Topic I. Similarity Measure (相似度量測)

2.0 Computer Vision (CV) and AI: Similarity Measure= Loss Function
2.1~2.3

National Cheng Kung University
Dept. of Computer Science and Information Engineering
Robotics Lab.
國立成功大學資訊工程系機器人實驗室

(Jenn-Jier James Lien / 連震杰)
http://robotics.csie.ncku.edu.tw
jjlien@csie.ncku.edu.tw

指導老師：連震杰 教授
學　生：簡佑軒

2022/10/17

Content: Topic1. Similarity Measure（相似度量測）

1.0 Computer Vision - Camera Model: Between 3D and 2D

- 1.1 2D Image
- 1.2 Camera Model Btw 3D Object and 2D Image
- 1.3 From Linear Combination to Homography Ax=b (Vector, Matrix)

2.0 Computer Vision (CV) and AI: Similarity Measure = Loss Function

- 2.1 Sum-of-Squared Differences (SSD) and Mean Squared Error (MSE)
- 2.2 Normalized Correlation Coefficient (NCC) and Cross Entropy (CE)
- 2.3 AI Bayesian Decision Rule - Maximum a Posteriori (MAP) Probability with Gaussian Model

3.0 Image Processing for Deep Learning: Filter, Correlation and Convolution

- 3.1 Low-Pass Filter - Denoise and High-Pass Filter - Edge Detection
- 3.2 Correlation Vs. Convolution
- 3.3 Multi-Resolution - Downsampling (Encoder) and Upsampling (Decoder)

2.1 Sum-of-Squared Differences (SSD) and Mean Squared Error (MSE)

2.1-1 Template Matching: Goal and Theory

Goal

In this tutorial you will learn how to:

- Use the OpenCV function matchTemplate to search for matches between an image patch and an input image
- Use the OpenCV function minMaxLoc to find the maximum and minimum values (as well as their positions) in a given array.

Location/position

Theory

What is template matching?

Template matching is a technique for finding areas of an image that match (are similar) to a template image (patch).

➢ Template Matching = Similarity Measure

2.1-1 Template Matching: Definition (1/3)

◆ How does it work? Match procedure:

1. Template Image and Source Image Collection: We need two primary components-
   1) **Source image (I)**: The image in which we expect to find a match to the template image
   2) **Template image (T)**: The patch image which will be compared to the template image

source image

template image

the matching region

2.1-1 Template Matching: Definition (2/3)

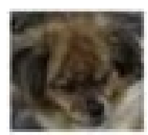
Template matching as convolution

◆ How does it work?

2. Template Search Sliding along Scanline:

- To identify the matching area, we have to compare the template image against the source image by sliding it:

Source Image



Template (patch)

Template 中心點對 image 左上

Compare the template image against the source image by sliding it.

Compare the template image against the source image by sliding it.

- By sliding (scanline): we mean moving the patch one pixel (called stride 1 pixel) at a time (left to right, up to down). At each location, a **metrics** is calculated so it represents how "good" or "bad" the match at that location is (or how similar the patch is to that particular area of the source image).

2.1-1 Template Matching: Definition (3/3)

3. Matching Metrics Matrix Image *R* Creation:

- For each location of *T* over *I*, you store the metrics in the result matrix *(R)*. Each location (x,y) in R contains the match metric:

R(x,y): The brightest location = the best matching location:
- Top-left corner of T, or
- Center of T

越白越matching

- The image is the result ***R*** of sliding the patch with a metric.
- The brightest locations indicate the best matching location.

4. Matching Location Decision:

- As you can see, the location marked by the red circle is probably the one with the best matching value, so that location (the rectangle formed by that point as a corner and width and height equal to the patch image) is considered the match.

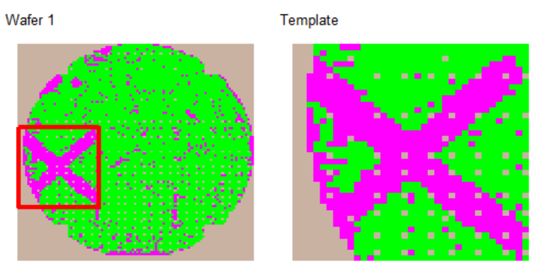
2.1-1 Template Matching: Definition ??

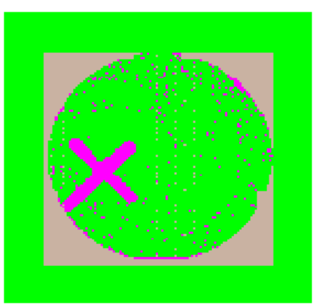
Image尺寸為W x H

Template尺寸為w x h

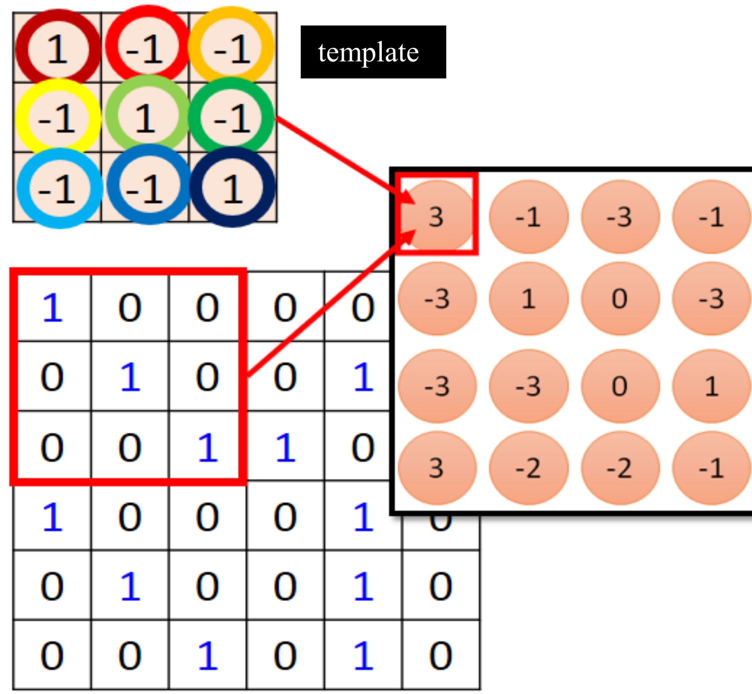
Output Image尺寸將為W-w + 1，H-h + 1

If 想獲得計算後輸出的相似度矩陣與掃描之原圖具有相同尺寸，須在欲掃描的原圖**四周圍以0 Padding**





四周填滿0的 image



6 x 6 image

2.1-2 Match Metric: SSD and MSE – SSD

數值越大相似度越小
如果值越接近0匹配程度越高

1. SSD: Sum-of-Squared Differences

- OpenCV implements template matching in the function matchTemplate:

1.1 CV_TM_SQDIFF: Sum-of-Squared-Differences (SSD)

$$\min R(x,y) = \sum_{x',y' \in w} \left(T(x',y') - I(x+x',y+y')\right)^2$$

x',y' belongs to window w

Sliding window w

$w(x',y')$: $x' \in 0 \sim (w'-1)$, $y' \in 0 \sim (h'-1)$

Window $w(x',y')$
Slide/shift/stide (x,y) over entire source image

$$\min R(x,y) = \sum_{x'=0}^{h'-1} \sum_{y'=0}^{w'-1} \left[\left(T(x',y')\right) - I(x+x',y+y')\right]^2$$

1.2 CV_TM_SQDIFF_NORMED: Normalized Sum-of-Squared Differences

$$\min R(x,y) = \frac{\sum_{x',y'} \left(T(x',y') - I(x+x',y+y')\right)^2}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x',y+y')^2}}$$

最佳匹配在結果為0處

目的是為了減少光照的影響

m: Mean or average value
光線

1.3 SQDIFF_NORMED – Normalized by Gaussian Model (Variance)

把絕對的關係變成相對的關係

$$\min R(x,y) = \frac{\sum_{y'=0}^{h'-1} \sum_{x'=0}^{w'-1} \left[\left(T(x',y') - m^T\right) - I(x+x',y+y') - m^I\right)\right]^2}{\sqrt{\sum_{y'=0}^{h'-1} \sum_{x'=0}^{w'-1} \left[\left(T(x',y') - m^T\right]^2\right.} \sqrt{\sum_{y'=0}^{h'-1} \sum_{x'=0}^{w'-1} \left[\left(I(x+x',y+y') - m^I\right]^2\right.}}$$

σ^2: Standard deviation

2.1-2 Match Metric: SSD and MSE – MSE

2. MSE: Mean Squared Error

Loss Function:

$$L_b = 1/N * \sum_{n=1}^{N} \{ \sum_{x'=0}^{h'-1} \sum_{y'=0}^{w'-1} [T(x', y') - I(x', y')]^2 \}$$

$$= 1/N * \sum_{n=1}^{N} \{ \sum_{u=0}^{h-1} \sum_{v=0}^{w-1} [y(u, v) - y'(u, v)]^2 \}$$

