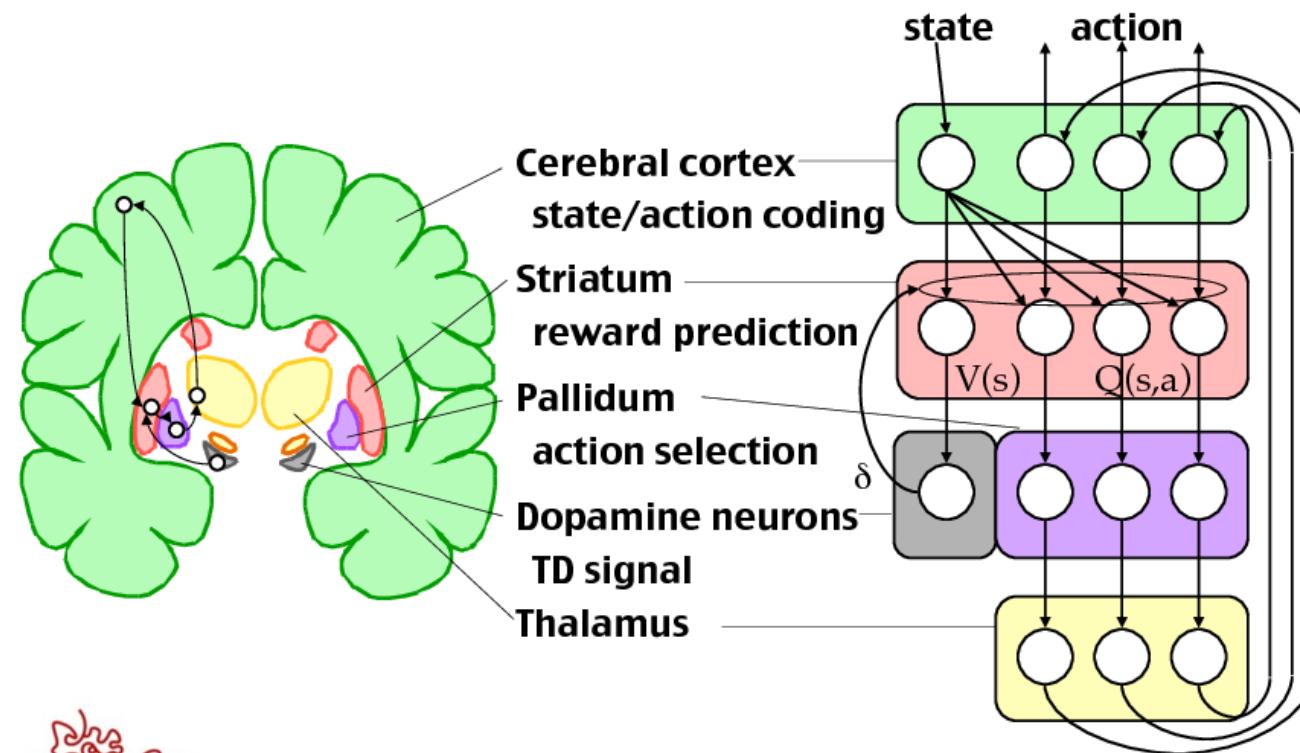
	0 0 0 1 0 0	0 0 0 0 1 0	0 0 0 0 0 1	1 0 0 0 0 0	0 1 0 0 0 0	0 0 1 0 0 0
↑	0.2	0.3	1.0	-0.22	-0.3	0.0
↓	-0.5	-0.4	-0.2	-0.04	-0.02	0.0
→	0.21	0.4	-0.3	0.5	1.0	0.0
←	-0.6	-0.1	-0.1	-0.31	-0.01	0.0

# 操作制約的 Q-Learning (2/2)

Q(s,a) 是狀態 s 與行動 a 的函數

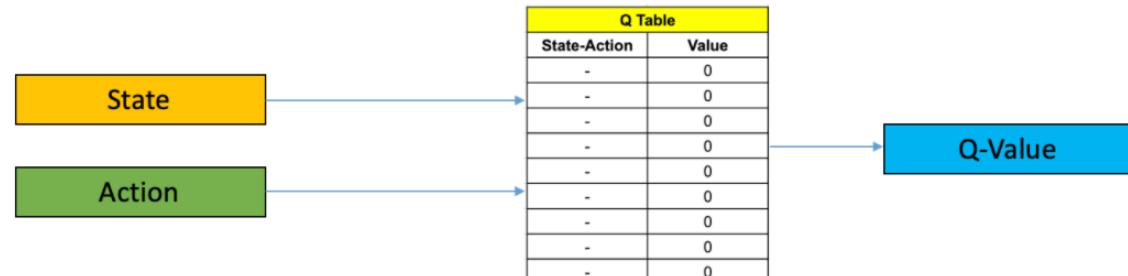
$$Q_{t+1}(s_t, a_t) = \underbrace{Q_t(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha_t(s_t, a_t)}_{\text{learning rate}} \cdot \left( \underbrace{R_{t+1} + \gamma \max_a Q_t(s_{t+1}, a)}_{\substack{\text{learned value} \\ \text{reward} \\ \text{discount factor} \\ \text{estimate of optimal future value}}} - \underbrace{Q_t(s_t, a_t)}_{\text{old value}} \right)$$

在 t 時要選什麼動作變成下一個狀態要根據既定 policy  
計算上常離散化 s 與 a 來做成一個 Q(s,a) 的查找表

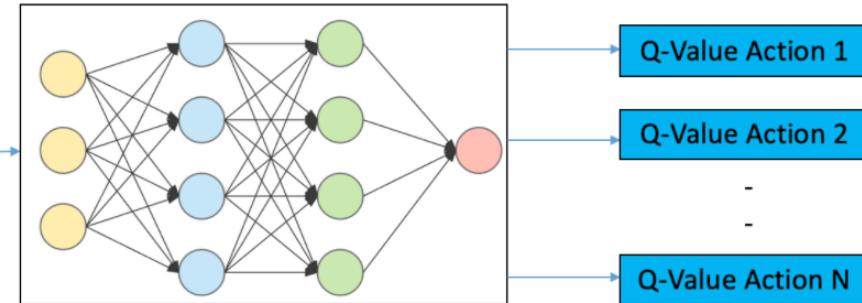


# Deep Q Network (1/5)

2015 年



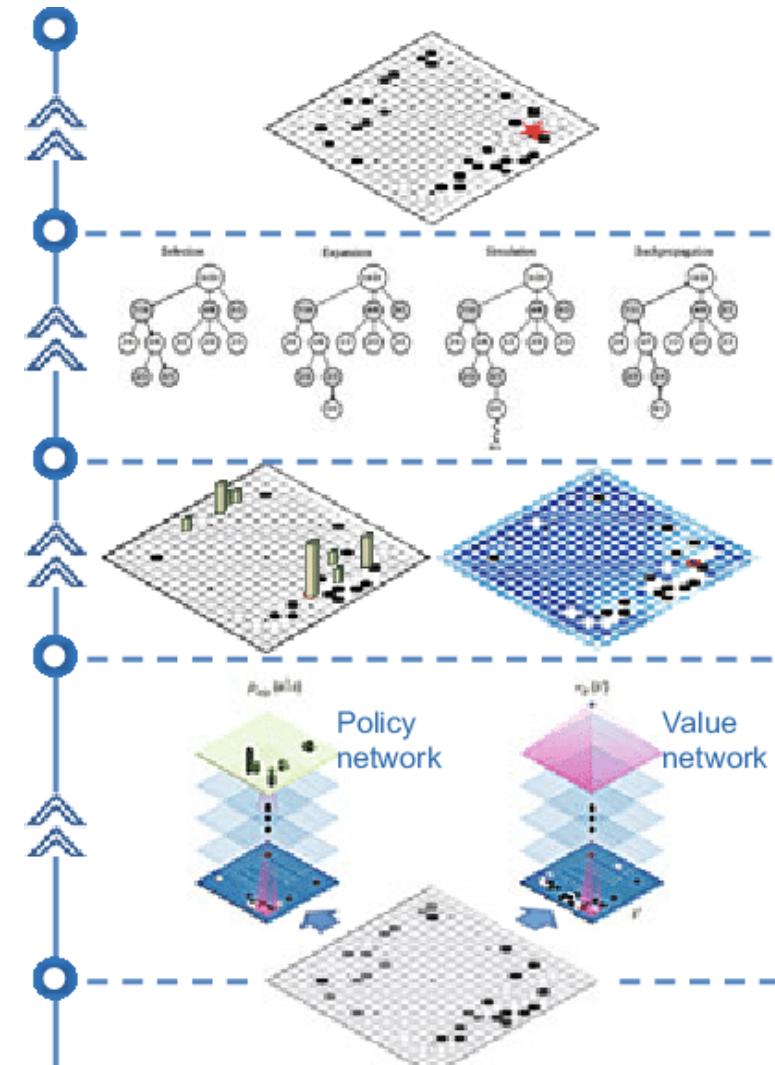
Q Learning



Deep Q Learning

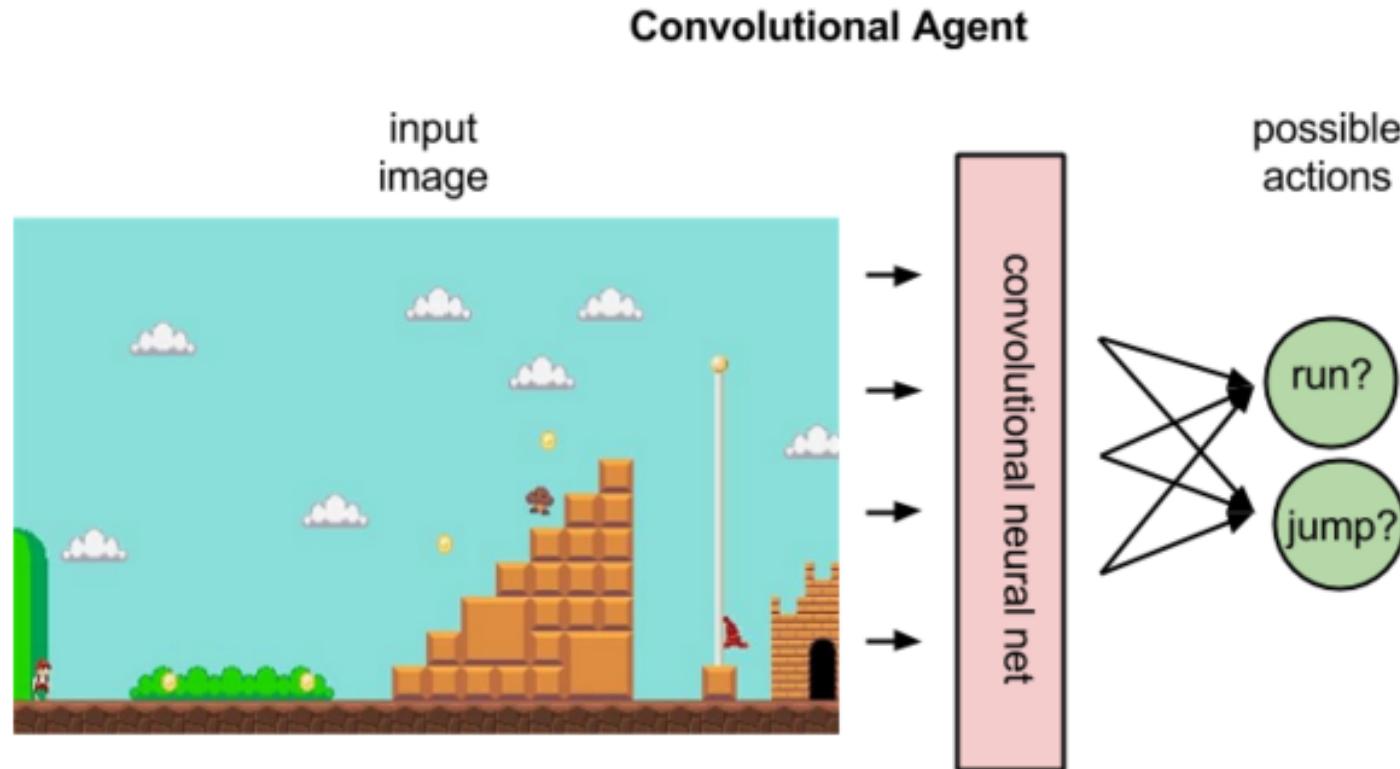
用一 NN 來逼近 Q Table  
(因為 state 不離散)

