人工智能
神經網絡深度學習
Deep Learning

Macau AI Challenge 2019

Fai Wong, Derek

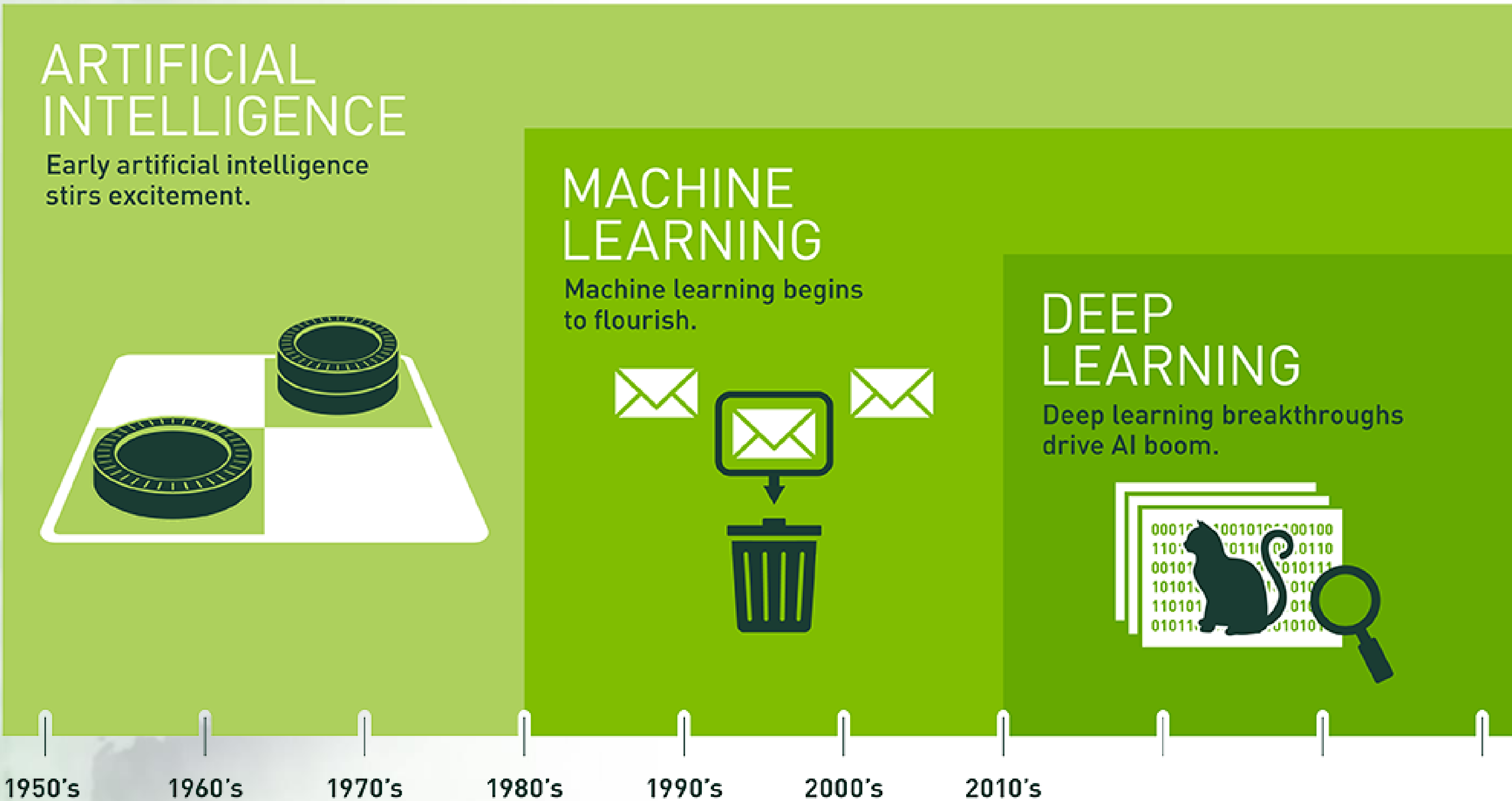
NLP²CT – *Natural Language Processing & Portuguese-Chinese Machine Translation Research Group*

University of Macau

derekfw@umac.mo

Natural Language Processing & Portuguese – Chinese Machine Translation Laboratory
自然語言處理與中葡機器翻譯實驗室

What is Deep Learning?



What is Machine Learning?

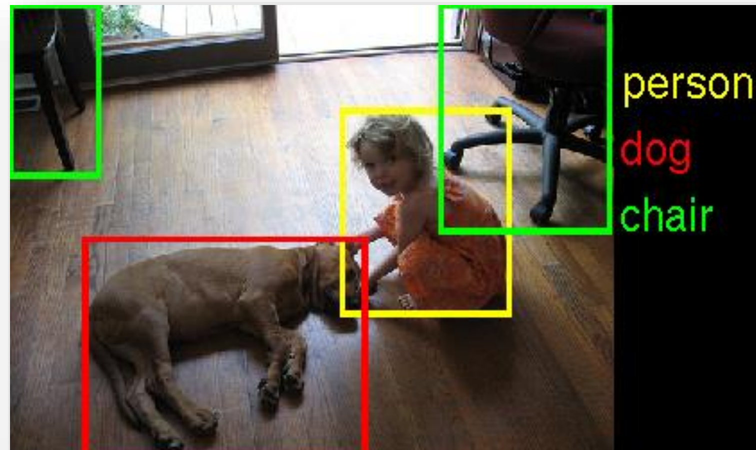
- Arthur Samuel (1959)
  - Field of study that *gives computers the ability to learn* without being *explicitly programmed*
- Tom Mitchell (1998)
  - A computer program is said to learn from *experience E* with respect to some *task T* and some performance *measure P*, if its performance on T, as measured by P, improves with experience E

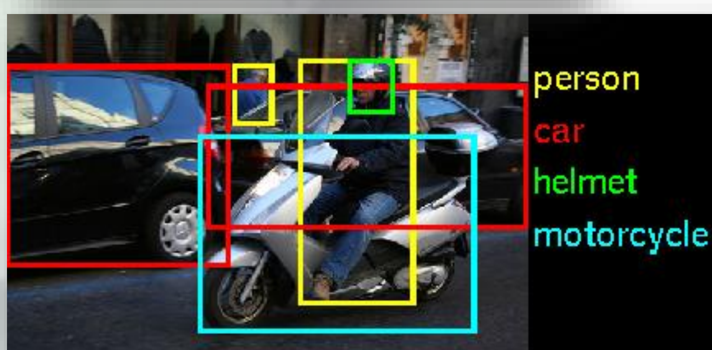
DEEP LEARNING in

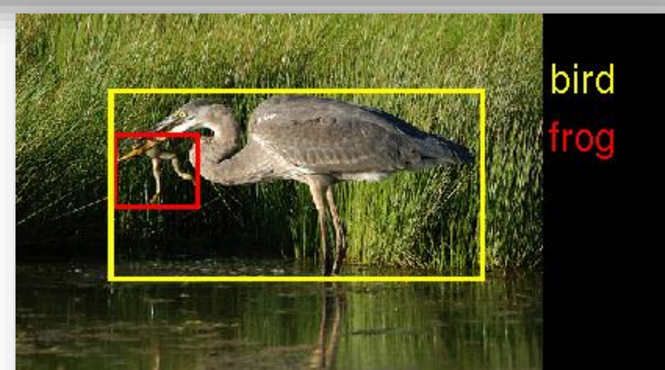
COMPUTER VISION

Deep Learning Success
Vision

Image Recognition



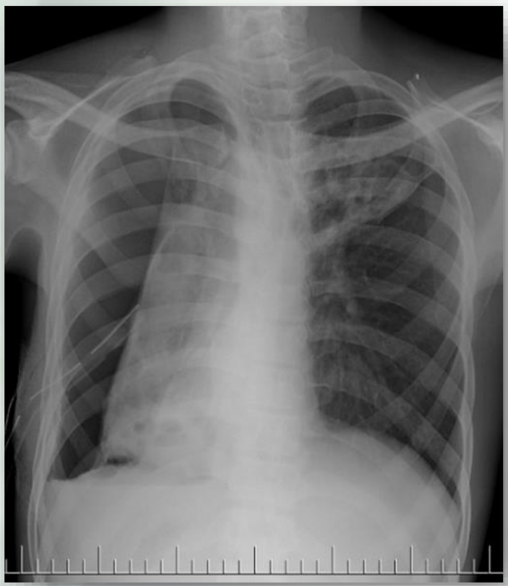
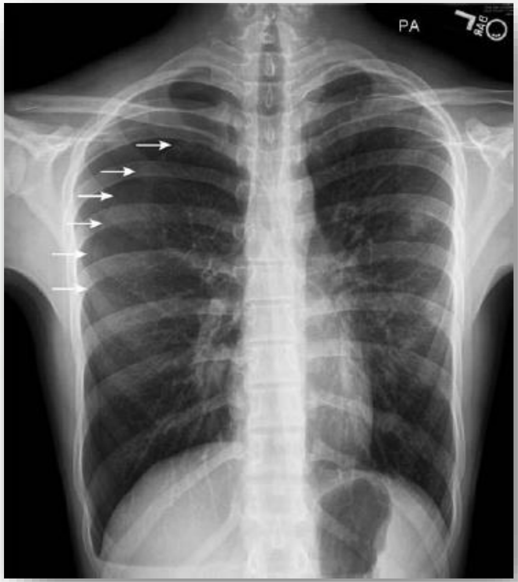




(ILSVRC 2014)

Deep Learning Success
Vision

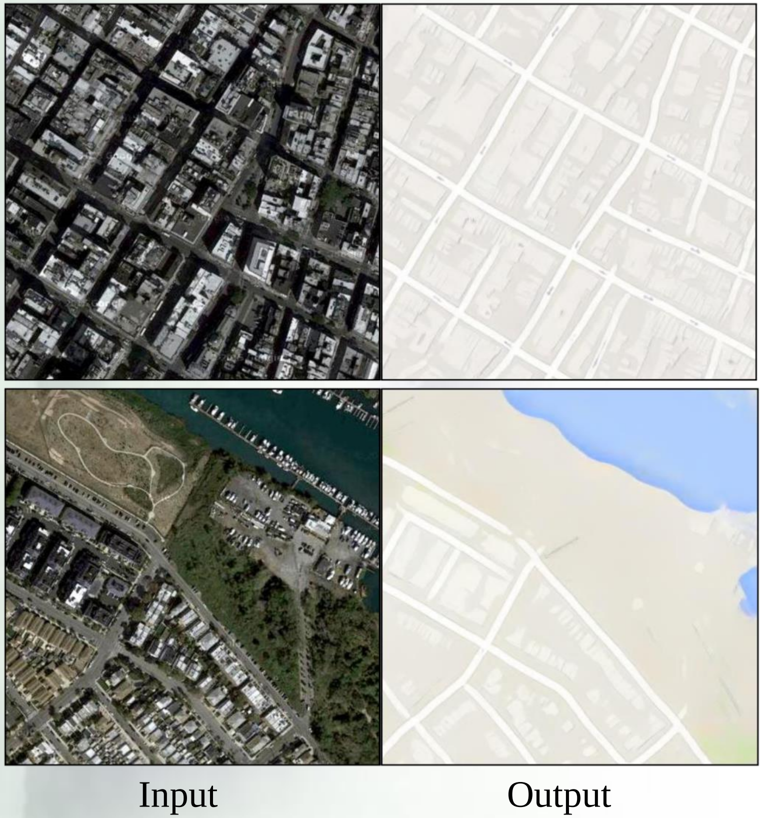
Detection of pneumothorax (氣胸) in X-Ray scans

(Alexander 2018)

Deep Learning Success
Vision

Image Translation



(Isola 2018)

Deep Learning Success
Vision

Image Generation

redshank ant Monastery

volcano

(Nguyen et al. 2016)

Why Impressed?

- Vision is very challenging
  - *For a small* 256×256 *resolution and for* 256 *pixel values*
  - *A total* $2^{524,288}$ *of possible images*
  - *In comparison, there are about* 10^{24} *stars in the universe*
- Visual object variations
  - *Different viewpoints* (視角), *scales* (尺度), *deformations* (變形), *occlusions* (遮擋)
- Semantic object variations
  - *Intra-class variation*
  - *Inter-class overlaps*

DEEP LEARNING in

ROBOTICS & AI

Deep Learning Success
Robotics & AI

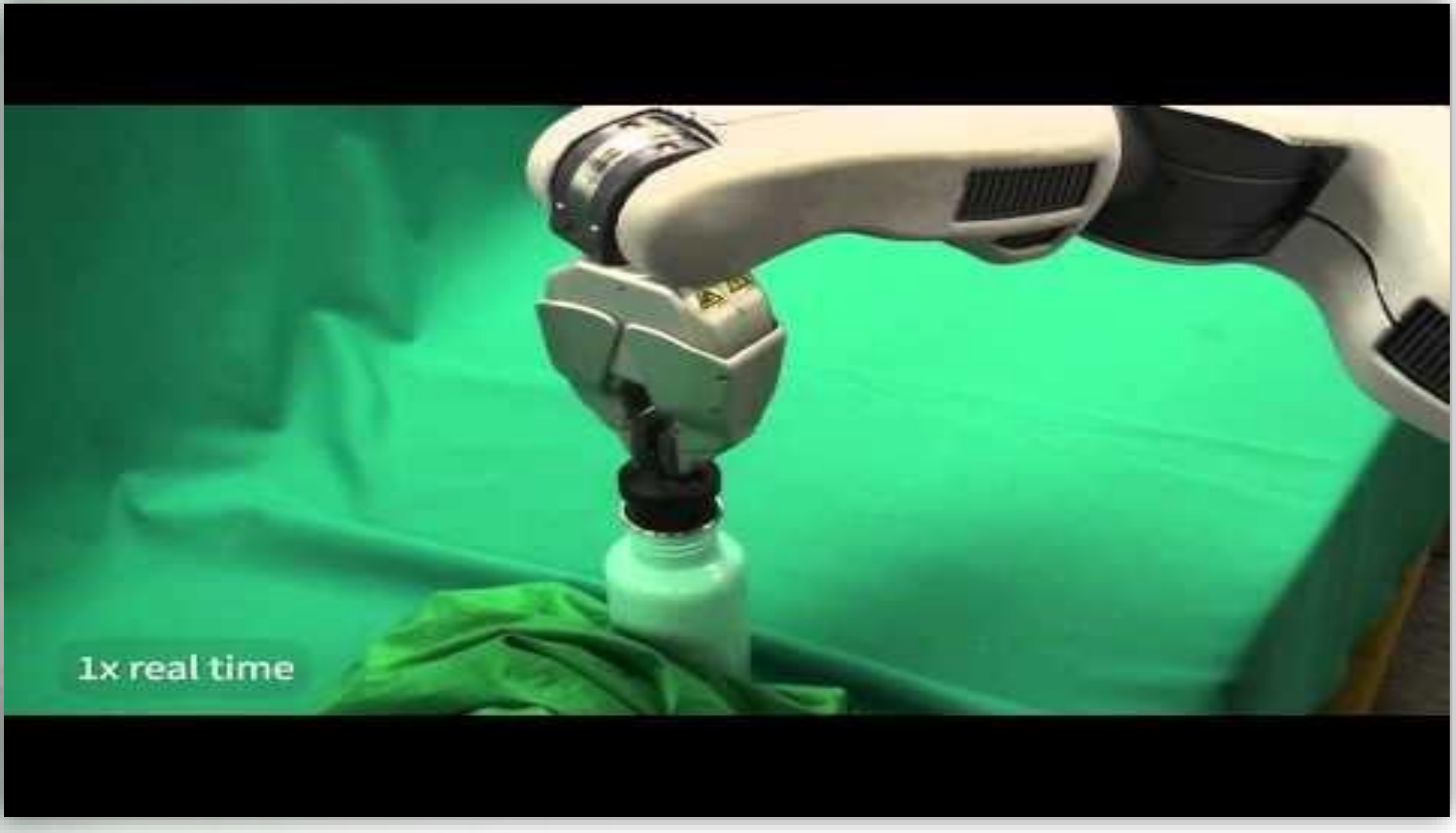
Self-Driving Cars https://www.youtube.com/watch?v=-96BEoXJMs0

Deep Learning Success
Robotics & AI

NLP²CT *Research Group*

Robots

https://www.youtube.com/watch?v=2hGngG64dNM



Deep Learning Success
Robotics & AI

Game AI – Playing Atari

https://www.youtube.com/watch?v=V1eYniJ0Rnk

Deep Learning Success

Robotics & AI

AlphaGo

vs

Lee Sedol

8 March 2016

Why Impressed?