b: b-th batch
N: Total number of images or Batch size

y: Ground truth values

y': Prediction or estimation values

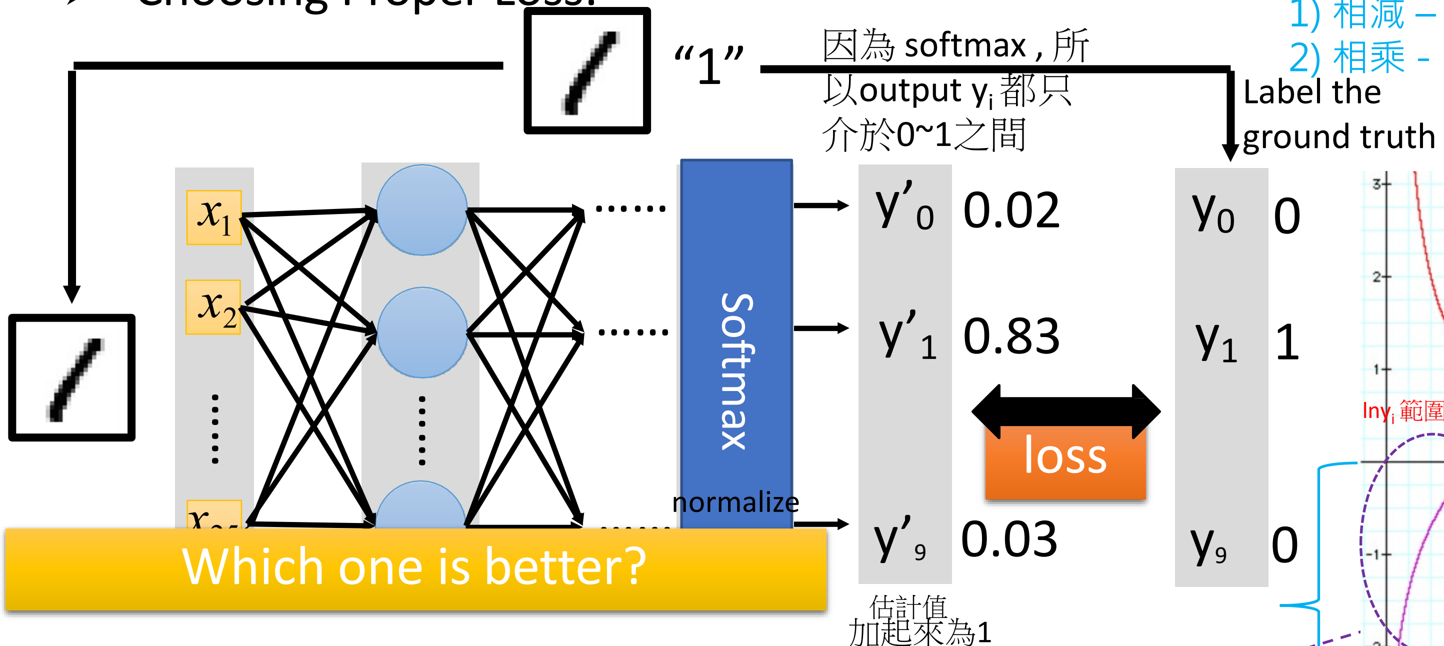
In DL data分成多batch，除以batch size

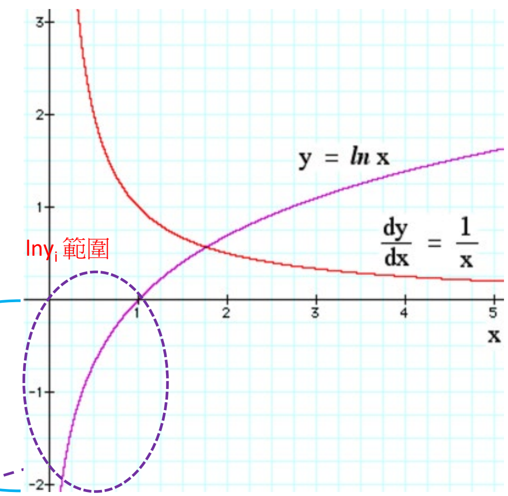
2.1-2 Match Metric: MSE Vs. CE – Loss Function

3. Loss Function at Deep Learning – 1) MSE and 2) Cross Entropy

- Choosing Proper Loss:

相似度判斷(similarity measure)：
1) 相減 – MSE, SSD, SAD
2) 相乘 - NCC





1) Mean Square Error: for continuous data, like detection Bounding Box (Bbox)

$$L_b = 1/N * \sum_{n=1}^{N} \{\sum_{i=0}^{9}(y_i - y'_i)^2\}$$

Why not L1 | |➔ why L2 Or Smooth L1

Not continuous, cannot derivative

Continuous, can optimize by derivative

2) Cross Entropy: for discrete data, like classification

$L_b = -\sum_{i=0}^{9}\{y_i \ln y'_i\}$, or $L_b=L_{b1}+L_{b2}$

$$\begin{cases} L_{b1} = -\sum_{i=0}^{9}\{y_i \ln y'_i\}, & \text{if } y_i = 1 \\ L_{b2} = -\sum_{i=0}^{9}\{(1-y_i)\ln y'_i\}, & \text{Otherwise } (y_i = 0) \end{cases}$$

- Before, have Softmax for normalizing all output values, so don't need 1/N
- Because y is prob. 0.0~1.0, so log y will be -, therefore - - = +

Content: Topic1. Similarity Measure（相似度量測）

1.0 Computer Vision - Camera Model: Between 3D and 2D

- 1.1 2D Image
- 1.2 Camera Model Btw 3D Object and 2D Image
- 1.3 From Linear Combination to Homography Ax=b (Vector, Matrix)

2.0 Computer Vision (CV) and AI: Similarity Measure = Loss Function

- 2.1 Sum-of-Squared Differences (SSD) and Mean Squared Error (MSE)
- 2.2 Normalized Correlation Coefficient (NCC) and Cross Entropy (CE)
- 2.3 AI Bayesian Decision Rule - Maximum a Posteriori (MAP) Probability with Gaussian Model

3.0 Image Processing for Deep Learning: Filter, Correlation and Convolution

- 3.1 Low-Pass Filter - Denoise and High-Pass Filter - Edge Detection
- 3.2 Correlation Vs. Convolution
- 3.3 Multi-Resolution - Downsampling (Encoder) and Upsampling (Decoder)

2.2 Normalized Correlation Coefficient (NCC) and Cross Entropy (CE)

2.2-1 From SSD to CC

- **From sum-of-squared difference (SSD) to cross correlation (CC)**

$$\min SSD = \min\left(T - I\right)^2 = \min\left(T^2 - 2TI + I^2\right)$$

$$\min SSD \rightarrow \left(T^2 + I^2\right) - 2\left(TI\right)$$

$$\therefore \max CC \rightarrow TI$$

$T * I$

➔Cross Correlation

2.2-1 Template Matching + Scanline: NCC (1/2)

DL uses Similarity measure: MSD (-), Cross Entropy (x) for loss function

1. Cross Correlation, R(x, y): The correlation (Linear combination) between two signals (cross correlation) is standard approach to feature detection as well as a component of more sophisticated techniques.

Similarity measure: MSD (minus: -), Cross Entropy (product: x)

越相似內積越高

$$R(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1}(T(u,v)\cdot I(x+u,y+v))$$

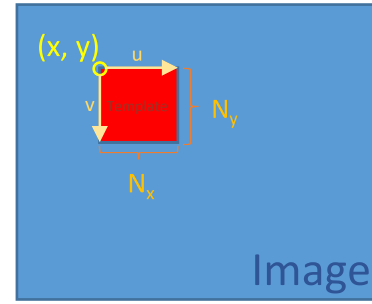
← 其中內積是取其投影量來求相似度

Inside the product is to take its projection amount to find similarity

← 計算的起始位置在左上角

The starting position of the calculation is at the top left corner

OpenCV



-Source image (I, MxM): The image in which we expect to find a match to the template image
Template image (T, NxN): The patch image which will be compared to the template image
Slide/shift (x, y); Window posited (u, v)
- Problem: R(x,y) range is not btw 0.0~1.0

EX: 1x9, 2x8, … 5x5(Max)
But where is the threshold? So we got the Normalization of cross correlation

2. Normalized Cross Correlation: But the normalized form of correlation (normalized correlation coefficient or unit vector) **preferred in template matching does not have a correspondingly simple and efficient frequency domain expression**. For this reason normalized cross-correlation has been computed in the spatial domain. 空間域

$$R(x,y) = \frac{\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}(T(u,v)\cdot I(x+u,y+v))}{\sqrt{[\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}(T(x,y))^2][\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}(I(x+u,y+v))^2]}}$$

← 沒有減掉mean不是NCC, 只是整理成同樣的scale

Not the NCC, but it's just the same scale.

←結果也是會被環境光影響

Sam

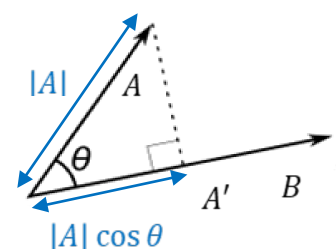
← Normalized Unity

-Source image (I, MxM): The image in which we expect to find a match to the template image
Template image (T, NxN): The patch image which will be compared to the template image
Slide/shift (x, y); Window posited (u, v)
- R(x,y) range is normalized btw -1.0~1.0

$$\cos\theta = \frac{A^T B}{\|A\|\|B\|},\ 0.0 \le \cos\theta \le 1.0$$

$$\vec{A}\cdot\vec{B} = |A|\cdot|B|\cdot\cos\theta$$

$$\cos\theta = \frac{\vec{A}\cdot\vec{B}}{|A||B|}$$



$$\text{similarity} = \cos(\theta) = \frac{A\cdot B}{|A||B|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2}\times\sqrt{\sum_{i=1}^{n}(B_i)^2}}$$

產生類似餘弦的相關係數

2.2-1 Template Matching + Scanline: NCC (2/2)

3. Matric: similarity measure - NCC (Normalized Correlation Coefficient)
   - At each location, a metric is calculated so it represents how "good" or "bad" the match at that location is (or how similar)
   - For each location of t over f, we store the metric in the result matrix R. Each location (x, y) in R contains the match metric:

     - R(x,y) range is normalized btw -1.0~1.0
     - NCC is a Gaussian Model T(,) and I(,)

$$max\, R(x,y) = \frac{\sum_{y'=0}^{h'-1}\sum_{x'=0}^{w'-1}[(T(x',y') - m^T) * I(x+x',y+y') - m^I)]^2}{\sqrt{\sum_{y'=0}^{h'-1}\sum_{x'=0}^{w'-1}[(T(x',y') - m^T]^2}\sqrt{\sum_{y'=0}^{h'-1}\sum_{x'=0}^{w'-1}[(I(x+x',y+y') - m^I]^2}}$$

←起始點在左上角，由左至右，由上至下
The starting point is in the upper left corner, from left to right, from top to bottom

The image is the result γ of sliding the patch with a metric TM_CCORR_NORMED.
The brightest locations indicate the highest matches