2016 年



# Deep Q Network (2/5)

學習上的問題：貴近賤遠 / 學近忘遠

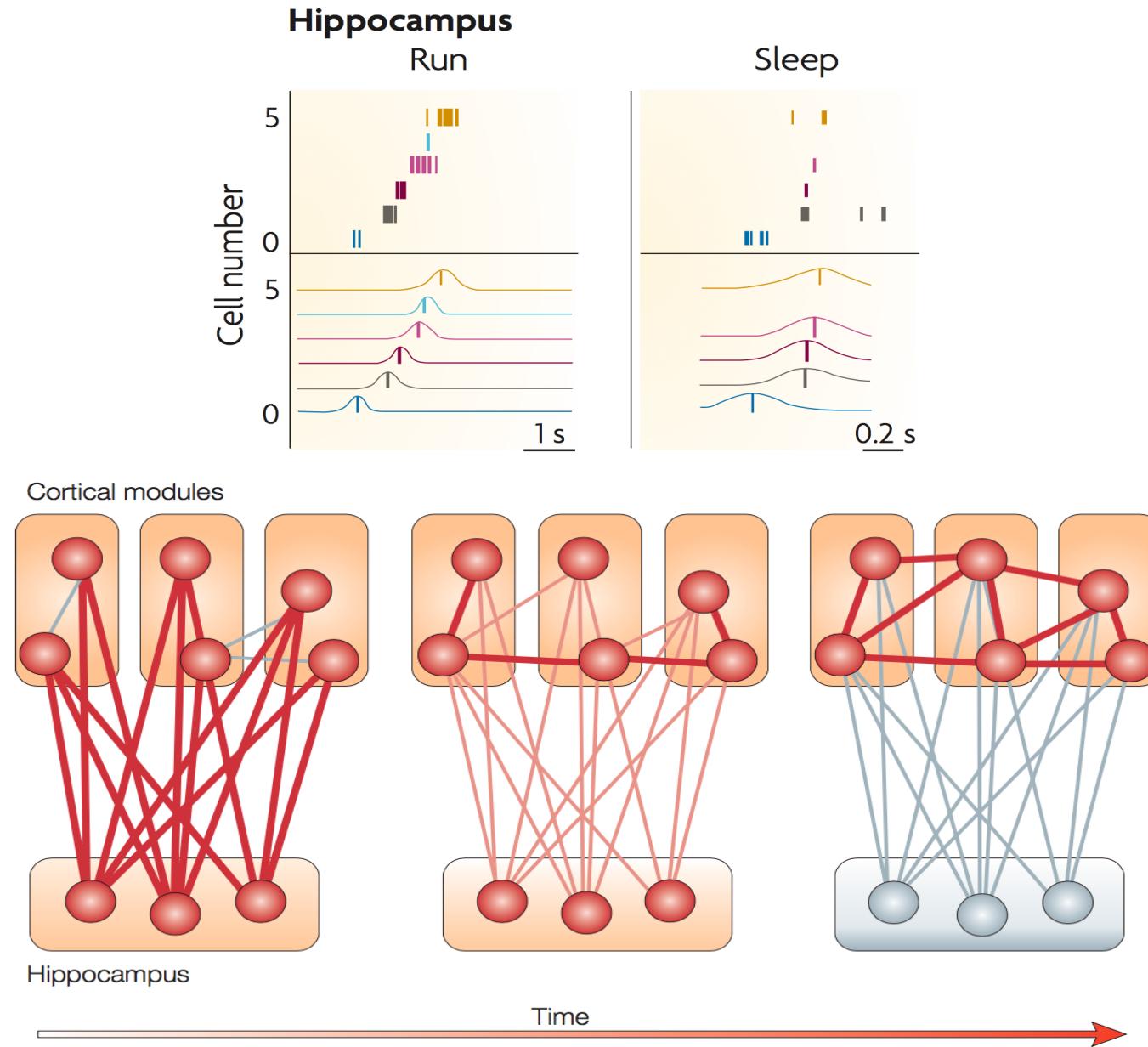


$$L_i(\theta_i) = \mathbb{E}_{(s,a,r,s') \sim \boxed{U(D)}} \left[ \left( r + \gamma \max_{a'} Q(s', a'; \theta_i^-) - Q(s, a; \theta_i) \right)^2 \right]$$

也讓快學的完美記憶教慢學的 Q-learning

# Deep Q Network (3/5)

引入海馬 Experience Replay 解決穩定性 - 彈性兩難



# Deep Q Network (4/5)

引入分開的 Target Network 隔一陣子才更新

**Algorithm 1:** deep Q-learning with experience replay.

Initialize replay memory  $D$  to capacity  $N$

Initialize action-value function  $Q$  with random weights  $\theta$

Initialize target action-value function  $\hat{Q}$  with weights  $\theta^- = \theta$

**For** episode = 1,  $M$  **do**

    Initialize sequence  $s_1 = \{x_1\}$  and preprocessed sequence  $\phi_1 = \phi(s_1)$

**For**  $t = 1, T$  **do**

        With probability  $\varepsilon$  select a random action  $a_t$    action selection

        otherwise select  $a_t = \operatorname{argmax}_a Q(\phi(s_t), a; \theta)$

        Execute action  $a_t$  in emulator and observe reward  $r_t$  and image  $x_{t+1}$

        Set  $s_{t+1} = s_t, a_t, x_{t+1}$  and preprocess  $\phi_{t+1} = \phi(s_{t+1})$

        Store transition  $(\phi_t, a_t, r_t, \phi_{t+1})$  in  $D$                   Experience Replay

        Sample random minibatch of transitions  $(\phi_j, a_j, r_j, \phi_{j+1})$  from  $D$

        Set  $y_j = \begin{cases} r_j & \text{if episode terminates at step } j+1 \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^-) & \text{target Network otherwise} \end{cases}$

        Perform a gradient descent step on  $(y_j - Q(\phi_j, a_j; \theta))^2$  with respect to the network parameters  $\theta$    更新Q-network权重

        Every  $C$  steps reset  $\hat{Q} = Q$  每隔C步更新target network

**End For**

**End For**

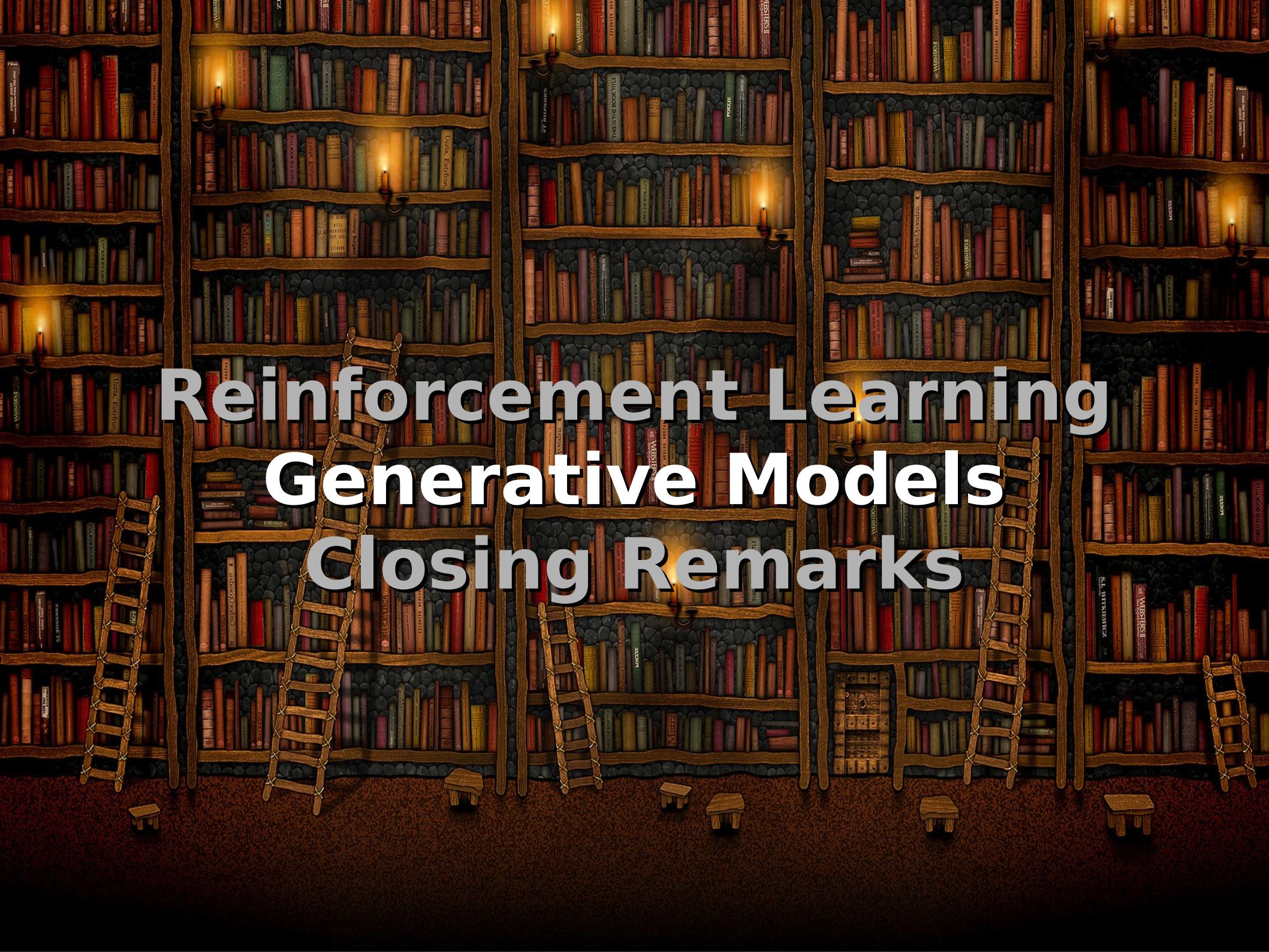
# Deep Q Network (5/5)

獨立的 Q-Net 與 Experience Replay 效果卓著

Extended Data Table 3 | The effects of replay and separating the target Q-network

Game	With replay, with target Q	With replay, without target Q	Without replay, with target Q	Without replay, without target Q
Breakout	316.8	240.7	10.2	3.2
Enduro	1006.3	831.4	141.9	29.1
River Raid	7446.6	4102.8	2867.7	1453.0
Seaquest	2894.4	822.6	1003.0	275.8
Space Invaders	1088.9	826.3	373.2	302.0