- Robotics are considered in controlled environments
  - *Lab settings, predictable positions, standardized tasks*
- In real life situations:
  - *Environments constantly change, new tasks need to be learnt without guidance, unexpected factors must be dealt with*
- Game AI
  - *At least* $10^{10^{48}}$ *possible GO games. Where do we even start?*

DEEP LEARNING in
LANGUAGE & SPEECH

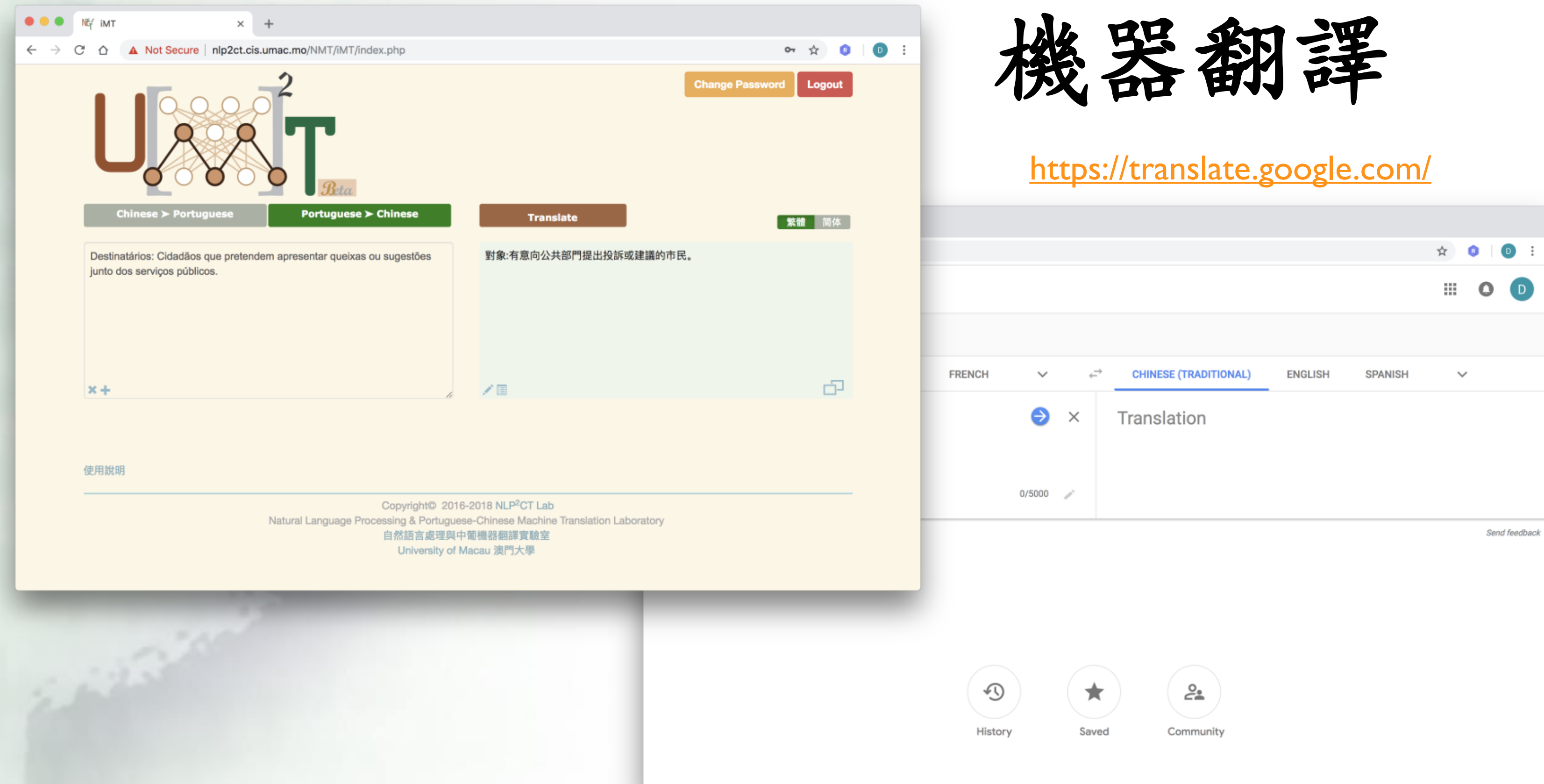
Deep Learning Success
Language & Speech

http://nlp2ct.cis.umac.mo/NMT/iMT/index.php

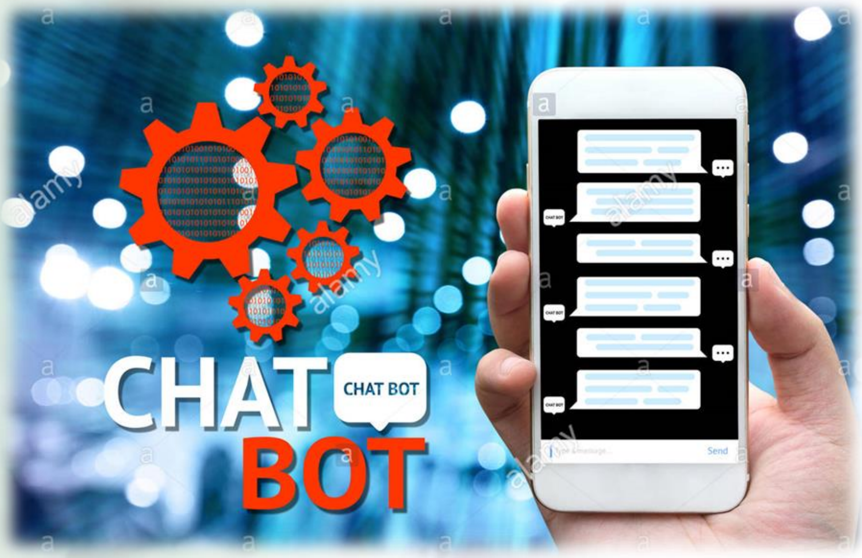
Machine Translation

機器翻譯

https://translate.google.com/



Deep Learning Success
Language & Speech



Chatbots

Deep Learning Success
Language & Speech

Image Captioning & Visual QA

C: Two **cats sit in** a room with a **cat**.
Q1: What animal is in the photo? **A**: cat
Q2: What are the cats doing? **A**: looking out window
Q3: Are these cats sitting or outside? **A**: inside
Q4: What are the cats looking at? **A**: window
GTQ: What animals are shown?
GTQ: How many cats are there?

C: A **train sitting on** the **tracks**.
Q1: What is the yellow object? **A**: train
Q2: Is this train moving or coming? **A**: going
Q3: Is the train in or outside? **A**: outside
Q4: Where is the train? **A**: station
GTQ: What color are the train doors on the right?
GTQ: What shape are the windows?

C: Three **people** are playing **with a large** frisbee.
Q1: Who is holding the frisbee? **A**: boy
Q2: What kind of game are they playing? **A**: frisbee
Q3: What is the man in the blue shirt holding? **A**: frisbee
Q4: What color is the frisbee? **A**: blue

C: A **cat** laying **on a bed** with a **pillow** and a pillow.
Q1: What is on top of the suitcase? **A**: cat
Q2: Is the cat inside or inside? **A**: inside
Q3: What kind of cat is on the left? **A**: gray
Q4: Where is the cat? **A**: suitcase
Q5: What is on the left of the suitcase? **A**: cat

Why Impressed?

- NLP is an extremely complex task

《季姬擊雞記》

季姬寂，集雞，雞即棘雞。棘雞饑嘰，季姬及箕稷濟雞。雞既濟，躋姬笈，季姬忌，急咭雞，雞急，繼圾幾，季姬急，即籍箕擊雞，箕疾擊幾伎，伎即齏，雞嘰集幾基，季姬急極屐擊雞，雞既殛，季姬激，即記《季姬擊雞記》。

Why Impressed?

NLP²CT *Research Group*

- NLP is an extremely complex task
  - *Synonym* (同義): "campus/university", "beautiful/pretty", "answer/reply"
  - *Ambiguities* (歧義): "I made her duck"
- NLP is very high dimensional
  - *Assuming* $100K$ *English words, we need to train* $100K$ *classifiers, with quite sparse data for most of them*
- Beating NLP feels the closest to achieving true AI
  - *Although true AI probably goes beyond NLP, Vision, Robotics,..., alone*

Deep Learning in

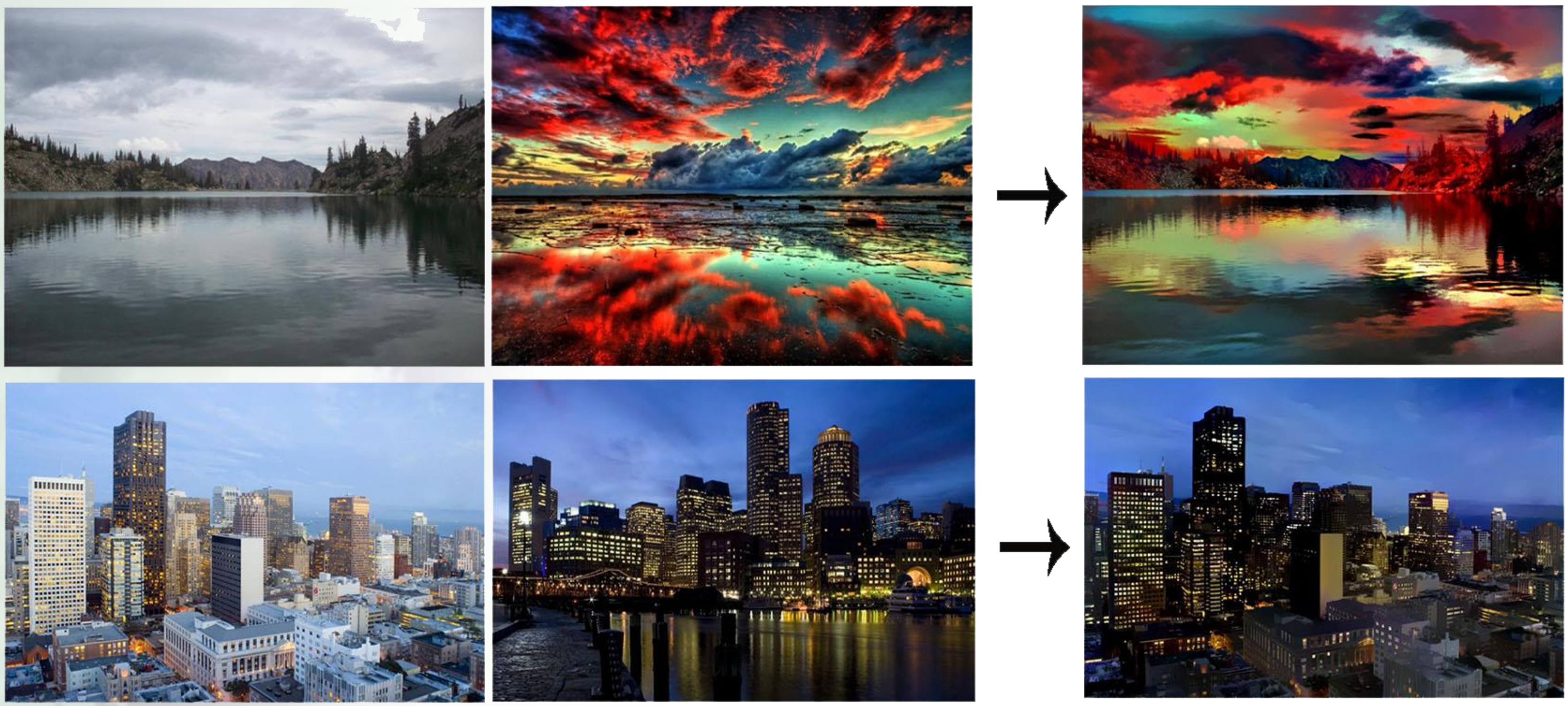
Music & Arts

Deep Learning Success
Music & Arts

Style Transfer

Deep Learning Success
Music & Arts

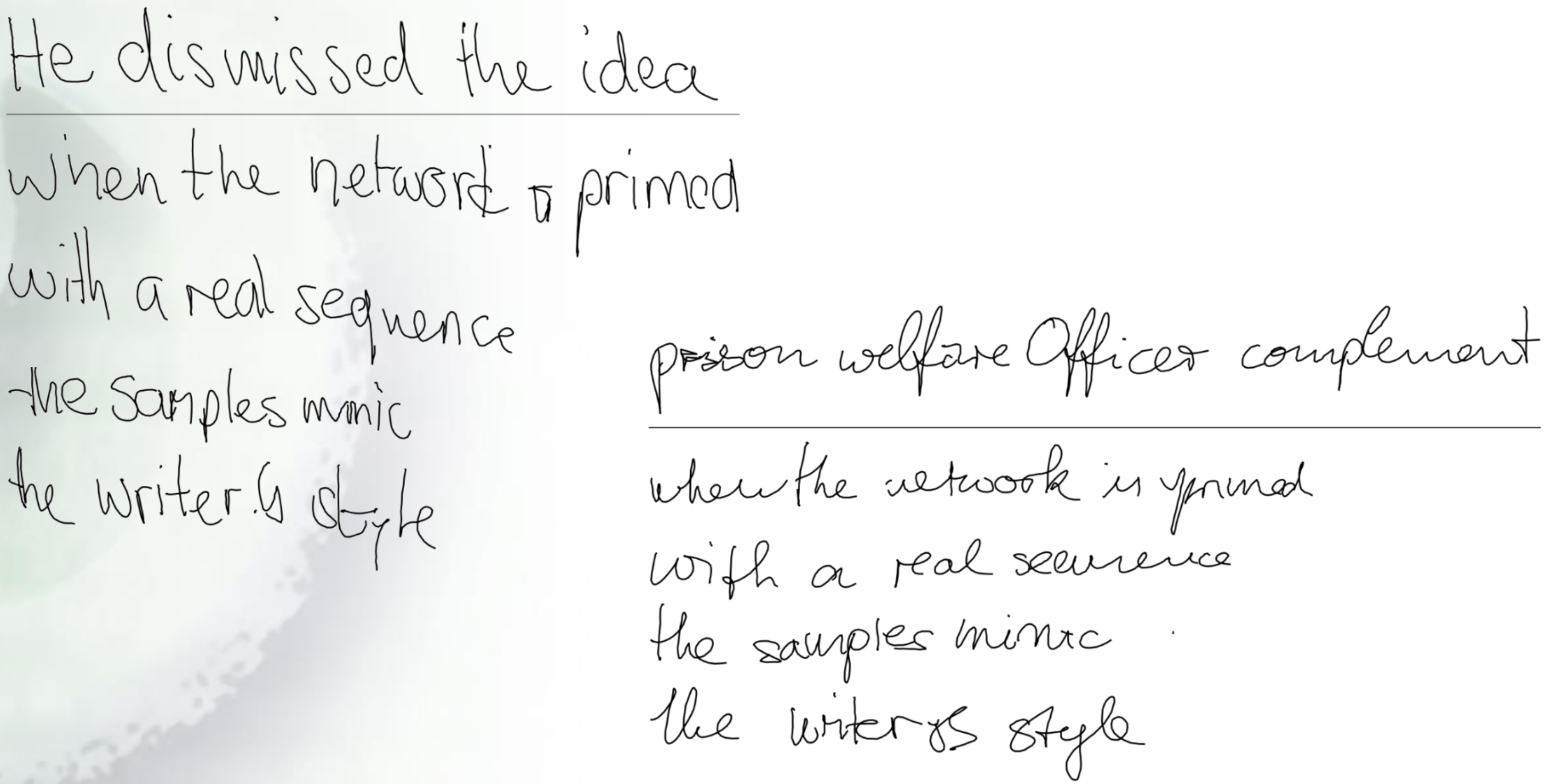
Style Transfer



Original Photo | Example Photo | Result

Deep Learning Success
Music & Arts

Handwriting

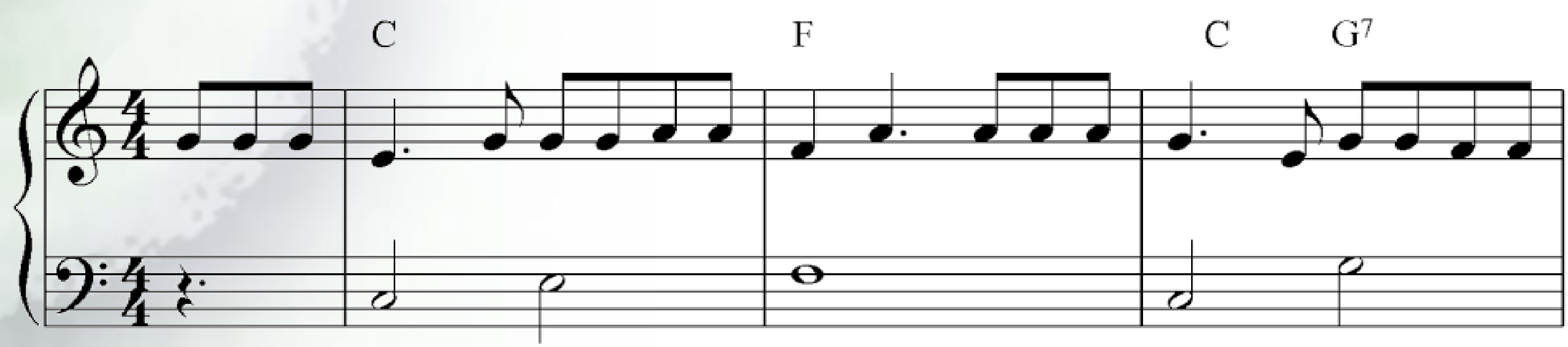


(Graves 2013)

Deep Learning Success
Music & Arts

NLP²CT *Research Group*

Music Generation



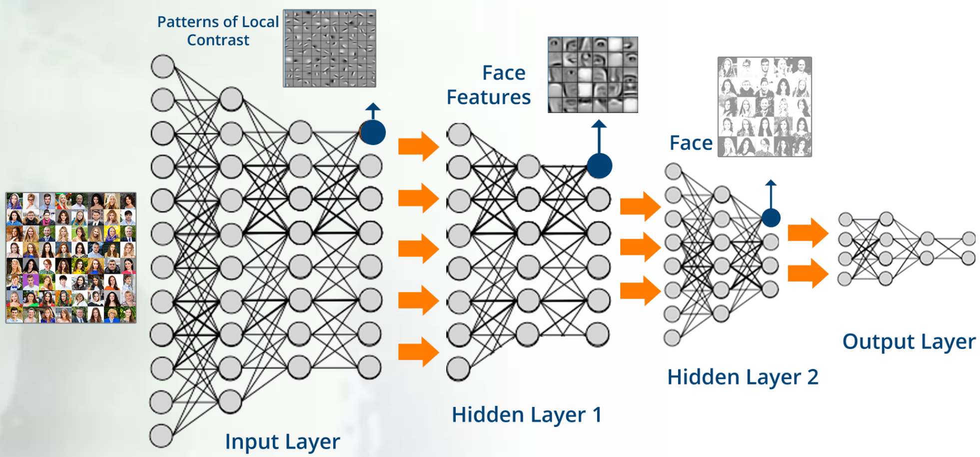
Why Impressed?

