



## How to Learn Deep Learning? (2/2) (如何學習人工智慧-深度學習?)

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](mailto:jjlien@csie.ncku.edu.tw)

Department of Computer Science and Information Engineering  
Robotics Lab



# 3.0 History: Computer Vision and Deep Learning

## 1. Industrial Revolution:

3)1969, Industry 3.0:  
Computing and Automation

4)2013, Industry 4.0:  
Smart Manufacturing

1)1981, 生產力1.0：程式化

2)1991, 生產力2.0：整線化 3)2001, 生產力3.0：電子化 4)2011, 生產力4.0：智慧化

1990

2000

2010

2020

## 2. OpenCV: Image Processing, Computer Vision, Machine Learning, and Deep Learning

0)1999, [OpenCV 0.0 Started](#) 1)2007, [OpenCV 1.0: IP+CV](#) 3)2016, [OpenCV 3.0: IP+CV+ML](#)  
2)2009, [OpenCV 2.0: IP+CV](#) 4)201x, [OpenCV 4.0: IP+CV+ML+DL?](#)

1)1999, OpenCV Supported by Intel

2)2010, OpenCV Supported by Nvidia  
3)2011, OpenCV Supported by Google

## 3. Deep Learning:

1)1989, LeCun tPami1989

2)1998, LeNet IEEE1998

3)2012, AlexNet NIPS2012

## 4. Robotics Lab.: Computer Vision and Machine Learning – 1991 ~ 2013

1)1993, Facial Expression Recognition at CMU, USA

2)1999, Face Detection and Recognition at a Distance Using Stereo at Visionics, USA

3)2004, Automatic Optical Inspection (AOI System Integration): TFT-LCD, Solar Cell...

4)2009, eCV Surveillance with TI & Faraday

## 5. Robotics Lab.: Computer Vision and Deep Learning - Since 2014: AI智慧製造+AI半導體+AI醫材+AI

# Content



- 0.1 [Profile](#)
- 0.2 [Robotics Lab. at NCKU](#)
- 1.0 [History: Computer Vision and Deep Learning](#)
- 1.1 [Industry 4.0 in 2013](#)
  - 1.2.1 [Application 1: eCV Surveillance System](#)
  - 1.2.2 [Application 2: Scraping System](#)
- 3.0.1 [Deep Learning – CNN: History](#)
- 3.0.2 [Deep Learning: Computer Vision Vs. Artificial Intelligence Network](#)
- 3.0.3 [Deep Learning – Basic: MLP-BP \(Multilayer Perceptron Back-Propagation\)](#)
- 3.1 [DL-Basic: LeNet 1998 – CNN](#)
- 3.2 [DL-Basic: AlexNet 2012 – Deep CNN](#)
- 3.3 [DL-Basic: VGG-16 2015 – Very Deep CNN](#)
- 3.4 [DL-Basic: ResNet 2016 – Deep Residual Learning](#)
- 3.5 [DL-Basic: Other Very Deep CNNs](#)
- 3.6.1 [DL-Basic: Application 1 – Body Joint Point Estimation Using VGG-16](#)
- 3.6.2 [DL-Basic: Application 2 – Scraping Segmentation Using U-Net](#)
- 4.1 [Multi-Task: Visual-Guided Robot Arm](#)
- 4.2 [Multi-Task: Detection, Segmentation, and Classification](#)
- 4.3 [Multi-Task: Faster R-CNN and Mask R-CNN](#)
- 4.4 [Multi-Input: RGB-D Camera](#)
- 4.5 [Multi-Input: Siamese Network for Image Fusion, Tracking, or Stereo](#)
- 5.0 [LSTM: Tool Life Signal Prediction](#)
- 6.0 [GAN: Augmented Data, or Image Deblurring](#)
- 7.0 [Reinforcement Learning: Robot Arm Grasp or Push, and Place](#)
- 8.0 [Embedded Deep Learning Using Nvidia Jetson TX2](#)

# 3.0 Deep Learning: 1. Basic CNN

Which one do I want to be?

1. Basic CNN Creation
2. Modified CNN
3. CNN Application: Data collection, organization and analysis / benchmark

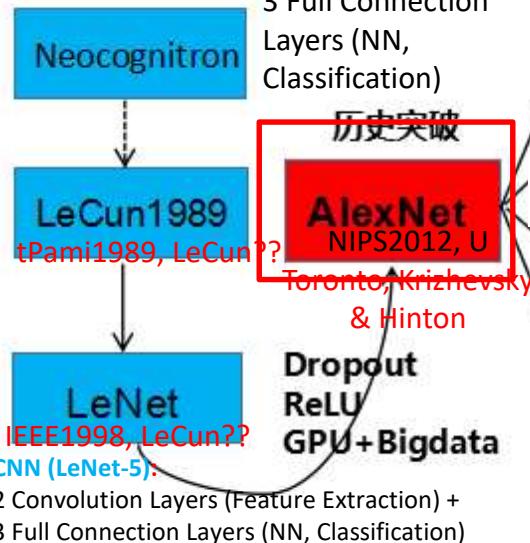
## 1. Basic CNN

Getting deeper:

Very Deep CNN, ICLR2015

Layer 16~19

**Deep CNN:**  
5 Convolution Layers  
(Feature Extraction)  
+  
3 Full Connection  
Layers (NN,  
Classification)



## 2. Multi-Task: Detection (ROI) + Classification

ECCV2014, MSR, He, Zhang, Ren, Sun  
SPP-Net  
R-CNN  
CVPR2014, tPami2015  
MSR, Girshick & Berkeley

增加新的功能单元

Inception V2

Fast R-CNN  
ICCV2015,  
MSR, Girshick

Input image size  
can be flexible

FCN

Fully Convolutional Networks

Faster R-CNN  
NIPS2015,  
MSR, Ren, He,  
Girshick, Sun

STNet

Mask R-CNN  
ICCV2017, FAIR,  
He, Gkioxari,  
Dollar, Girshick,

## 3. Temporal Domain

CNN+RNN/LSTM  
ICRA2017, Google,  
Berkeley, Finn, Levine

1. CVPR2016, Girshick

YOLO V1 V2 V3

Retina

## 4. GAN, AutoEncoder:

Generative Adversarial Network Vs. PCA

## 5. Reinforcement Learning

Unsupervised

tPAMI: IEEE Transactions on Pattern Analysis and Machine Intelligence

CVPR: Conference on Computer Vision and Pattern Recognition

NIPS: Conference on Neural Information Processing Systems

ICLR: International Conference on Learning Representation

NIN: Network in Network

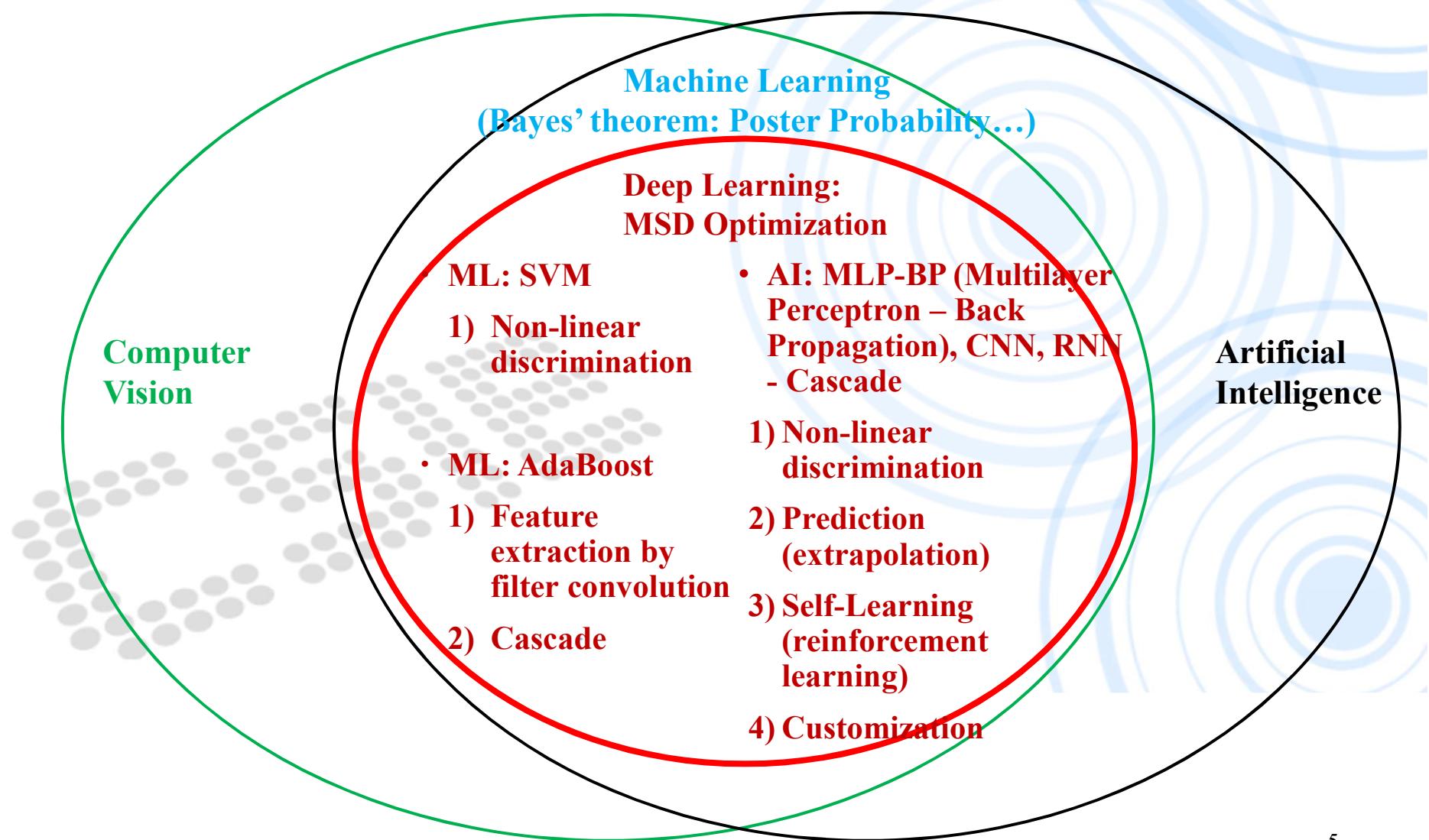
R-CNN: Region-based Convolutional Network method

SPP-Net: Spatial Pyramid Pooling networks

### 3.0.0 Deep Learning: Computer Vision Vs. Artificial Intelligence Network

Machine Learning: Need Data + Label

Deep Learning: Need Big Data + Label



# 3.0.0 Deep Learning: AI MLP-BP (Multilayer Perceptron Back-Propagation)

寫程式不會以equation的方式，會以vector, matrix ( $Ax = b$ ), homogenous matrix, 的方式撰寫。

$$WX = Y'$$

Batch size  $N$

## Loss Function: 1) MSE:

$$l = \min \frac{1}{N} \sum_{j=1}^N |y_j - y'_j|^2 + \text{Continu regressi}$$

2) Cross-Entropy:  $I = \min - [\sum_{j=1}^N (y_j \log y'_j) + ...]$

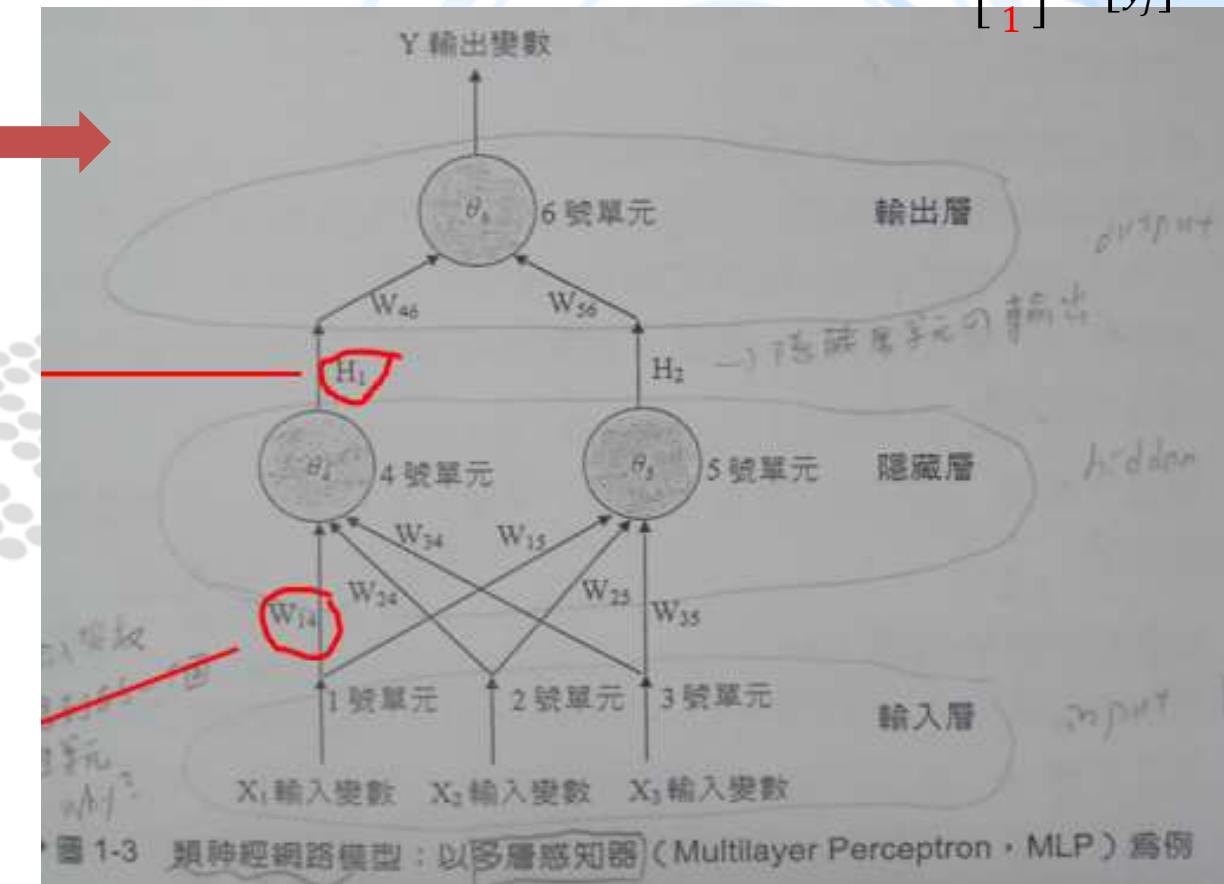
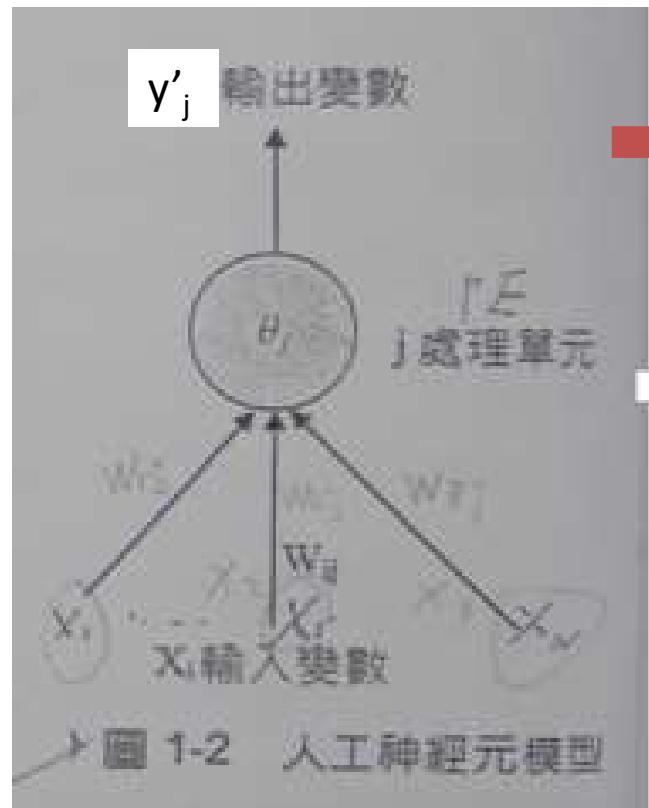
### 3) Softmax