4. Matching Result:
   - One target per source image: At the image above, the location marked by the red circle is probably the one with the highest value, so that the rectangle formed by that point as a corner is considered the match.
   - N targets per source image: Need to consider the mergence problem, such as subpixel problem can using weighted mergence
     ➔Non-Maximum Suppression for mergence

← 若出現許多matching score超過threshold的目標，必須選擇一個最大的目標，並整合周圍的目標。
If there are many goals with matching scores that exceed threshold, you must choose the largest goal and integrate the surrounding goals

2.2-2 Match Metrics: NCC and CE - NCC

1,2,..,4,5,6,...8,9
9,8,..,6,5,4,...2,1

1. NCC - Normalized Correlation Coefficient:

Deep Learning:	min	max
1) MSE	SSD	correlation
2) Cross Entropy	9-1 =8	9*1 =9
	5-5 =0	5*5 =25

1.1 CV_TM_CCORR: Cross Correlation

較大的數表示匹配程度較高

$$\max R(x,y) = \sum_{x',y'} \left(T(x',y') \cdot I(x+x', y+y') \right)$$

x',y' belong to window w

Sliding window w
$w(x',y')$: $x' \in 0 \sim (w'-1)$, $y' \in 0 \sim (h'-1)$

1.2 CV_TM_CCORR_NORMED: Normalized Cross Correlation

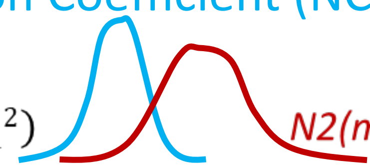
$$\max R(x,y) = \frac{\sum_{x',y'} \left(T(x',y') \cdot I(x+x', y+y') \right)}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x', y+y')^2}}$$

OpenCV

- -Is NCC a Gaussian Model? Yes,
- -Is Gaussian model a good model?
- \- Slow
- ➔ Modify to SSD or MSD
- \- Faster, without normalization
- ➔ Batch Normalization

1.3 CV_TM_CCOEFF_NORMED: Normalized Correlation Coefficient (NCC) by Gaussian Model

– NCC Range [-1.0, 1.0]



減少光照的影響

$$\max R(x,y) = \frac{\sum_{y'=0}^{h'-1}\sum_{x'=0}^{w'-1}[(T(x',y') - m^T) * I(x+x',y+y') - m^I)]^2}{\sqrt{\sum_{y'=0}^{h'-1}\sum_{x'=0}^{w'-1}[(T(x',y') - m^T]^2}\sqrt{\sum_{y'=0}^{h'-1}\sum_{x'=0}^{w'-1}[(I(x+x',y+y') - m^I]^2}}$$

m: Mean or average value

光線

σ^2: Standard deviation

– NCC Range [0.0, 1.0] if |R(x,y)|, for similarity Measure

Similar measure btw two Gaussian distributions

2.2-2 Match Metrics: NCC and CE - NCC

Opencv 官網所提供NCC作法

TM_CCORR_NORMED Python: cv.TM_CCORR_NORMED	$R(x,y)=\frac{\sum_{x',y'}(T(x',y')\cdot I(x+x',y+y'))}{\sqrt{\sum_{x',y'}T(x',y')^2\cdot\sum_{x',y'}I(x+x',y+y')^2}}$

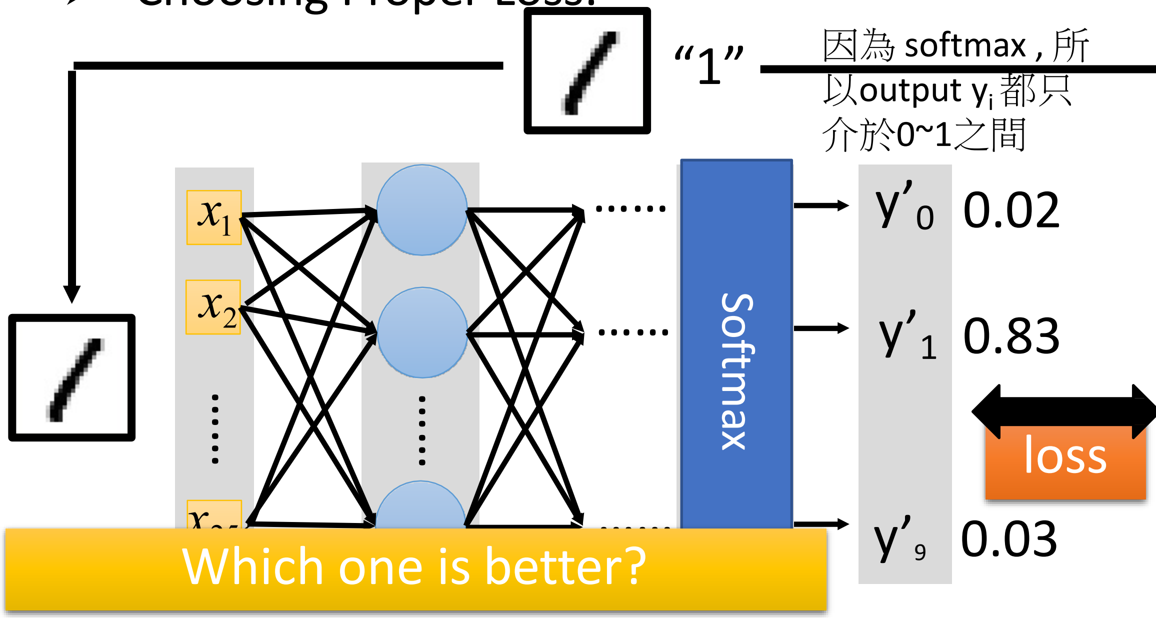
2.2-2 Match Metric: MSE Vs. CE – Loss Function

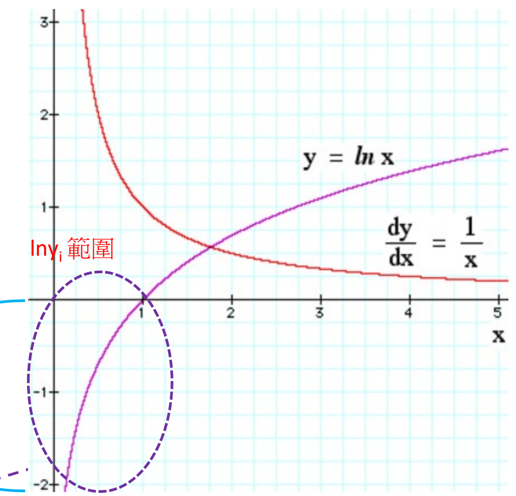
2. CE – Cross Entropy

3. Loss Function at Deep Learning – 1) MSE and 2) Cross Entropy

相似度判斷(similarity measure)：
1) 相減 – MSE, SSD, SAD
2) 相乘 - NCC

- Choosing Proper Loss:





1) Mean Square Error: for continuous data, like detection Bounding Box (Bbox)

$$L_b = 1/N * \sum_{n=1}^{N} \{\sum_{i=0}^{9} (y_i - y'_i)^2\}$$

Why not L1 | |➔ why L2 Or Smooth L1

Not continuous, cannot derivative

Continuous, can optimize by derivative

2) Cross Entropy: for discrete data, like classification

$$L_b = -\sum_{i=0}^{9} \{y_i \ln y'_i\}, \text{ or } L_b = L_{b1} + L_{b2}$$

$$\begin{cases} L_{b1} = -\sum_{i=0}^{9} \{y_i \ln y'_i\}, & \text{if } y_i = 1 \\ L_{b2} = -\sum_{i=0}^{9} \{(1 - y_i) \ln y'_i\}, & \text{Otherwise } (y_i = 0) \end{cases}$$

- Before, have Softmax for normalizing all output values, so don't need 1/N
- Because y is prob. 0.0~1.0, so log y will be -, therefore - - = +

2.2-3 Correlation Vs. Covariance (1/5) 相關係數與共變異數 Tommy

We have to consider mean value, because transfer from absolute difference to relative difference.

主要衡量兩變數間「線性」關聯性的高低程度。

1) Correlation:

$$\boxed{cor(X,Y) = \sum_{i=0}^{N} X_i \boxed{\cdot} Y_i}$$

inner product = projection = cosθ

N terms

Dot product
Measure similarity

2) Covariance: correlation of two variances

$$\boxed{cov(X,Y) = \frac{1}{N}\sum_{i=0}^{N}(X_i - \mu_X)\cdot(Y_i - \mu_Y)}$$

- When mean value = 0,
  Covariance matrix = Correlation matrix

N terms
μ_X: mean of x
μ_y: mean of y

Covariance(共變異數)
其實就等於在算x和y
的相關程度

$$\boxed{cov(X,Y) = E[(X-\mu_X)(Y-\mu_Y)]}$$

E: Expected value = average

Tommy

while $\mu_x = E(X)$ and $\mu_Y = E(Y)$

(= Correlation Variance)

- **Covariance**: is a measure of relation between two variables.

1-D

$$\boxed{(variance)cov(X,X) = \frac{1}{n}\sum_{i=0}^{n}(X_i - \bar{X})}$$

2-D => Covariance Matrix

$$\boxed{cov(X,Y) = \frac{1}{n}\sum_{i=0}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}$$

> Covariance is influenced by the overall magnitudes of X and Y.

> Linear combination ➔ Matrix

2.2-3 Correlation Vs. Covariance (1/5) Tommy

We have to consider mean value, because transfer from absolute difference to relative difference.