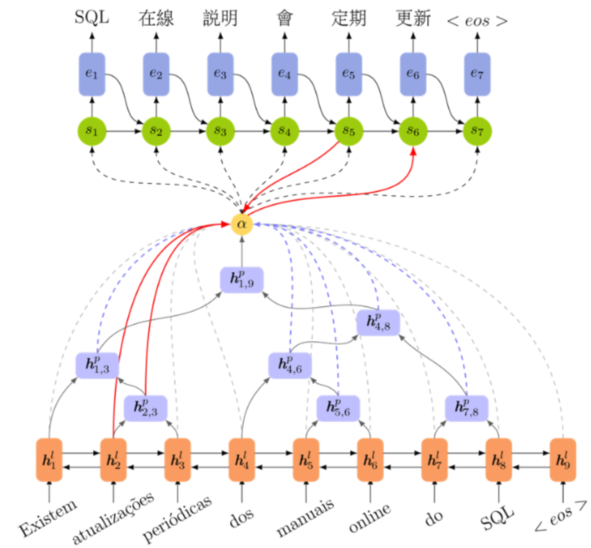
- Music, painting, writing, etc. are uniquely human
  - *Difficult to model*
  - *Even more difficult to evaluate*
- If machines can generate novel pieces even resembling art, they must have understand something about "*beauty*", "*harmony*", etc.
- Have they really learned to generate new art?
  - *Or do they just fool us with their tricks?*

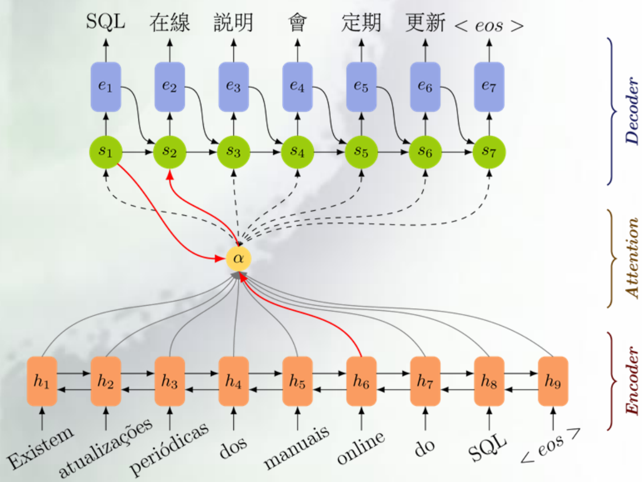
The Magic Behind the Stories

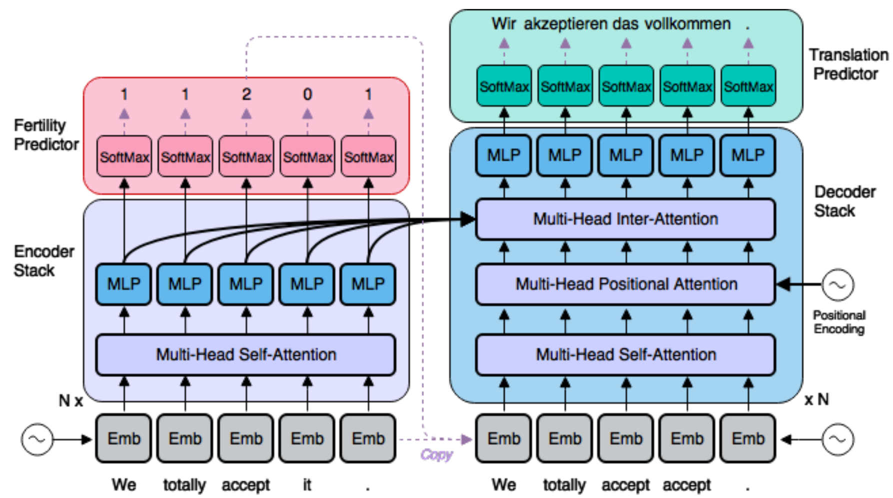
Deep Neural Networks

Deep Neural Networks



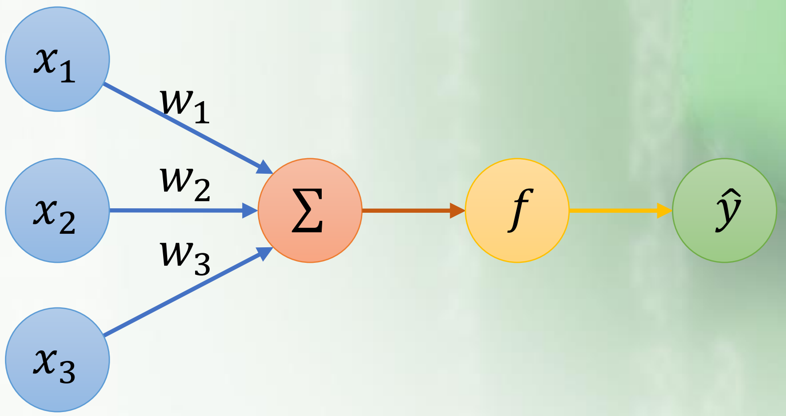






Neural Networks

The Neuron (神經元)

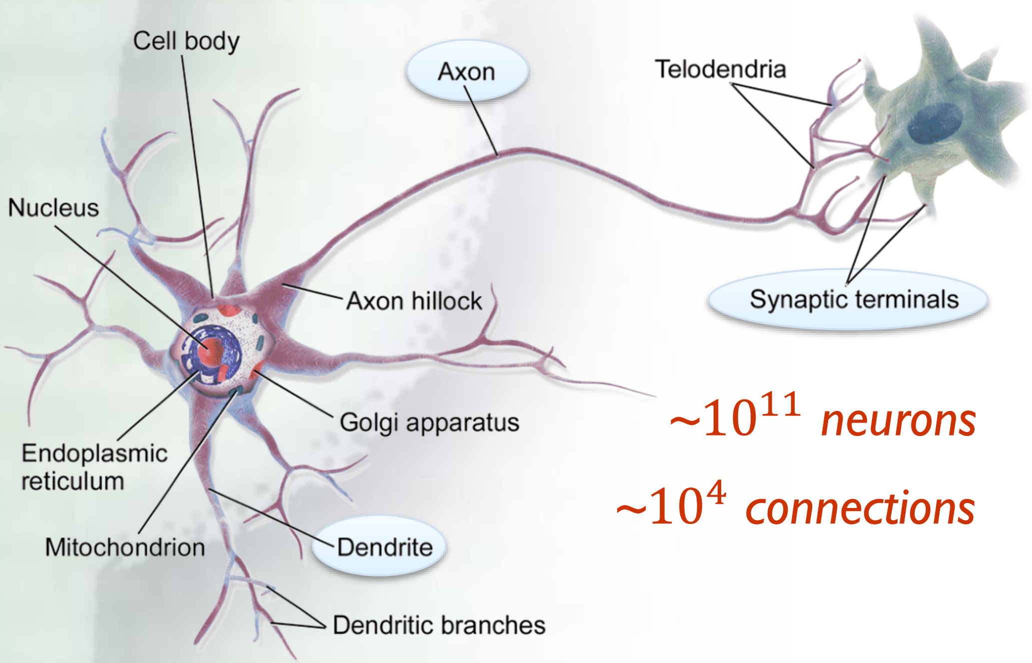


Nervous System

- The nervous system (神經系統) coordinates the actions of an animal
- Brain $\xleftrightarrow{\text{messages}}$ body parts
- The basic unit of the nervous system is the *neuron* (神經元)

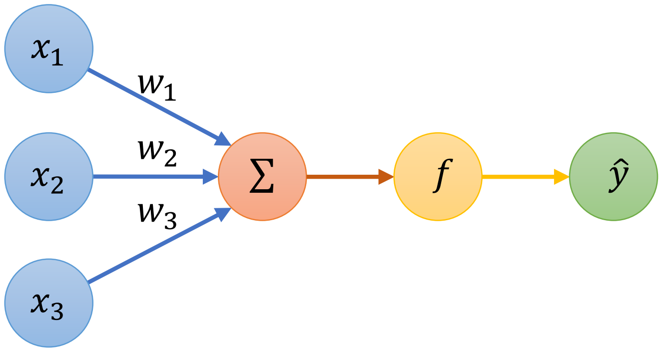
Neurons

- A neuron receives input from other *neurons* (generally thousands) from its *synapses* (突觸)
- Inputs are approximately *summed*
- When the input exceeds a *threshold* the neuron *sends an electrical spike* that travels from the body, down the *axon* (軸突), to the next neuron(s)

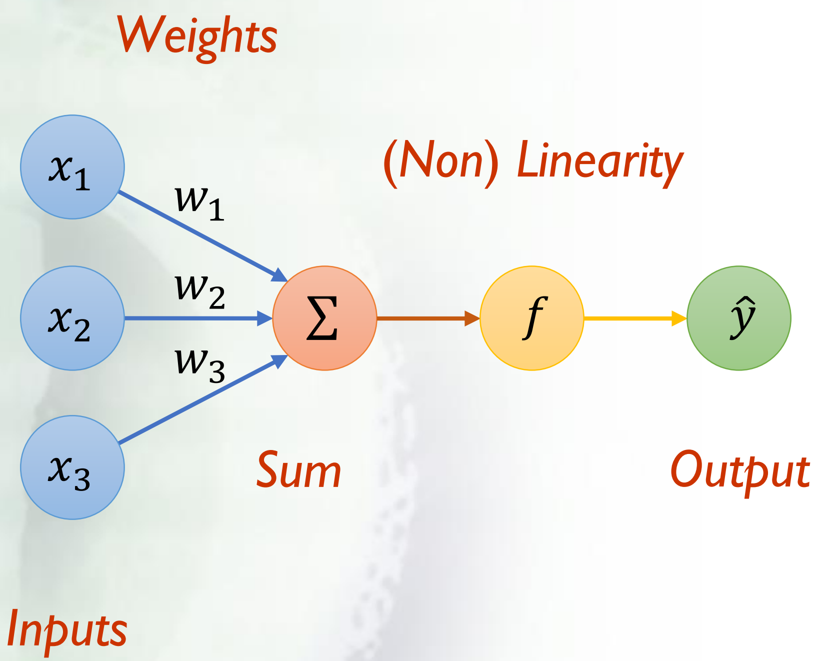


$\sim 10^{11}$ *neurons*

$\sim 10^{4}$ *connections*



The Perceptron



Linear combination of inputs

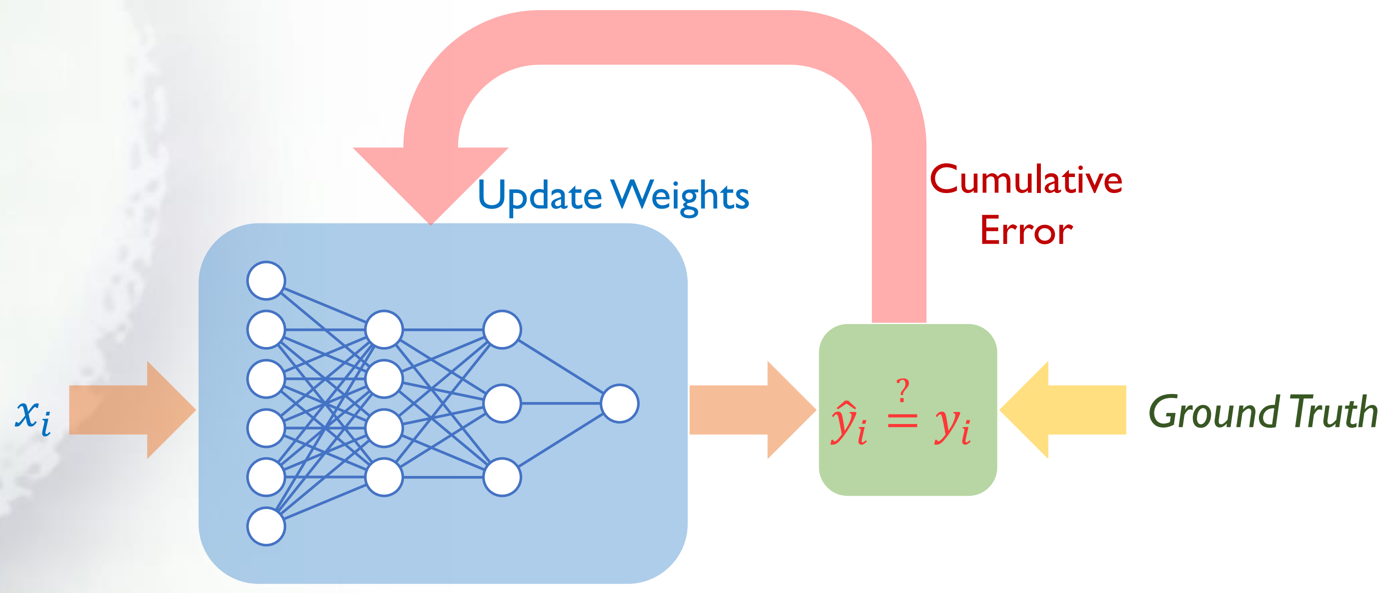
$$\hat{y} = f\left(\sum_{i=1}^{3} x_i\, w_i\right)$$

(Non) linear activation function

Supervised Learning (監督學習)

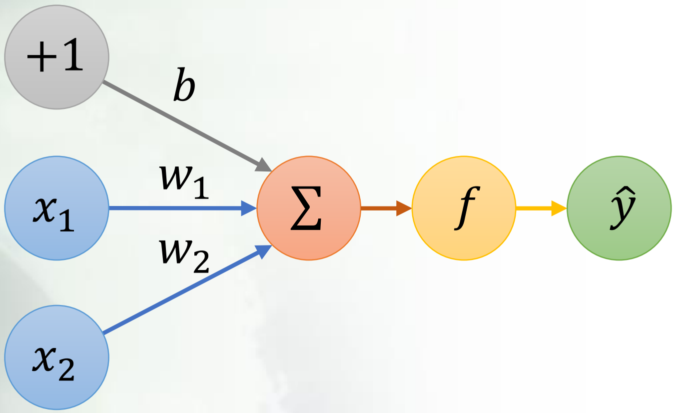
NLP²CT *Research Group*

- *Training by example*, i.e., priori known desired output for each input pattern
- Particularly useful for *feedforward networks* (前饋網路)



Training data: $\mathcal{D} = \langle x_1, y_1 \rangle, \ldots, \langle x_m, y_m \rangle$

Supervised Learning
An Example

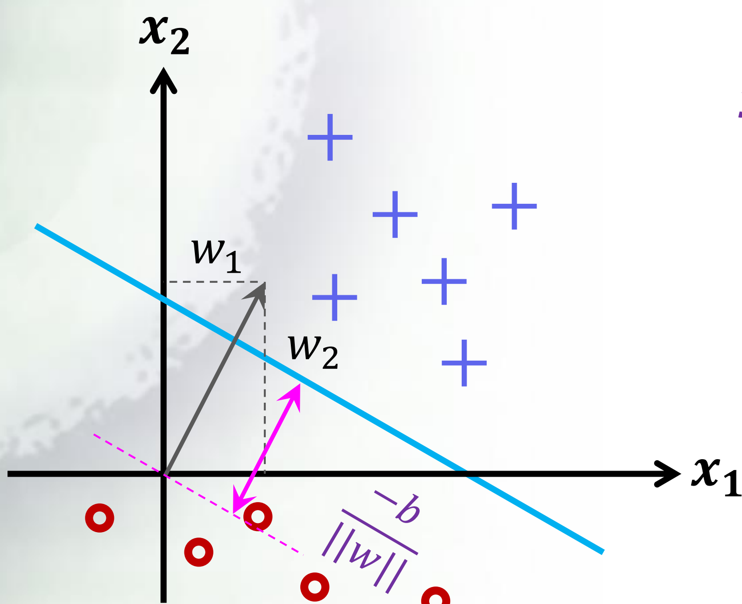


$$\hat{y} = f\left(b + \sum_{i=1}^{2} x_i w_i\right)$$

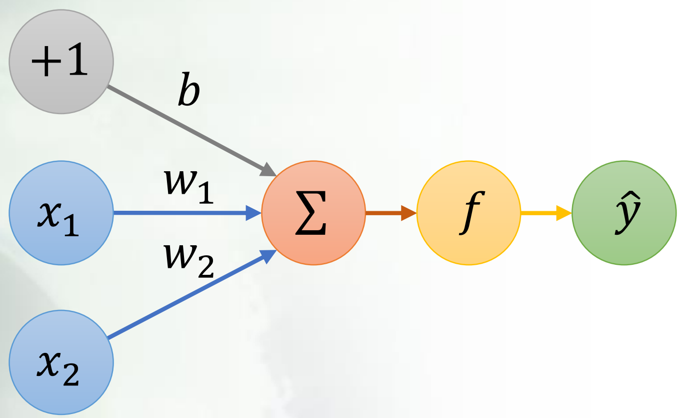
$$f(s) = \begin{cases} +1 & \text{if } s > 0 \\ -1 & \text{otherwise} \end{cases}$$