-One weak classifier is one haar filter is also as one neuron unit

-One strong classifier is as one layer

$$\begin{matrix} & \text{Homogenous Matrix} \\ \begin{matrix} \text{ous,} \\ \text{on} \\ \text{crete,} \\ \text{specification} \end{matrix} & \left[ \begin{matrix} w_{11} w_{21} \cdots w_{k1} \cdots w_{N1} & b_1 \\ w_{12} w_{22} \cdots w_{k2} \cdots w_{N2} & b_2 \\ \vdots & \\ w_{1j} w_{2j} \cdots w_{kj} \cdots w_{Nj} & b_j \end{matrix} \right] \end{matrix}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \\ \vdots \\ x_N \\ 1 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \\ \vdots \\ y_j \end{bmatrix}$$



$$y_j' = f(\sum_i w_i x_i - \theta_j)$$

W: As coefficient of edge detection/feature extraction filter

b: As the threshold of edge detection/feature extraction filter, it is a bias



### 3.0.1 Deep Learning with Machine Learning – Non-Linear Discrimination: As Learnable Kernel (SVM)

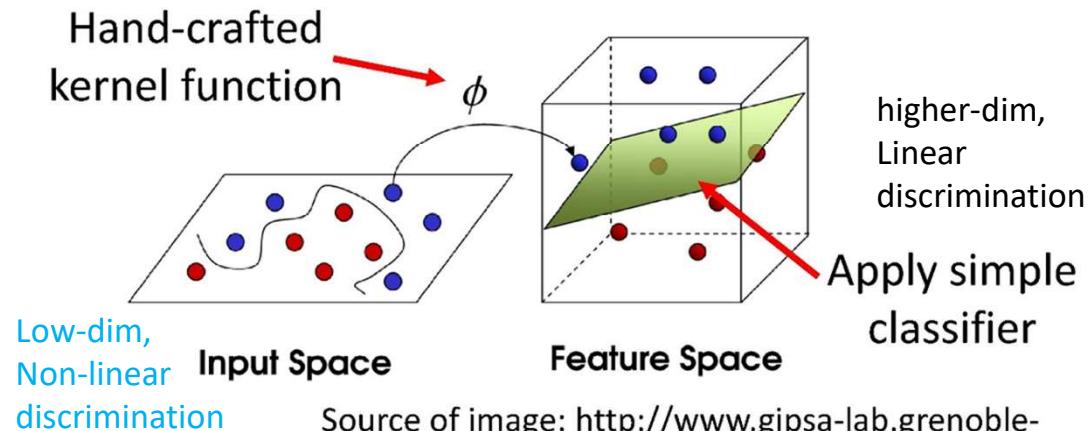
Deep Learning from ML:

Networks as

- 1) Non-linear discrimination: As learnable kernel (SVM)  $\phi()$
- 2) AdaBoost Cascade: As boosted weak / strong classifiers

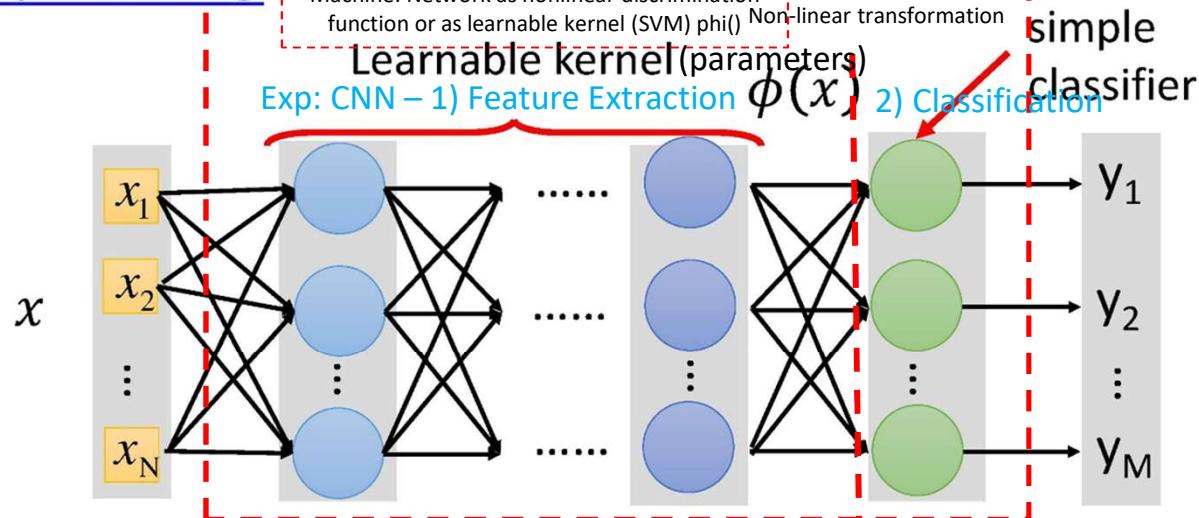
#### SVM

J: Without cascade concept



J: With cascade concept

#### Deep Learning



# 3.0.1 Deep Learning with Machine Learning – AdaBoost Cascade: As boosted weak/strong classifiers

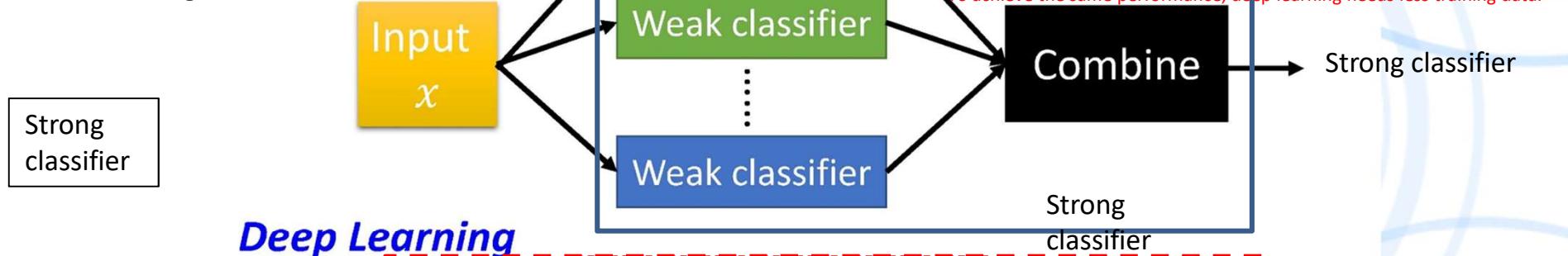
Deep Learning from ML:

Networks as

- 1) Non-linear discrimination: As learnable kernel (SVM)  $\phi$ )
- 2) AdaBoost Cascade: As boosted weak / strong classifiers

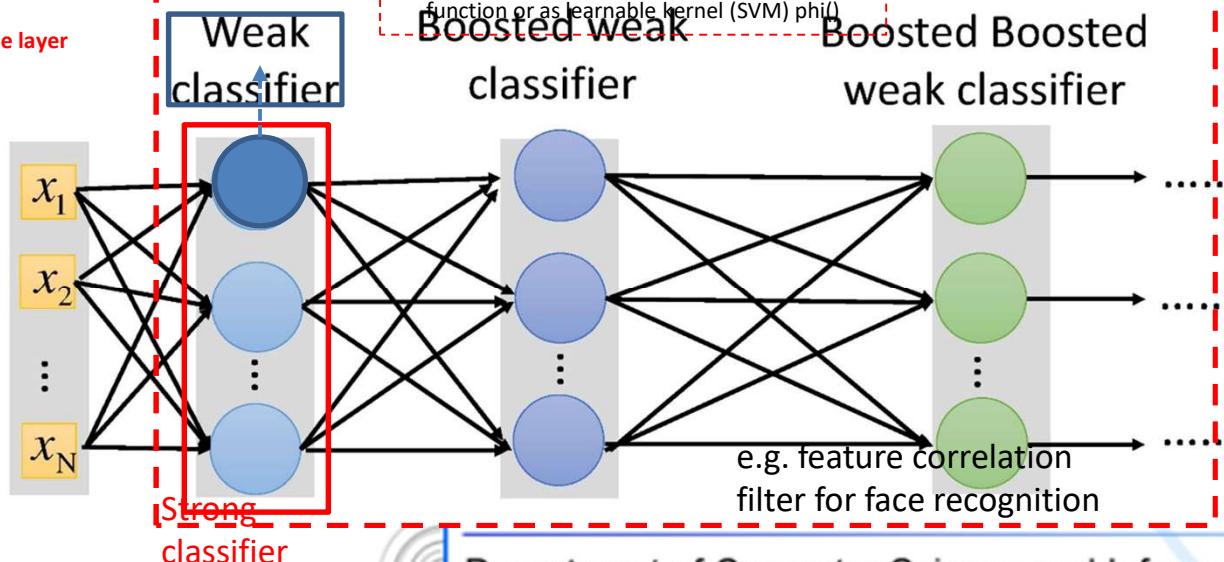
## Boosting

Cascade: Strong classifiers are cascaded



## Deep Learning

- One weak classifier is one haar filter is also as one neuron unit
- One strong classifier is as one layer