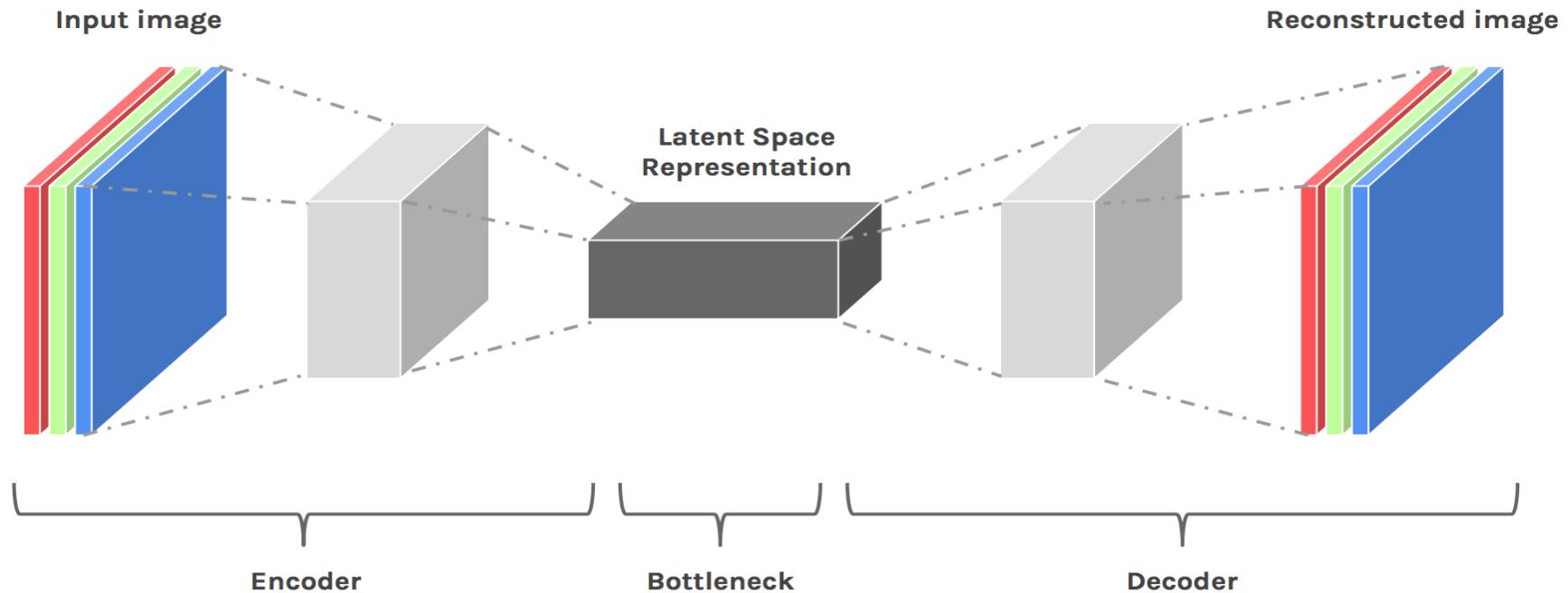
DQN agents were trained for 10 million frames using standard hyperparameters for all possible combinations of turning replay on or off, using or not using a separate target Q-network, and three different learning rates. Each agent was evaluated every 250,000 training frames for 135,000 validation frames and the highest average episode score is reported. Note that these evaluation episodes were not truncated at 5 min leading to higher scores on Enduro than the ones reported in Extended Data Table 2. Note also that the number of training frames was shorter (10 million frames) as compared to the main results presented in Extended Data Table 2 (50 million frames).



# Reinforcement Learning Generative Models Closing Remarks

# AutoEncoder

可以產生一個 latent/embedding representation



Label "7"- inlier

Input:	7 7 7 7 7 7 7 7
Reconstruction:	7 7 7 7 7 7 7 7

Label "0"- outlier

Input:	0 0 0 0 0 0 0 0
Reconstruction:	0 0 0 0 0 0 0 0

# Latent Space Sampling (1/2)

若 Latent Space 有好的結構則可以更好地 editing

Interpolation in Latent Space



Arithmetic in Latent Space

Latent  
space

$$\begin{array}{c} \textcolor{blue}{\square} \\ - \\ \textcolor{blue}{\square} \end{array}$$

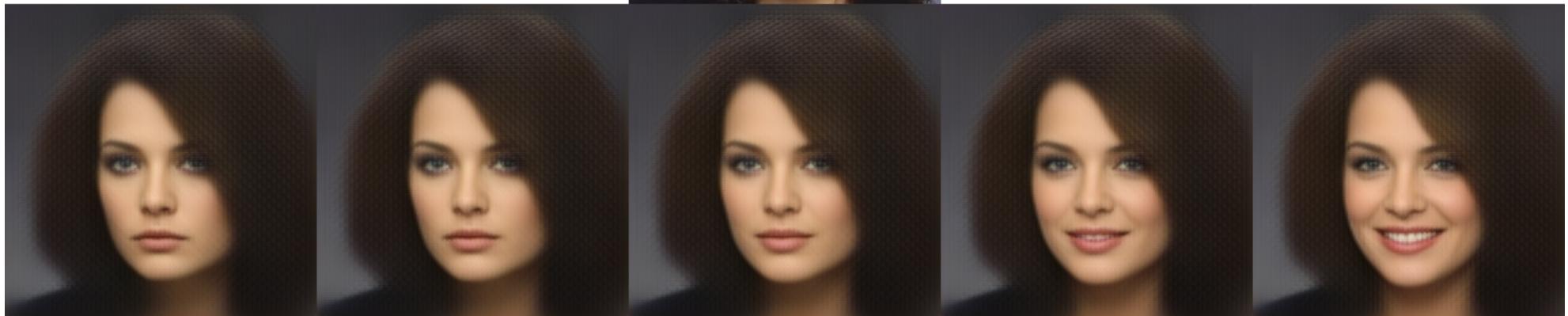
$$\downarrow \quad \downarrow$$

Shape  
space

$$\begin{array}{c} \textcolor{red}{\square} \\ - \\ \textcolor{red}{\square} \end{array}$$

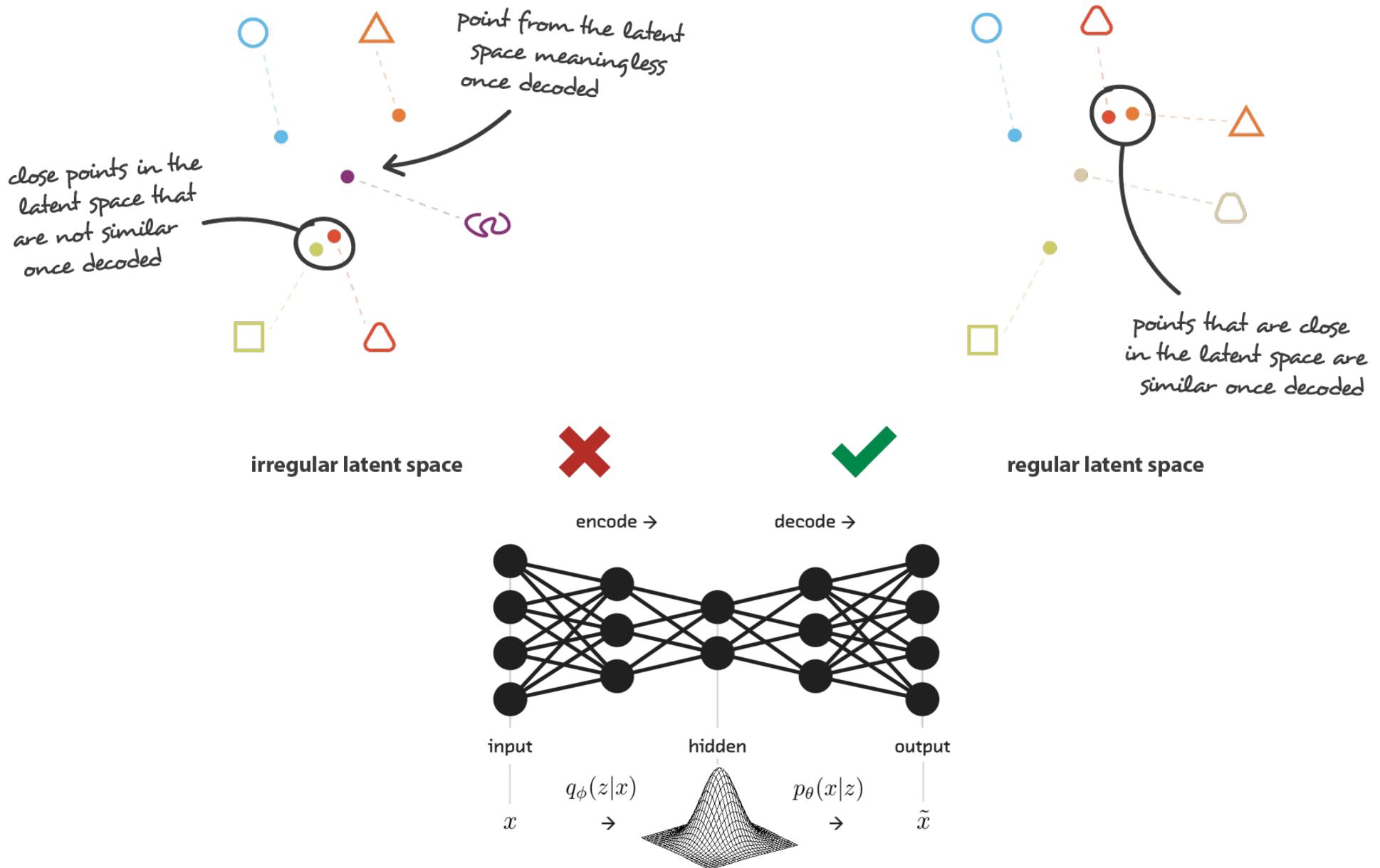
# Latent Space Sampling (2/2)