3) The normalized correlation coefficient (NCC) *r* between X and Y: the covariance of the variables (X, Y) divided by each of their standard deviations.

$$\text{NCC } r(X,Y) = \frac{\frac{1}{n}\sum_{i=0}^{n}(X_i-\bar{X})(Y_i-\bar{Y})}{\sigma_X\sigma_Y}$$

→ **Translation, shift or bias term**

→ **Normalization term as scaling term**

$$= \frac{\frac{1}{n}\sum_{i=0}^{n}(X_i-\bar{X})(Y_i-\bar{Y})}{\sqrt{\frac{\sum(X_i-\bar{X})^2}{n}}\sqrt{\frac{\sum(Y_i-\bar{Y})^2}{n}}} = \frac{\sum_{i=0}^{n}(X_i-\bar{X})(Y_i-\bar{Y})}{\sqrt{\sum(X_i-\bar{X})^2}\sqrt{\sum(Y_i-\bar{Y})^2}}$$

❖ Standard deviation of X, Y

$$\sigma_X = \sqrt{\frac{\sum(X_i-\bar{X})^2}{n}}$$

$$\sigma_Y = \sqrt{\frac{\sum(Y_i-\bar{Y})^2}{n}}$$

-Why this is Gaussian model? Because *N(u,σ^2))*

-*Why this can present Affine? Affine: Translation + Rotation + Scaling (+ Shearing)*

Covariance共變異數除上兩個變數間的標準差，值會落在正負1之間

$$\boxed{r = \frac{cov(X,Y)}{\sigma_X\sigma_Y}}$$

- Translation, shift or bias term
- Normalization term as scaling term

σ_X: standard deviation of x

σ_Y: standard deviation of y

2.2-3 Correlation Vs. Covariance – Speed Up (3/5)

Tommy

Use Integral Image to solve it

$$\frac{1}{N_xN_y}\Sigma_{x,y}[f(x,y)-\bar{f}_{u,v}]^2 = \frac{1}{N_xN_y}\left(\Sigma_{x,y} f^2(x,y) - \frac{\Sigma_{x,y} f(x,y)^2}{N_xN_y}\right)$$

$$cov[x] = E[x^2] - (E[x])^2 = var[x]$$

單個變異數的cov=var

Cov(X,Y)共變異數

Var(X)變異數

Aim: $\sum_{x,y}[f(x,y)-\bar{f}_{u,v}]^2 = \sum_{x,y} f^2(x,y) - \frac{\sum_{x,y} f(x,y)^2}{N_xN_y}$

左式: $\sum_x\sum_y (f(x,y)-\bar{f})^2 = \sum_x\sum_y f^2(x,y) - 2\bar{f}_{u,v}\sum_x\sum_y f(x,y) + \sum_x\sum_y \bar{f}^2_{u,v}$ ...... (1)

又 $\boxed{\sum_x\sum_y \bar{f}^2_{u,v} = N_xN_y\left(\frac{1}{N_xN_y}\sum_x\sum_y f(x,y)\right)^2 \text{ 代入(1)}}$

Tommy.

$\Rightarrow \sum_x\sum_y f^2(x,y) - 2\bar{f}_{u,v}\sum_x\sum_y f(x,y) + \left(\sum_x\sum_y f(x,y)\right)^2 \times \frac{1}{N_xN_y}$ ...... (2)

又 $\bar{f}_{u,v} = \frac{1}{N_xN_y}\sum_x\sum_y f(x,y)$ 代入(2)

$\Rightarrow \sum_x\sum_y (f(x,y)-\bar{f})^2 = \sum_x\sum_y f^2(x,y) - \frac{2}{N_xN_y}\left(\sum_x\sum_y f(x,y)\right)^2 + \left(\sum_x\sum_y f(x,y)\right)^2 \times \frac{1}{N_xN_y}$

$= \boxed{\sum_x\sum_y f^2(x,y) - \frac{1}{N_xN_y}\left(\sum_x\sum_y f(x,y)\right)^2 \text{ ※.}}$

➔ $\frac{1}{N_xN_y}\{\Sigma \quad \Sigma f^2(x,y)\}$ -{$\frac{1}{N_xN_y}$(Σ   Σ $f(x,y)$}2

Integral Image

使用Integral Image求「區域變異量」

$$Var(x)=\frac{1}{n}\sum_{i=1}^{n}(x_i-\mu)^2 \qquad \mu=\frac{1}{n}\sum_{i=1}^{n}x_i$$

$$\begin{aligned}
Var(x)&=\frac{1}{n}\sum_{i=1}^{n}\left(x_i^2-2x_i\mu+\mu^2\right)\\
&=\frac{1}{n}\sum_{i=1}^{n}x_i^2-\frac{1}{n}\sum_{i=1}^{n}2x_i\mu+\frac{1}{n}\sum_{i=1}^{n}\mu^2\\
&=\frac{1}{n}\sum_{i=1}^{n}x_i^2-\frac{1}{n}\sum_{i=1}^{n}2x_i\mu+\frac{1}{n}n\mu^2\\
&=\frac{1}{n}\sum_{i=1}^{n}x_i^2-\frac{1}{n}\sum_{i=1}^{n}2x_i\mu+\mu^2\\
&=\frac{1}{n}\sum_{i=1}^{n}x_i^2-\frac{1}{n}\sum_{i=1}^{n}2x_i\mu+\left(\frac{1}{n}\sum_{i=1}^{n}x_i\right)\times\left(\frac{1}{n}\sum_{i=1}^{n}x_i\right)\\
&=\frac{1}{n}\sum_{i=1}^{n}x_i^2-\frac{2\mu}{n}\sum_{i=1}^{n}x_i+\frac{1}{n^2}\left(\sum_{i=1}^{n}x_i\right)^2\\
&=\frac{1}{n}\sum_{i=1}^{n}x_i^2-\frac{2}{n^2}\left(\sum_{i=1}^{n}x_i\right)^2+\frac{1}{n^2}\left(\sum_{i=1}^{n}x_i\right)^2\\
&=\frac{1}{n}\sum_{i=1}^{n}x_i^2-\left(\frac{1}{n}\sum_{i=1}^{n}x_i\right)^2\\
&=\text{像素值平方的平均}-\text{均值平方}
\end{aligned}$$

2.2-3 Correlation Vs. Covariance (4/5)

Tommy

4) Pearson's Correlation Coefficient (探討線性關係): NCC r

$$r = \frac{\sum_{i=1}^{n}(x_i-\mu_x)(y_i-\mu_y)}{\sqrt{\sum_{i=1}^{n}(x_i-\mu_x)^2\sum_{i=1}^{n}(y_i-\mu_y)^2}} = \frac{x\text{和}y\text{的共變異數}}{x\text{的標準差}\times y\text{的標準差}}$$

其中 $-1 \le r \le 1$

μ_x, μ_y: mean of x, y

$$\text{共變異數(covariance)}: \operatorname{cov}(x,y) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i-\mu_x)(y_i-\mu_y)$$

$$\text{變異數(variance)}: \operatorname{var}(x) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i-\mu_x)^2$$

$$\text{標準差(standard deviation)}: std(x) = \sqrt{\operatorname{var}(x)}$$

~~n-1: 防止self correlation~~

n-1:只用少部分樣本在推論母體時因為偏量(bias)的關係，在推論時樣本推估會少一個自由度。

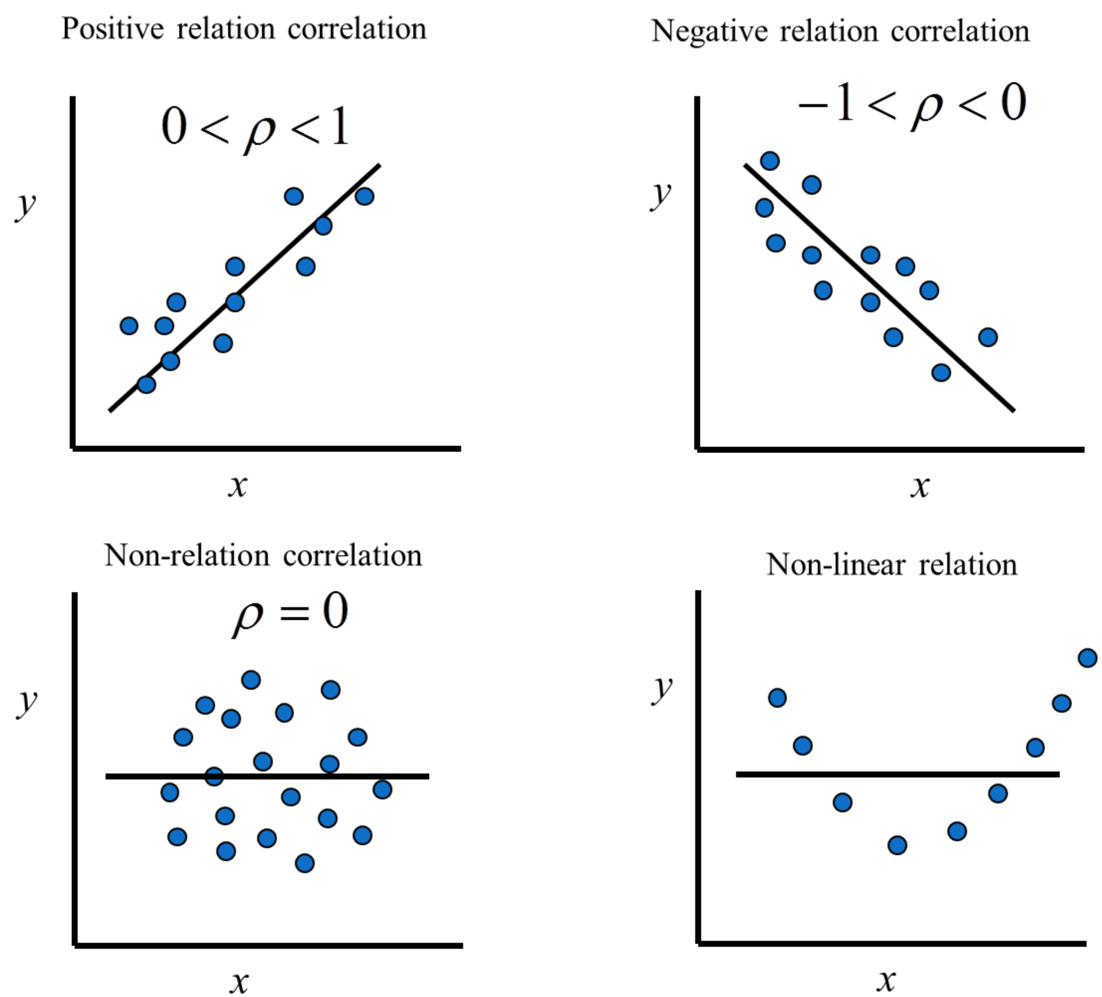
n-1: With only a small number of samples inferred the parent because of the relation of partiality (bias), the sample inferred less than one degree of freedom at the time of inference

https://medium.com/@chih.sheng.huang821/%E7%9B%B8%E9%97%9C%E4%BF%82%E6%95%B8%E8%88%87%E5%85%B1%E8%AE%8A%E7%95%B0%E6%95%B8-correlation-coefficient-and-covariance-c9324c5cf679