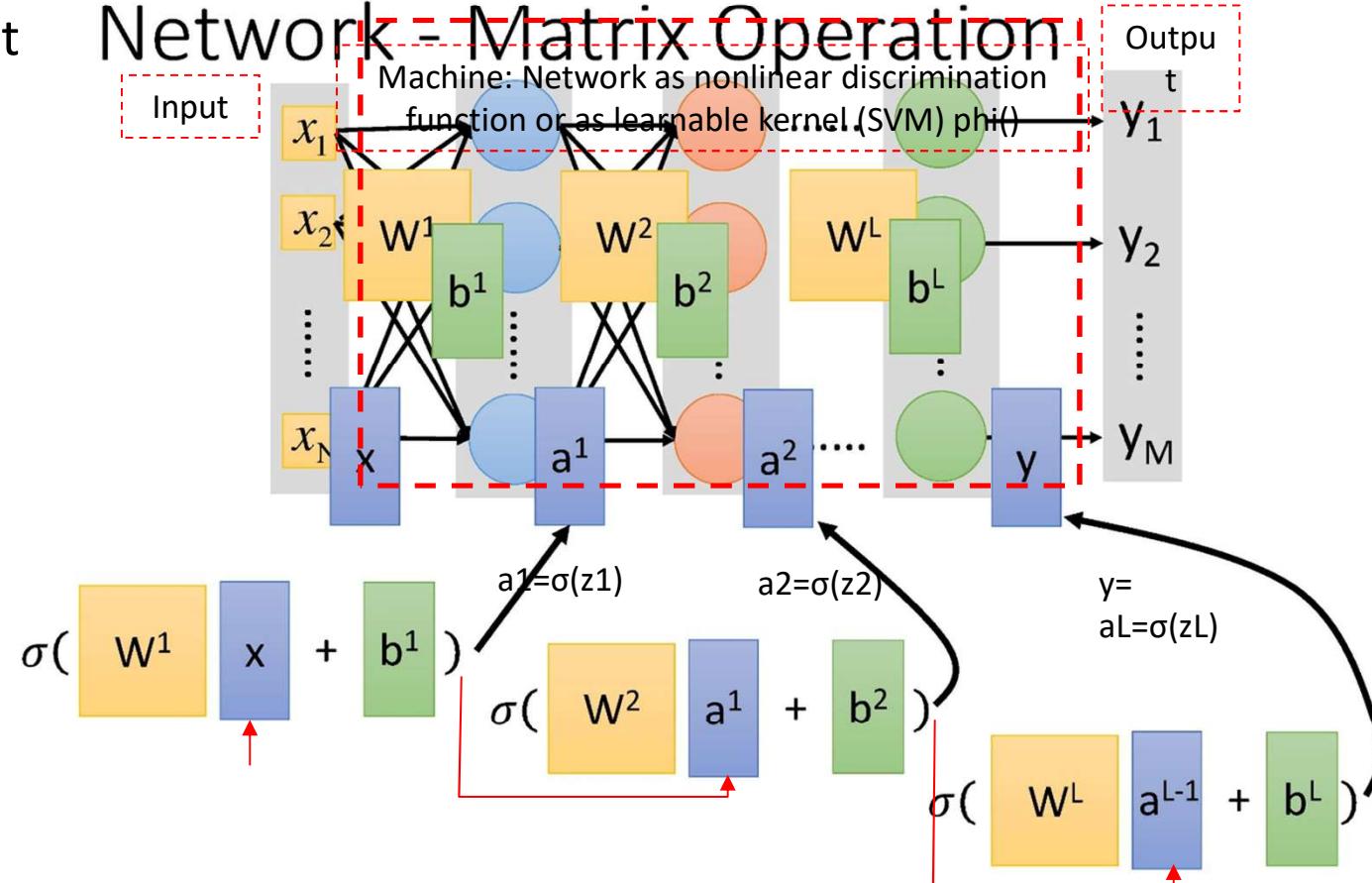


國立成功大學

### 3.0.2 Deep Learning with Machine Learning – AX=b

: AdaBoost  
- Cascade

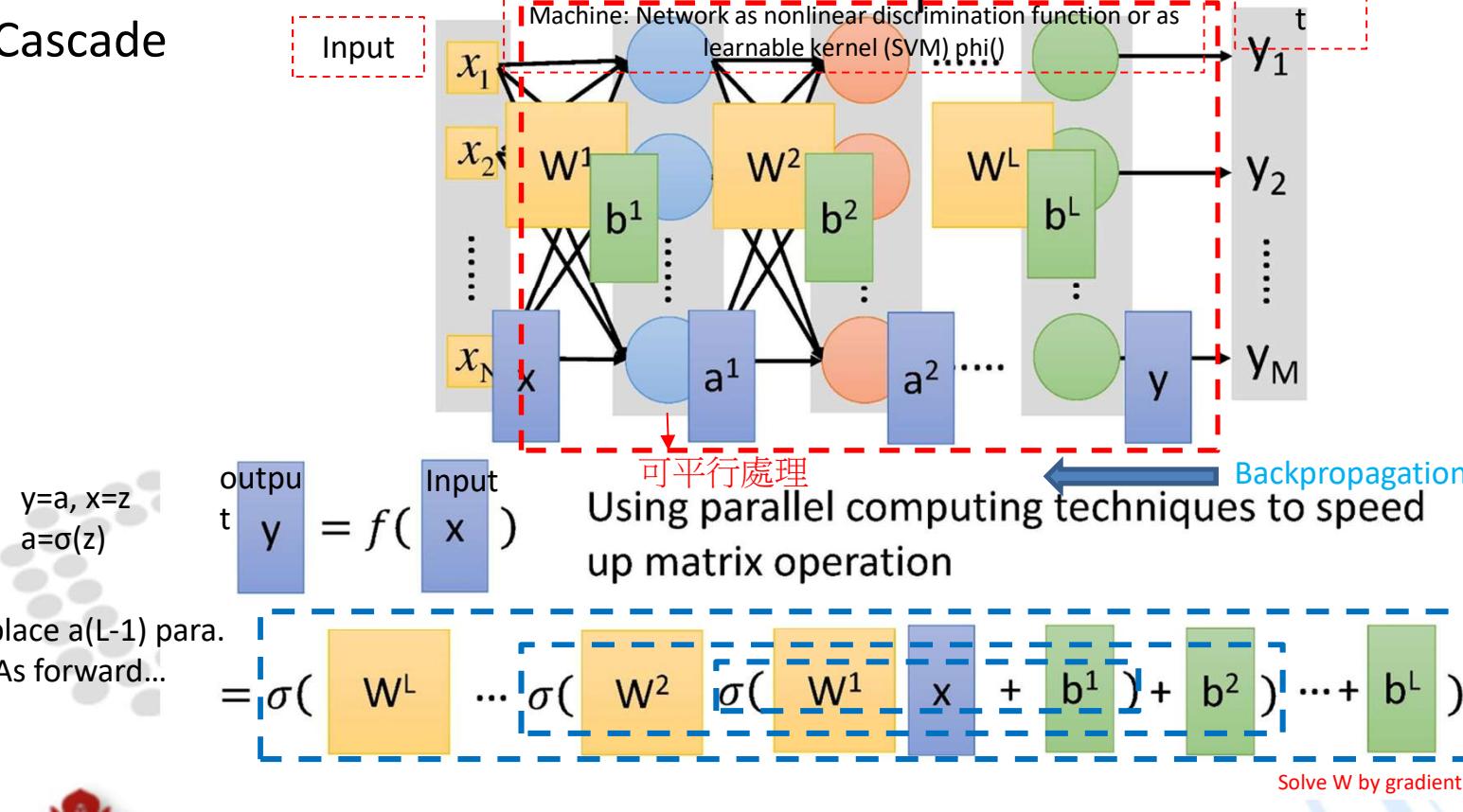
#### Fully Connect Feedforward Network - Matrix Operation



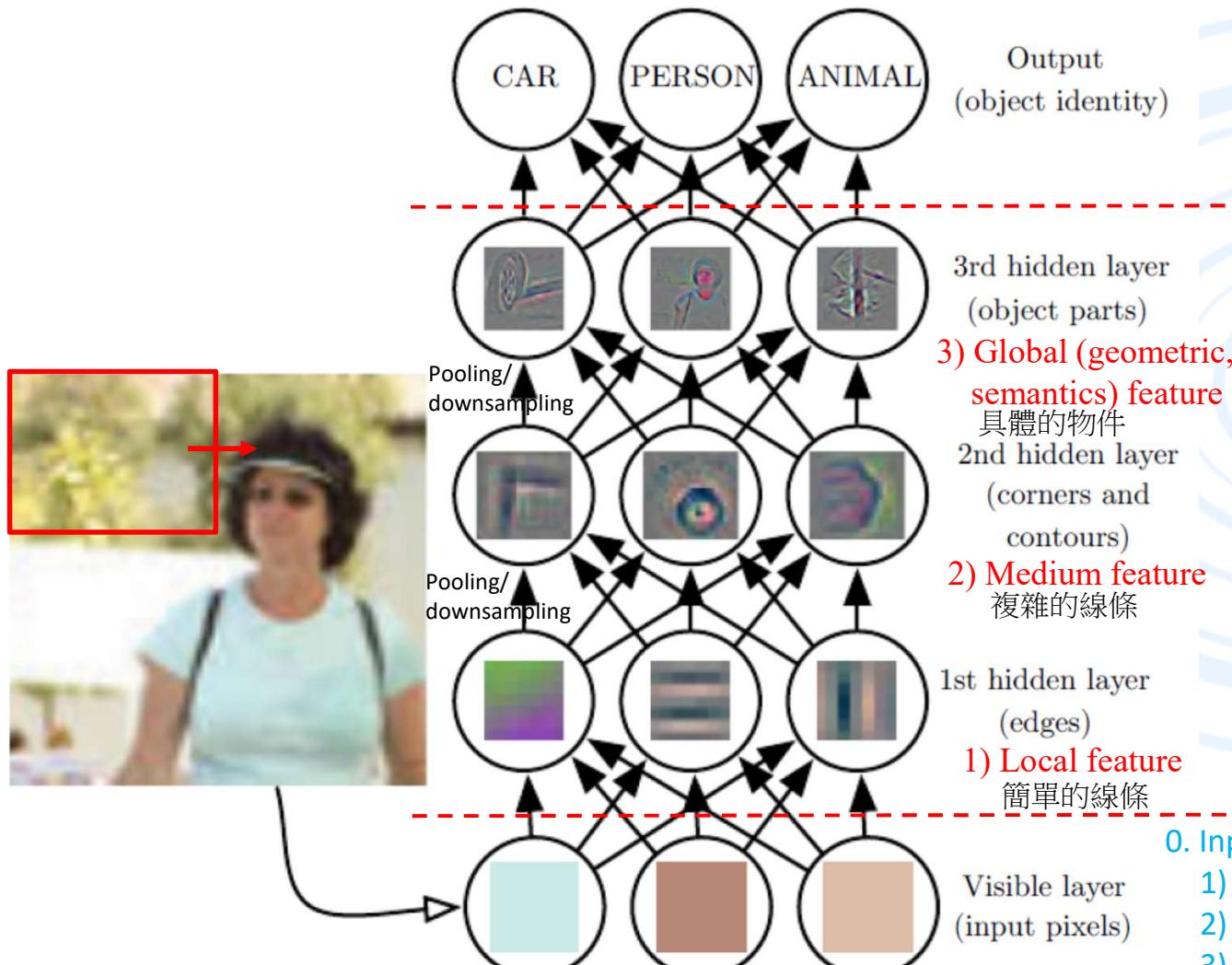
### 3.0.2 Deep Learning with Machine Learning – AX=b

: AdaBoost  
- Cascade

## Fully Connect Feedforward Network - Matrix Operation



### 3.0.2 Deep Learning: CNN Classification



#### 2. Classification Layers (NN)

#### 1. CNN: Feature Extraction Layers



#### 0. Input Data:

- 1) Texture > Sensitive to light
- 2) Shape > Sensitive to noise
- 3) Color > RGB to HSV  
> RGB to YUV for compression
- 4) RGB+Depth



## 3.2 AlexNet: 5 + 3 Fully Connectional Layers (1/2)

$\delta(j, \theta_i) = 1$  when gripper angle  $\theta_i$  corresponds to jth bin for ith image patch  
 $\delta(j, \theta_i) = 0$ , otherwise  
 $B$ : batch size,  $J$ : exists image patch  $i=1 \sim B$   
 $A_{ji}$ : forward pass binary activations  
 $l_i$ : the label corresponding to angle  $\theta_i$   
 AlexNet: 2012年, ImageNet比赛冠军的模型  
 $i$  : current iteration ???

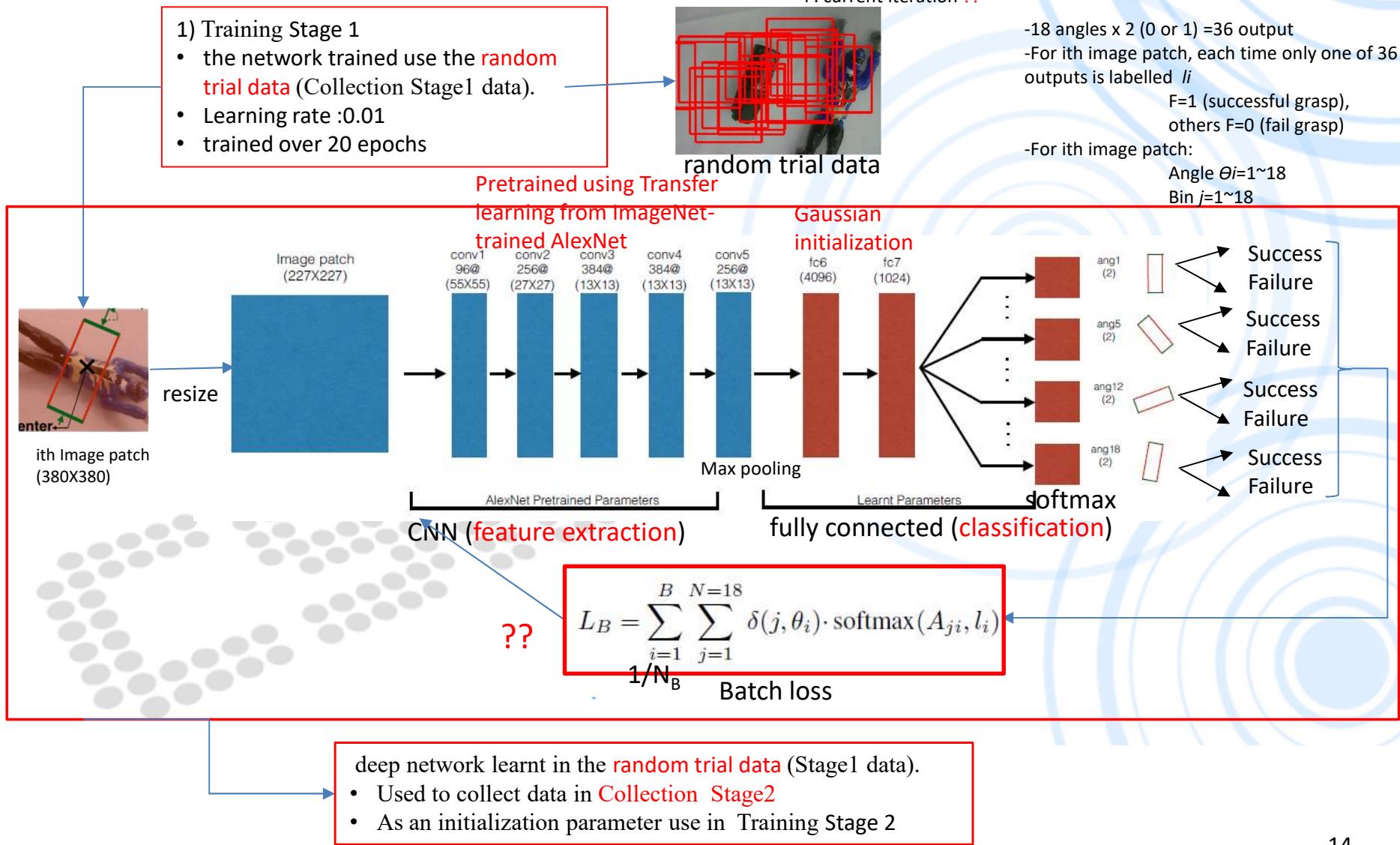


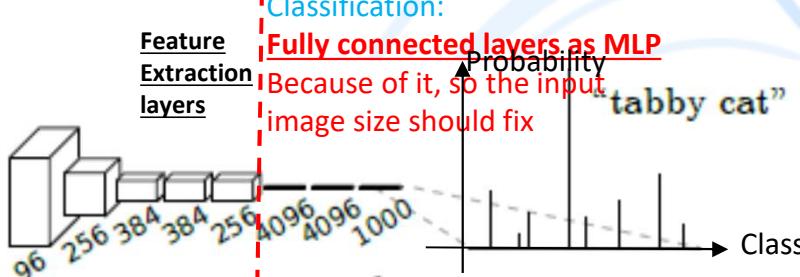
Fig. 5. Our CNN architecture is similar to AlexNet [6]. We initialize our convolutional layers from ImageNet-trained Alexnet.

## 3.2 Modified Alex: 5 + 3 Fully Convolutional Layers/Networks (FCN) (2/2)

Typical recognition nets: [LeNet](#) [23], [AlexNet](#) [22].

1. fixed-sized inputs
2. non-spatial outputs

[CNN: AlexNet](#)



[FCN: Modified AlexNet](#)

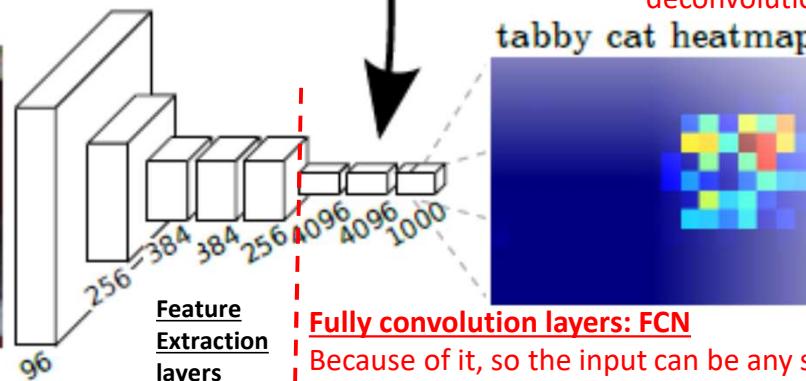


Figure 2 from [30-Cvpr2015, tPami2017]

Figure 2. Transforming fully connected layers into convolution layers **enables a classification net to output a heatmap**. Adding layers and a spatial loss (as in Figure 1) produces an efficient machine for end-to-end dense learning.