+/- smile vector



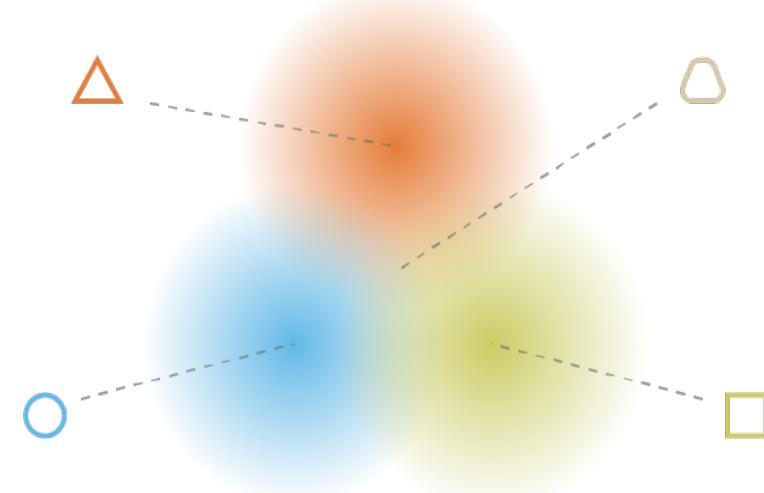
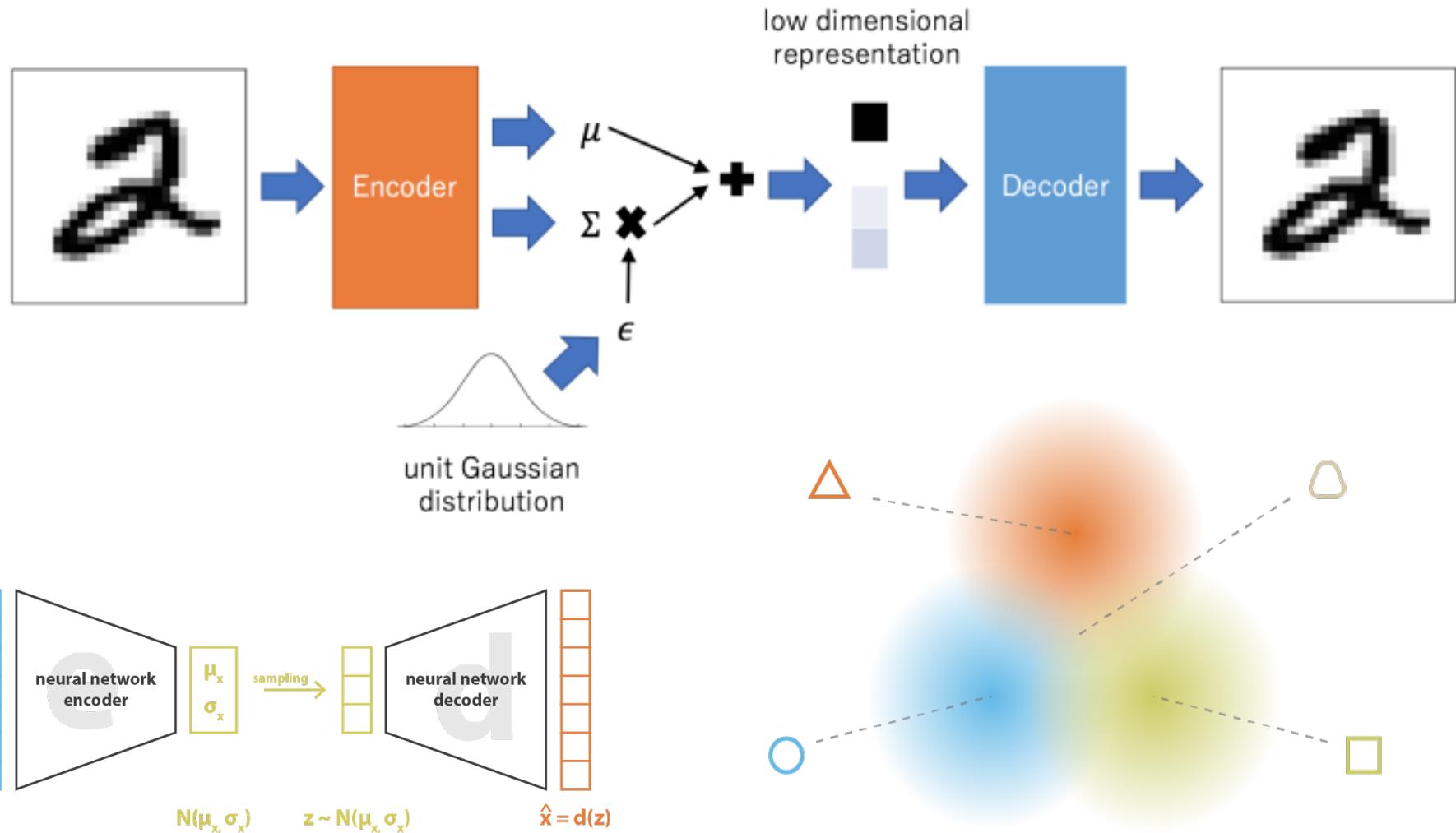
# Variational AutoEncoder (1/3)

強迫 latent space 有連續變化 (cf. Kohonen SOM)



# Variational AutoEncoder (2/3)

具體而言讓一 Gaussian dist. 映射到同一個 output



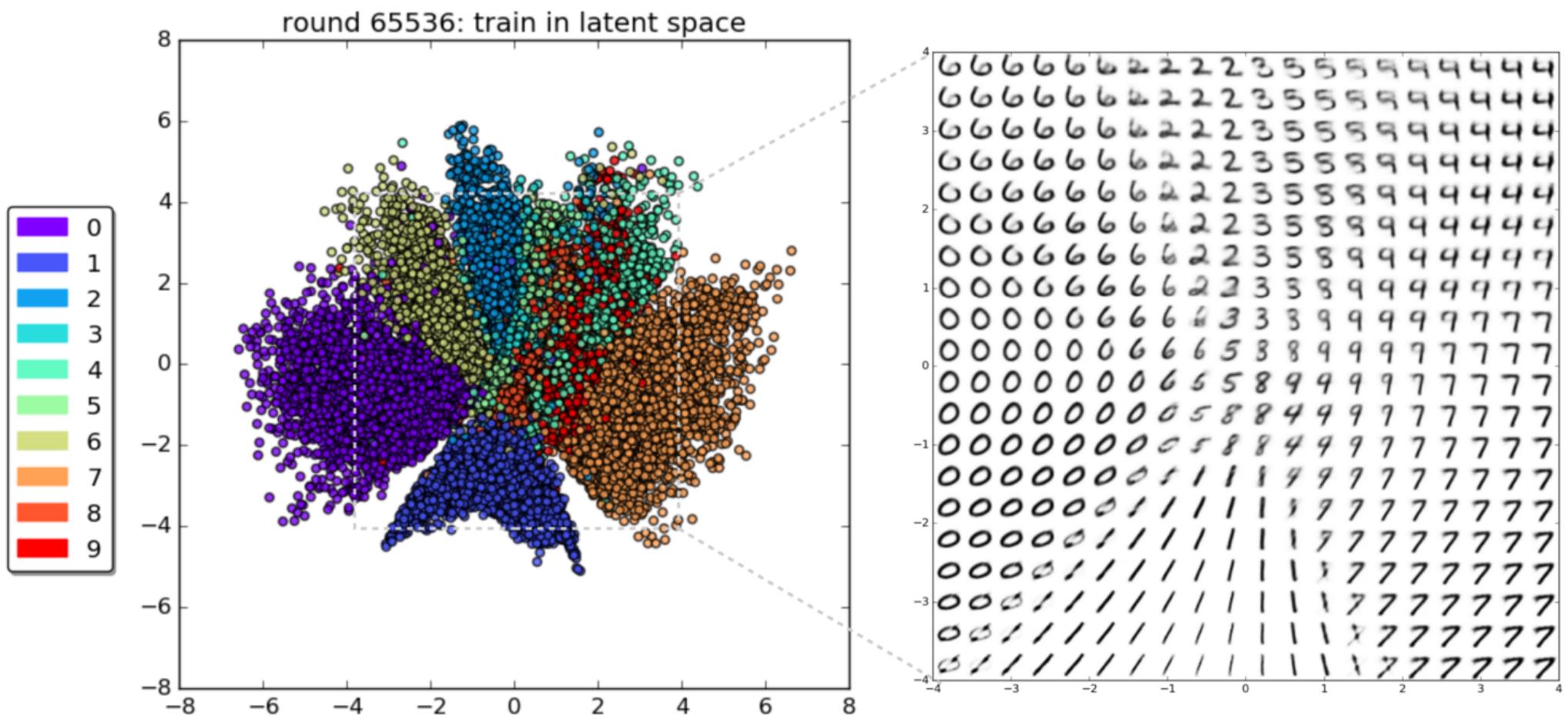
$$\text{loss} = \|x - \hat{x}\|^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)] = \|x - d(z)\|^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)]$$



what we want to obtain with regularisation

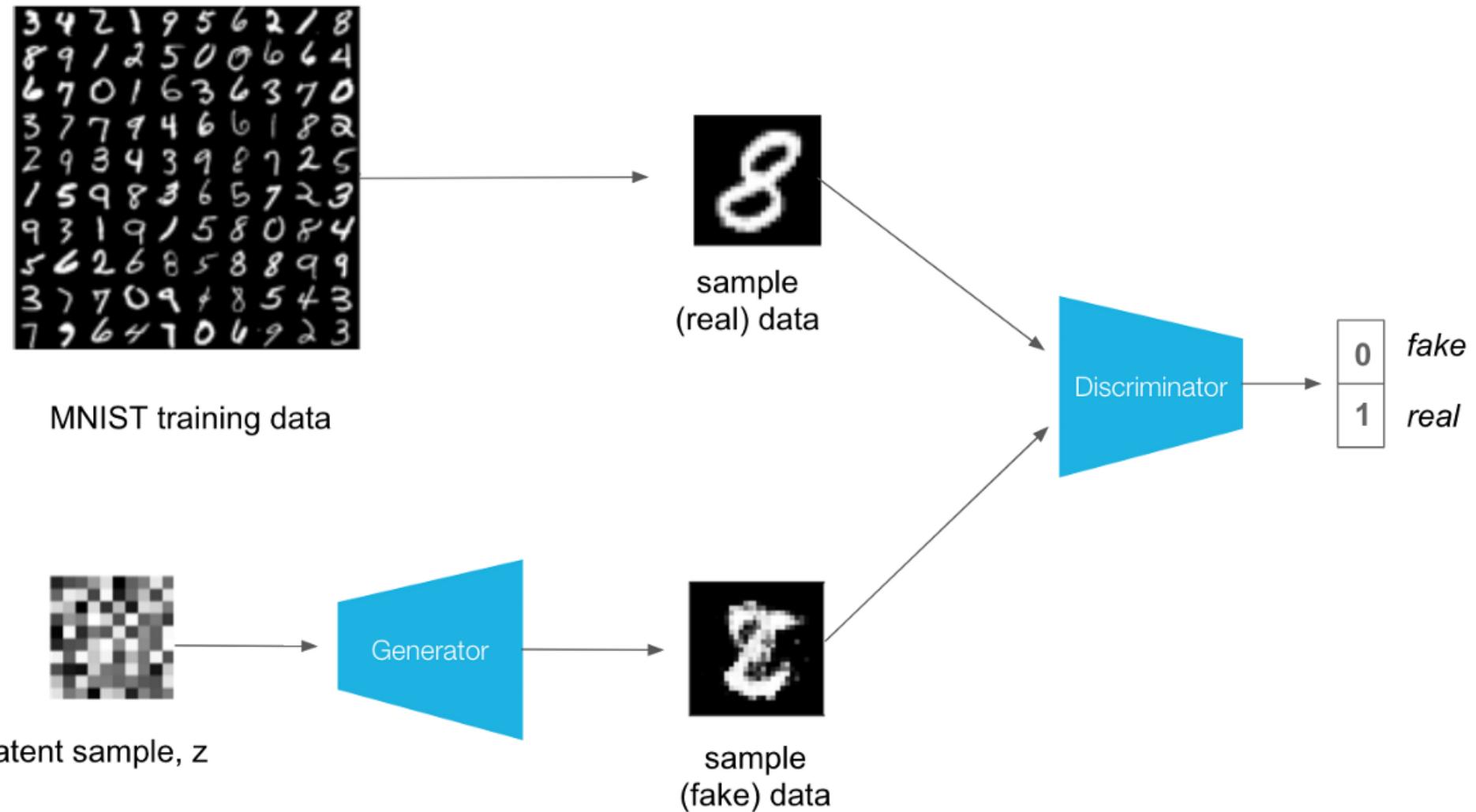
# Variational AutoEncoder (3/3)