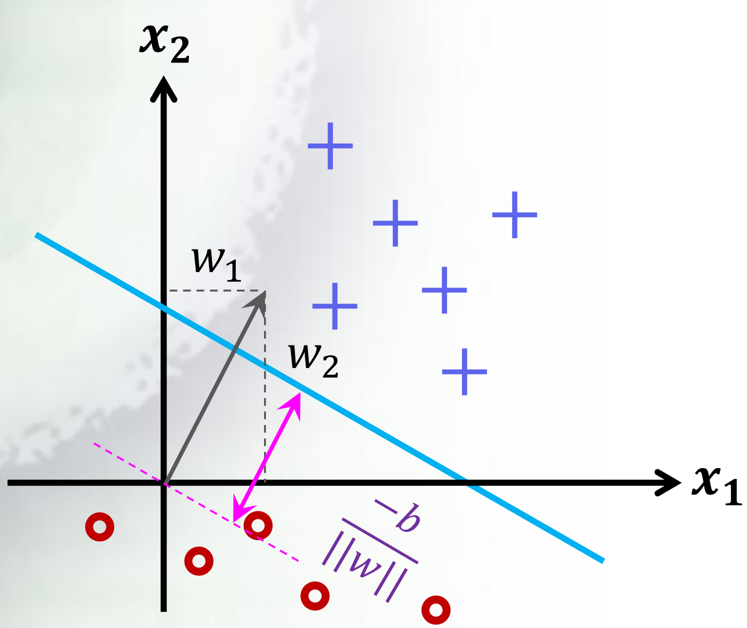
$$x_1 w_1 + x_2 w_2 + b = 0$$

$$x_2 = -\frac{w_1}{w_2} x_1 - \frac{b}{w_2}$$



Supervised Learning
An Example





Update the weights

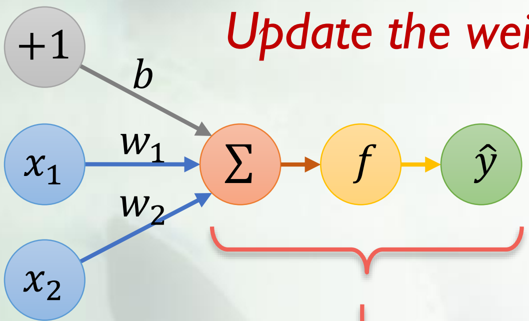
$$w_i(t+1) = w_i(t) + \Delta w_i(t)$$

$$\Delta w_i(t) = y \cdot x_i$$

$$b(t+1) = b(t) + \Delta b(t)$$

$$\Delta b(t) = \begin{cases} 0 & \text{if } \hat{y} = y \\ y & \text{otherwise} \end{cases}$$

Supervised Learning
An Example



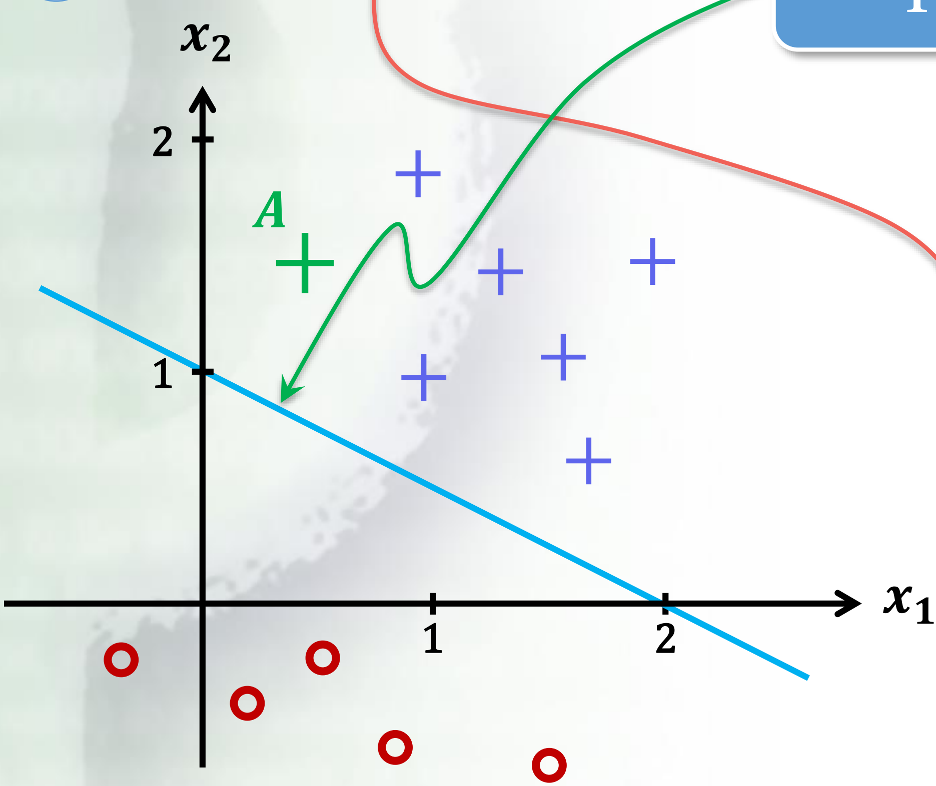
Update the weights $w_i(t+1) = w_i(t) + \Delta w_i(t)$ $\quad b(t+1) = b(t) + \Delta b(t)$

$\Delta w_i(t) = y \cdot x_i$

$$\Delta b(t) = \begin{cases} 0 & \text{if } \hat{y} = y \\ y & \text{otherwise} \end{cases}$$

$\mathbf{w_1 = 1,\ w_2 = 2,\ b = -2}$

$\mathbf{x_A = (0.5, 1.5),\ y = +1}$

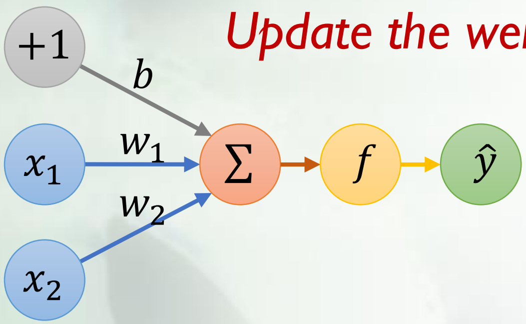


$$\hat{y} = f\left(b + \sum_{i=1}^{2} x_i w_i\right)$$

$$\hat{y} = f(-2 + 0.5 \times 1 + 1.5 \times 2)$$

$$\hat{y} = +1 \quad \boxed{= y}$$

Supervised Learning
An Example



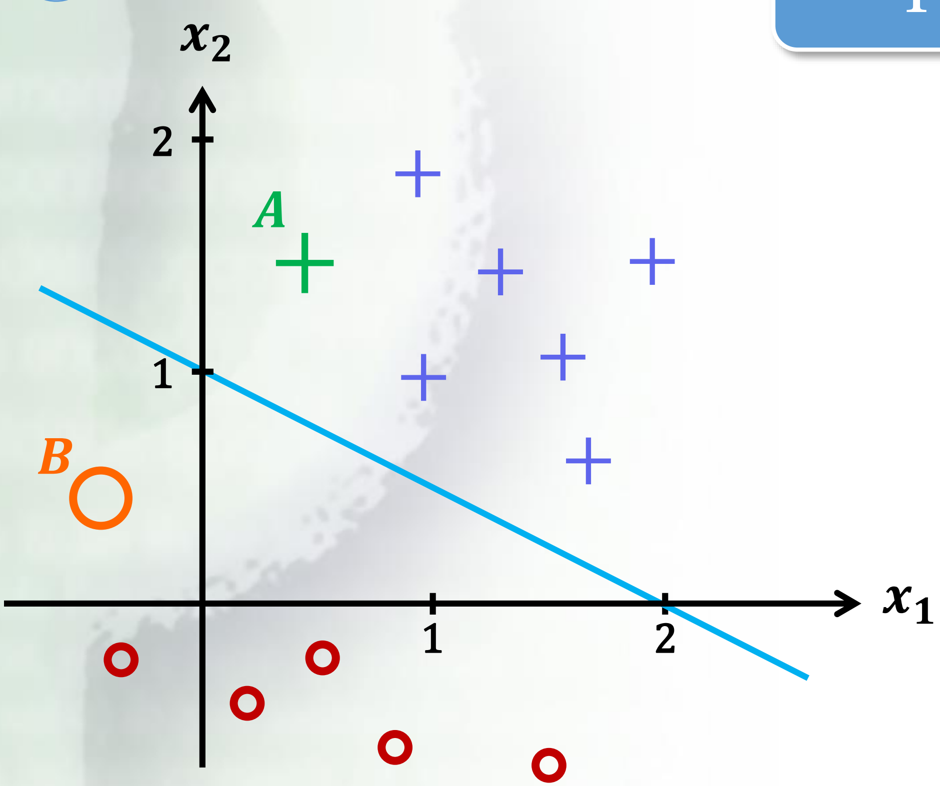
Update the weights $w_i(t+1) = w_i(t) + \Delta w_i(t)$ $\quad b(t+1) = b(t) + \Delta b(t)$

$$\Delta w_i(t) = y \cdot x_i$$

$$\Delta b(t) = \begin{cases} 0 & \text{if } \hat{y} = y \\ y & \text{otherwise} \end{cases}$$

$\boldsymbol{w_1 = 1,\ w_2 = 2,\ b = -2}$

$\mathbf{x}_B = (-0.5, 0.5),\ y = -1$

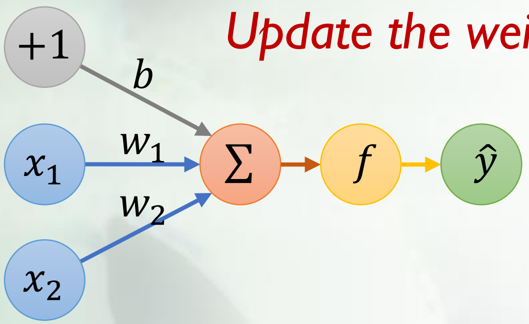


$$\hat{y} = f\left(b + \sum_{i=1}^{2} x_i w_i\right)$$

$$\hat{y} = f(-2 - 0.5 \times 1 + 0.5 \times 2)$$

$$\hat{y} = -1 \ \boxed{= y}$$

Supervised Learning
An Example



Update the weights $w_i(t+1) = w_i(t) + \Delta w_i(t)$ $\quad b(t+1) = b(t) + \Delta b(t)$

$$\Delta w_i(t) = y \cdot x_i$$

$$\Delta b(t) = \begin{cases} 0 & \text{if } \hat{y} = y \\ y & \text{otherwise} \end{cases}$$

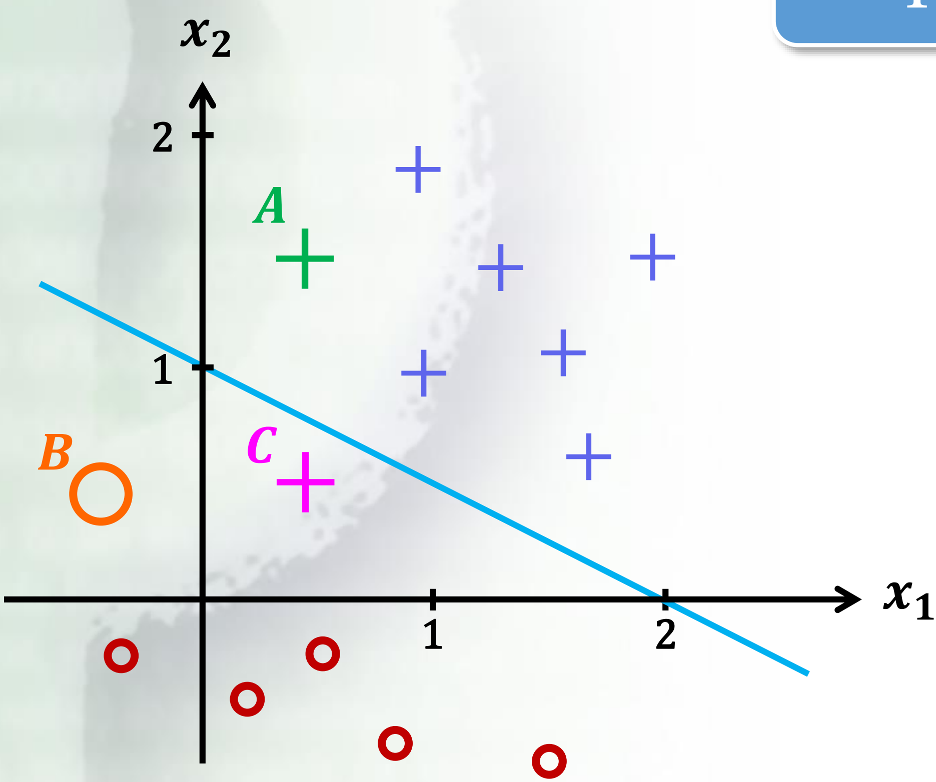
$$\boldsymbol{w_1 = 1,\ w_2 = 2,\ b = -2}$$

$$\mathbf{x}_C = (0.5, 0.5),\ y = +1$$

$$\hat{y} = f\left(b + \sum_{i=1}^{2} x_i w_i\right)$$

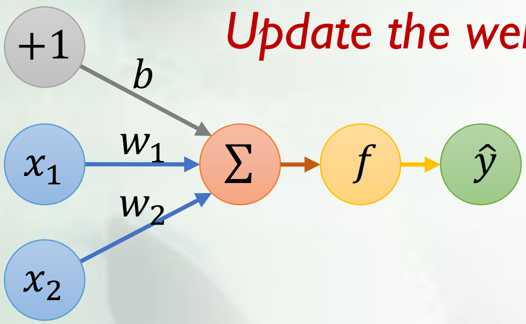
$$\hat{y} = f(-2 + 0.5 \times 1 + 0.5 \times 2)$$

$$\hat{y} = -1 \ \boxed{\neq y}$$



Supervised Learning

An Example



Update the weights $w_i(t+1) = w_i(t) + \Delta w_i(t)$ $b(t+1) = b(t) + \Delta b(t)$

$$\Delta w_i(t) = y \cdot x_i$$

$$\Delta b(t) = \begin{cases} 0 & \text{if } \hat{y} = y \\ y & \text{otherwise} \end{cases}$$

$$\boldsymbol{w_1 = 1.5,\ w_2 = 2.5,\ b = -1}$$

$$\mathbf{x}_C = (0.5, 0.5),\ y = +1$$

$$\Delta w_1(t) = y \cdot x_1 = +1 \times 0.5 = 0.5$$

$$w_1(t+1) = 1 + 0.5 = 1.5$$

$$\Delta w_2(t) = y \cdot x_2 = +1 \times 0.5 = 0.5$$

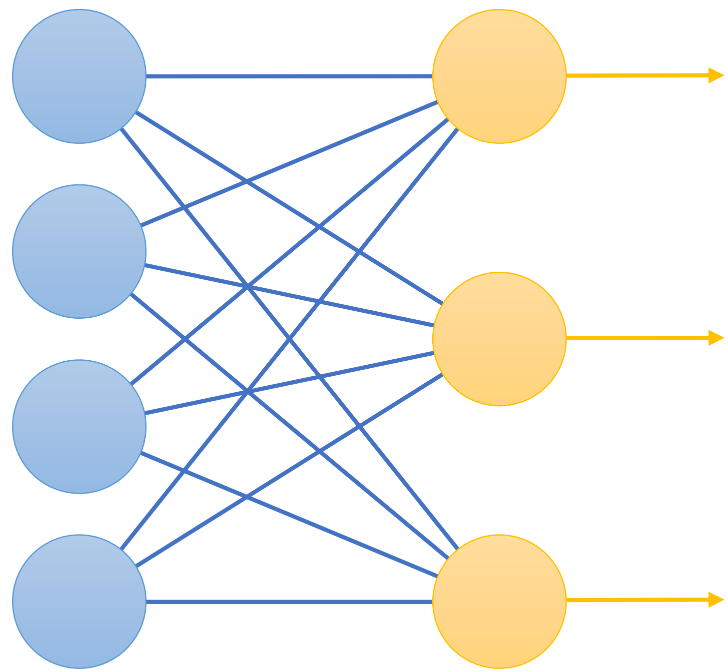
$$w_2(t+1) = 2 + 0.5 = 2.5$$

$$\Delta b(t) = y = 1$$

$$b(t+1) = -2 + 1 = -1$$

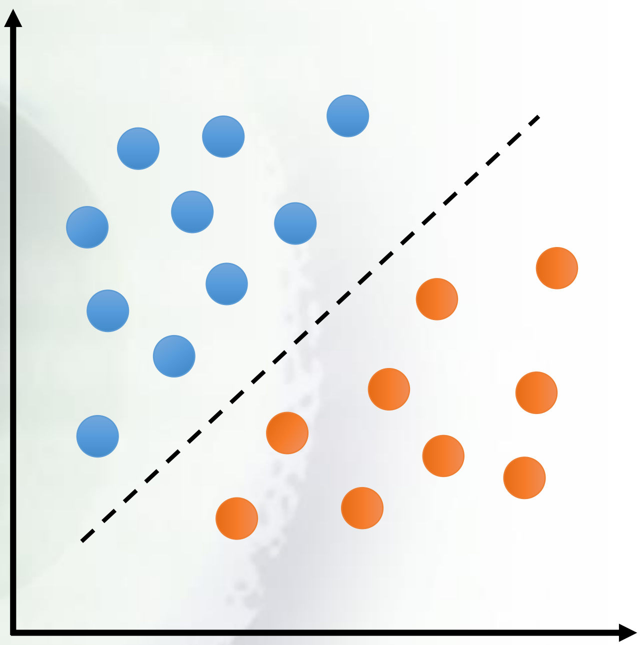
From Perceptron
To a Neural Network

- One *perceptron* = one *decision*
- What about *multiple decisions*?
  - *e.g. Digit classification, letter classification*
- Stack as *many outputs* as the possible *outcomes* into a layer
  - *Neural network*
- User *one* layer as input to the *next* layer
  - *Multi-layer perceptron (MLP – 多層感知神經網絡)*