国立成功大學

J: FCN -

Because during the training process,

- the input is any size of image,
- the output has 1000 classes, each of which (each class) contains pre-labeling classification image set.

- > Learnable upsampling (or deconvolution) segmentation heatmaps (which is the Gaussian distribution - probability) through
- > transpose convolution (initialized with bilinear interpolation) > Fixed-size feature map

> Each pre-classified object / target region at image is also labeled its RoI.

> That is, the output target is labeled 1) classification result with 2) target RoI at image.

> FCN does NOT reduce total parameter number compared with Fully connectional network (MLP)??

J: AlexNet -  
Because during the training process,  
- the input is fixed size of image,  
- the output has 1000 classes, each of which (each class) contains pre-labeling classification image set

### 3.3 VGG16 or VGG19 2015 – Very Deep CNN (13+3 or 16+3 Layers (as 5 + 3 Layers)) (1/2)

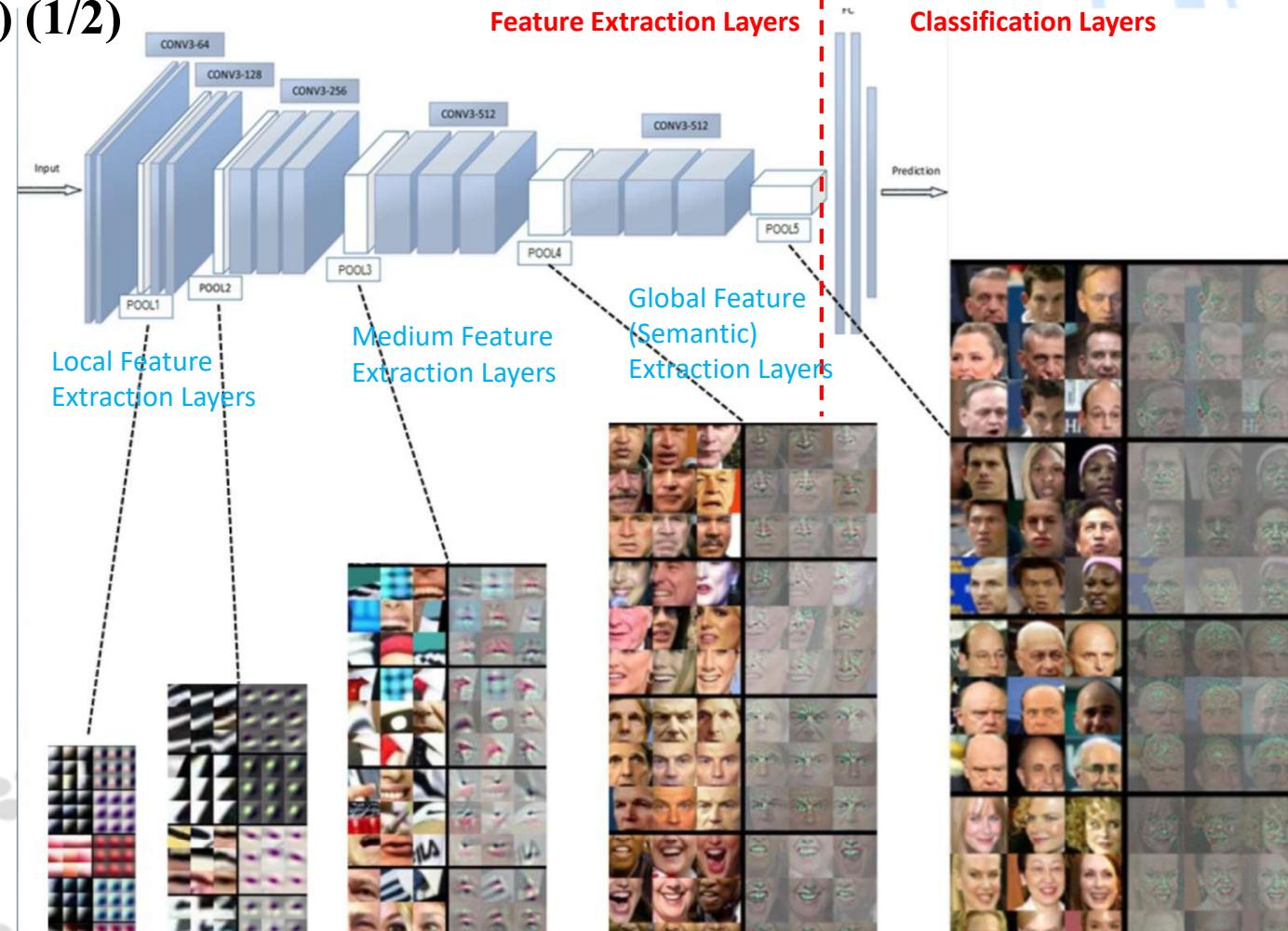


Fig. 2. The hierarchical architecture that stitches together pixels into invariant face representation. Deep model consists of multiple layers of simulated neurons that convolute and pool input, during which the receptive-field size of simulated neurons are continually enlarged to integrate the low-level primary elements into multifarious facial attributes, finally feeding the data forward to one or more fully connected layer at the top of the network. The output is a compressed feature vector that represent the face. Such deep representation is widely considered the state-of-the-art technique for face recognition.

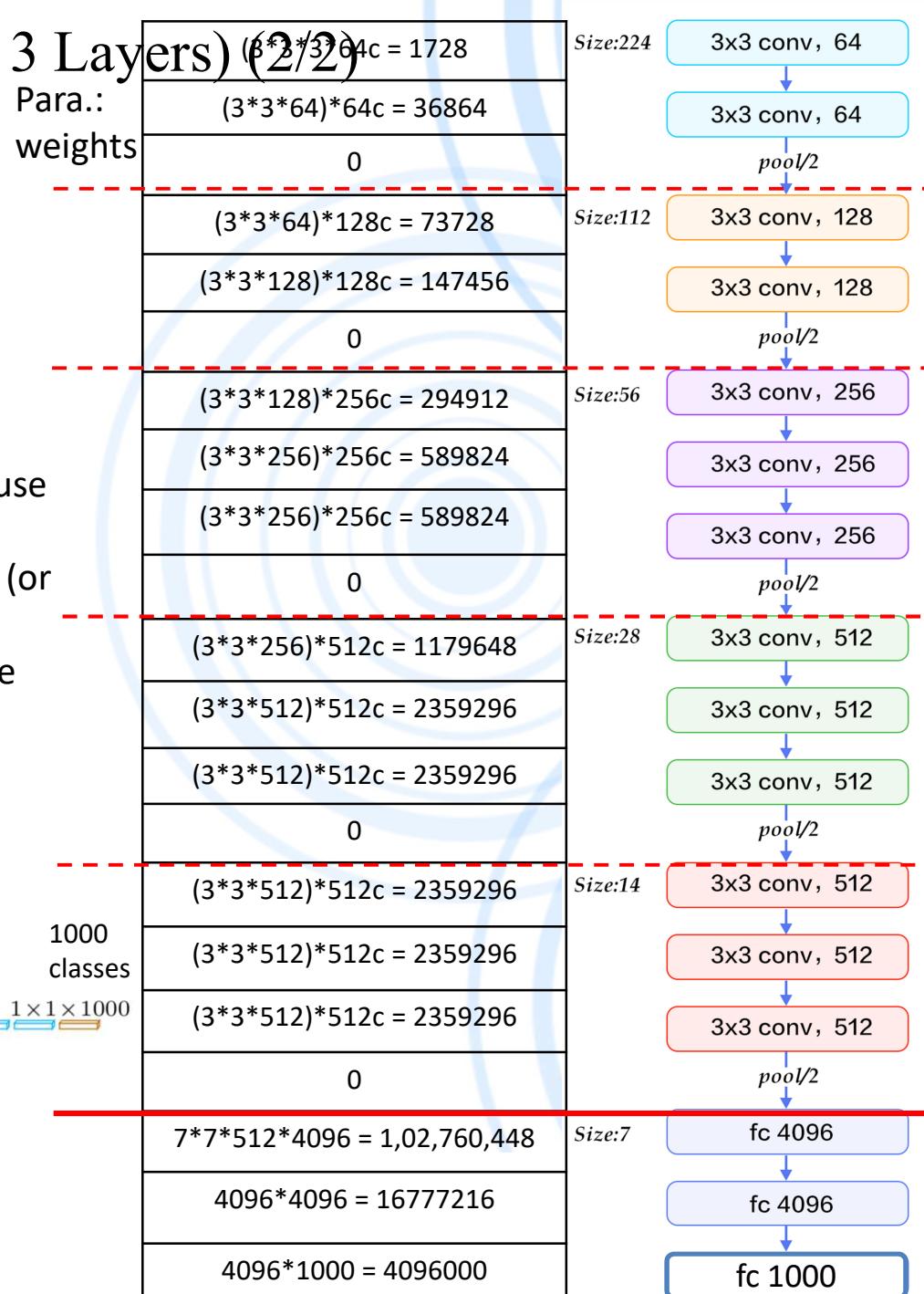
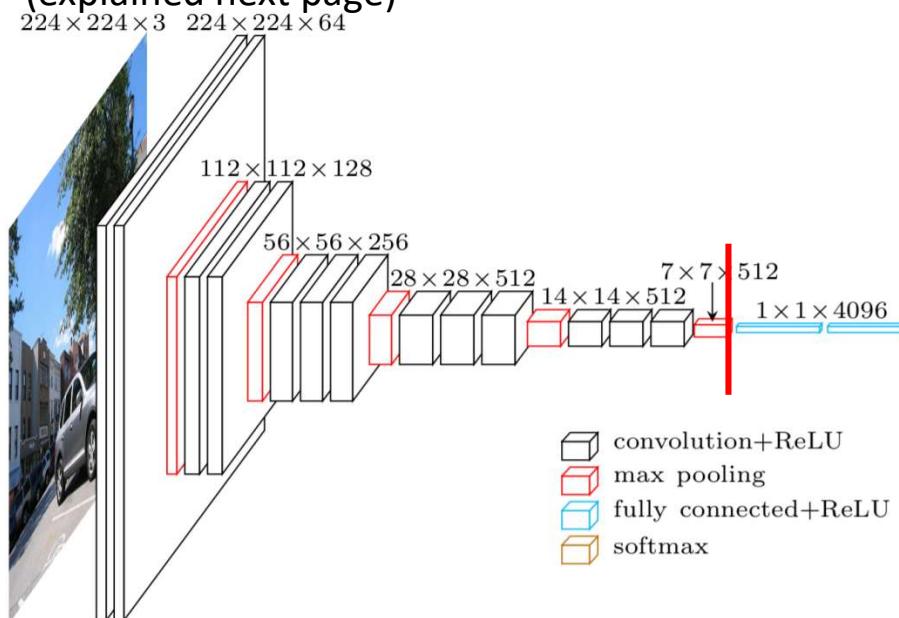
### 3.3 VGG16: 13 + 3 Layers (as 5 + 3 Layers) (2/2)

- Spatial filter size:  $3 \times 3$
- Stride: 1
- Padding: 1
- Non-overlap pooling
- Total parameters:  

$$\text{params} = \text{weights} + \text{biases} \text{ (in FC)}$$
  

$$= 138344128 + 4096 * 3 = 138,356,416$$

- Instead of using large spatial filter size ( $7 \times 7$ ), use small one  $3 \times 3$
- reduce parameters, and increase the net depth (or # of ReLU)
- make the decision boundary more discriminative (explained next page)