MNIST 經過 VAE 學習 2 維 embedding 後



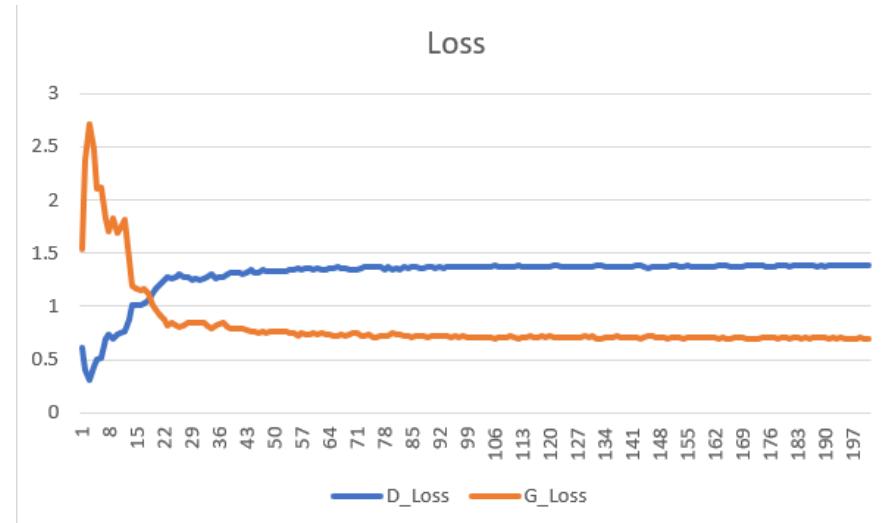
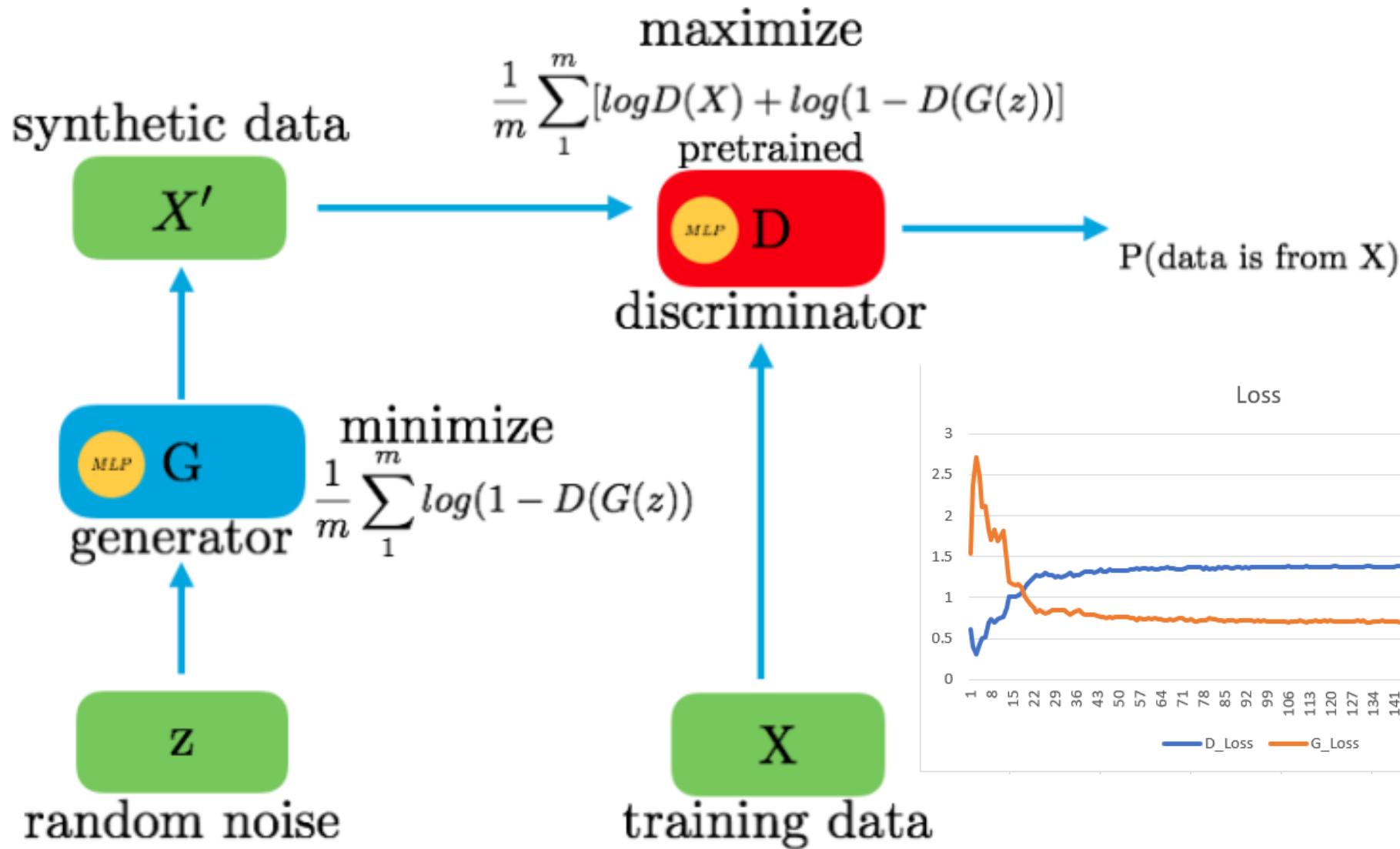
# Generative Adversarial Network (1/2)

可看成更進階的 agent-based modeling



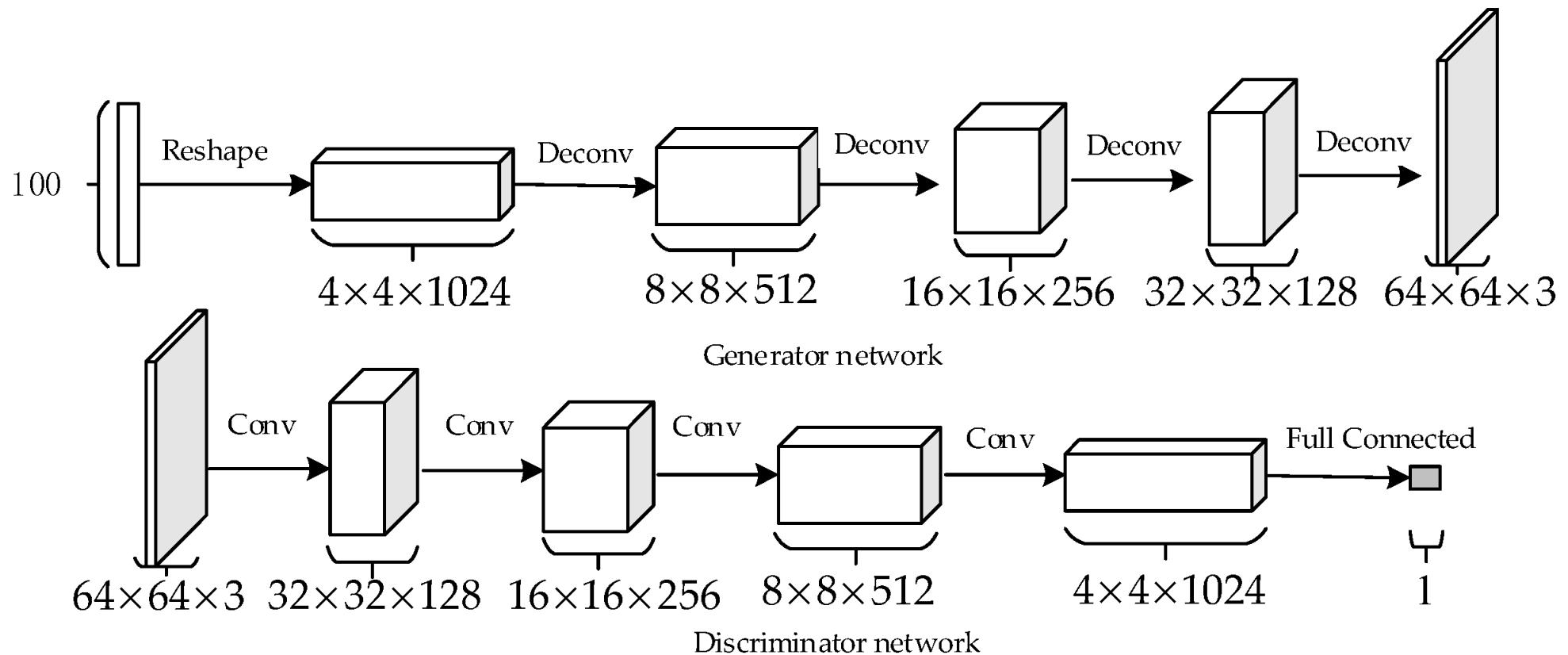
# Generative Adversarial Network (2/2)

關鍵在設計好 loss functionS 與 D/G 的非同步學習



# Deep Convolutional GAN (DCGAN)

Convolution & Deconvolution



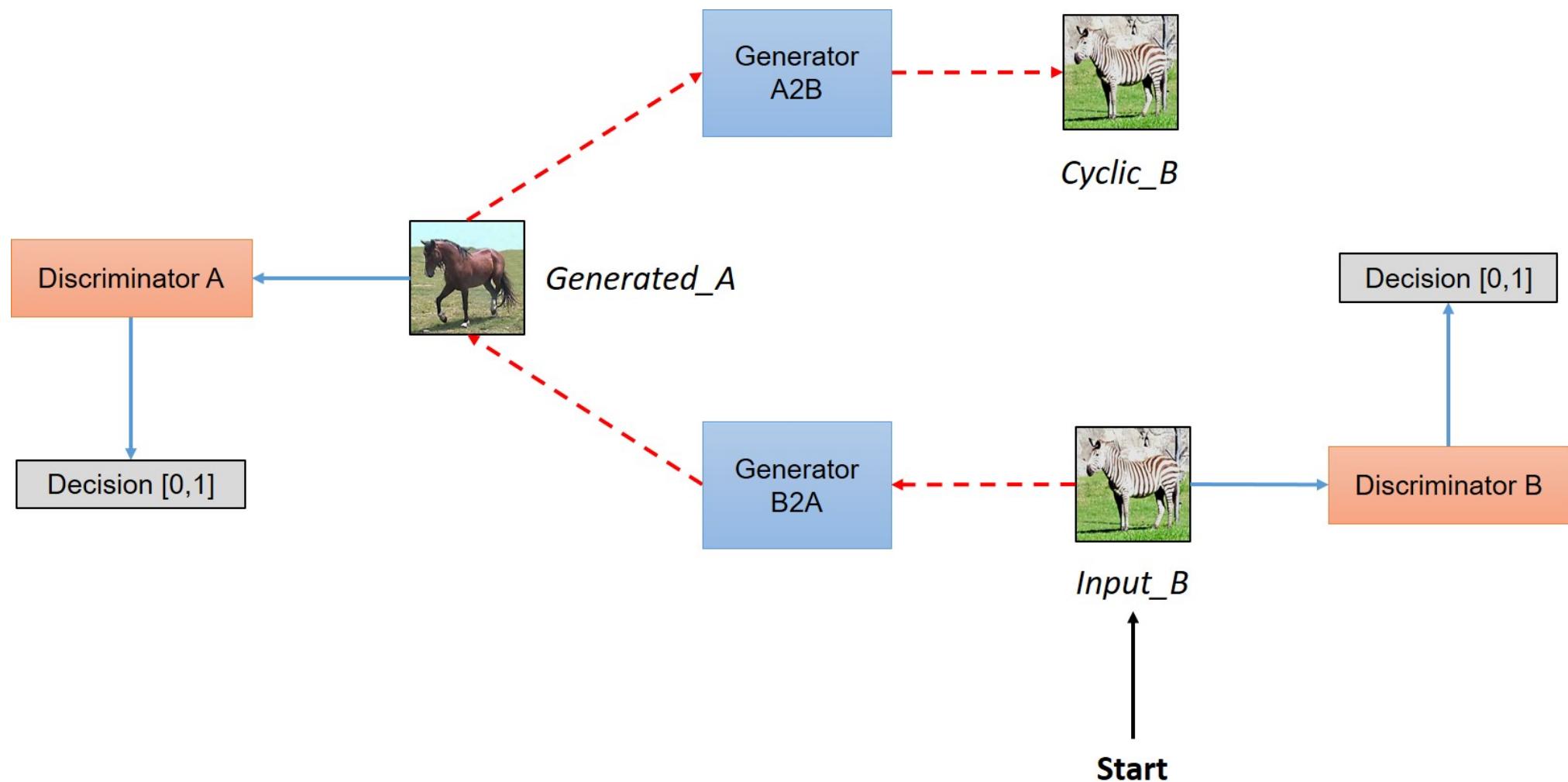
# DCGAN (續)