2.2-3 Correlation Vs. Covariance

4) Pearson’s Correlation Coefficient (探討線性關係): NCC r

r，有的時候會用ρ來表示



2.2-3 Correlation Vs. Covariance (5/5)

Tommy

- When mean value = 0, Covariance matrix = Correlation matrix

5) Correlation Matrix Vs. Covariance Matrix

- The covariance matrix of 2 random variables is given by

$$\Sigma = C_{XY} = \begin{bmatrix} Cov(X,X) & Cov(X,Y) \\ Cov(Y,X) & Cov(Y,Y) \end{bmatrix}$$

where Cov(X, X) = Var(X), Cov(Y, Y) = Var(Y)
Cov(X, Y) = Cov(Y, X),
> it is a symmetric matrix

- The sample covariance matrix can be computed using:

$$Cov(X,Y) = \frac{1}{n-1}\sum_{i=1}^{n-1}(x_i - \bar{x})(y_i - \bar{y})$$

-The covariance matrix of n random variables is given by

$$C_X = \begin{bmatrix} Cov(X_1,X_1) & Cov(X_1,X_2) & \cdots & Cov(X_1,X_n) \\ Cov(X_2,X_1) & Cov(X_2,X_2) & \cdots & Cov(X_2,X_n) \\ \cdots & \cdots & \cdots & \cdots \\ Cov(X_n,X_1) & Cov(X_n,X_2) & \cdots & Cov(X_n,X_n) \end{bmatrix}$$

where Cov(Xi, Xj) = Cov(Xj, Xi)

Σ是多維隨機變量的covariance matrix
μ爲樣本均值

Covariance can represent??
- When X and Y are dissimilar
  - ➔ Diagonal has higher values, cross has lower value
- When X and Y are **similar**
  - ➔ **each element has high value**

單個數據點 Gaussian model

- **Mahalanobis Distance**

$$((x-\mu)^T \Sigma^{-1}(x-\mu))^{0.5}$$

包括旋轉

- Inverse of covariance matrix is like putting covariance matrix to denominator (Scaling)
- If the correlation between variables are strong, the larger value in each element of covariance matrix. variables高度相關

Hence, smaller value of Mahalanobis Distance

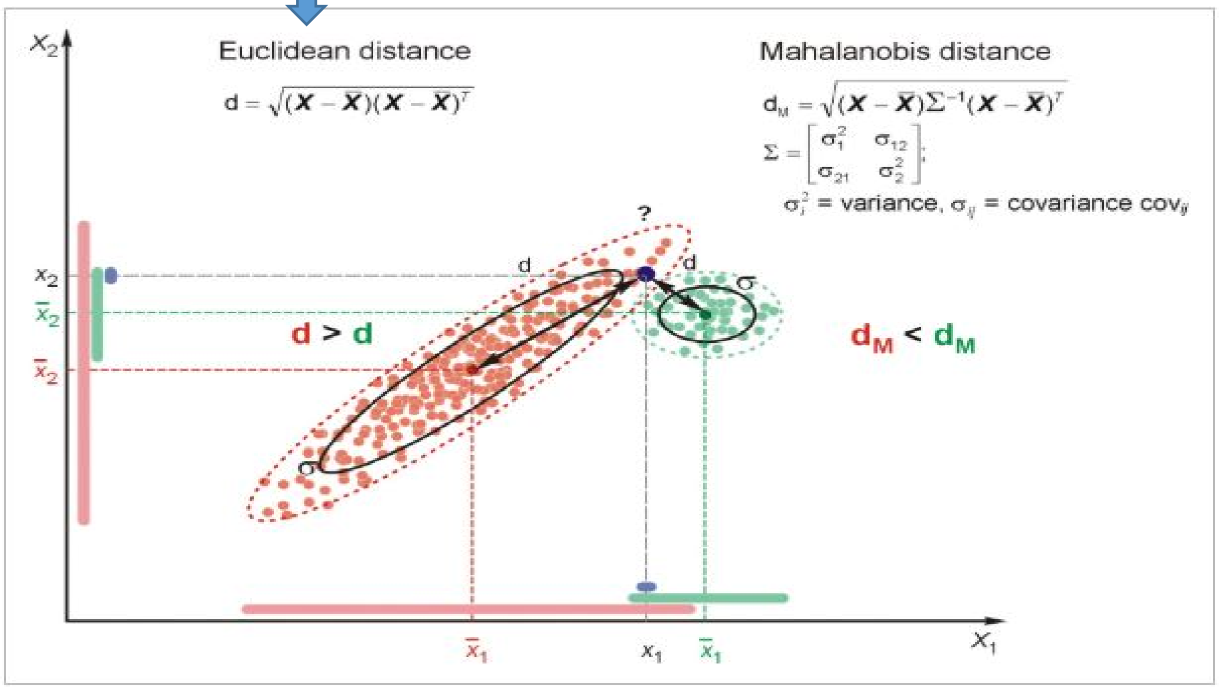
- (x −u) is essentially the distance of the vector from the mean
- divide this by the covariance matrix(or multiply by the inverse of the covariance matrix)

Problems of **scale** as well as the correlation of the variables變量的相關性問題

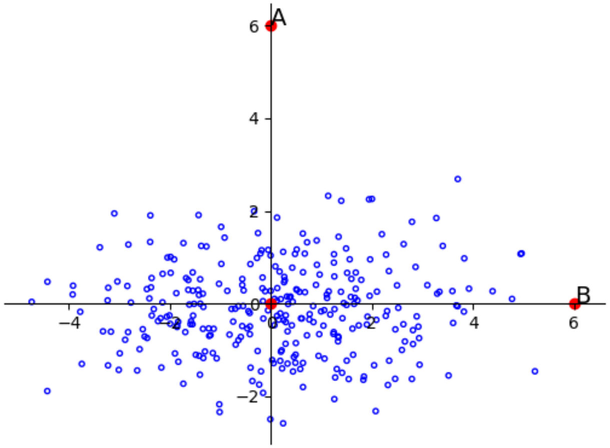
2.2-3 Correlation Vs. Covariance (5/5)

Mahalanobis 少 standard deviation

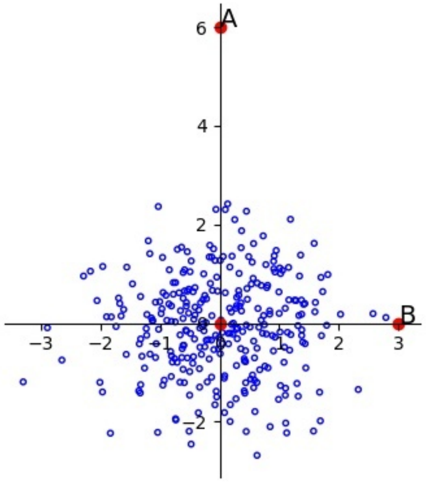
距離越遠越不相似



2.2-3 Correlation Vs. Covariance (5/5)



Euclidean distance



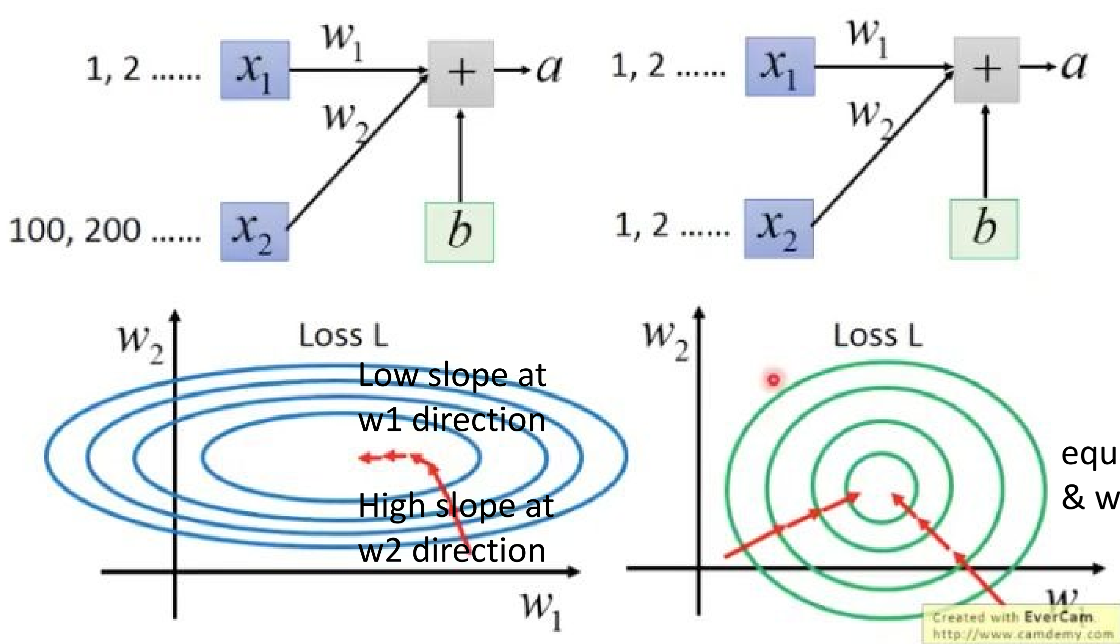
Mahalanobis Distance

Mahalanobis Distance 是基於樣本分布的一種距離。
Mahalanobis Distance 是一種距離的度量，可以看作是歐氏距離的一種修正，修正了歐式距離中各個維度尺度不一致且相關的問題。
物理意義就是在規範化的主成分空間中的歐氏距離。
所謂規範化的主成分空間就是利用PCA對一些數據進行主成分分解。
再對所有主成分分解**軸**做normalize，形成新的坐標軸。
由這些坐標軸張成的空間就是規範化的主成分空間。
PCA就是把橢球分布的樣本改變到另一個空間裡，使其成為球狀分布。
而Mahalanobis Distance就是在樣本呈球狀分布的空間裡面所求得的歐式距離。

4.3-3.2 Batch Normalization: Feature Scaling

Feature Scaling

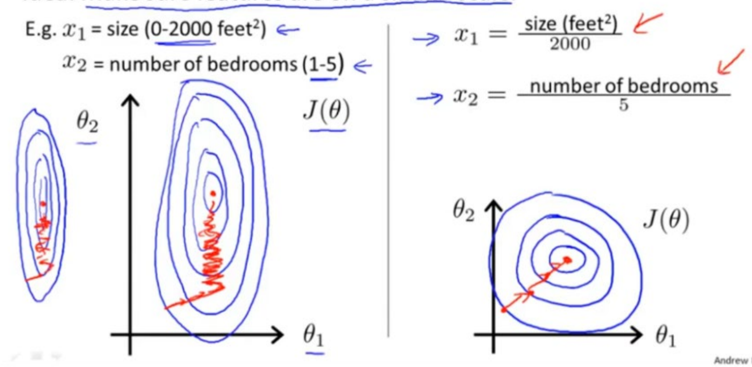
Make different features have the same scaling



Batch Normalization

Feature Scaling

Idea: Make sure features are on a similar scale.