# 3.4 ResNet: Deep Residual Learning – 50, 101...

Network Architectures: /2: Stride 2

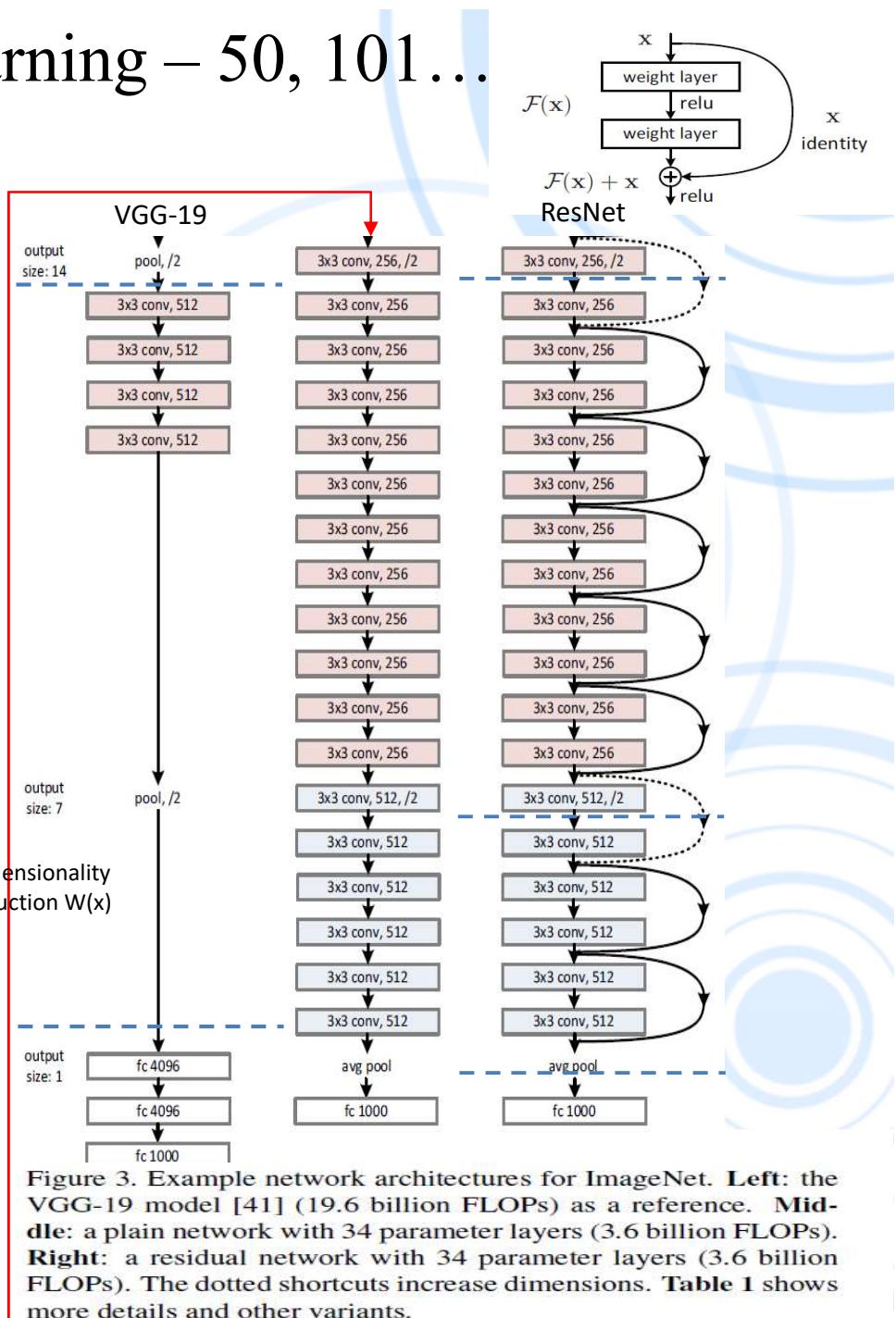
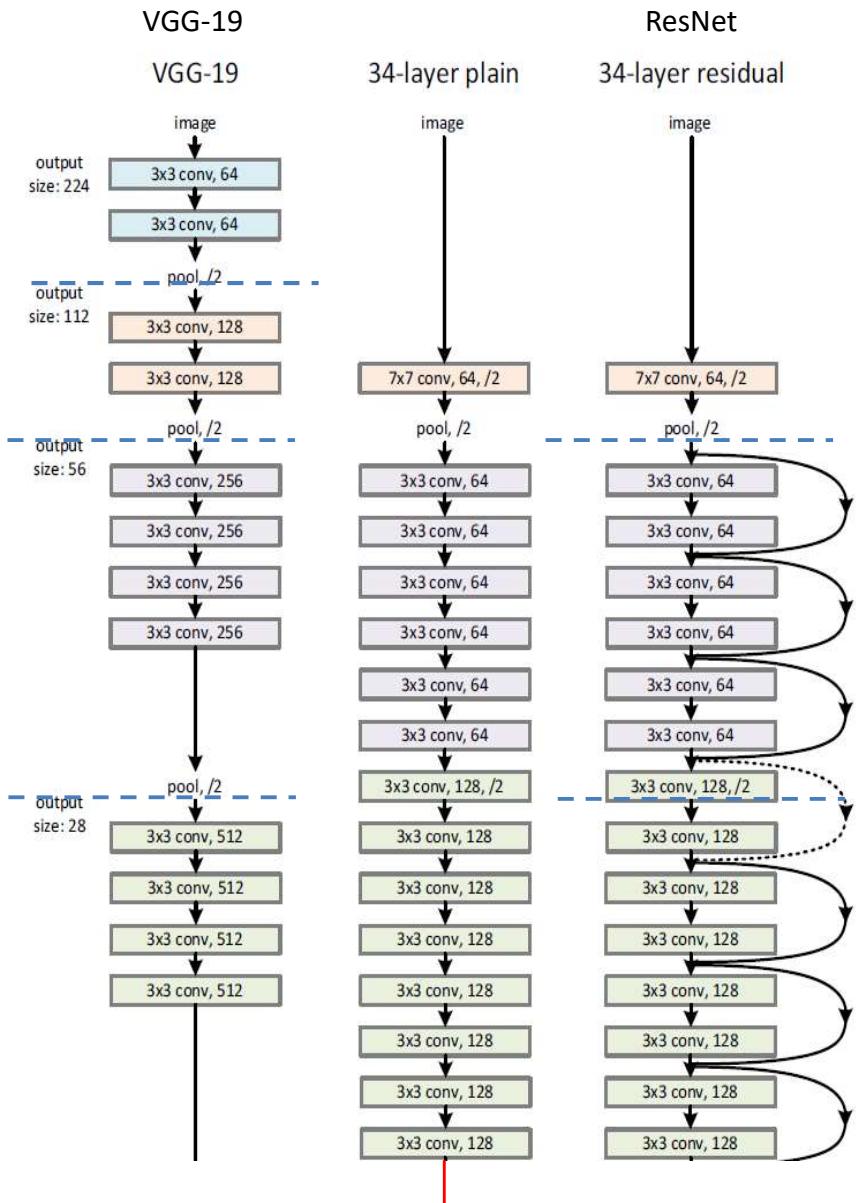


Figure 3. Example network architectures for ImageNet. **Left:** the VGG-19 model [41] (19.6 billion FLOPs) as a reference. **Middle:** a plain network with 34 parameter layers (3.6 billion FLOPs). **Right:** a residual network with 34 parameter layers (3.6 billion FLOPs). The dotted shortcuts increase dimensions. **Table 1** shows more details and other variants.

### 3.5 Other Very Deep CNNs

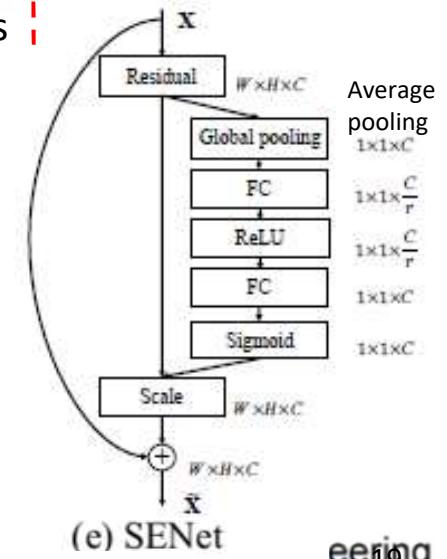
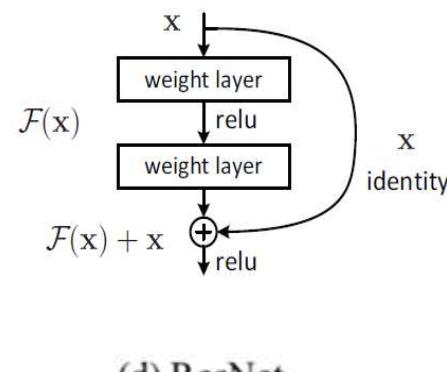
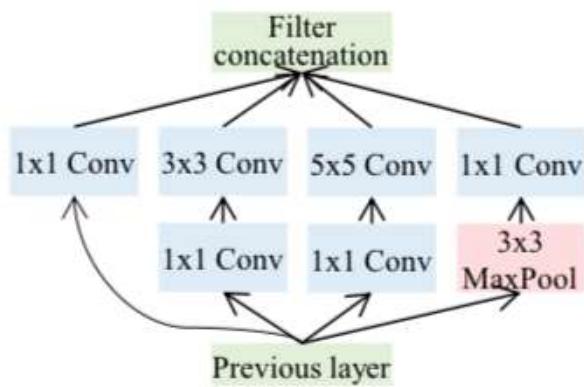
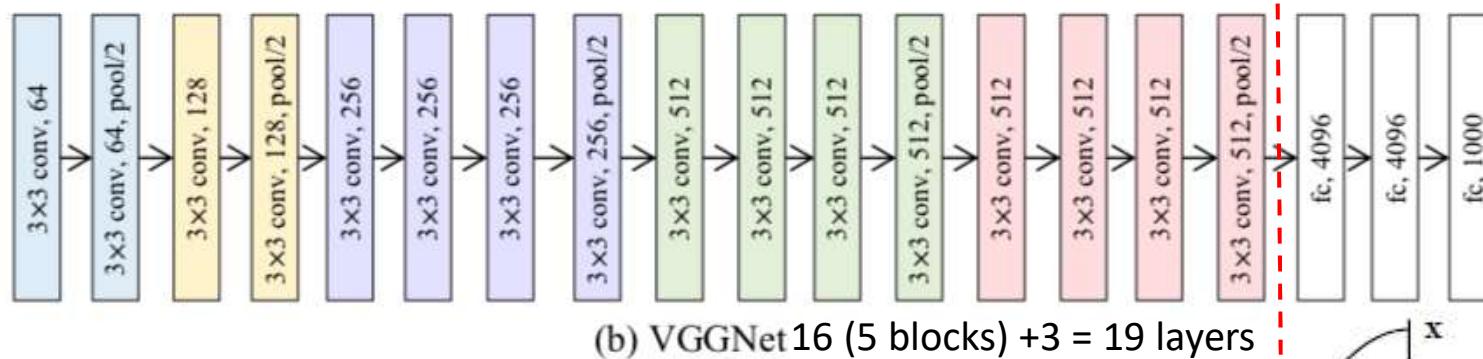
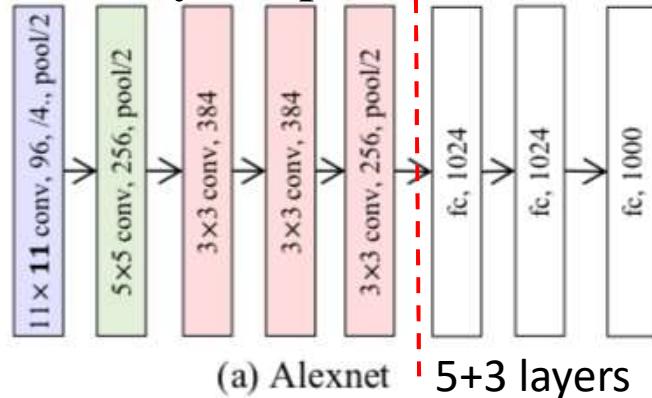
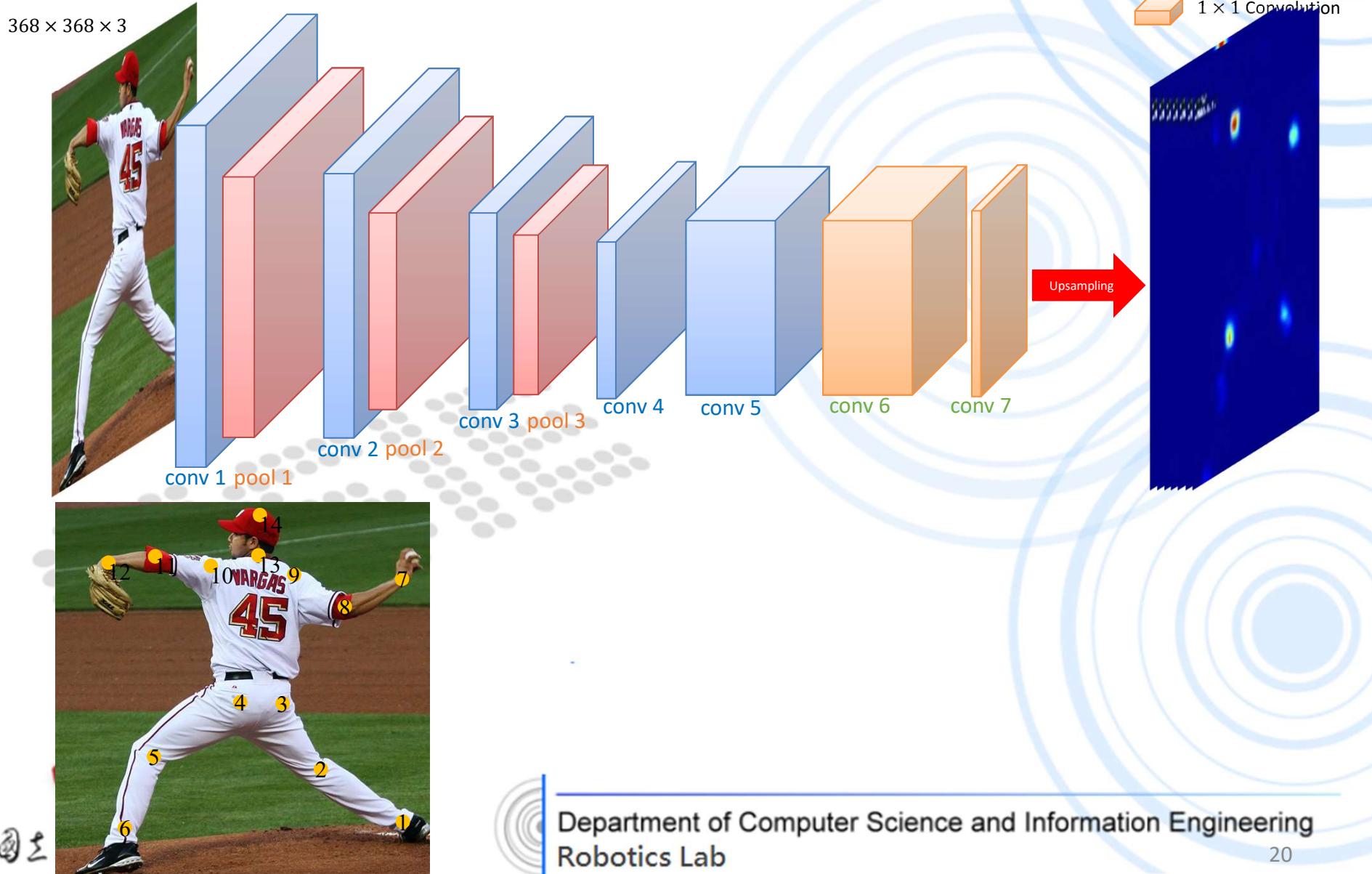


Fig. 9. The architecture of Alexnet, VGGNet, GoogleNet, ResNet, SENet.