Generator 最後可以以假亂真



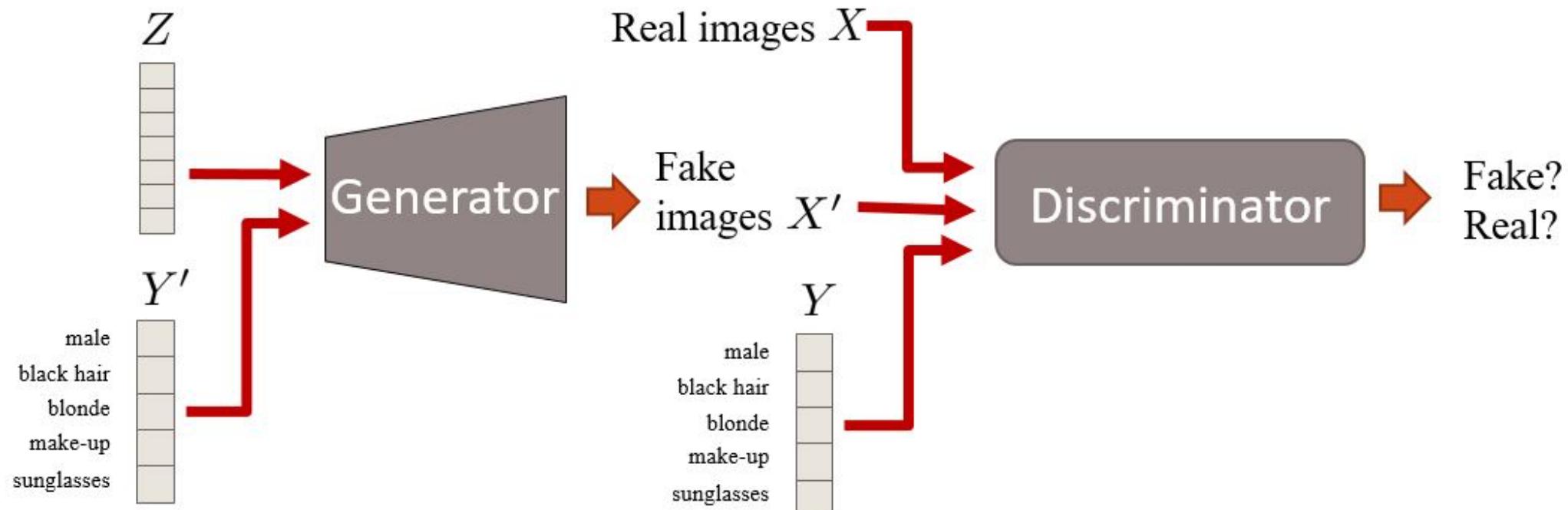
# CycleGAN

調包遊戲



# Conditional GAN

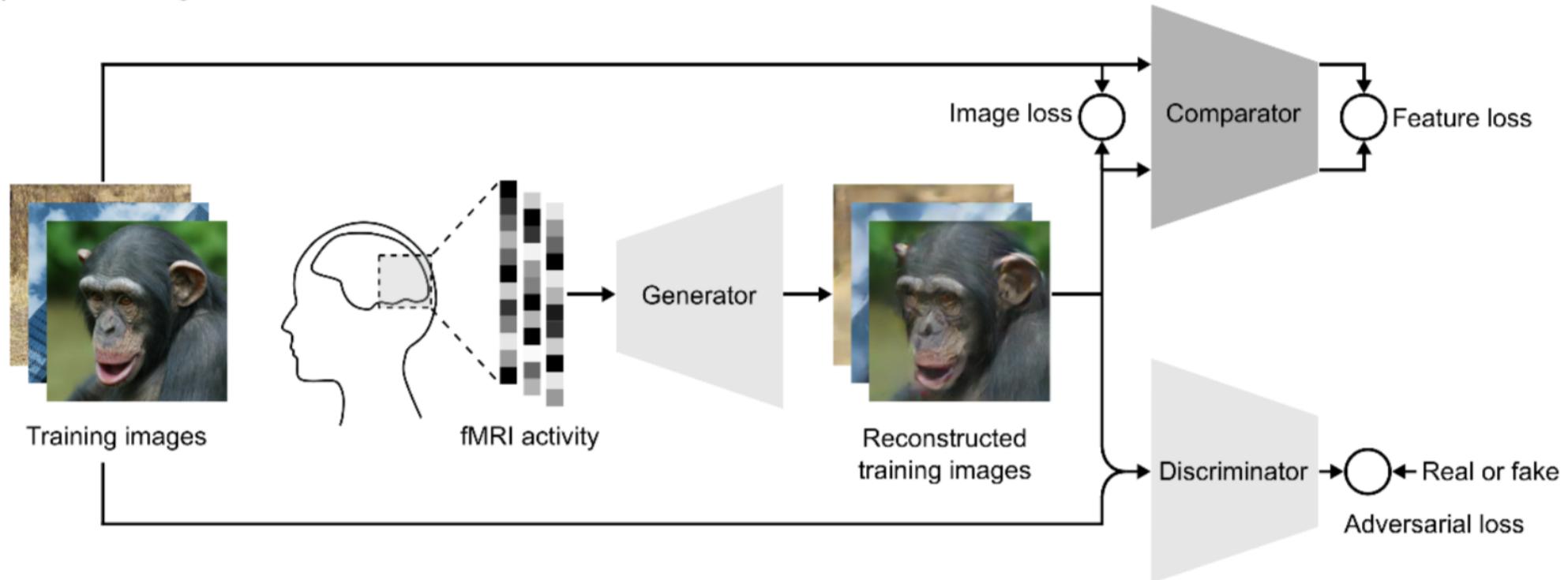
更能根據需要產生樣本  
(cf. latent space editing)



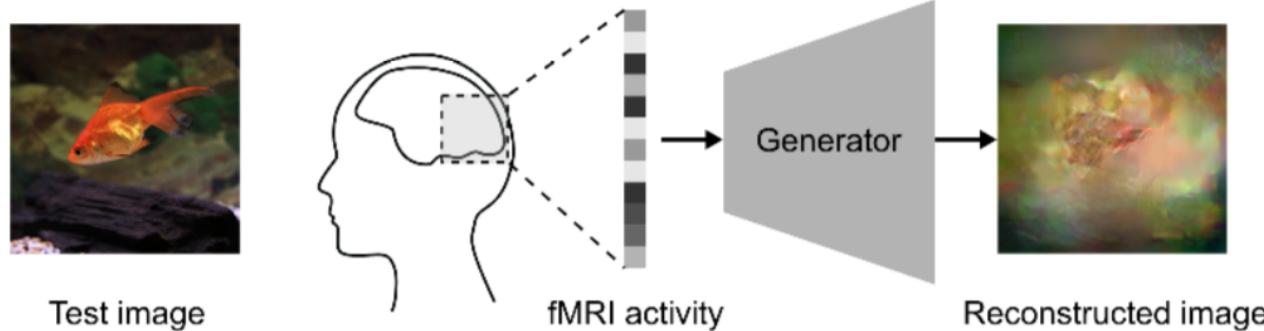
# Applications of GAN

例如用來從腦訊號重建外接刺激

(A) Model training



(B) Model test



# Neural Style Transfer (1/4)

如果風格與內容能夠分開

## 1 Upload photo

The first picture defines the scene you would like to have painted.



## 2 Choose style

Choose among predefined styles or upload your own style image.



## 3 Submit

Our servers paint the image for you. You get an email when it's done.



# Neural Style Transfer (2/4)

Styles 藏在細節裡 ( 即空間中的高頻資訊 )

新細明體

焉能辨我是宋明？

華康粗明體

焉能辨我是宋明？

華康儷中宋

焉能辨我是宋明？

文鼎中明

焉能辨我是宋明？

文鼎粗標準宋體

焉能辨我是宋明？

蒙納繁中宋

焉能辨我是宋明？

Broad SF



High SF

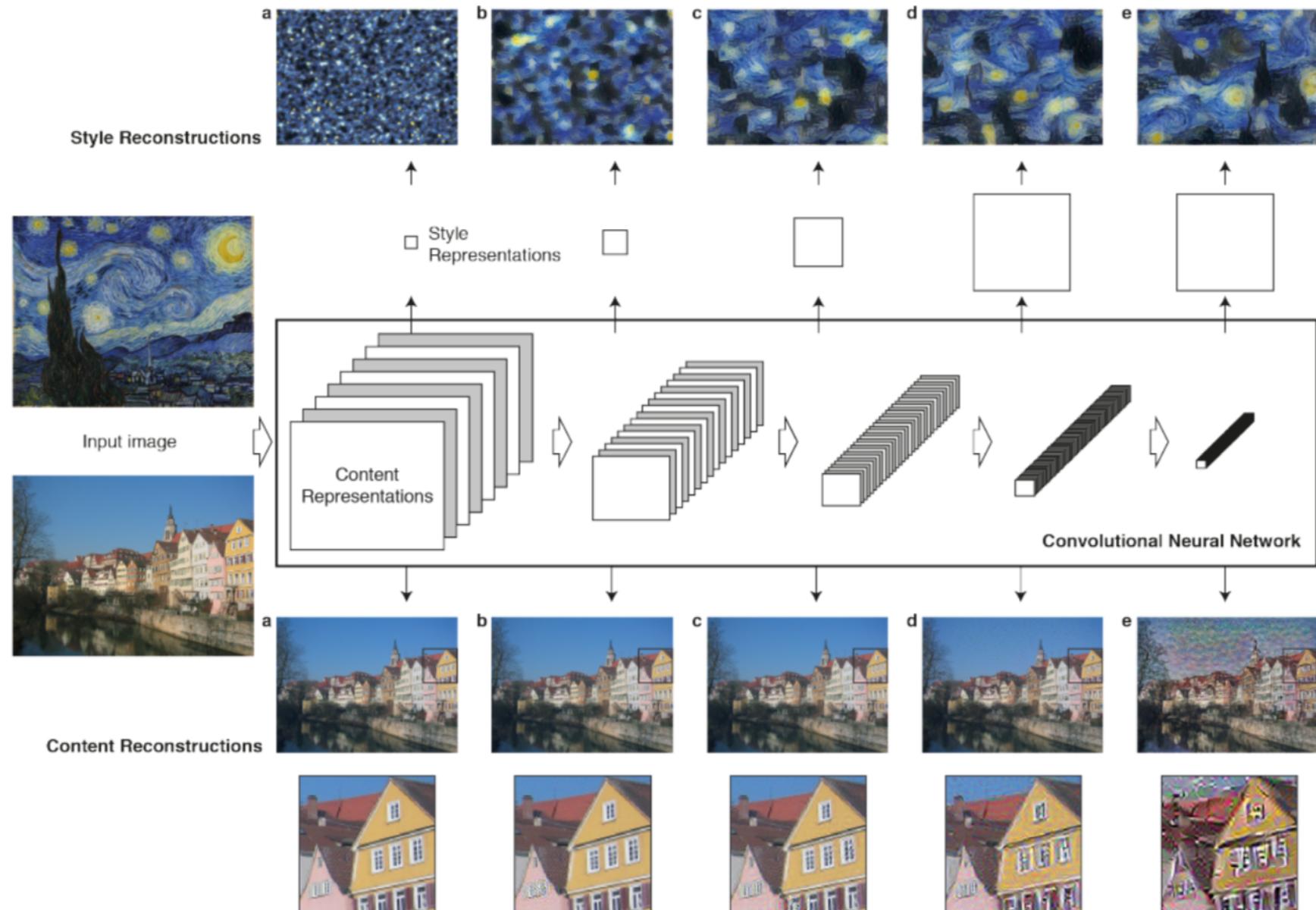


Low SF



# Neural Style Transfer (3/4)

CNN 的下游層比較有內容 (gist)



# Neural Style Transfer (4/4)

$$L = D(\text{content}(G) - \text{content}(P)) + D(\text{style}(G), \text{style}(A))$$

$$\mathcal{L}_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{content}(\vec{p}, \vec{x}) + \beta \mathcal{L}_{style}(\vec{a}, \vec{x})$$