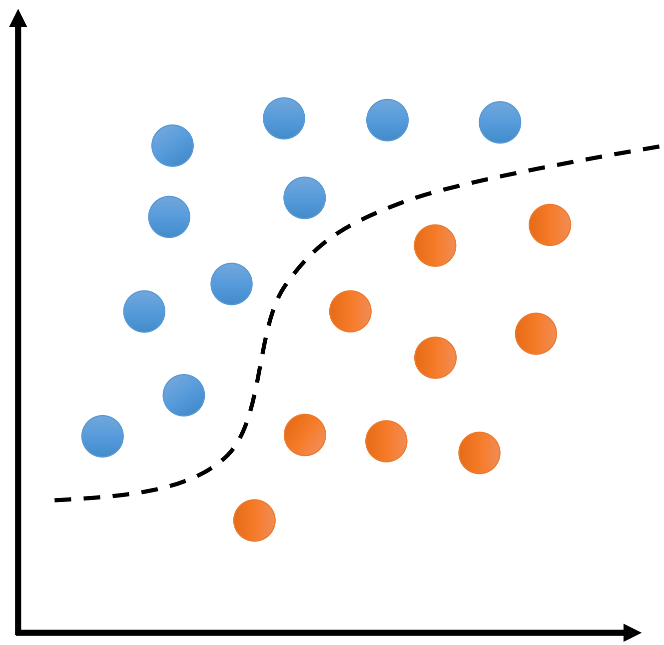


Linear *vs* Non-linear

Linear Separable



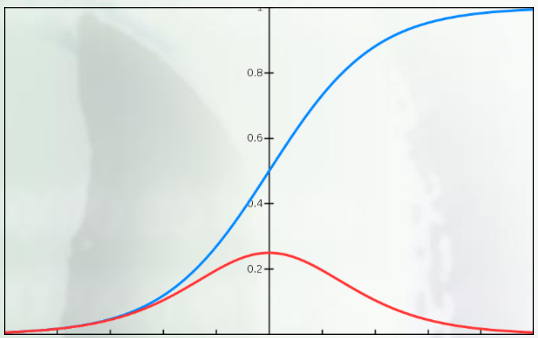
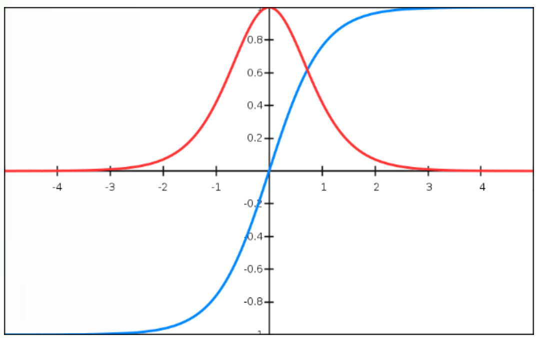
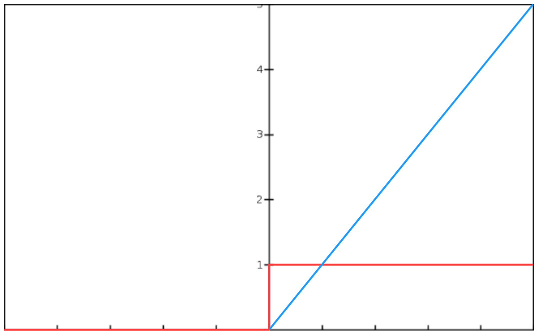
Linear Non-separable



$$\hat{y} = f\left(\sum_{i=1}^{m} x_i\, w_i\right) = \begin{cases} +1 & \text{if } z = \sum_{i=1}^{m} x_i\, w_i > 0 \\ -1 & \text{otherwise} \end{cases}$$

$f(\cdot) = \text{Signum}$

Common Activation Functions

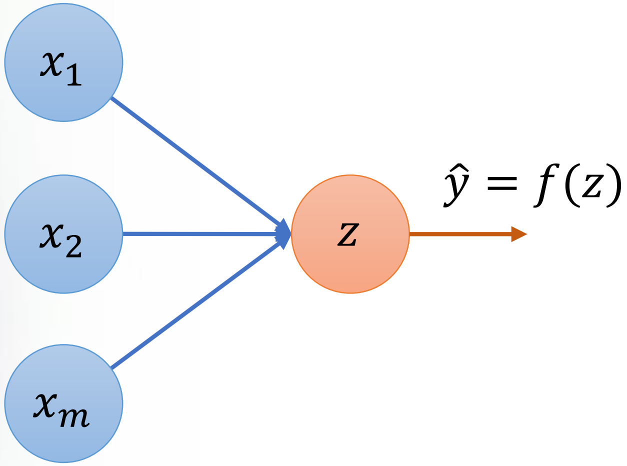
Sigmoid Function	*Hyperbolic Tangent* (雙曲正切)	*Rectified Linear Unit (ReLU)* (線性整流函數)
		
$f(z) = \dfrac{1}{1+e^{-z}}$	$f(z) = \dfrac{e^{z}-e^{-z}}{e^{z}+e^{-z}}$	$f(z) = \max(0, z)$
$f'(z) = f(z)\big(1 - f(z)\big)$	$f'(z) = 1 - f(z)^2$	$f'(z) = \begin{cases} 1 & \text{if } z > 0 \\ 0 & \text{otherwise} \end{cases}$

All activation functions are non-linear

Building Neural Network

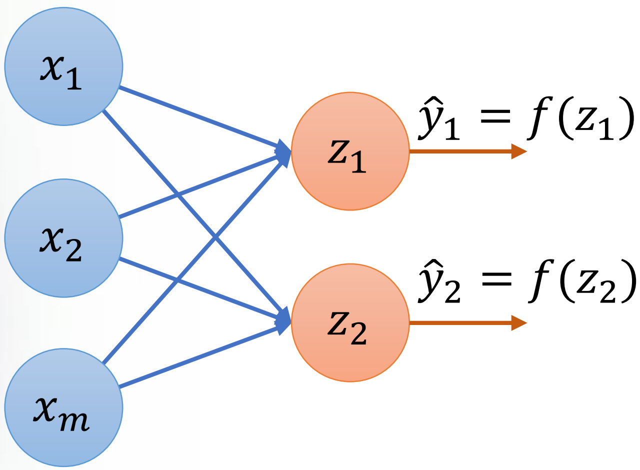
with Perceptron

The Perceptron Simplified



$$z = w_0 + \sum_{i=1}^{m} x_i \, w_i$$

Multi Output Perceptron



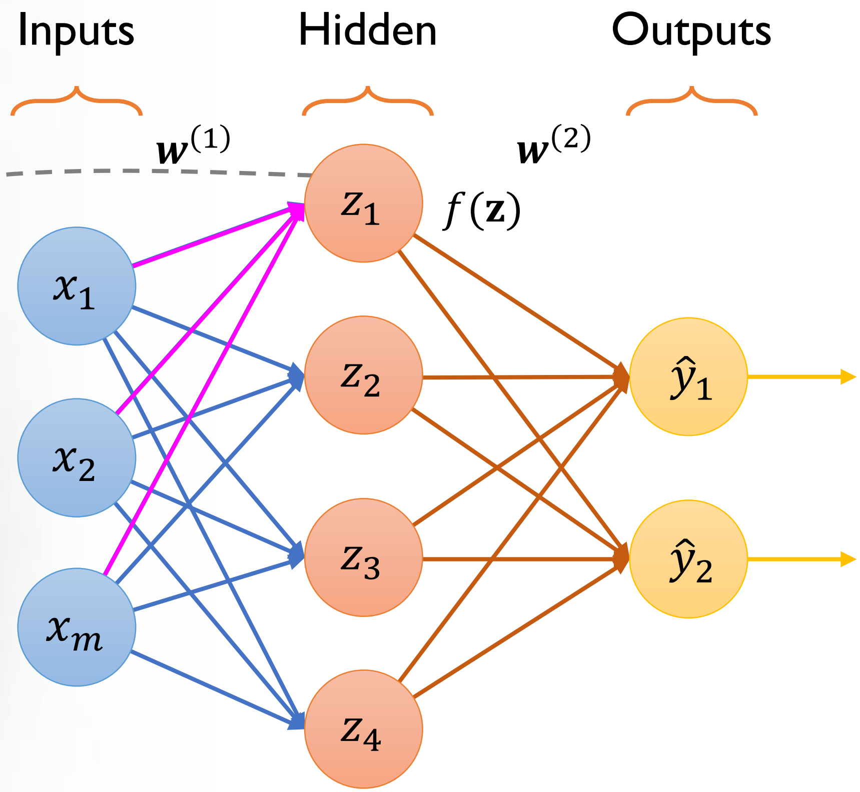
$$z_j = w_{0,j} + \sum_{i=1}^{m} x_i \, w_{i,j}$$

Single Layer Network

$$z_{1} = w_{0,1}^{(1)} + \sum_{i=1}^{m} x_i w_{i,1}^{(1)}$$

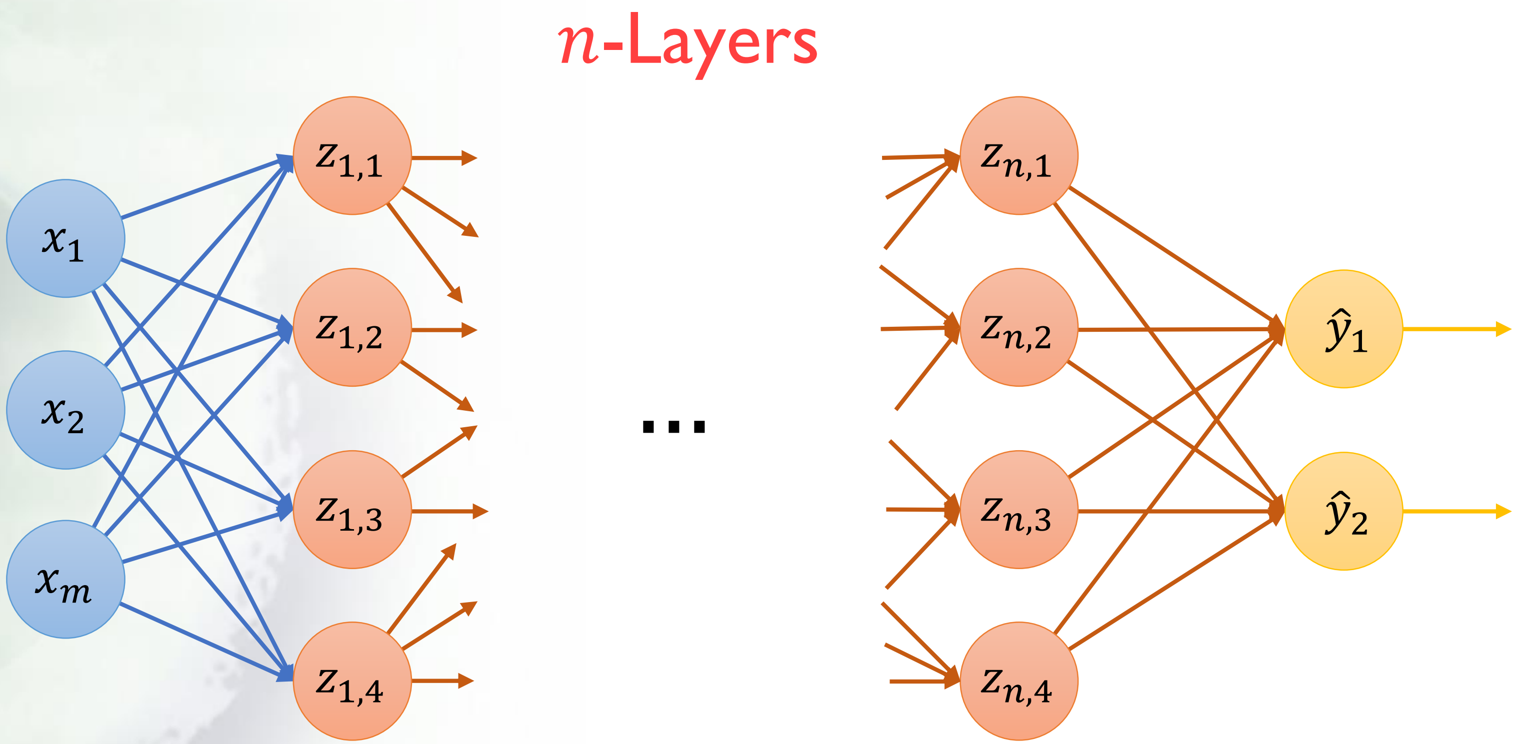
$$= w_{0,1}^{(1)} + x_1 w_{1,1}^{(1)}$$

$$+ x_2 w_{2,1}^{(1)} + x_3 w_{3,1}^{(1)}$$



$$z_j = w_{0,j}^{(1)} + \sum_{i=1}^{m} x_i w_{i,j}^{(1)} \qquad y_j = w_{0,j}^{(2)} + \sum_{i=1}^{m} x_i w_{i,j}^{(2)}$$

Deep Neural Network



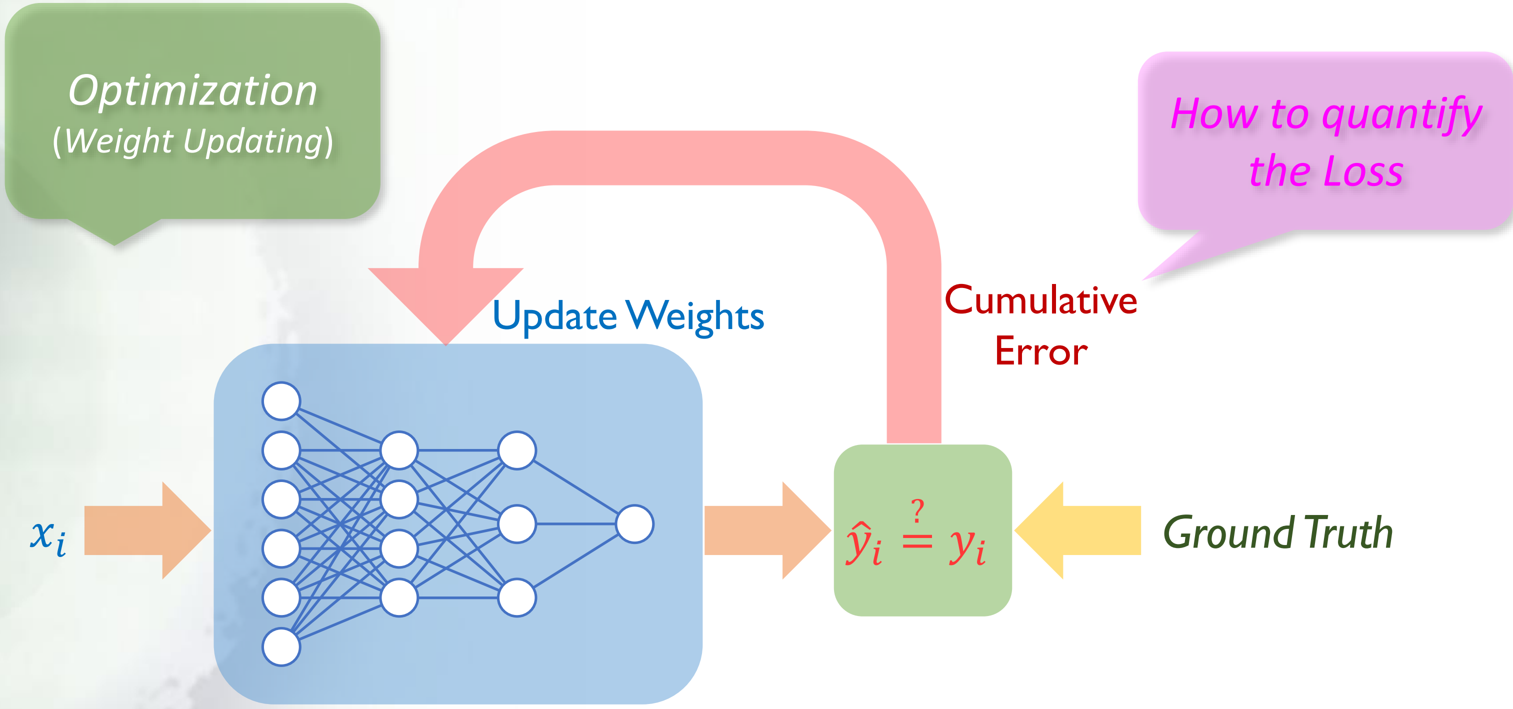
$$z_j = w_{0,j}^{(k)} + \sum_{i=1}^{m_{k-1}} f(z_{k-1}, i) w_{i,j}^{(k)}$$

Training the Model

Loss Function & Optimization

Training Algorithm

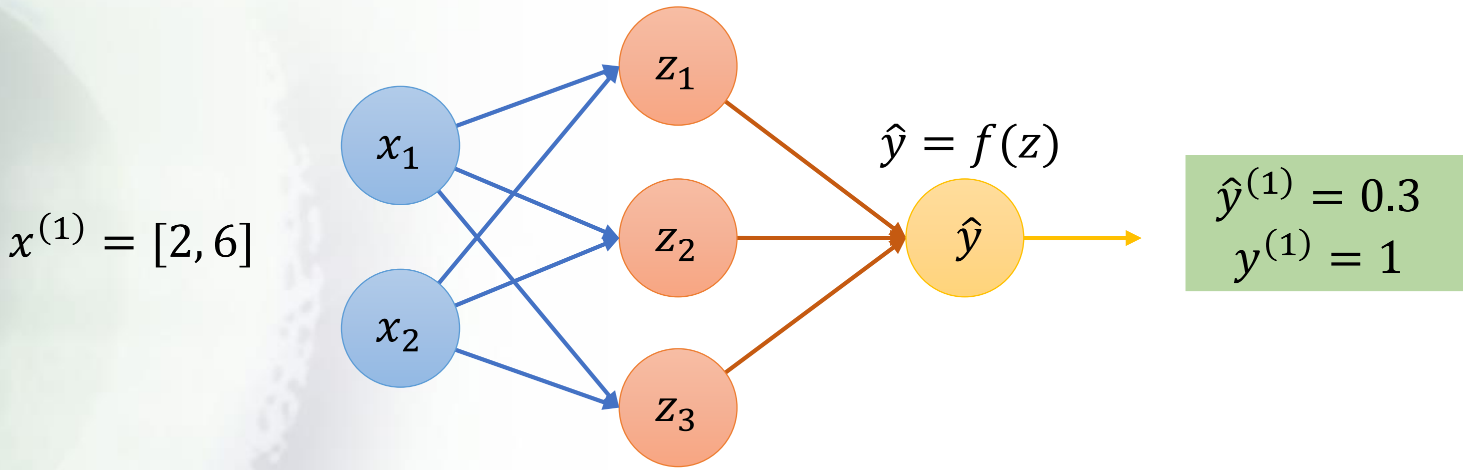
The Deep Networks



Training data: $\mathcal{D} = \langle x_1, y_1 \rangle, \dots, \langle x_m, y_m \rangle$