### 3.6.1 Application 1 - Body Joint Point Estimation Using VGG-16 (1/2)

#### ❑ CNN-based VGG-16



# Joint Estimation



### 3.6.1 Application 1 - Body Joint Point Estimation Using Deep Learning

## Real-time Multi-Person 2D Pose Estimation Using Part Affinity Fields

Zhe Cao, Tomas Simon, Shih-En Wei, Yaser Sheikh

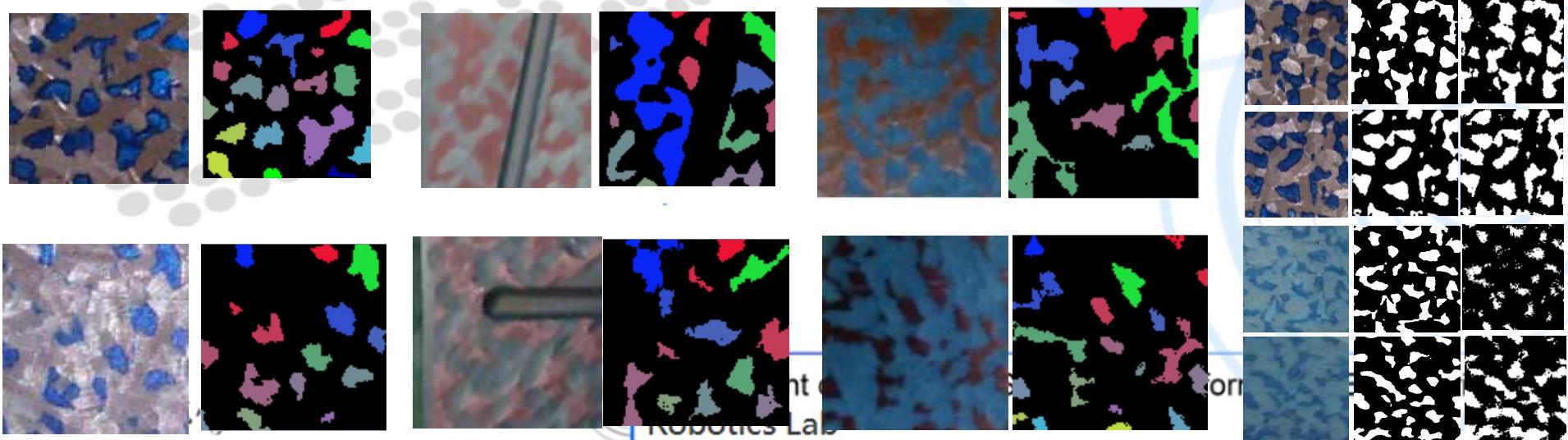
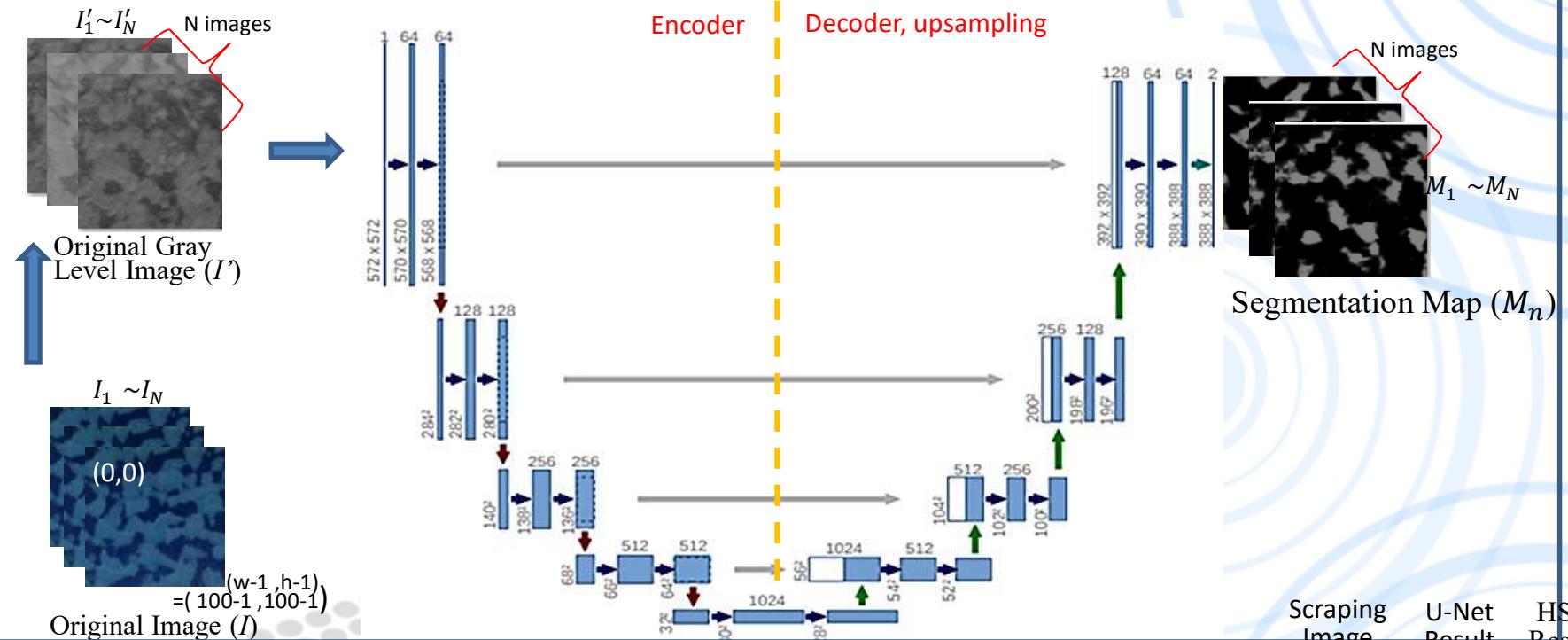
Carnegie Mellon University



Department of Computer Science and Information Engineering  
Robotics Lab

## 3.6.2 Basic: Application 2 - Scraping Segmentation Using U-Net

### Scraping Segmentation Using U-Net (Vs. AutoEncoder)



# 4.0 Deep Learning: 2. Multi-Task

Which one do I want to be?

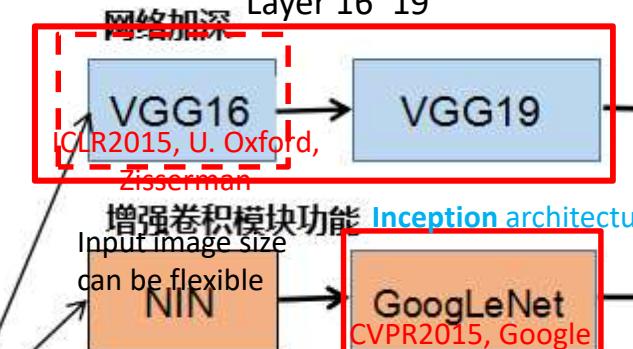
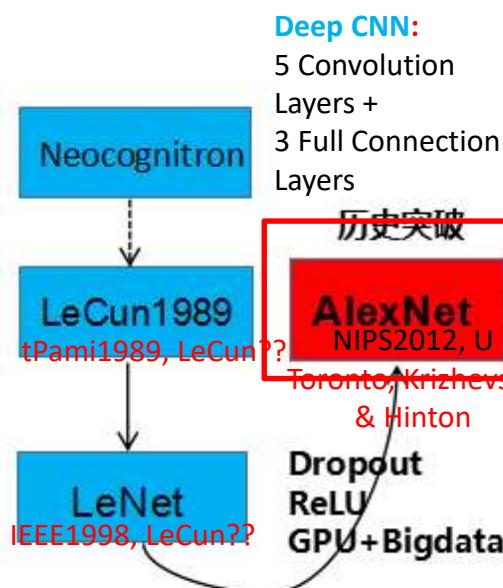
1. Basic CNN Creation
2. Modified CNN
3. CNN Application: Data collection, organization and analysis / benchmark

## 1. Basic CNN

Getting deeper:

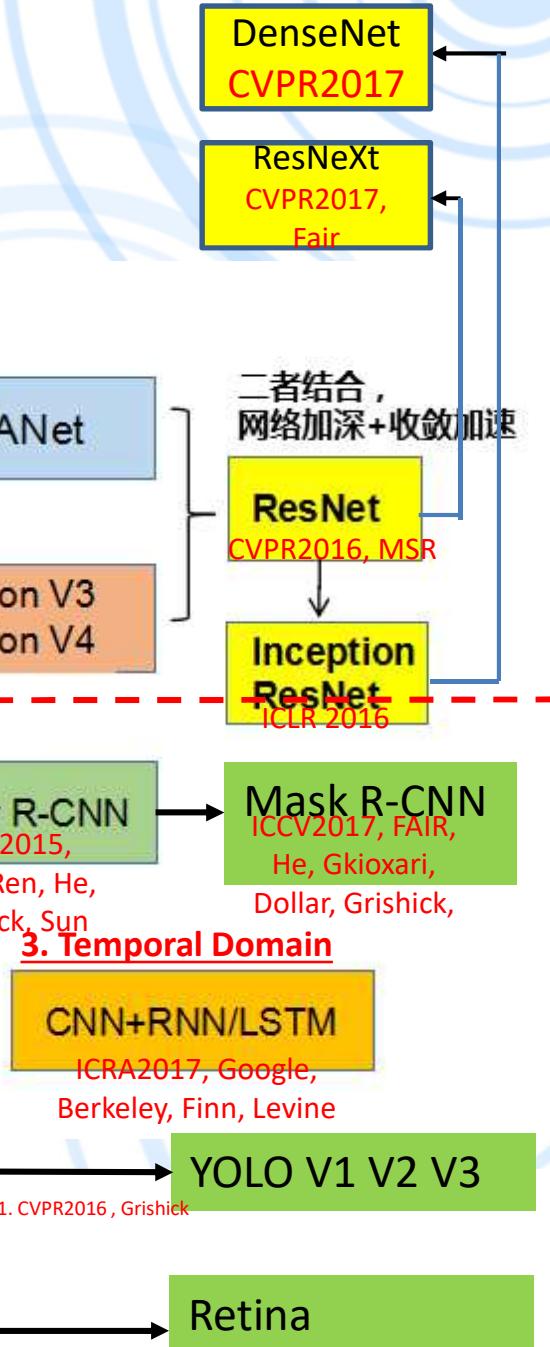
Very Deep CNN, ICLR2015

Layer 16~19



## 2. Multi-Task: Detection (ROI) + Classification

ECCV2014, MSR, He, Zhang, Ren, Sun  
SPP-Net  
R-CNN  
CVPR2014, tPami2015  
MSR, Girshick & Berkeley  
增加新的功能单元 (Add new functional units)



## 4. GAN, AutoEncoder:

Generative Adversarial Network Vs. PCA

## 5. Reinforcement Learning

Unsupervised

tPAMI: IEEE Transactions on Pattern Analysis and Machine Intelligence

CVPR: Conference on Computer Vision and Pattern Recognition

NIPS: Conference on Neural Information Processing Systems

ICLR: International Conference on Learning Representation

NIN: Network in Network

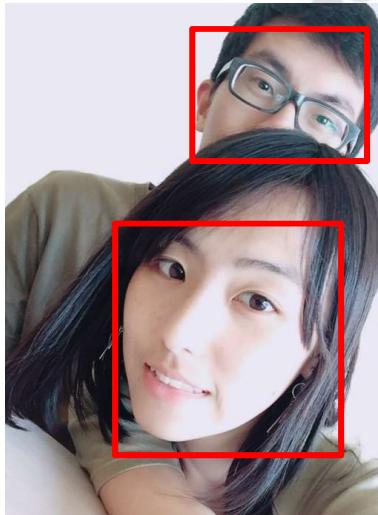
R-CNN: Region-based Convolutional Network method