左圖未經過normalization X_2特徵範圍 variance 遠大於 X_1，收斂時無法直接朝圓心前進

右圖經過normalization reference X_1特徵範圍等於X_2，收斂時直接朝圓心前進

- 通過左圖發現loss在W_1方向上的梯度變化緩慢，而在W_2方向上的梯度變化劇烈
- 如果在training過程中，為了避免這種情況發生，可以在W_1方向上使用較大learning rate,而在W_2方向上使用較小的learning rate but this is not an easy way to process
- Therefore, some methods: BN, ….

右圖為做完feature scaling後，不管在哪個方向上的梯度都是一樣的，在training過程中，model很快就能收斂了。

- If not normalize to same unit for x1 and x2, that is, without BN, then need to have different learning rates

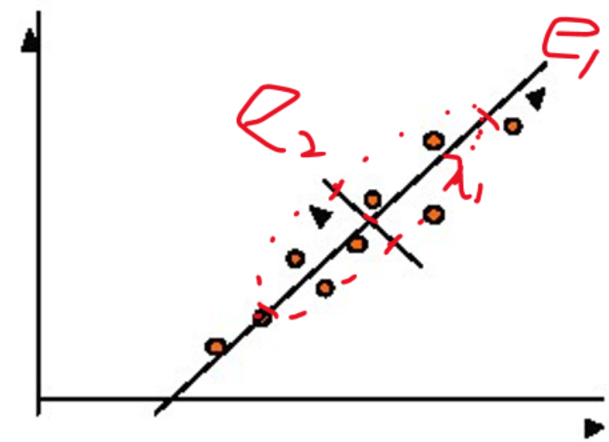
2.2-4 Principal Component Analysis (PCA) - Definition

多了 mean　　　　　　　　　　　　　　　　　　　　主成分分析

1. Covariance Matrix A Vs. Correlation Matrix A'　　Like Gaussian Model

Factorization:　萃取重要特徵
SVD (Singular Value Decomposition)

Estimate the shortest distance (errors) (point to plane) over all samples



2. $A=(A'-m)=UWV^T$
Xy original domain　　eigen domain
   - Gaussian distribution $N(m,\lambda^2)$,
   - as affine transform

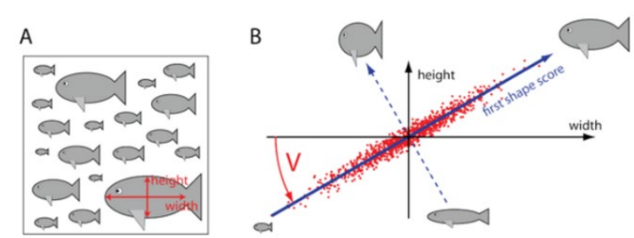
- eigenvalue就是變異量(variance)
- eigenvector就是讓資料投影下去會有最大變異量的投影軸。

3. Affine Transform –

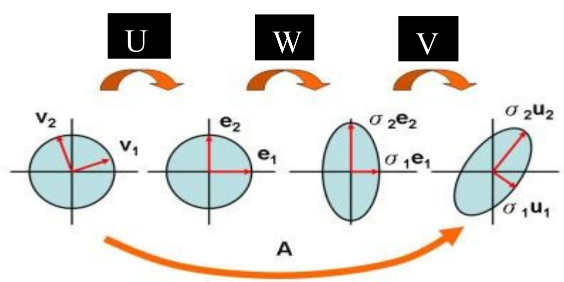
m: Mean as the translation terms

U: Eigenvector matrix as the rotation matrix　Like unit vector

W: Eigenvalues (λ^2) as the variance/scaling terms at denomination for normalization.　Like amplitude



V: Rotation matrix
V可以換到前面



Variance越大，差異越大，
例如人臉掃描中的眼睛

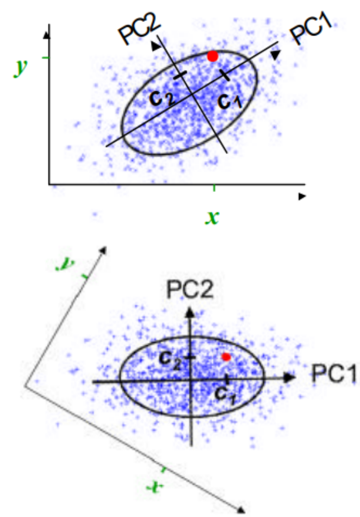
2.2-4 Principal Component Analysis (PCA) - Definition

- PCA 特徵量分析 多元統計分布的方法。
- 結果可以理解為對原數據中的變異數做出解釋：哪一個方向上的**數據值對變異數的影響最大**？換而言之，**PCA提供了一種降低數據維度的有效辦法**；如果分析者在**原數據中除掉最小的特徵值**所對應的成分，那麼所得的**低維度數據必定是最優化的**
- 有效的減少維度數，但整體變異量並沒有減少太多（這樣降低維度必定是**失去訊息最少的方法**）
- 主成分分析在分析複雜數據時尤為有用。

- 把維度數降到最低的同時，維持最多的資訊量(變異數)
- PCA是一個將原本維度投影到幾個存在最大的變異數的維度，來達到既降維，又維持足夠的資訊量(變異數)

Principle component analysis (PCA) finds the **directions of the axes** of the ellipsoid.

There are two ways to think about what PCA does next:
– Projects every point perpendicularly(垂直) onto the axes of the ellipsoid.
– Rotates the ellipsoid so its axes are parallel to the coordinate axes, and translates the ellipsoid so its center is at the origin.
(旋轉橢球，使其軸平行於坐標軸，並平移橢球，使其中心在原點)



Two views of what PCA does

- Transforms $\underset{\mathbf{x}}{\begin{bmatrix} x \\ y \end{bmatrix}} \rightarrow \underset{\mathbf{c}}{\begin{bmatrix} c_1 \\ c_2 \end{bmatrix}}$
  – Projects every point perpendicularly onto the axes of the ellipsoid.
  – Rotates space so that the ellipsoid lines up with the coordinate axes, and translates space so the ellipsoid's center is at the origin.

2.2-4 PCA – Domain Knowledge

1. Factorization Using SVD for space transformation – feature principal extraction
2. PCA for dimensionality reduction + LDA (Linear Discrimination Analysis) for classification

PCA目標是希望找到投影軸讓資料投影下去後分散量最大化，但PCA不需要知道資料的類別 (unsupervised learning)

LDA是希望資料投影下去後分散量最大，這個分散量是希望「不同類別之間的分散量」越大越好。(supervised learning)

3. Deep Learning: Autoencoder and GAN for encoding and decoding (reconstruction, synthesis合成)

4. PCA: Orthonormal原始數據降維並提取出不相關的屬性

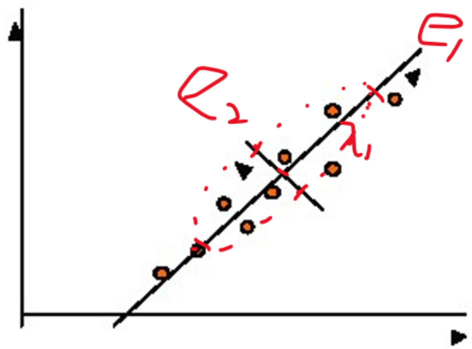
   (目的是找到這樣一組分量表示，使得重構誤差最小，即最能代表原事物的特徵。

   PCA要求找到方差最大化的方向，各個成分是Orthonormal正交的)