Quantifying Loss

- *The loss of network measures the cost incurred from incorrect predictions*



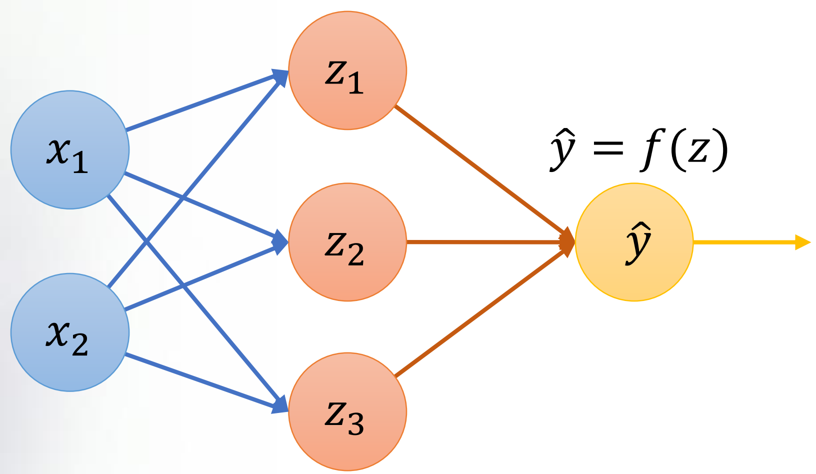
$$\mathcal{L}\Big(\underbrace{f\big(x^{(i)}; \boldsymbol{w}\big)}_{\text{Predicted}}, \underbrace{y^{(i)}}_{\text{Actual}}\Big)$$

Quantifying Loss

Empirical Loss

- *The* ***empirical loss*** *(經驗損失) measures the total loss over the entire dataset*

$$x = \begin{bmatrix} 2 & 6 \\ 9 & 4 \\ 3 & 1 \end{bmatrix}$$



$$\begin{array}{ccc} \hat{y} & & y \\ \begin{bmatrix} 0.2 \\ 0.9 \\ 0.7 \end{bmatrix} & \begin{array}{c} ✘ \\ ✘ \\ ✓ \end{array} & \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \end{array}$$

As known as:

- *Objective Function*
- *Cost Function*
- *Empirical Risk*

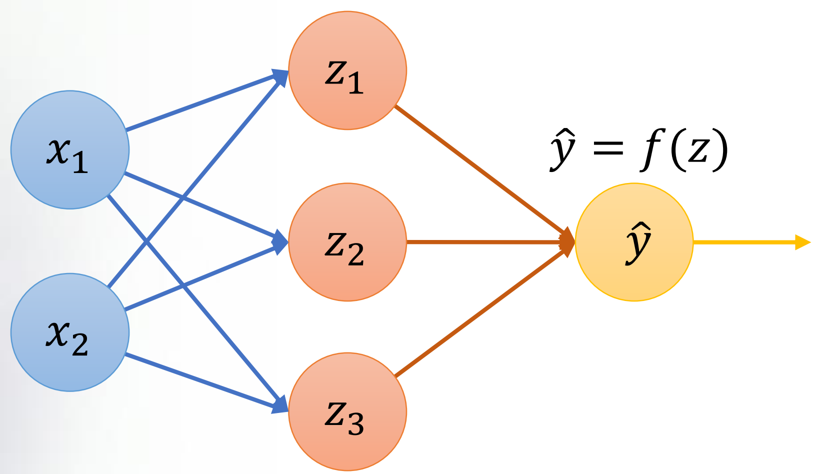
$$\mathcal{J}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}\Big(\underbrace{f\big(x^{(i)}; \boldsymbol{w}\big)}_{\text{Predicted}}, \underbrace{y^{(i)}}_{\text{Actual}}\Big)$$

Quantifying Loss

Binary Cross Entropy Loss

- *Cross Entropy Loss (交叉熵) can be used with models that output a probability between 0 and 1*

$$x = \begin{bmatrix} 2 & 6 \\ 9 & 4 \\ 3 & 1 \end{bmatrix}$$



$$\hat{y} = \begin{bmatrix} 0.2 \\ 0.9 \\ 0.7 \end{bmatrix} \quad \begin{matrix} ✘ \\ ✘ \\ ✓ \end{matrix} \quad y = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$\mathcal{J}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \underbrace{y^{(i)}}_{\text{Actual}} \log\left(\underbrace{f\left(x^{(i)}; \boldsymbol{w}\right)}_{\text{Predicted}}\right) + \left(1 - \underbrace{y^{(i)}}_{\text{Actual}}\right) \log\left(1 - \underbrace{f\left(x^{(i)}; \boldsymbol{w}\right)}_{\text{Predicted}}\right)$$

Predicted

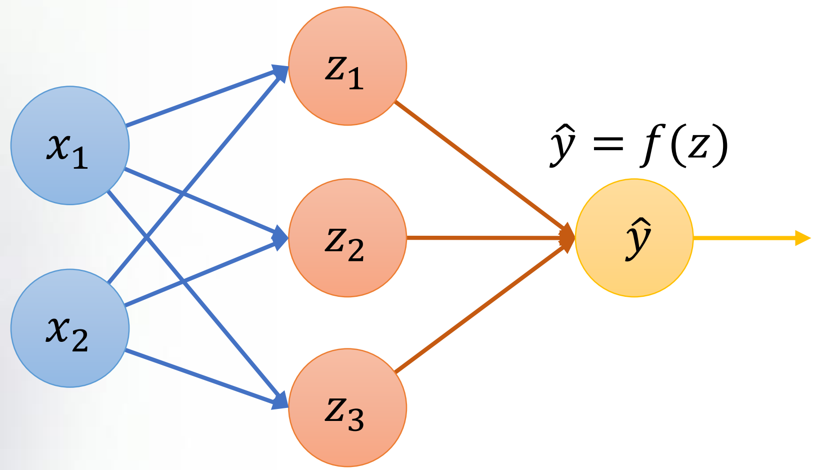
Actual

Quantifying Loss

Mean Squared Error Loss

- *Mean Squared Error Loss (均方差) can be used with regression models that output continuous real numbers*

$$x = \begin{bmatrix} 2 & 6 \\ 9 & 4 \\ 3 & 1 \end{bmatrix}$$

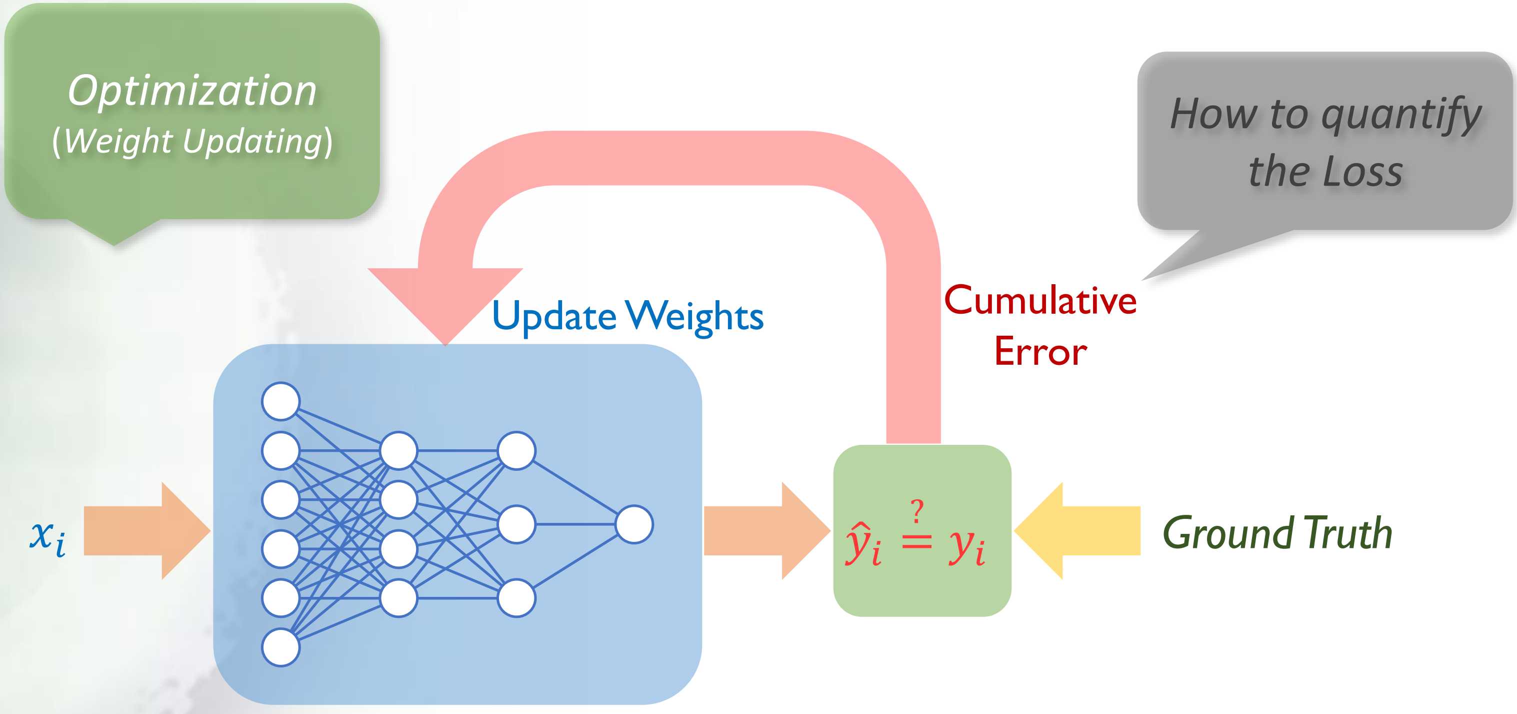


$$\begin{array}{ccc} \hat{y} & & y \\ \begin{bmatrix} 20 \\ 90 \\ 70 \end{bmatrix} & \begin{matrix} ✘ \\ ✘ \\ ✔ \end{matrix} & \begin{bmatrix} 90 \\ 30 \\ 75 \end{bmatrix} \end{array}$$

$$\mathcal{J}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\Big(\underbrace{y^{(i)}}_{\text{Actual}} - \underbrace{f\big(x^{(i)}; \mathbf{w}\big)}_{\text{Predicted}}\Big)^2$$

Training Algorithm

The Deep Networks



Training data: $\mathcal{D} = \langle x_1, y_1 \rangle, \dots, \langle x_m, y_m \rangle$

Loss Optimization

- *We want to find the network weights that achieve the lowest loss*

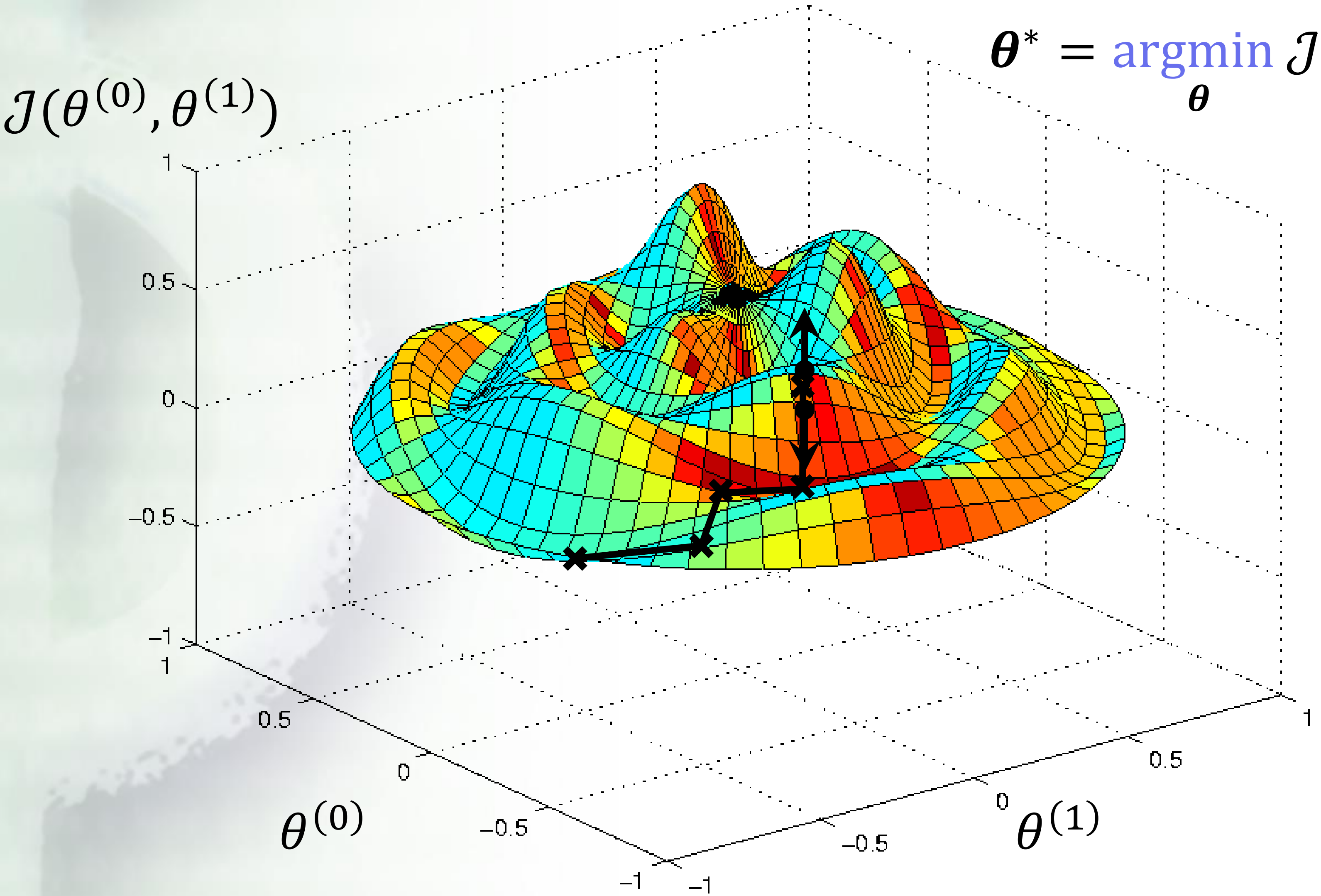
$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}\left(f\left(x^{(i)}; \boldsymbol{\theta}\right), y^{(i)}\right)$$

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, \mathcal{J}(\boldsymbol{\theta})$$

$$\boldsymbol{\theta} = \{\theta^{(0)}, \theta^{(1)}, \dots\}$$

Loss Optimization

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \mathcal{J}(\boldsymbol{\theta})$$

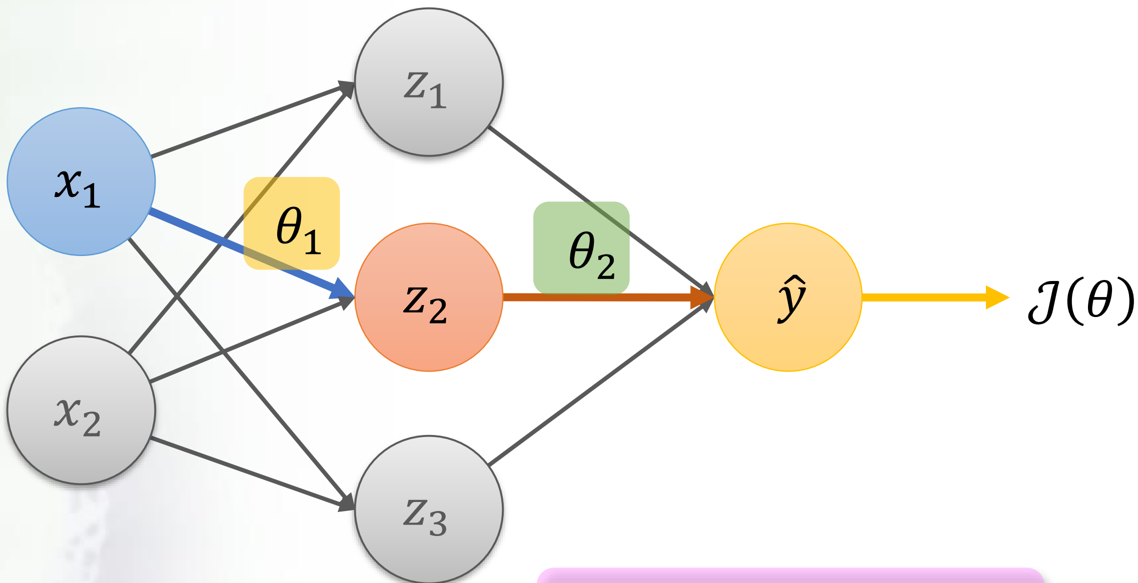


Gradient Descent 梯度下降

Algorithm

- Initialize weights randomly $\sim\mathcal{N}(0,\sigma^2)$
- Loop until convergence
  - Compute gradient: $\frac{\partial \mathcal{J}(\theta)}{\partial \theta}$
  - Update weights: $\theta \leftarrow \theta - \alpha \frac{\partial \mathcal{J}(\theta)}{\partial \theta}$
- Return weights

Compute Gradients
Backpropagation (反向傳播)



Chain Rule (連鎖法則)!