$\alpha$  and  $\beta$  are the weighting factors for content and style reconstruction

$$\mathcal{L}_{content}(\vec{p}, \vec{x}, l) = \frac{1}{2} \sum_{i,j} (F_{ij}^l - P_{ij}^l)^2$$

$\vec{p}$  The original image

$\vec{x}$  The generated image

$l$  Layer

$F_{ij}^l$  Activation of the  $i^{th}$  filter at position j in the feature representation of  $\vec{x}$  in  $l$

$P_{ij}^l$  Activation of the  $i^{th}$  filter at position j in the feature representation of  $\vec{p}$  in  $l$

$$\mathcal{L}_{style}(\vec{a}, \vec{x}) = \sum_{l=0}^L w_l E_l$$

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G_{ij}^l - A_{ij}^l)^2$$

$N_l$  number of distinct feature maps

$M_l$  the height times the width of the feature map

$G_{ij}^l$  pairwise feature maps i and j in the style representation of  $\vec{x}$  in  $l$

$A_{ij}^l$  pairwise feature maps i and j in the style representation of  $\vec{a}$  in  $l$

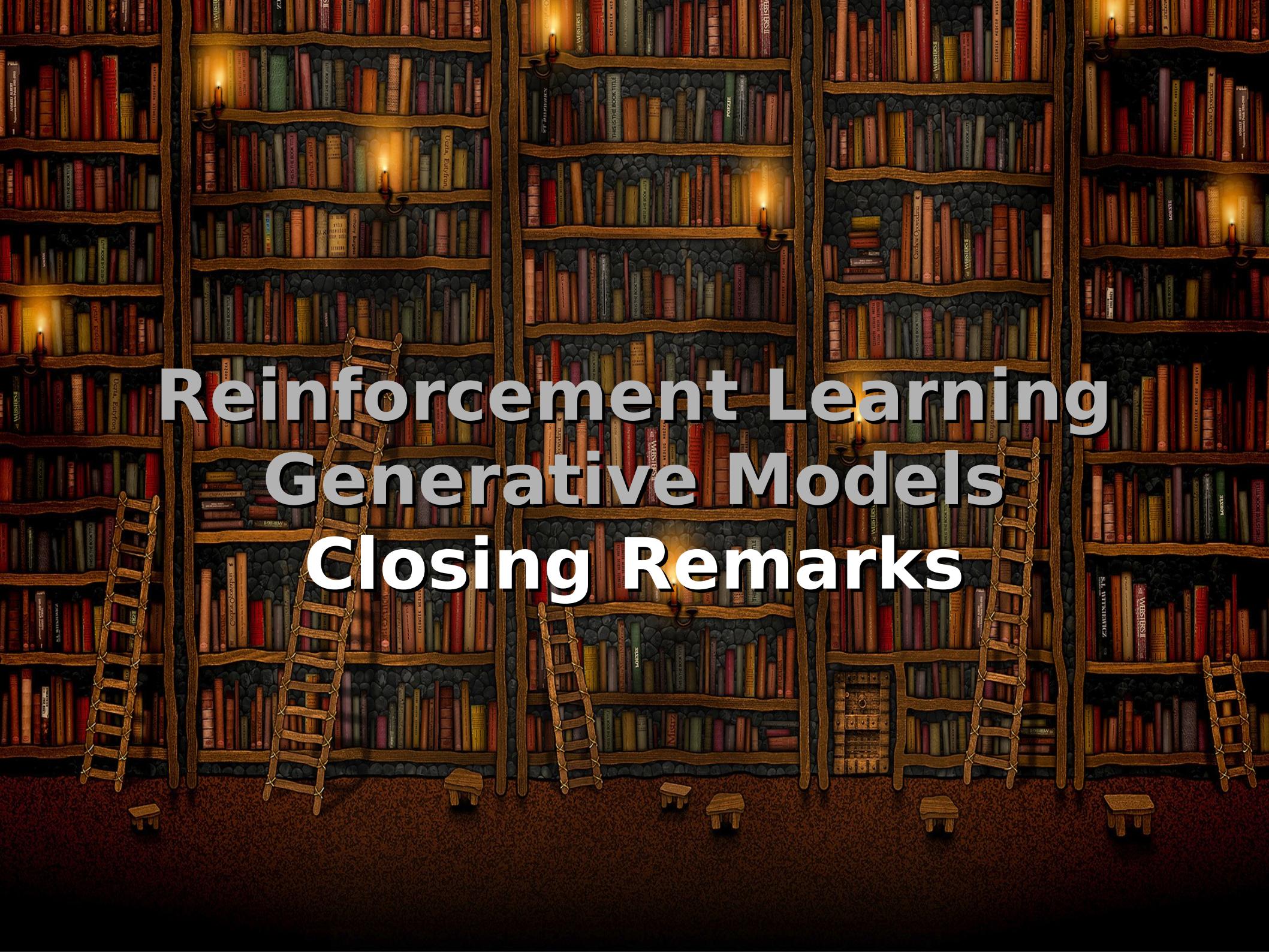
$\vec{a}$  The original image

$\vec{x}$  The generated image