   ICA: Independent原始數據降維並提取出相互獨立的屬性

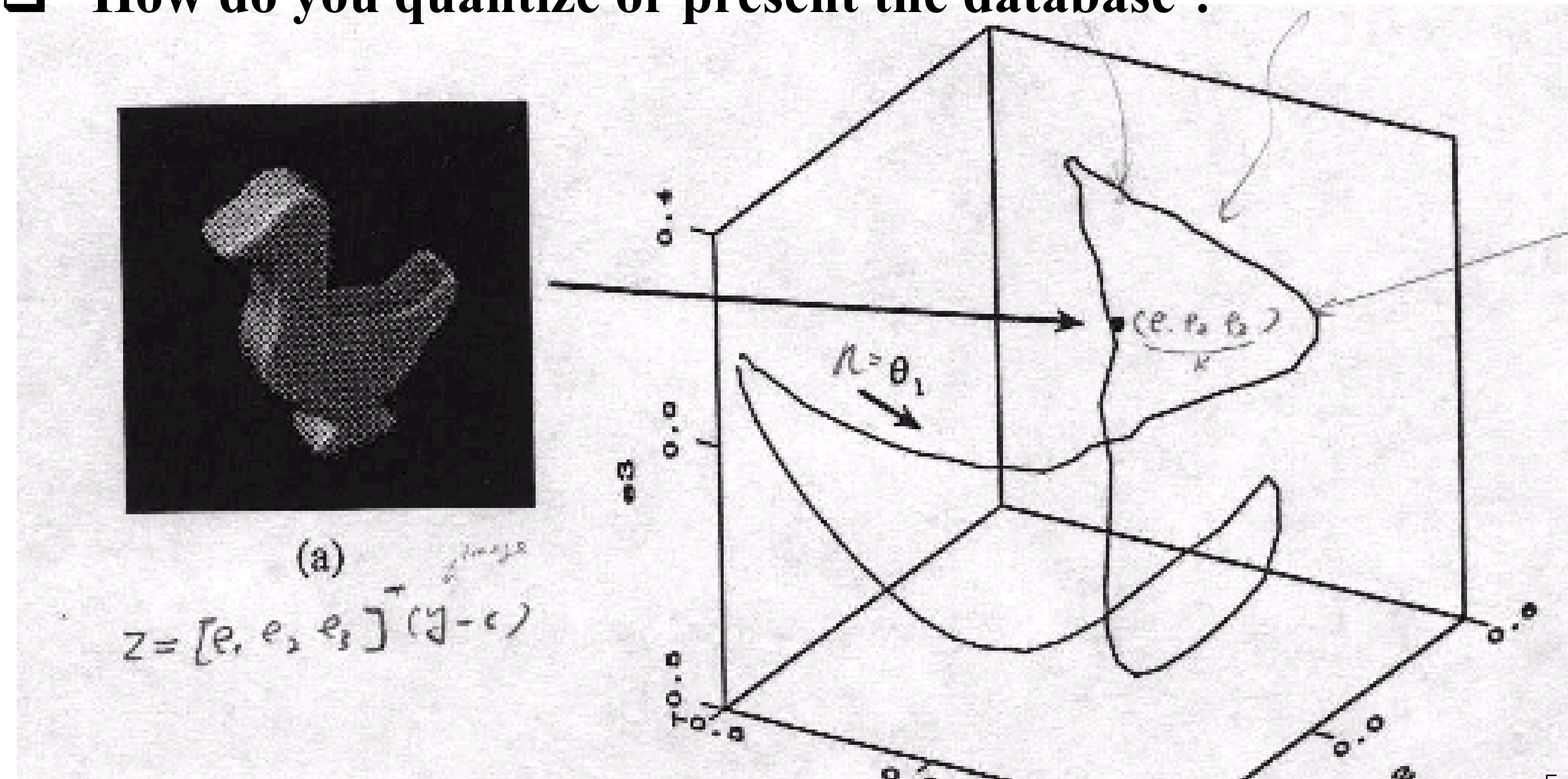
   (目的是找到這樣一組分量表示，使得每個分量最大化獨立，能夠發現一些隱藏因素。

   ICA要求找到最大獨立的方向，各個成分是Independent獨立的)



2.2-4 PCA – Domain Knowledge

- ❑ **How do you quantize or present the database ?**



(b)

Ax=b
Linear combination
Vector matrix

Manifold (subspace): Inter-interpolation 插值

$$[1 \quad 2 \quad 3]\begin{bmatrix} 11 & 12 & 13 \\ 21 & & \\ 31 & & 33 \end{bmatrix}$$

Eigen(cos,sin)

Linear Combination (= $\sum$) of basis functions

(=>Homogenous Matrix or Matrix * [1 1 … 1]^T):

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}\begin{bmatrix} 11 & 12 & 13 \\ 21 & & \\ 31 & & 33 \end{bmatrix}$$

1*v1+1*v2+…+1*v_M=landa1*e1+...+landaM'*eM'+…+landaM*eM

given any vi = $\boxed{\text{w1*e1+w2*e2+…+wM'*eM'}}$+…+w_M*e_M

Independent Vs. Orthogonal

2.2-4 PCA - Solution/Optimization of Homogenous Matrix

Local optimization (0 moment): 1 and 2
Global optimization: 3

X=input data，不等於0
A≅ 0

1. Closed Form Solution:

$Ax = 0$ ➔ A^TA = Covariance Matrix => PCA => Smallest not-zero eigenvalue => eigenvector

Homogenous Matrix symmetric matrix

2. Pseudo Inverse:

1 and 2 first order for local minima

$Ax = b$ ➔ $x=(A^TA)^{-1}(A^Tb)$

3. Sum of Squared Difference: (max likelihood – exponential term)

$\min\ E = \sum [Ax - b]^2$ Second order

3.1 $Ax = b'$: estimation value. b: ground truth, $E = \sum [b - b']^2$

a. Initial value estimation => Pseudo Inverse

b. L-M (non-linear approach) Global minima

b.1 First order Taylor series expansion

b.2 2nd order Taylor series expansion (sensitive to noise)

3.2 $Ax = b'$: estimation value. b'': estimation value, $E = \sum [b'' - b']^2$ 沒有Ground truth

a. EM (Expected-Maximization), initial b = average value

average value

Ex. Unsupervised learning(要做預測)

4. Lagrange Approach (outlier)

$\min\ E = \sum [Ax - b]^2 + \lambda (x^2+y^2)^2$

Weighting
Delete noise(outlier)

Content: Topic1. Similarity Measure（相似度量測）

1.0 Computer Vision - Camera Model: Between 3D and 2D

- 1.1 2D Image
- 1.2 Camera Model Btw 3D Object and 2D Image
- 1.3 From Linear Combination to Homography Ax=b (Vector, Matrix)

2.0 Computer Vision (CV) and AI: Similarity Measure = Loss Function

- 2.1 Sum-of-Squared Differences (SSD) and Mean-Squared Error (MSE)
- 2.2 Normalized Correlation Coefficient (NCC) and Cross Entropy (CE)
- 2.3 AI Bayesian Decision Rule - Maximum a Posteriori (MAP) Probability with Gaussian Model

3.0 Image Processing for Deep Learning: Filter, Correlation and Convolution

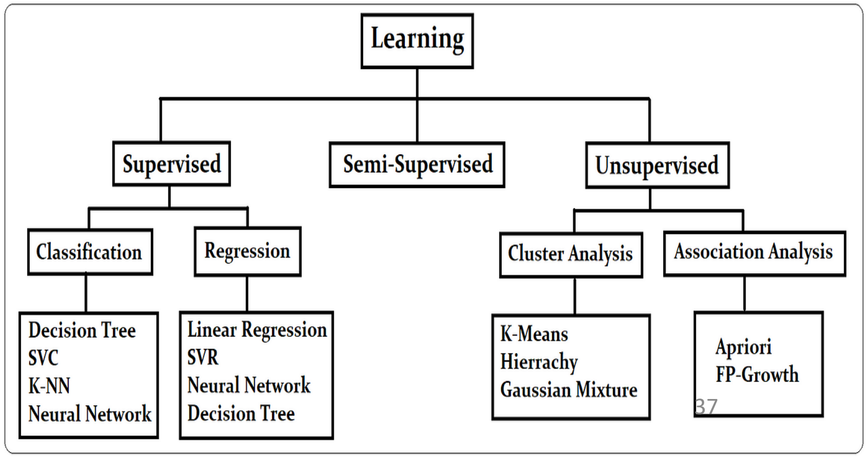
- 3.1 Low-Pass Filter - Denoise and High-Pass Filter - Edge Detection
- 3.2 Correlation Vs. Convolution
- 3.3 Multi-Resolution - Downsampling (Encoder) and Upsampling (Decoder)

2.3 AI Bayesian Decision Rule - Maximum a Posteriori (MAP) Probability with Gaussian Model

I/O Data and Loss Function: Data Collection and Labeling (1/2)

1. Supervised Learning: Label
   - Is a problem with labeled data, expecting to develop predictive capability
   - ➔ Self-supervised learning: Makes use of the structure within the data to generate its own labels.
2. Unsupervised Learning: Unlabeled
   - All the observations in the dataset are unlabeled and the algorithms learn to inherent structure from the input data
   - Is discovering process, diving into unlabeled data to capture hidden information.
   - ➔ Supervised machine learning is generally used to classify data or make predictions,
   - ➔ Whereas unsupervised learning is generally used to understand relationships within datasets.
   - ➔ Supervised machine learning is much more resource-intensive because of the need for labelled data.
3. Semi-Supervised Learning:
   - Some of the observations of the dataset are labeled but most of them are usually unlabeled.
   - So, a mixture of supervised and unsupervised methods are usually used.
4. Reinforcement Learning:
   - Differs from supervised learning in a way that in supervised learning the training data has the answer key with it so the model is trained with the correct answer itself
   -whereas in reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task.

Type	Supervised	Unsupervised	Semi-supervised
Input Data	Labelled	Unlabelled	Partially Labelled
Computational Complexity	Simpler	Computationally Complex	Depends on use of Supervised or unsupervised
Accuracy	Higher	Lesser	Lesser



I/O Data and Loss Function: Data Collection and Labeling (2/2)

0.1 (1/4): Reinforcement Learning – Scenario detail

P: (from reinforcement learning book:
-State is different from observation
-Example: Go
> State: Probably consist of 棋譜 和 position of black-and-white 棋子... all information has only one status even under diff. view images
> Observation: Different images from different view angles, but having the same (only one) status

Find or define policy for reinforcement learning!
Reinforcement: Markov decision, consider only current t and previous statement t-1

1) Discrete actions (video game, GO)
2) Continuous actions (robot arm)

(One Episode) MDP (Markov Decision Process)