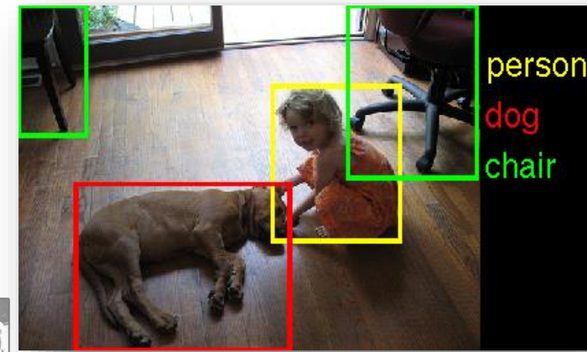
$$\frac{\partial \mathcal{J}(\boldsymbol{\theta})}{\partial \theta_2} = \frac{\partial \mathcal{J}(\boldsymbol{\theta})}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial \theta_2}$$

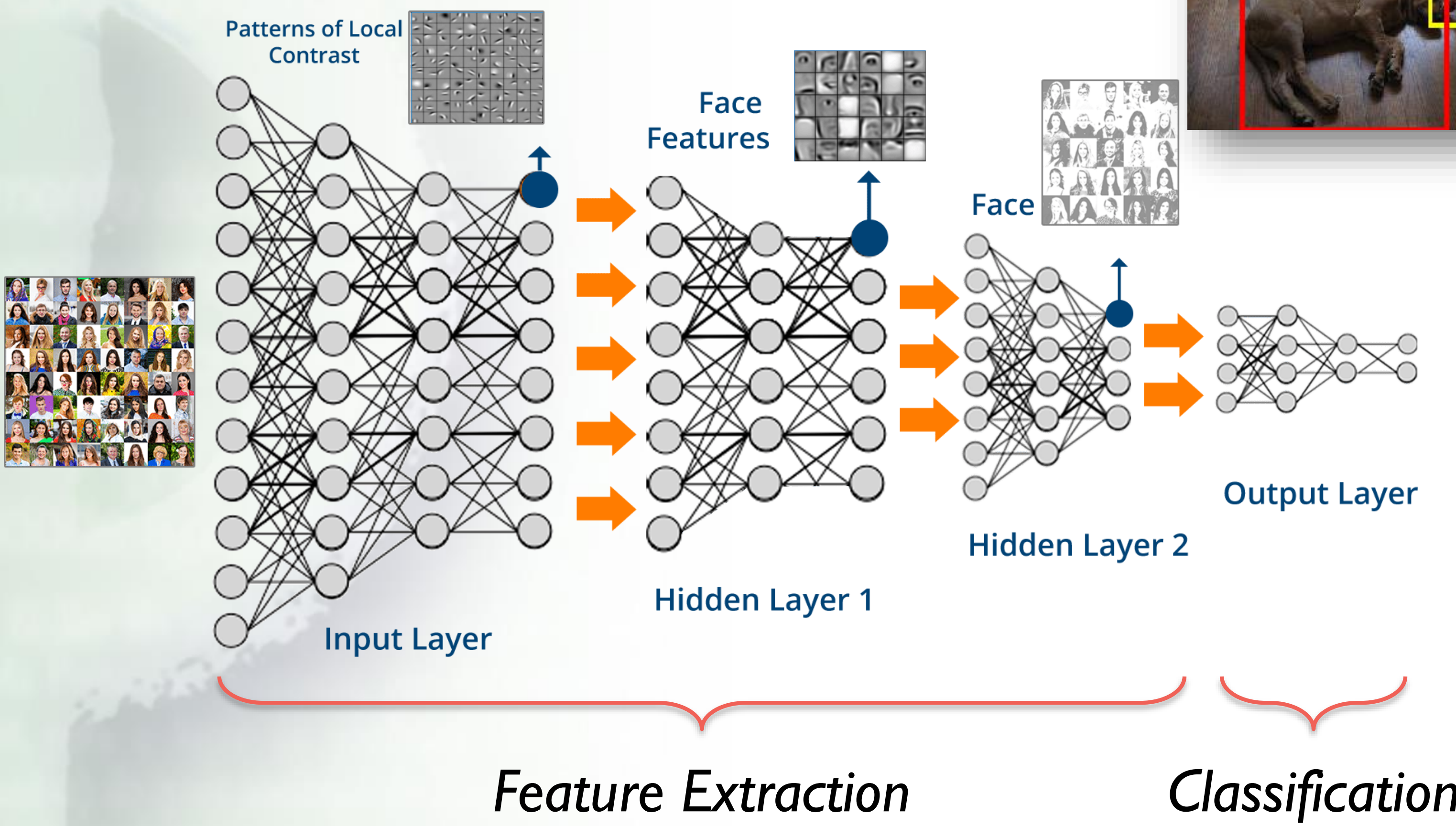
$$\frac{\partial \mathcal{J}(\boldsymbol{\theta})}{\partial \theta_1} = \frac{\partial \mathcal{J}(\boldsymbol{\theta})}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial \theta_1} = \frac{\partial \mathcal{J}(\boldsymbol{\theta})}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial z_2} \times \frac{\partial z_2}{\partial \theta_1}$$

Deep NN for Visual Recognition

Convolutional Neural Networks (CNN/ConvNet)

Visual Recognition

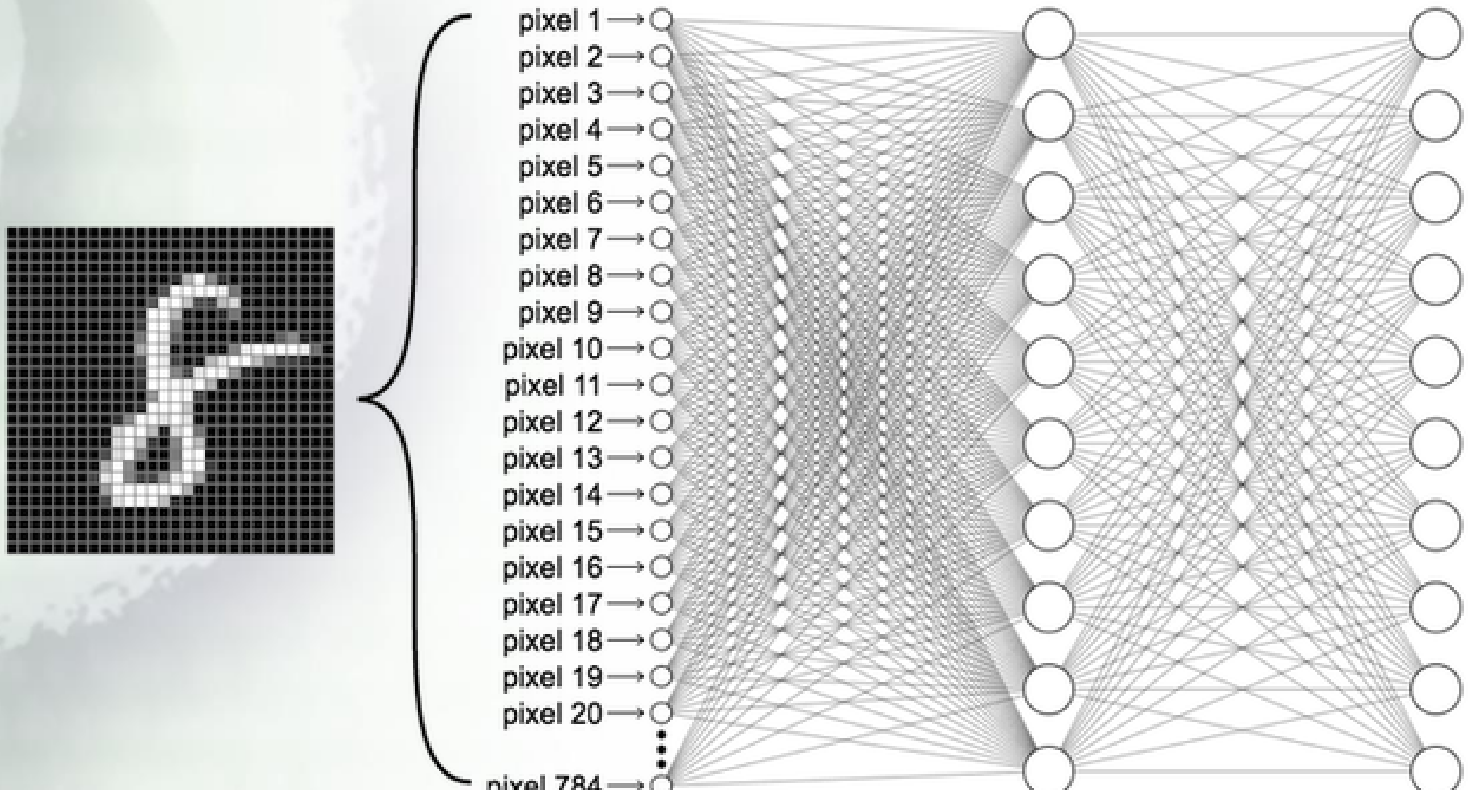




CNN or ConvNet

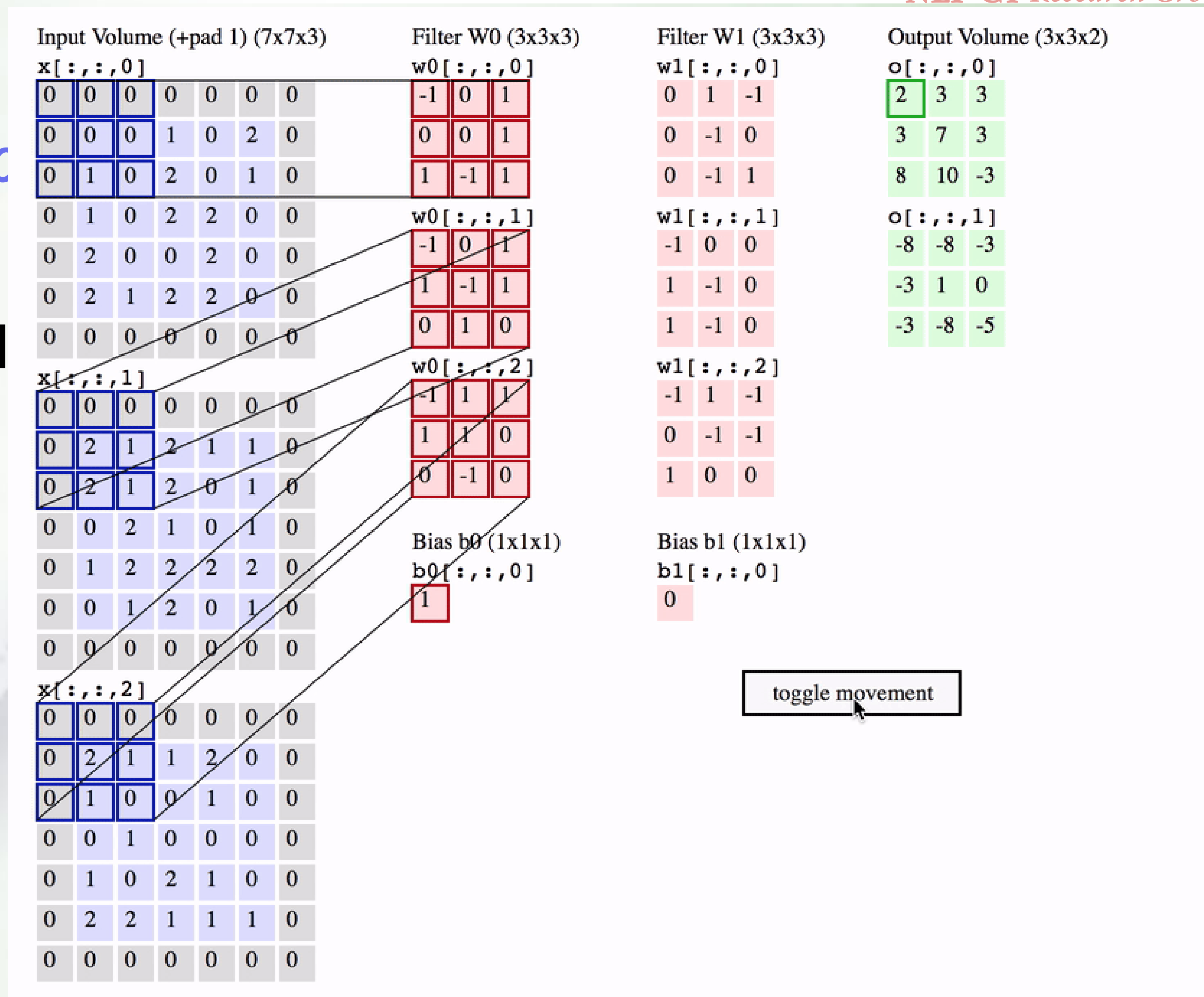
Convolutional Neural Network

- CNN is the *extension* of traditional *Multi-layer Perceptron*, based on 3 ideas:
  - Sparse connectivity



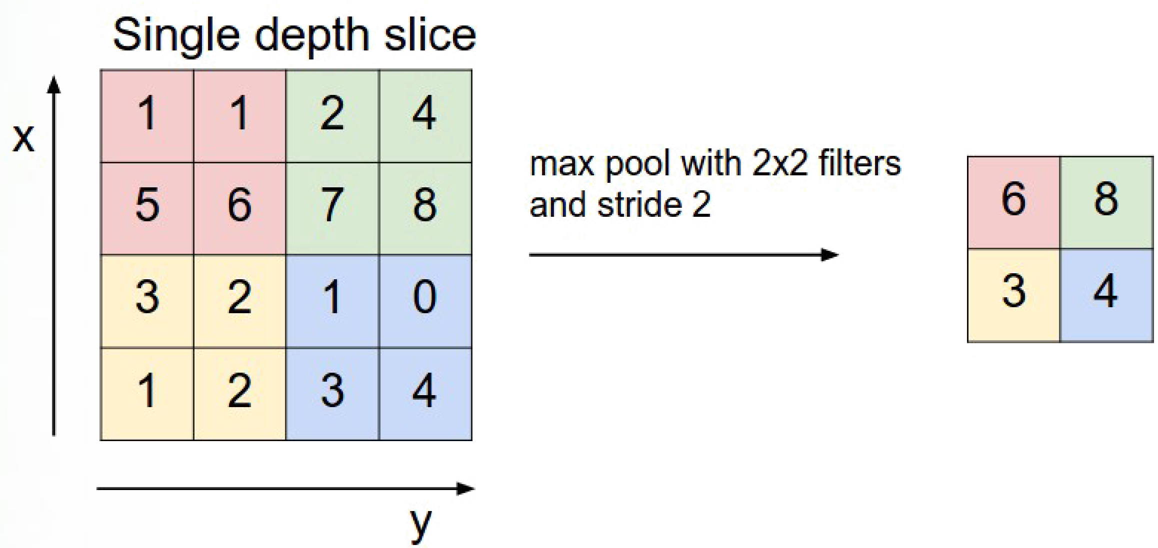
Convolutional Neural Network
CNN *or* ConvNet

- CNN is
  Perceptr
  - Sparse
  - Shared

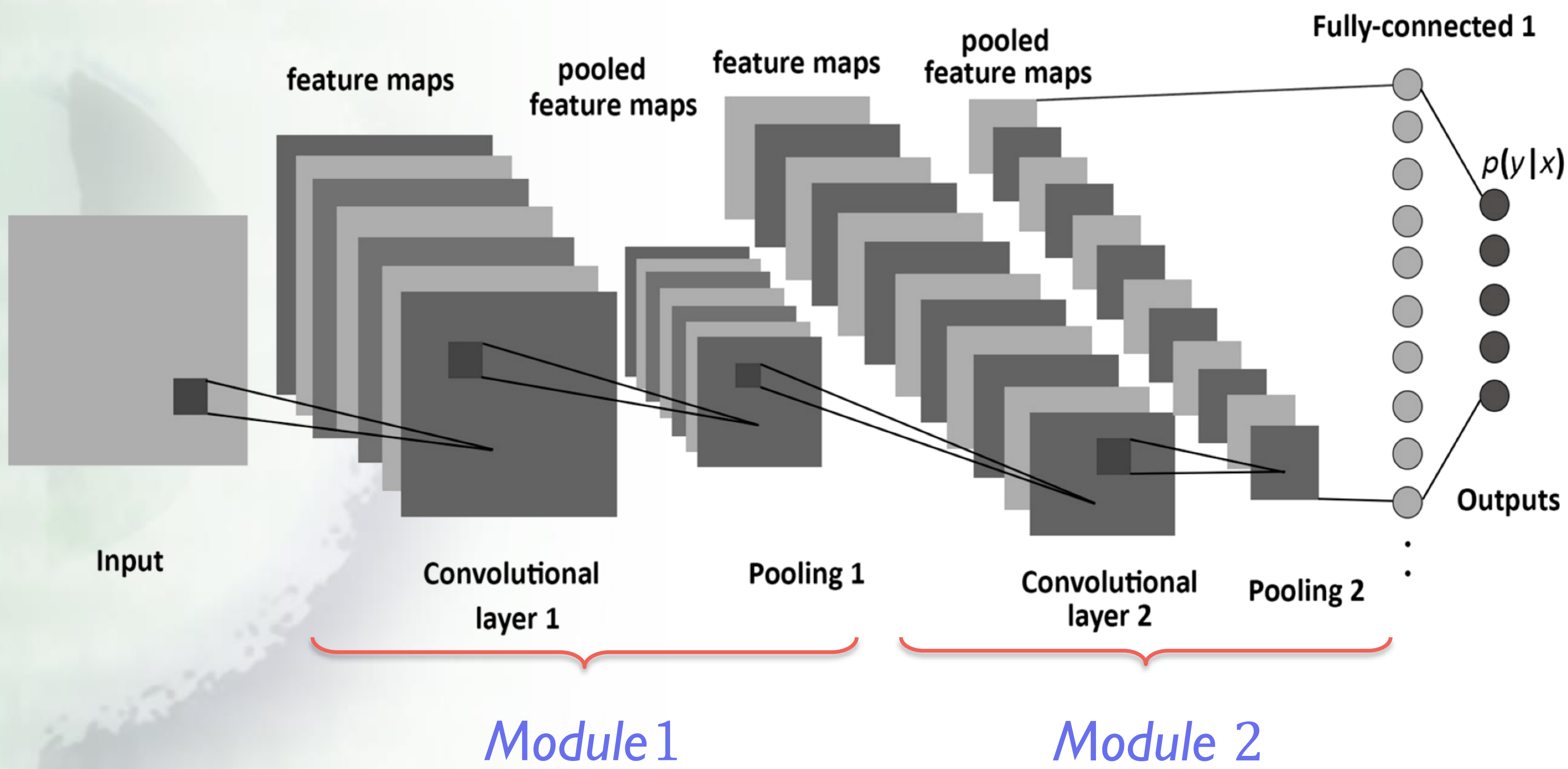


Convolutional Neural Network
CNN *or* ConvNet

- CNN is the *extension* of traditional *Multi-layer Perceptron*, based on 3 ideas:
  - Sparse connectivity
  - Shared weights
  - Spatial/temporal sub-sampling