$L$  Total number of all layers

$w_l$  Weighting factors of the contribution of each layer to the total loss

$E_l$  loss in layer  $l$

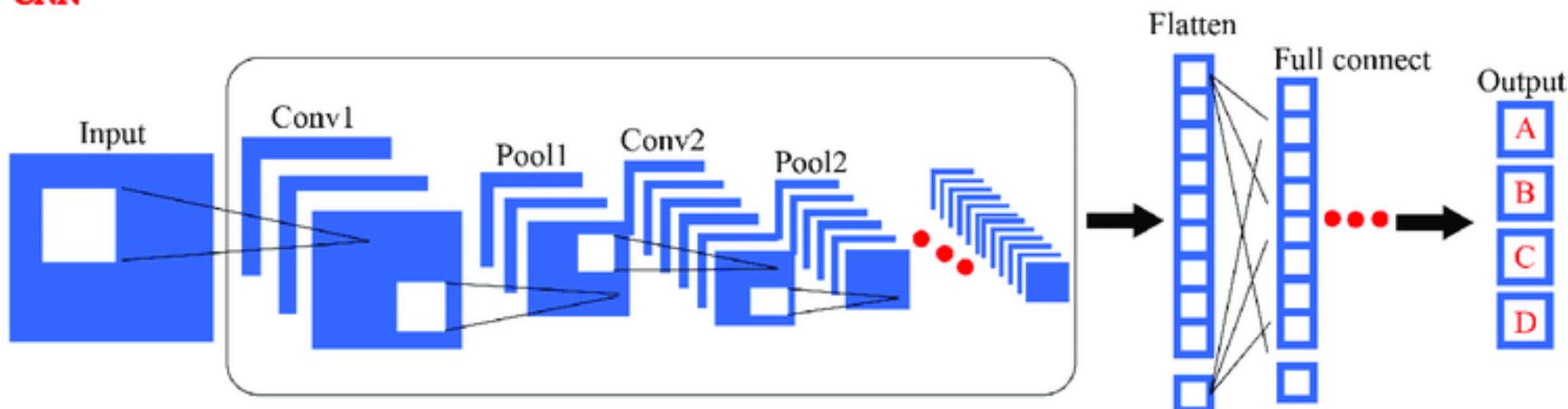


# Reinforcement Learning Generative Models Closing Remarks

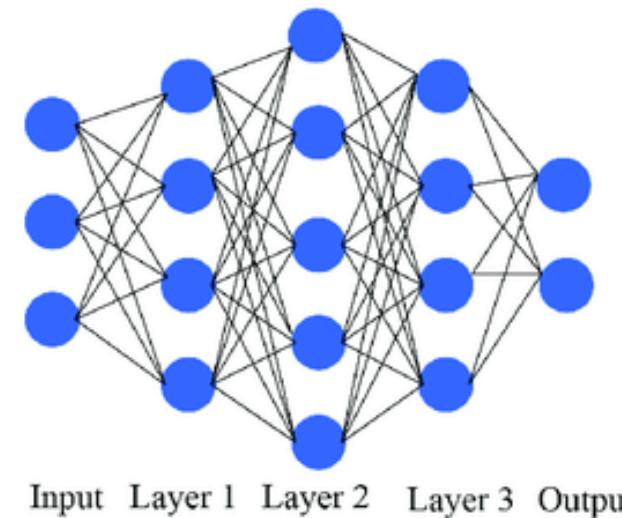
# Deep Neural Networks 的分類

CNN 通常處理影像資料；RNN 通常處理語言資料

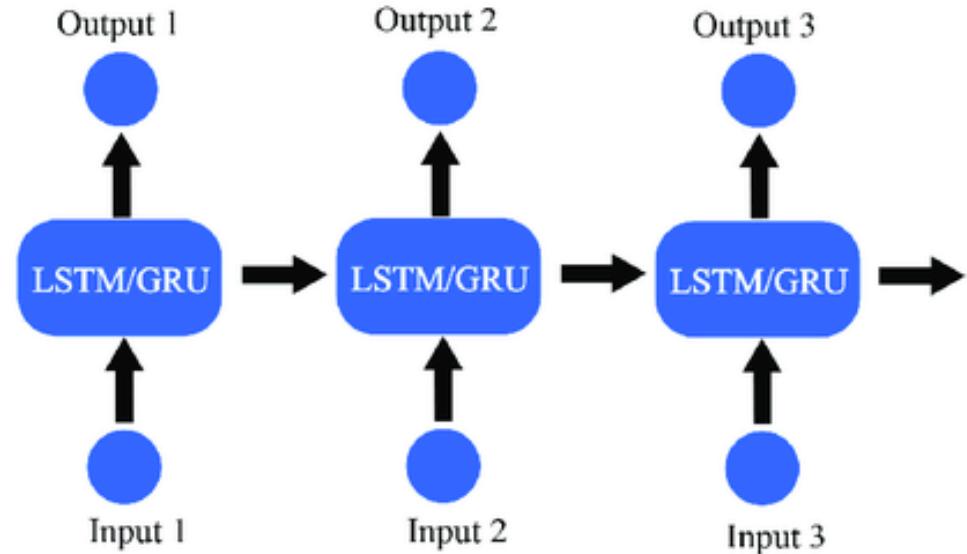
**CNN**



**DNN**



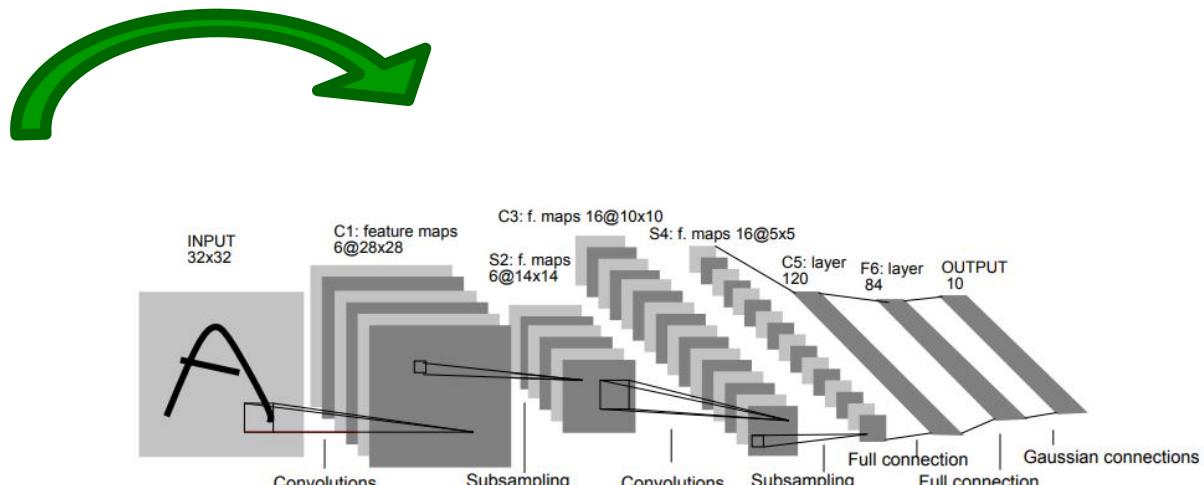
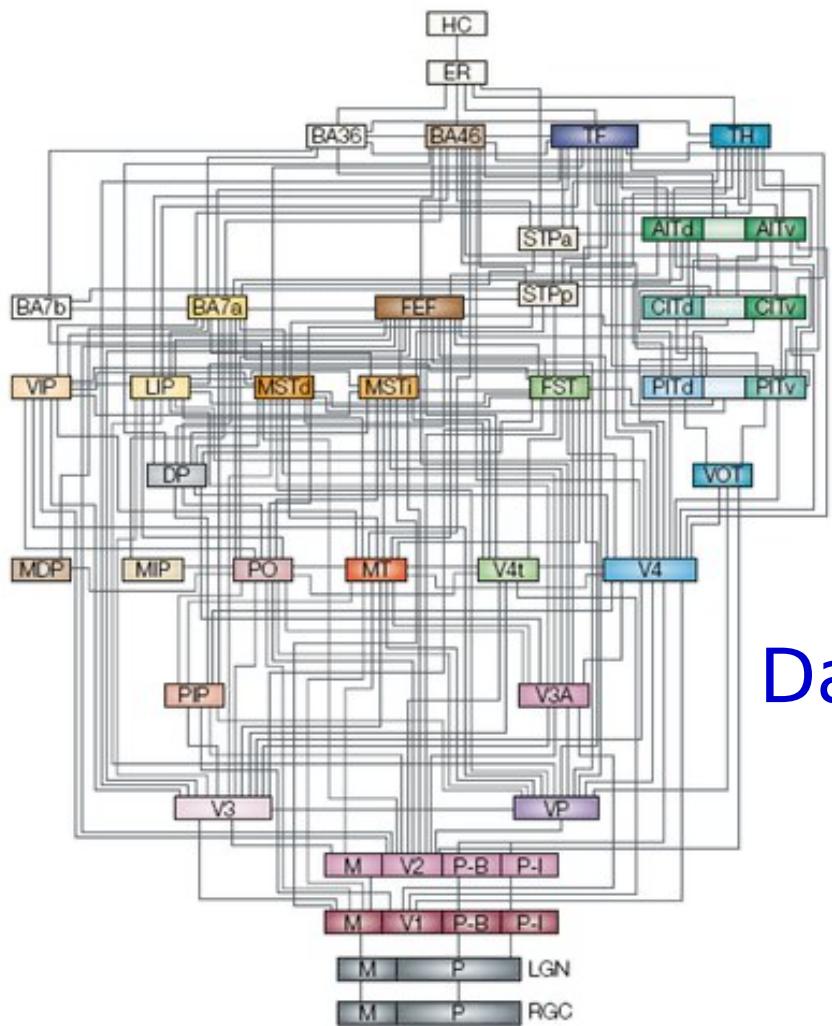
**RNN**



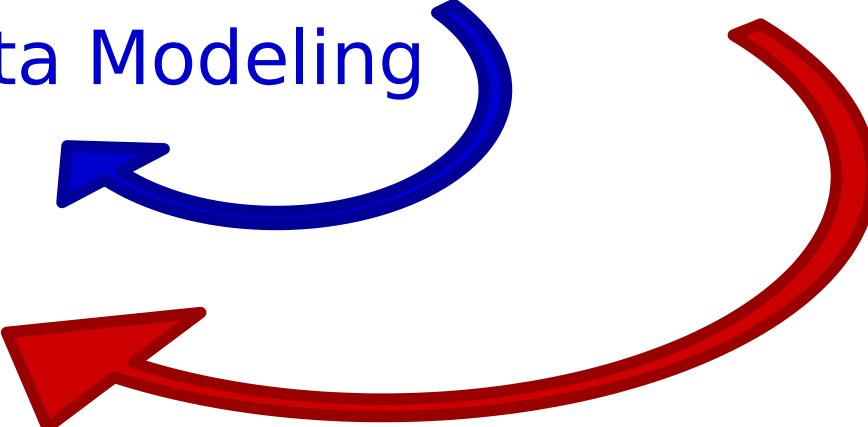
# 生物神經與類神經的三種關係

預測性模型也可被解釋或幫助解釋

## Architectural Metaphors



## Data Modeling

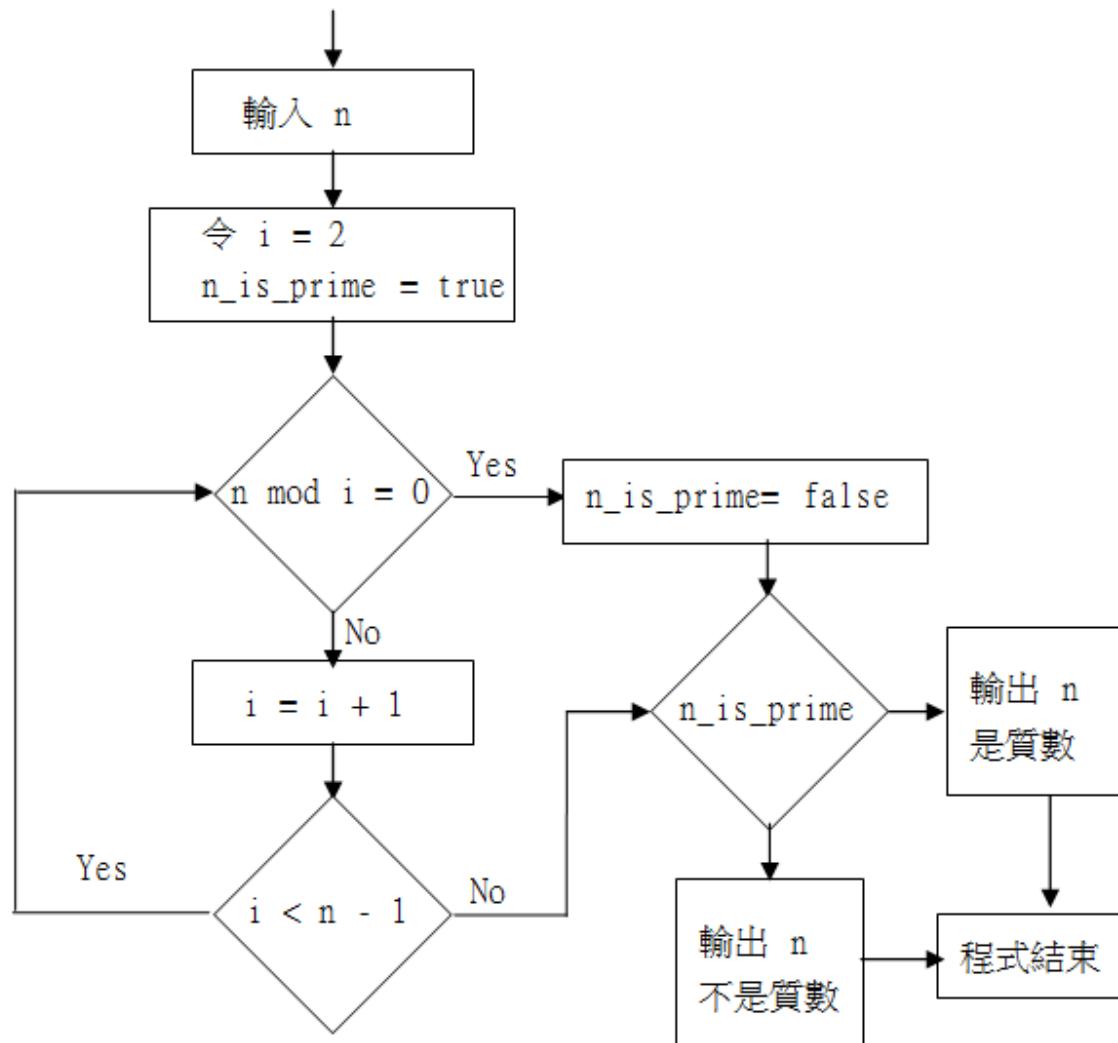


## Functional Metaphors

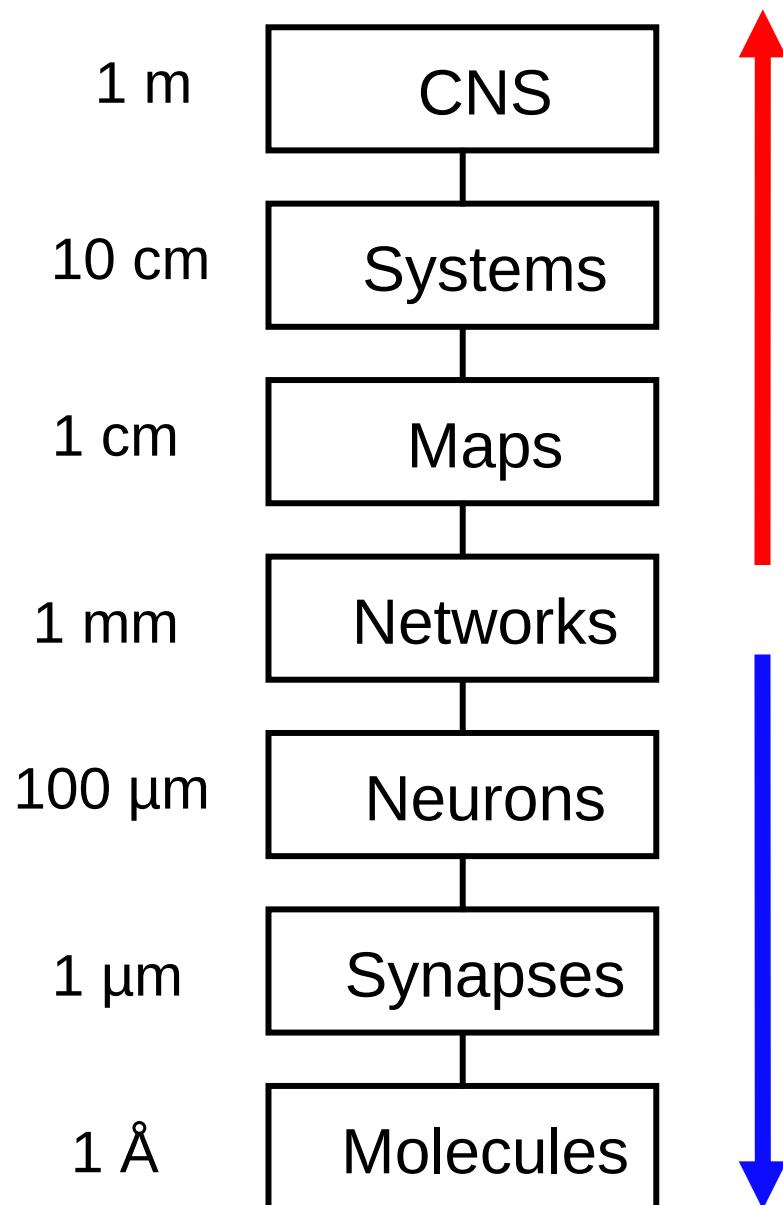


# Limits of Deep Learning

整體架構還是歸納學習與內插法預測



# Research Scales



**Computational  
Cognitive Science**  
Bayesian perception  
Diffusion models

**Computational  
Cognitive Neuroscience**  
Connectivity analysis  
Neural networks

**Computational  
Neuroscience**  
Neural coding  
Neuronal models

# Game Over