~~Trajectory $\tau = \{s_0, a_0, r_1, s_1, a_1, r_2, \ldots, s_{T-1}, a_{T-1}, r_T, s_T\}$~~

joint determined

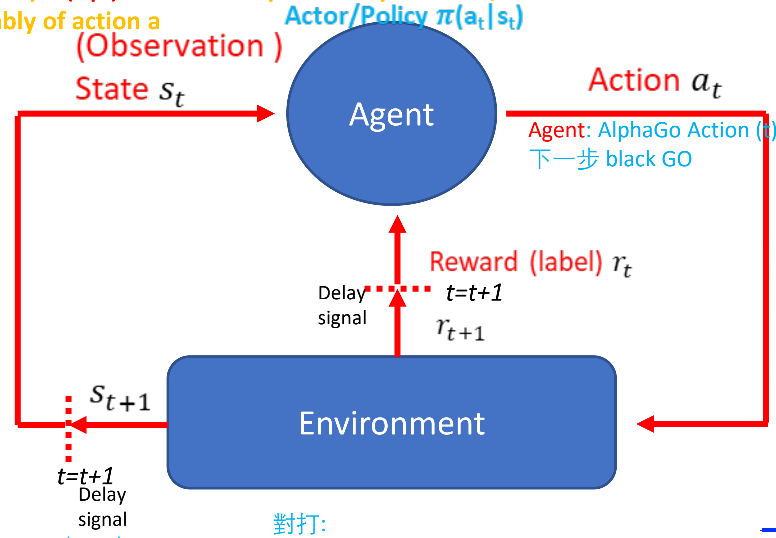
T: termination

Trajectory $\tau = \{\boxed{s_1, a_1}, r_1, s_2, a_2, r_2, \ldots, \boxed{s_T, a_T}, r_T\}$

joint determined s & a

T: termination

-P: Actor follows a **policy as a function as the NN**. If NN is a deep NN model, then it calls **deep reinforcement learning**.
-Actor/Policy π(a|s): Conditional probability. Given s, take the probably of action a



對打:
Agent AlphaGo Action – Black 下一步
Env – White 下另一步

Sometimes multiply discount factor γ (0~1)
短視近利or重視長遠
Exp. Exponential reduce along the time

This is the reward for only one trajectory:

Total reward: (long-term reward)

$$R(\tau) = \sum_{t=1}^{T} r_t$$

This is average reward over *N* different trajectories:

over τ (all)
J: Margin??

$$\bar{R}_\theta = \boxed{\sum_{\tau}} R(\tau) \boxed{P(\tau|\theta)} \approx \boxed{\frac{1}{N}\sum_{n=1}^{N}} R(\tau^n)$$

As weight

Expected cumulative reward per episode

Agent learns to take actions to maximize expected reward.

$$\theta^* = arg \max_{\theta} \bar{R}_\theta$$

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla \bar{R}_\theta \approx \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n}(R(\tau^n) - b)\nabla logp(a_t^n|s_t^n, \theta)$$

An agent (actor) like a function (Neural Network)
Action = π (Observation)
~~機率分布: π(a|s)~~

Q: How to choose a good actor (policy)?
A: an actor follow the policy which expected reward is maximum than other policy.
That is, the reaction with the environment.

J: Similar to HMM: dynamic programming - shortest path

2.3-1 Similarity Measure: Likelihood Probability [0.0, 1.0]

- From sum-of-squared differences (SSD) to cross correlation (CC)

$$\min SSD = \min(T - I)^2 = \min(T^2 - 2TI + I^2)$$

$$\min SSD \rightarrow (T^2 + I^2) - 2(TI)$$

$$\therefore \max CC \rightarrow TI$$

$T * I$

Un-correlated:

$$\sum = \begin{pmatrix} \sigma_1^2 & 0 & 0 \\ 0 & \sigma_2^2 & 0 \\ 0 & 0 & \sigma_3^2 \end{pmatrix}$$

1. Likelihood Probability [0.0,1.0]:
   From sum-of-squared differences (SSD) to Gaussian model (normal distribution) $N(\mu,\sigma^2)$

0~1之間

$$P(x) = \frac{1}{\sqrt{(2\pi)^D|\Sigma|}} \exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))$$

$$P(I|x) = \frac{1}{\sqrt{(2\pi)^D|\Sigma|}} \exp(-\frac{1}{2}(x-I)^T\Sigma^{-1}(x-I))$$

除以covariance
空間單位一樣(Scaling)

距離越小exp越接近0，機率接近1

Covariance Matrix

$$\Sigma = \begin{bmatrix} (x_{11}-\mu)^2 & \cdots & (x_{1D}-\mu)^2 \\ \vdots & \ddots & \vdots \\ (x_{D1}-\mu)^2 & \cdots & (x_{DD}-\mu)^2 \end{bmatrix}$$

➔The similarity btw **input unknown patch** ***I*** and **ground truth face** ***x***
I: 24x24-pixels patch; *x*: Face, ~*x*: Non-Face

2. Likelihood Probability [0.0,1.0]:
   From **cross correlation** (CC) to normalized correlation coefficient (NCC) *R*

$$|R(x,y)| = \left| \frac{\sum_{y'=0}^{h'-1}\sum_{x'=0}^{w'-1}[(T(x',y')-m^T)*I(x+x',y+y')-m^I)]^2}{\sqrt{\sum_{y'=0}^{h'-1}\sum_{x'=0}^{w'-1}[(T(x',y')-m^T]^2}\sqrt{\sum_{y'=0}^{h'-1}\sum_{x'=0}^{w'-1}[(I(x+x',y+y')-m^I]^2}} \right|$$

2.3-1 Likelihood Probability by Gaussian Model

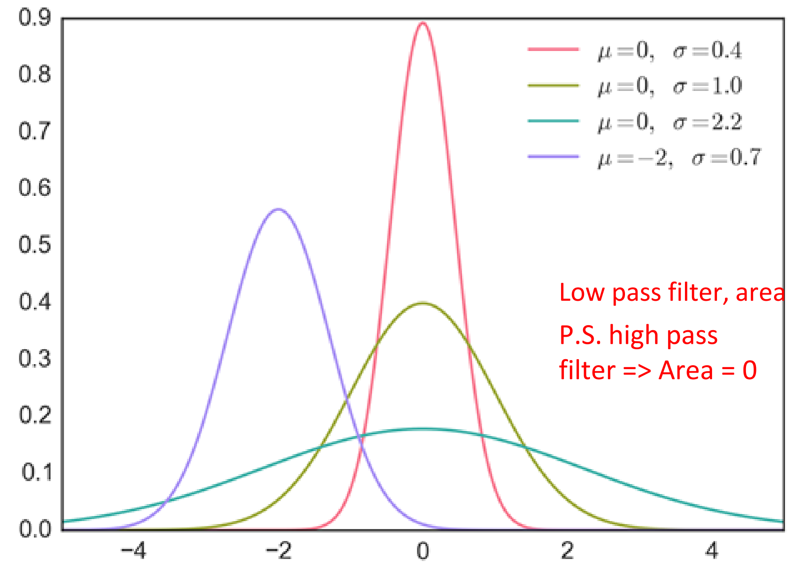
1. Gaussian Distribution (Normal Distribution):
   1) 當樣本為1-D (univariate)，高斯分佈遵守機率密度函數 (Probability Density Function - PDF):

$$\boxed{\mathrm{p}(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}$$

$x \in \{x_1, x_2, \cdots, x_n\}$

$\mu = \frac{1}{n}\sum_{i=1}^{n} x_n$: mean (平均值 or 期望值): Average template

$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2}$: standard deviation (標準差)



- 右圖為不同平均數與標準差，所對應的機率密度函數曲線。

Gaussian model 機率0~1之間

往兩端推開，避免卡在中間

- Gaussian Distribution 物理意義:

- *distance of x from μ*
- *degree of spread*

Shift/Translation SSD 移到中間

$$\mathrm{p}(x) = \frac{1}{\boxed{\sigma\sqrt{2\pi}}}\exp\left(-\frac{(\boxed{x-\mu})^2}{\boxed{2\sigma^2}}\right)$$

Like **Activation Function of deep learning**

Affine translation:
- "- "Shift/translation,
- σ normalization/scaling

variance

Normalization/Scaling高矮調成一致

Normalization term: to between 0.0 ~ 1.0

- 位移: 回歸基準
- Exponential: 使大小值拉開
- 除標準差: 使 term 統一單位

- *μ: Depend on Center-Position by* μ *(average)*,
- *σ Depend on height and degree of spread by* σ *(standard deviation)*

a) Gaussian distribution (Probability) 1被、2被、3被 標準差面積

m+-σ (66%), m+-2σ(95%?), m+-3σ(99%)

b) Gaussian Mixture Model (GMM)

➔ Weighted GMM estimated by EM

(x-u)^2=x1*x2=(x1-u1)*(x2-u2) 兩個不同物體，內積越大越相似

Exp(曲線切一半，把值逼近兩端)相似度量測，越接近1越相似

目的Detection、recognition

http://work.thaslwanter.at/Stats/html/statsDistributions.html

2.3-2 AI Bayesian Decision Rule –
Maximum a Posteriori (MAP) Probability with Gaussian Model

1. Maximum a Posteriori (MAP) Probability with Gaussian Model

1) Posteriori Prob. ∝ Likelihood Prob. * Priori Prob.

Max:

MAP: *Prob. Of existed face?* Given x

$$p(\Omega_f \mid x) = \frac{p(x \mid \Omega_f)\, p(\Omega_f)}{p(x \mid \Omega_f)\, p(\Omega_f) + p(x \mid \Omega_n)\, p(\Omega_n)}$$

$$p(\Omega_n \mid x) = \frac{p(x \mid \Omega_n)\, p(\Omega_n)}{p(x \mid \Omega_f)\, p(\Omega_f) + p(x \mid \Omega_n)\, p(\Omega_n)}$$

x: 24*24-pixels patch

f: 24*24-pixels face template

n = ~f: 24*24-pixels non-face template

2) Likelihood Prob. by Gaussian Model

> The similarity btw *x* and *f*

$$p(x|f) = \frac{\exp[-\frac{1}{2}(x-f)^T \Sigma^{-1}(x-f)]}{(2\pi)^{1/2}|\Sigma|^{1/2}}$$

3) Priori Prob. *$p(f_t)$!= 1/2*

> Example: Location of face appearance probability at time *t*

2. Bayesian Decision Rule

$$x \in \begin{cases} \Omega_f & \begin{cases} if \;\; p(\Omega_f \mid x) \geq p(\Omega_n \mid x) \\ If \;\; p(f|x) > threshold \end{cases} \\ \Omega_n & otherwise \end{cases}$$

$$\frac{p(x \mid \Omega_f)}{p(x \mid \Omega_n)} \underset{nonface}{\overset{face}{\underset{<}{\geq}}} \frac{p(\Omega_n)}{p(\Omega_f)} = \gamma$$

Constant by experiment