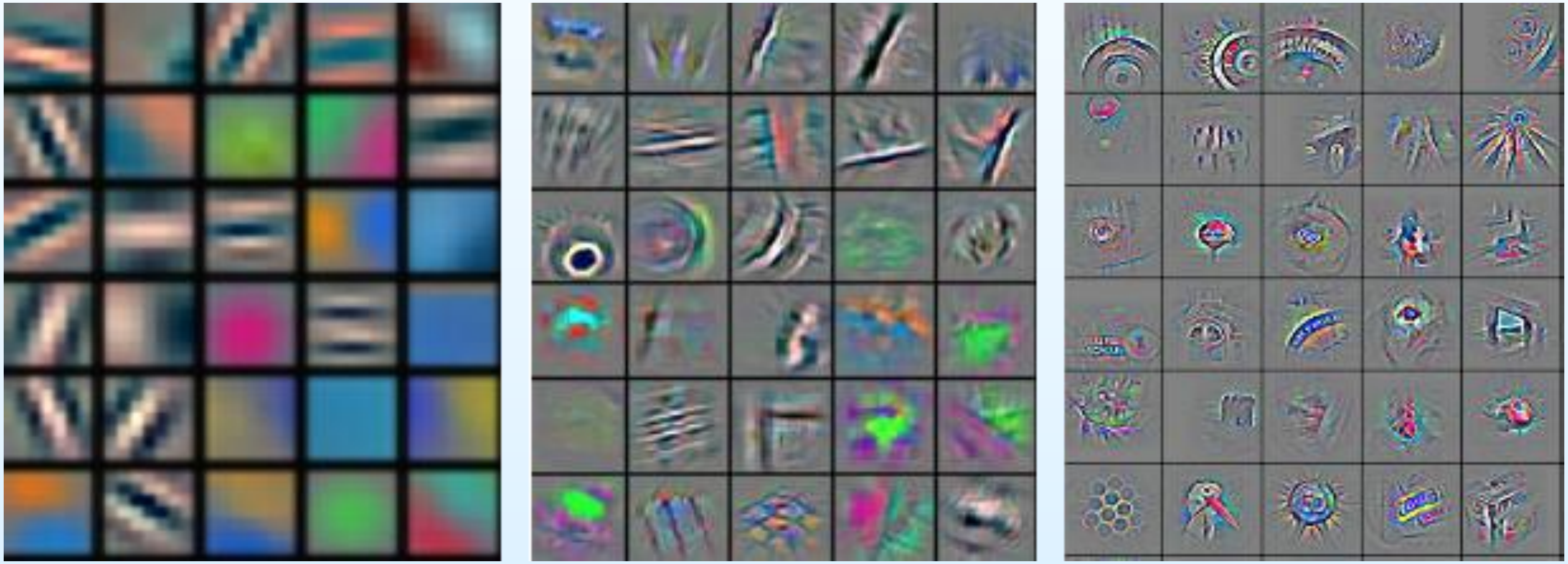


Convolutional Neural Network
The Architecture



Convolutional Neural Network

The Architecture



Input

Convolutional layer 1

Pooling 1

Convolutional layer 2

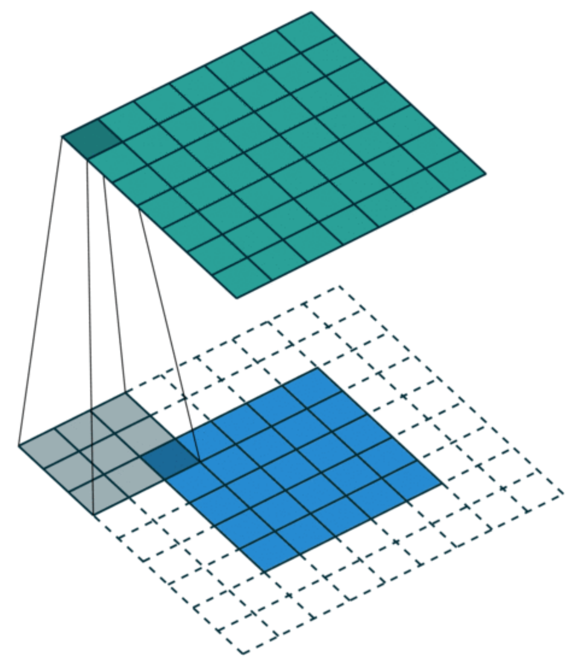
Pooling 2

Module 1

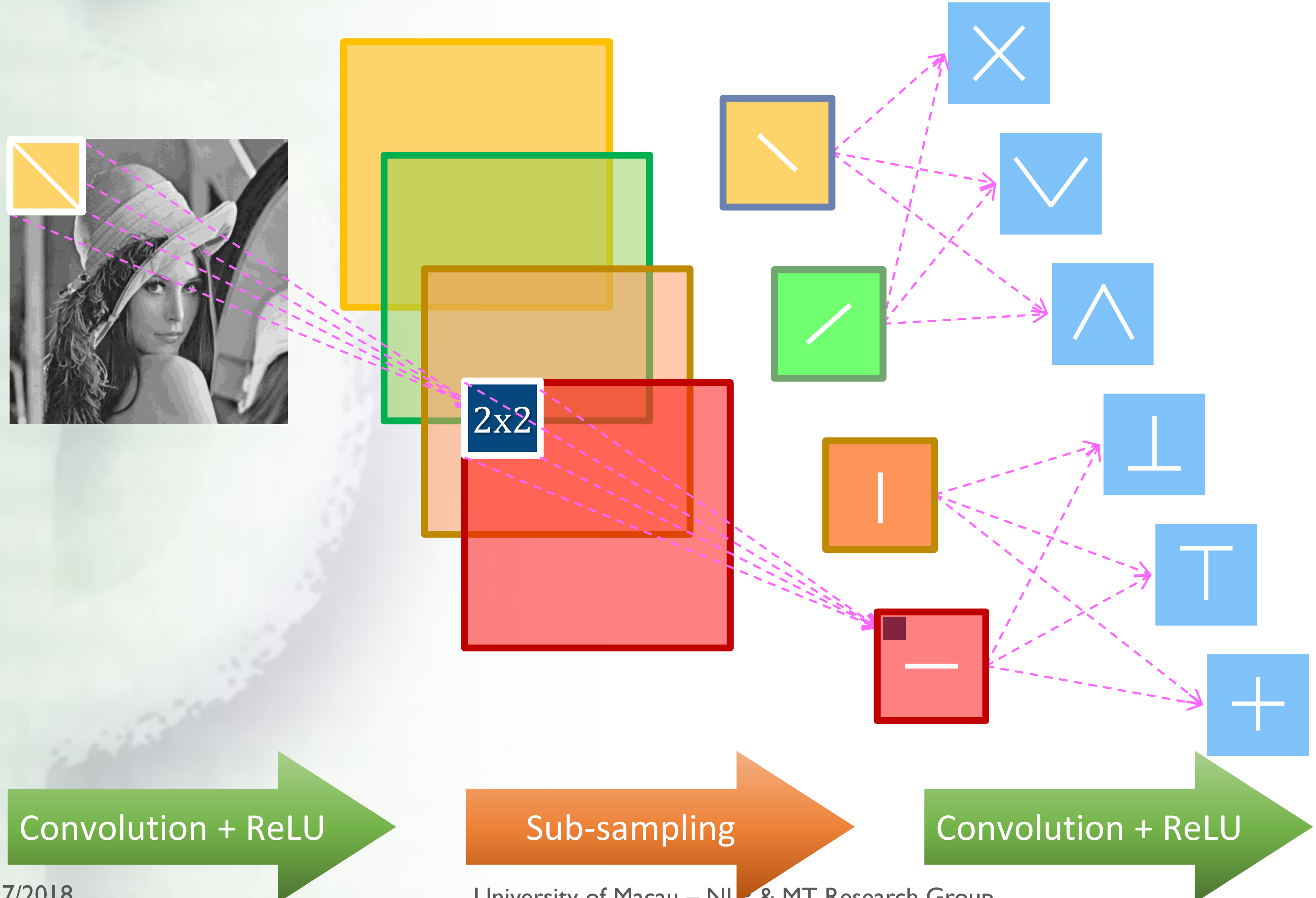
Module 2

The Filters

$$\begin{bmatrix} -1 & 1 & 0 \\ -1 & 1 & 0 \\ -1 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & -1 \\ 0 & 1 & -1 \\ 0 & 1 & -1 \end{bmatrix}$$



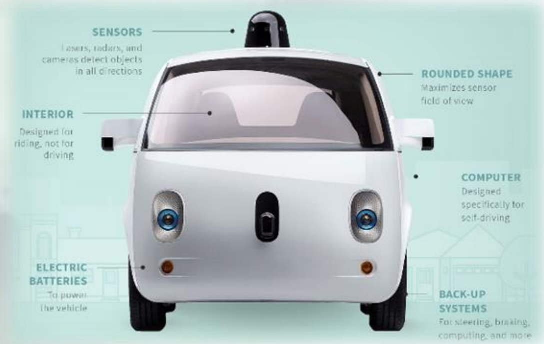
ConvNet
The Whole Process

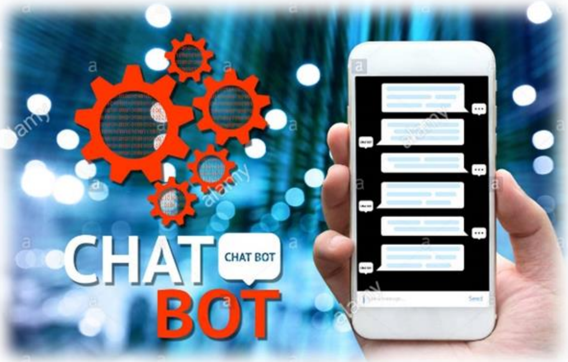


Take Away From
This Lecture

Main Points

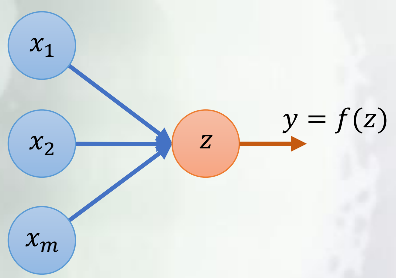
- What is Deep Learning? Its successfully story
  - *Very Deep Neural Network*
  - *Vision, Robotics, Game AI, Language & Speech, Music & Arts*

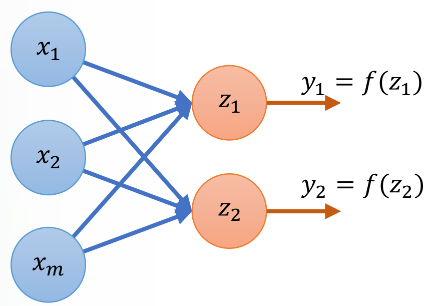


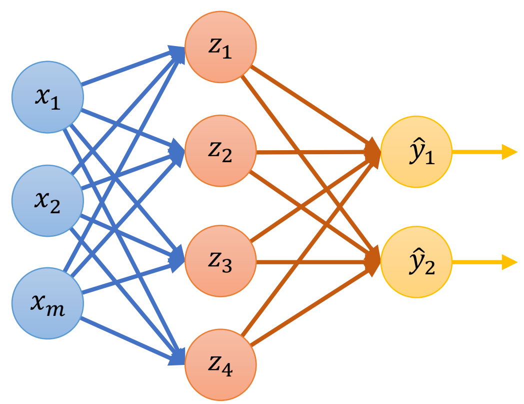


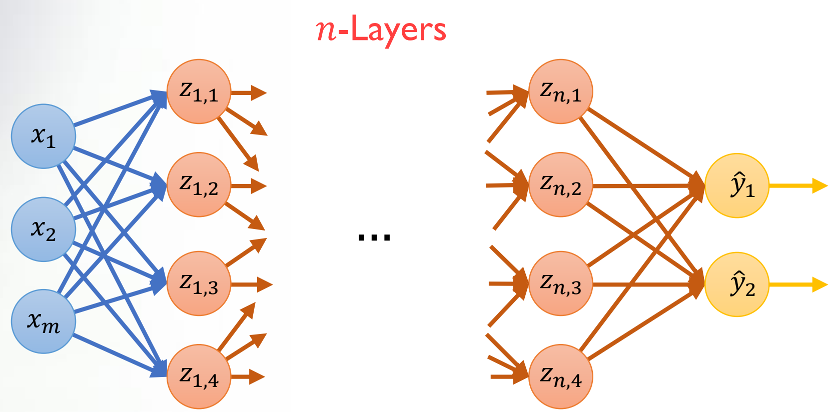
Main Points

- What is Artificial Neural Network?
  - *From Neuron → Perceptron → Single Layer Network → Multiple Layer Network → Deep Network*

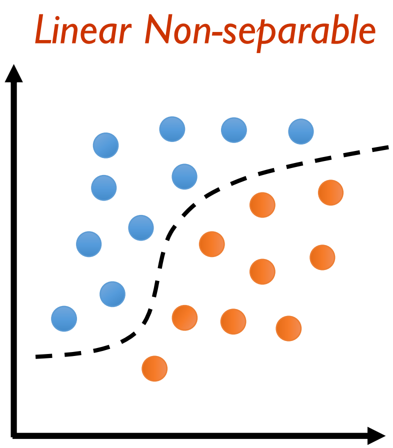








Main Points



- Activation functions
  - *Signum, Sigmoid, Tangent, ReLU*
- Loss functions
  - *Empirical Loss, Cross Entropy Loss, Mean Squared Loss*
- Optimization
  - *Backpropagation, Chain Rule*
- Training Algorithm
- Deep Model
  - *Convolutional Neural Network*

Hands on Experiment

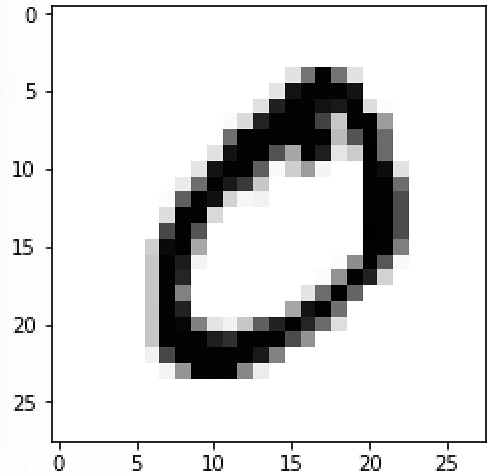
Digits Classification with ConvNet

Digits Classification

MNIST Handwritten Digits

Dataset: http://yann.lecun.com/exdb/mnist/

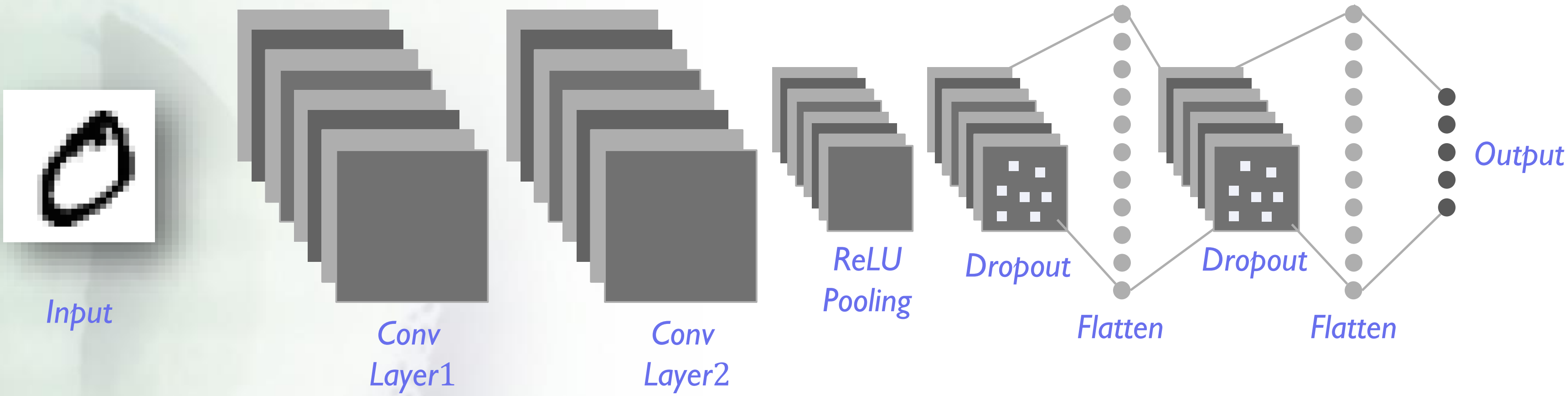
Digits Classification
The Image



28×28

Monochrome (1-channel)

Digits Classification
The ConvNet Model



Digits Classification
The Notebook

Download the Notebook

https://drive.google.com/open?id=1S_dePZu9L4tQMkOsI2W0VKmOx8GmUN9u

Digits Classification References

- Learn from the champion is the best way to success
  - http://blog.kaggle.com/category/winners-interviews/
- There are many winner interviews we can learn for competition methods and tricks
  - https://ndres.me/kaggle-past-solutions/
  - http://shujianliu.com/kaggle-winning-code.html

Acknowledgements

Some of the presentation contents are derived from the following materials:

- *An Introduction to Neural Network*, Ben Kröse & Patrick Smagt
- *Introduction to Neural Networks and Deep Learning*, Efstratios Gavves
- *Introduction to Deep Learning*, Alexander Amini
- *Convolutional Neural Networks (CNNs)*, Deeplizard