SPP-Net: Spatial Pyramid Pooling networks

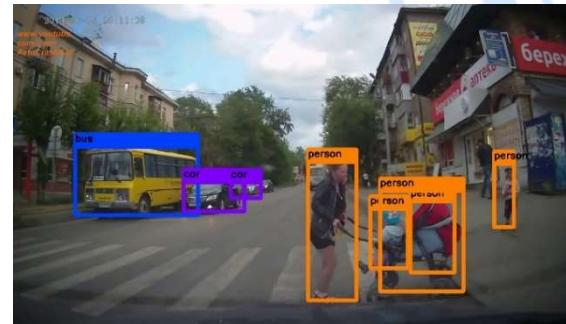
# 4.0 Multi-Task: Application



5. Sports: Detect and classify people and sports balls



4. Face recognition:  
includes both detection  
and recognition



1. ADAS: Detect and classify cars, people, scooters...

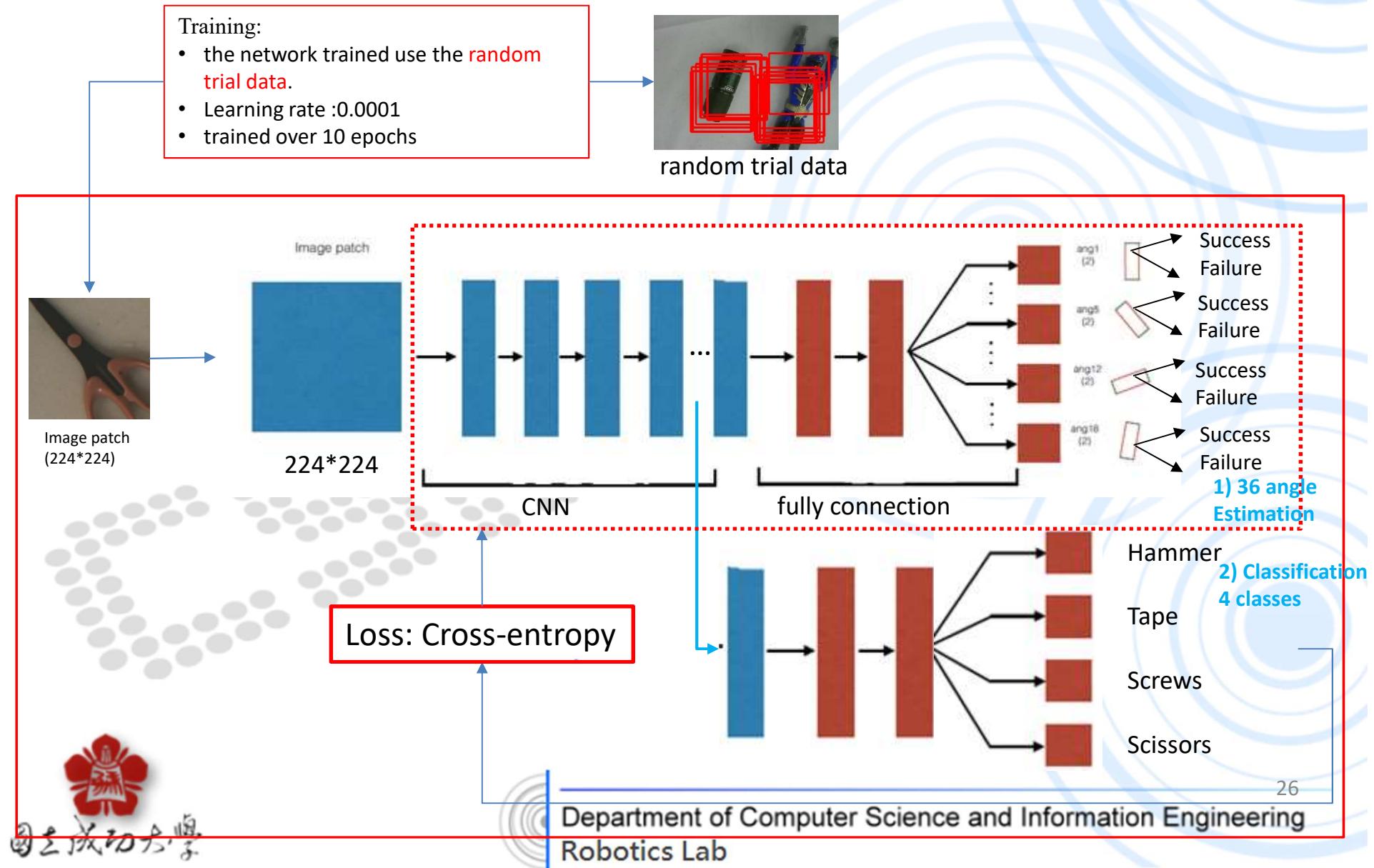


2. Robot Arm Pick and Place: Detect and classify objects (ex: screws, screwdrivers...)

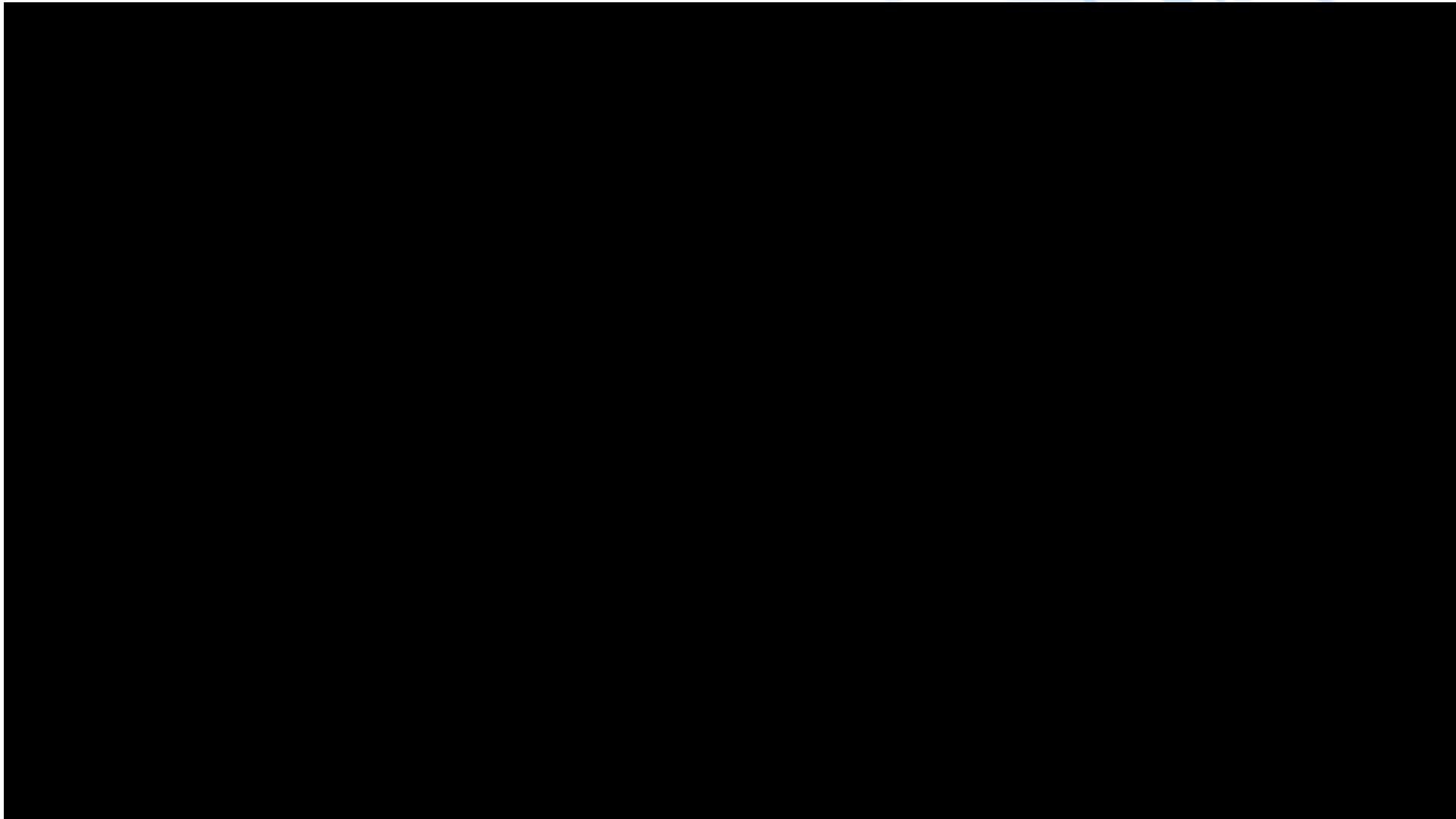


3. Surveillance:  
Detect and classify  
people, cars...

## 4.1 Multi-Task: Visual-Guided Robot Arm Using VGG-16 – Object Detection, Classification and Picking Angle Estimation

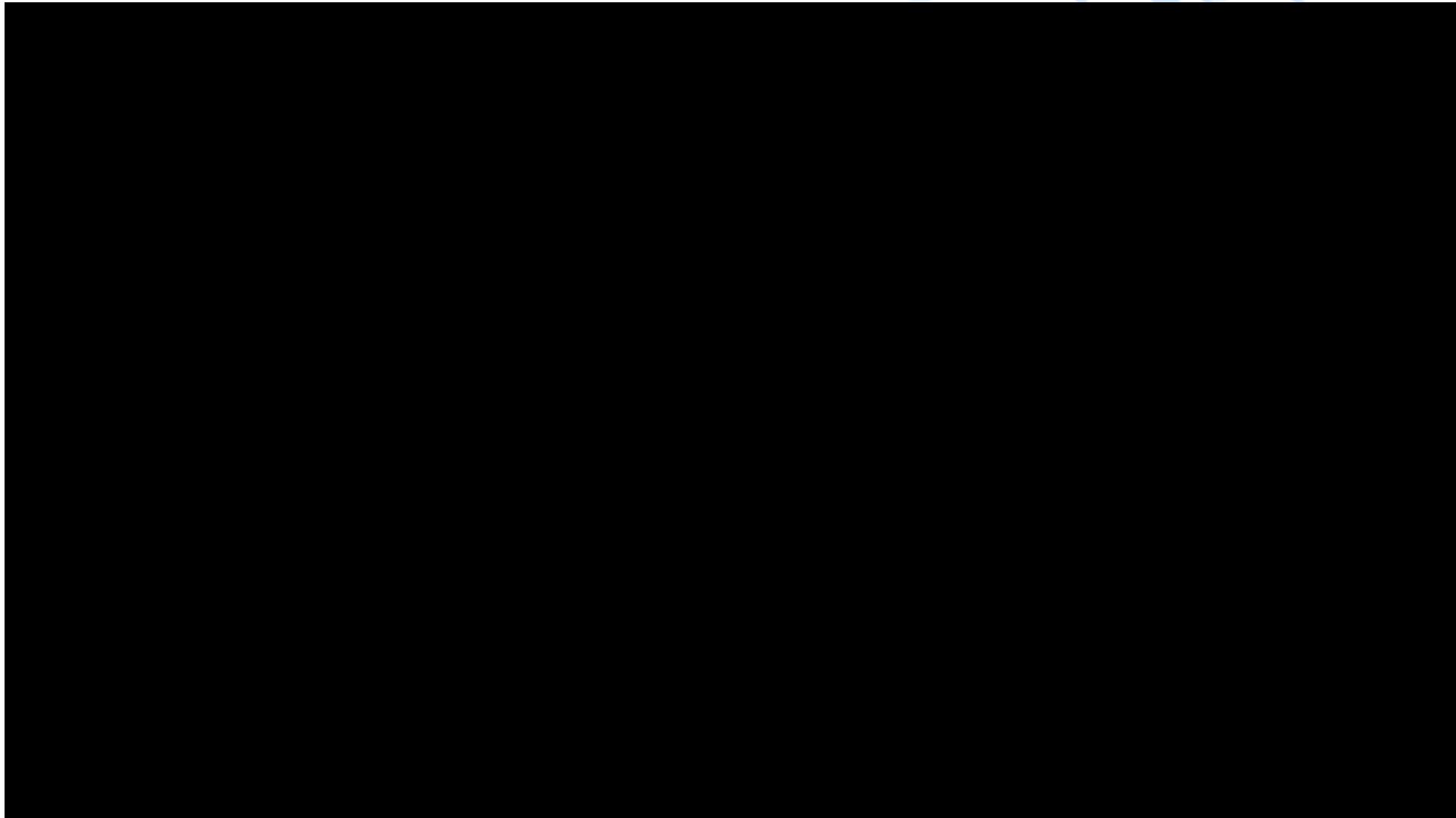


## 4.1 Multi-Task: Visual-Guided Robot Arm Using VGG-16 - Object Detection, Classification and Picking Angle Estimation: Training Process 01:31.61

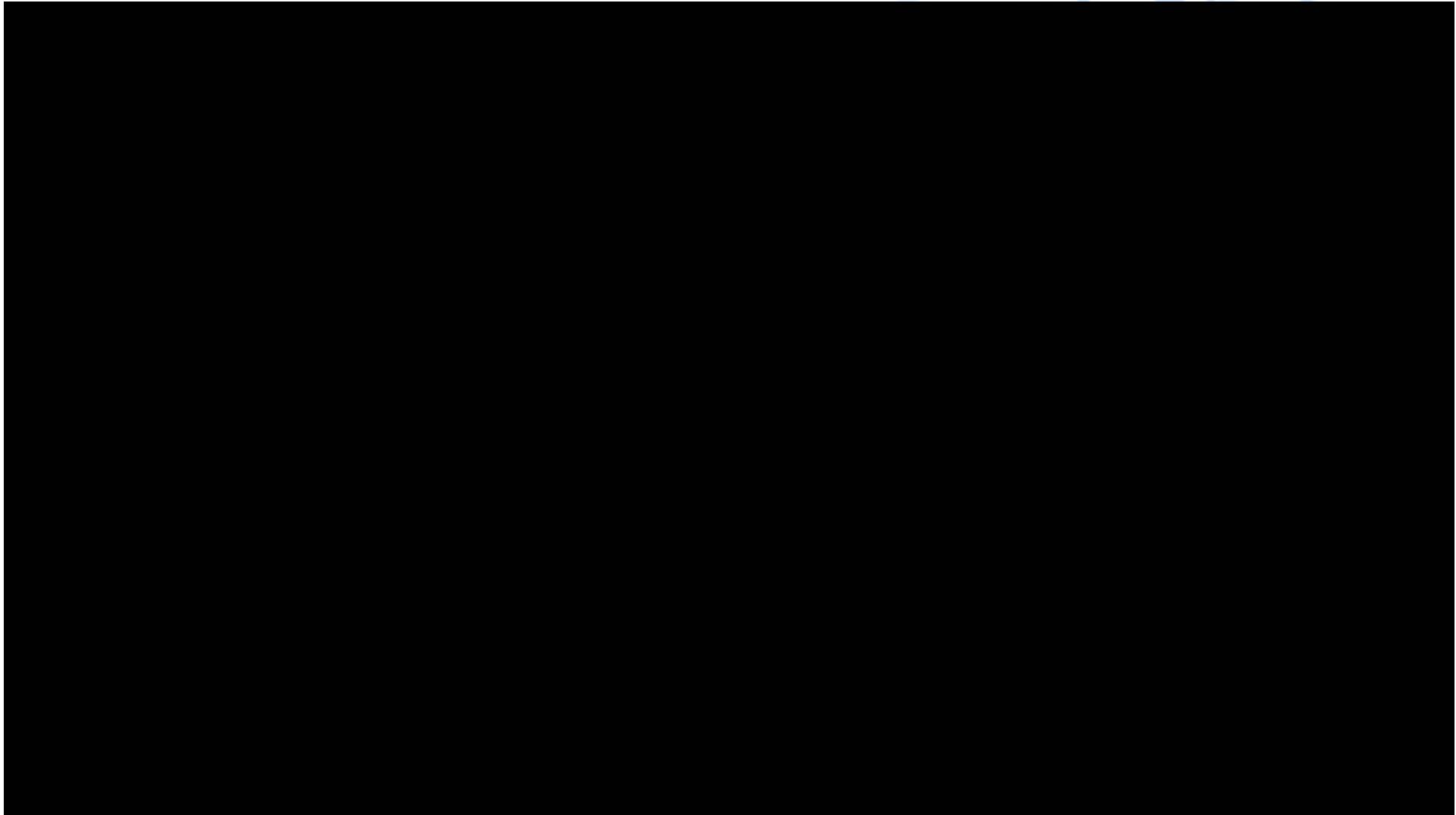


## 4.1 Multi-Task: Visual-Guided Robot Arm Using VGG-16 - Object Detection, Classification and Picking Angle Estimation: Test Process

02:14.35



## 4.1 Visual-Guided Robot Arm Using Machine Learning: Static Object



## 4.1 Multi-Task: Visual-Guided Robot Arm Using Deep Learning - Object Detection, **Tracking** and Picking Angle Estimation 02:14.35



## 4.2 Multi-Task: Detection, Segmentation, and Classification

Which order is better?

- 1) BBox 2) Mask 3) Classification
- 1) BBox 2) Classification 3) Mask

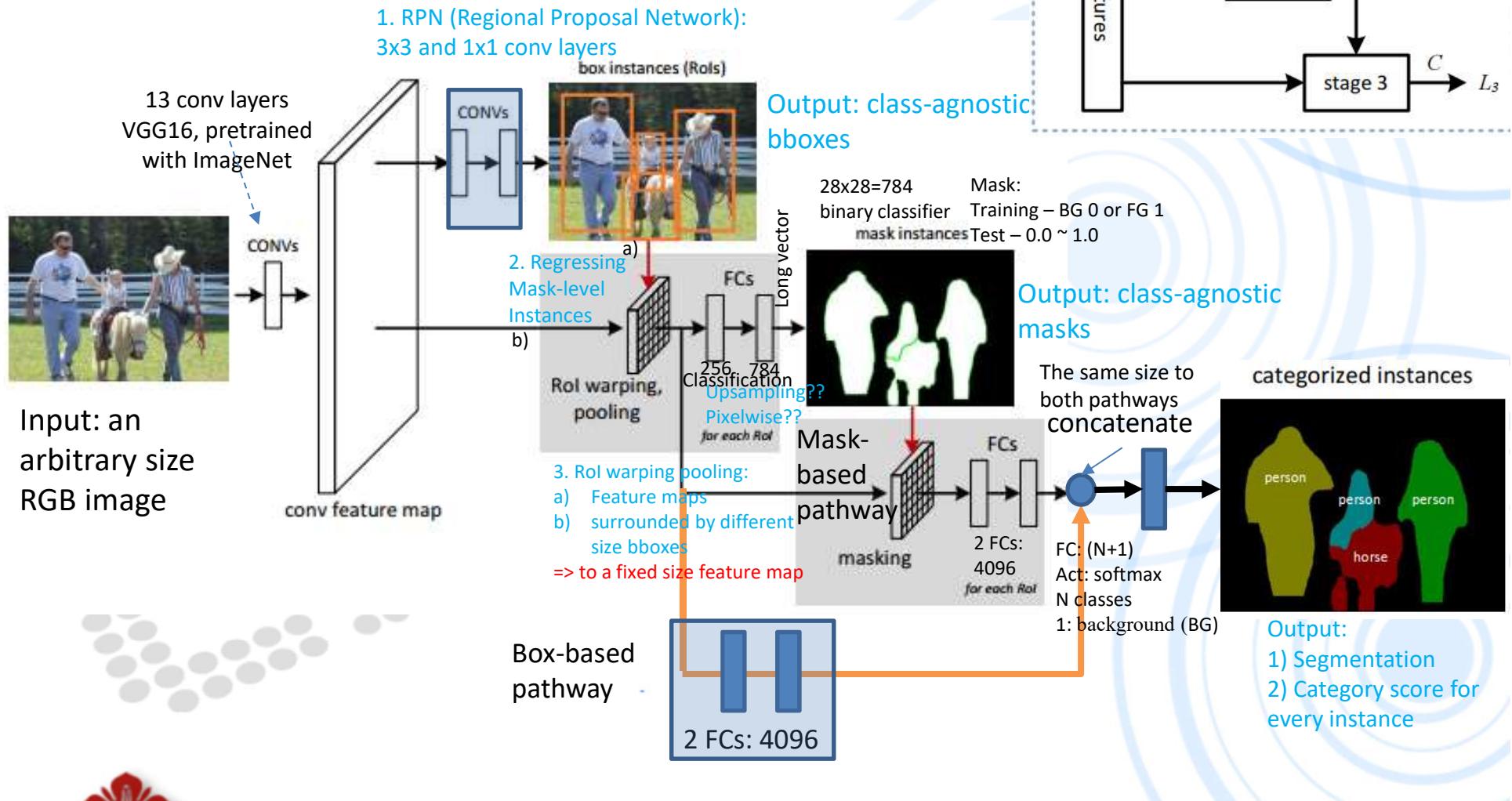


Figure 2. Multi-task Network Cascades for instance-aware semantic segmentation. At the top right corner is a simplified illustration.

## 4.3 R-CNN Background Knowledge and Comparison ??

### 1.1) R-CNN (Region-CNN):

- Use Selective Search + CNN (AlexNet)
- RoI: Bounding Box Segmentation

### 1.2) SPPNet (Spatial Pyramid Pooling Network):

### 2) Fast R-CNN:

- Use Selective Search + FCN (Modified AlexNet or VGG16)
- RoI: Bounding Box Segmentation

### 3) Faster R-CNN:

- Use RPN + FCN (Modified ??)
- RoI: Bounding Box Segmentation

### 4) Mask R-CNN:

- Use RPN + FCN (Modified ResNet)
- RoI: Pixel-wise Segmentation

For later layers:

(1) CNN - AlexNet: Fully Connectional Network (MLP) for classification

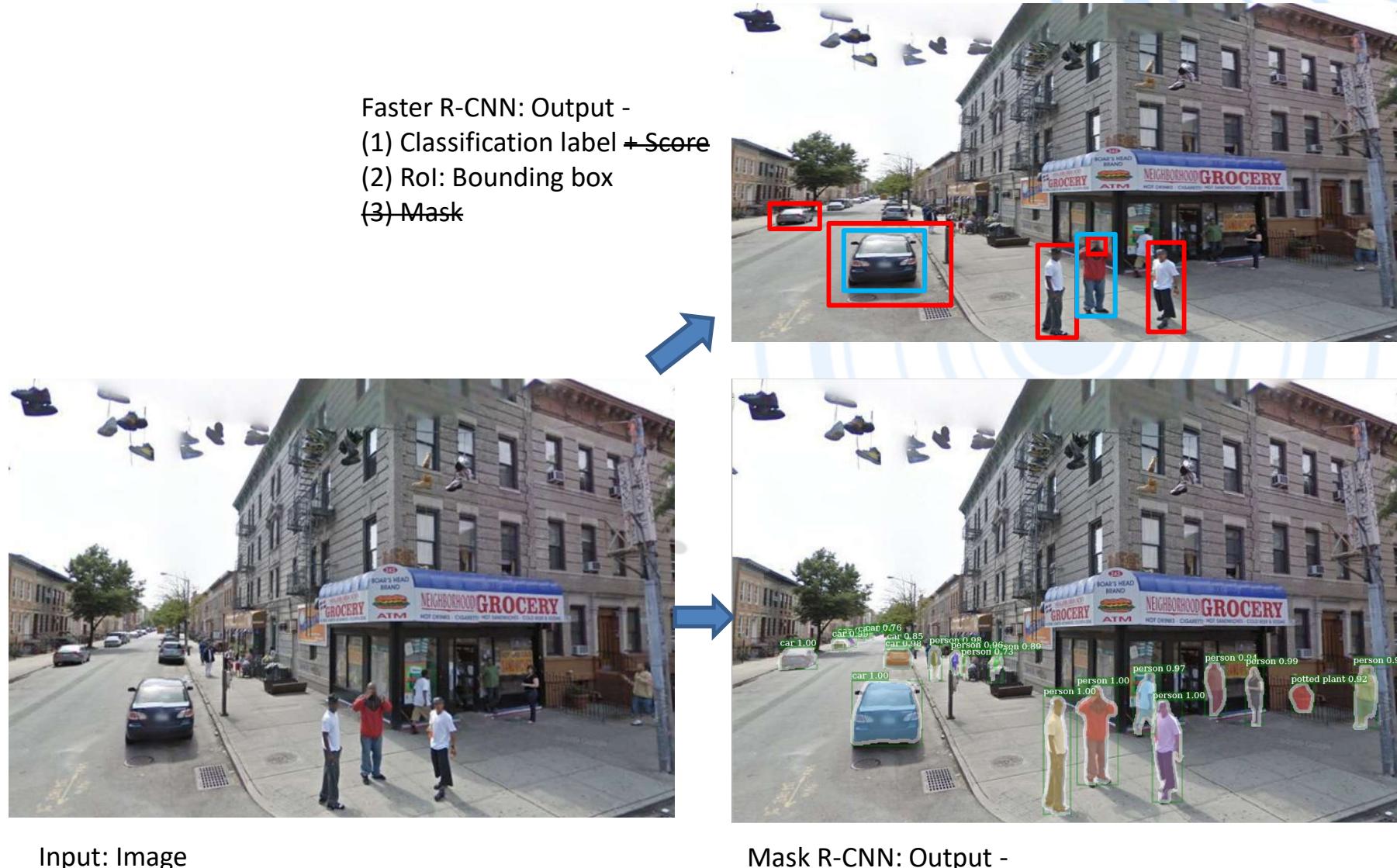
(2) FCN – Modified AlexNet or VGG16: Fully Convolutional Network for classification + RoI

**RPN: Region Proposal Network**

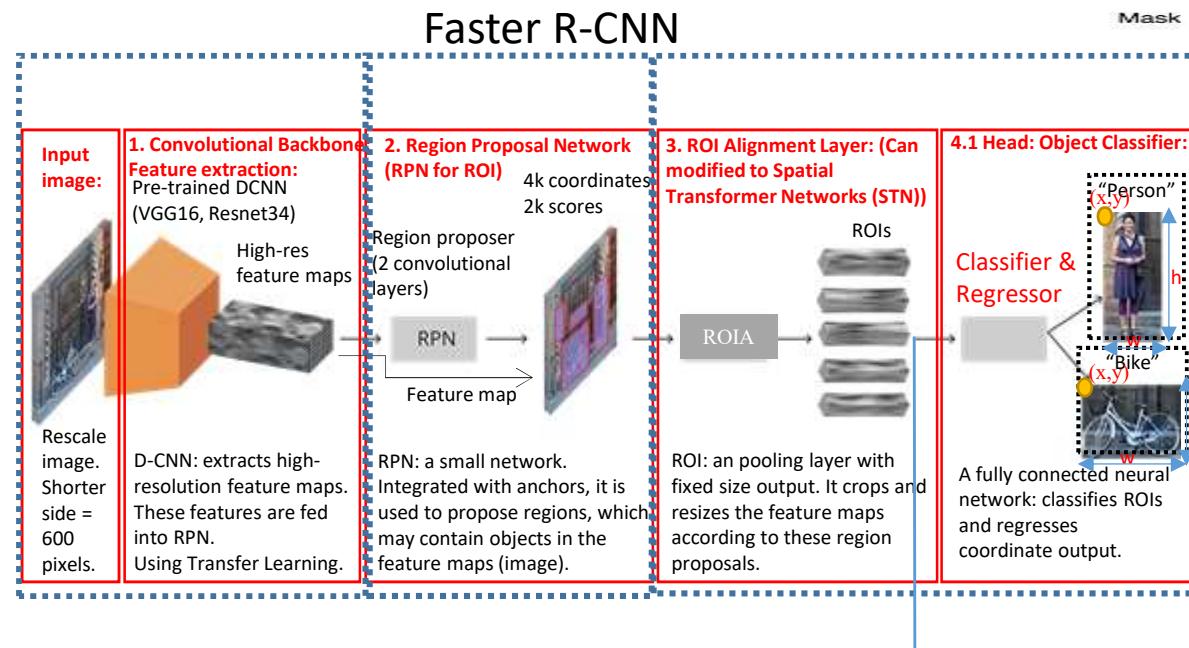
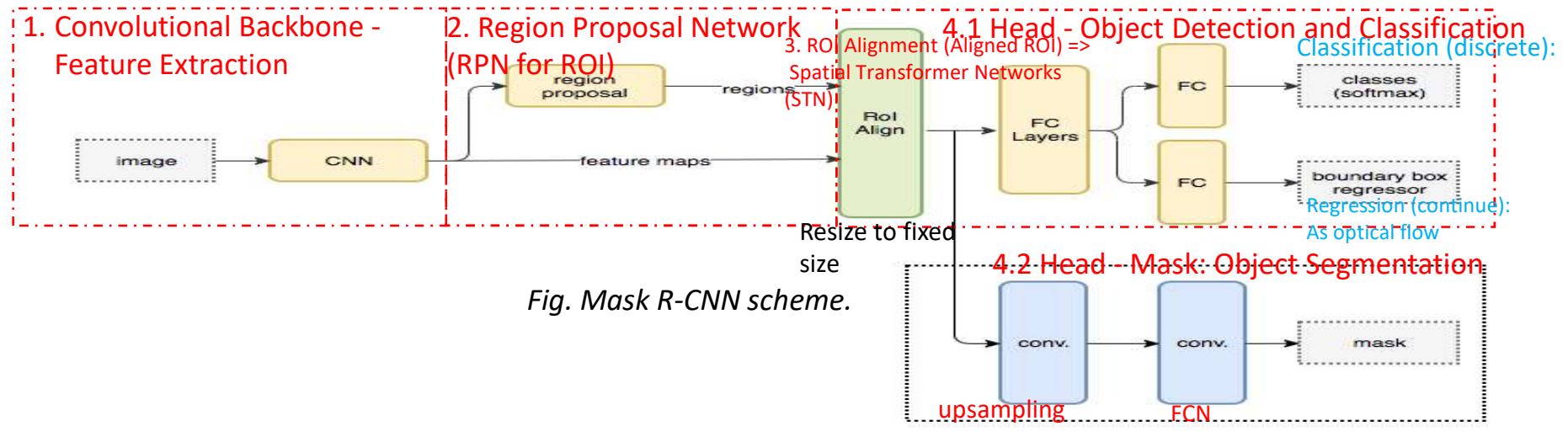
-Input: An image (of any size)

-Output: A set of rectangular object proposals (RoI), each (RoI) with an objectness score

## 4.3 Multi-Task: Faster R-CNN: Goal – Detection + Bbox Segmentation + Classification Mask R-CNN: Goal – Detection + Pixel-Wise Segmentation + Classification



## 4.3 Multi-Task: Faster R-CNN: Goal – Detection + Bbox Segmentation + Classification Mask R-CNN: Goal – Detection + Pixel-Wise Segmentation + Classification



## 4.3 One- Vs. Two-Stages: SSD, Yolo, RetinaNet

-RPN: Region Proposal Network

**One-stage**

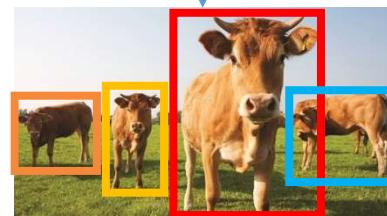
-SSD: Single Shot MultiBox Detector    **Two-stage**

input image



Dense samples

CNN



In general,

input image

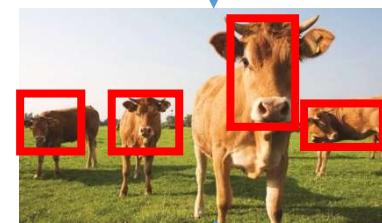


(or Selective  
Search, etc...)

CNN

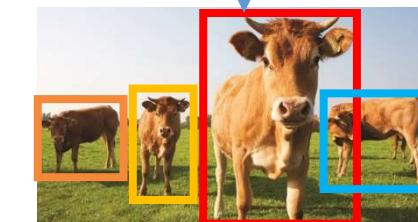
region  
proposals

RPN



Sparse samples after  
RPN selection, so this  
will reduce many  
background regions

classifier



	Speed	Accuracy	Examples
One-stage	faster	Lower ??	Yolo, SSD
Two-stage	slower	higher	Fast R-CNN, Faster R-CNN

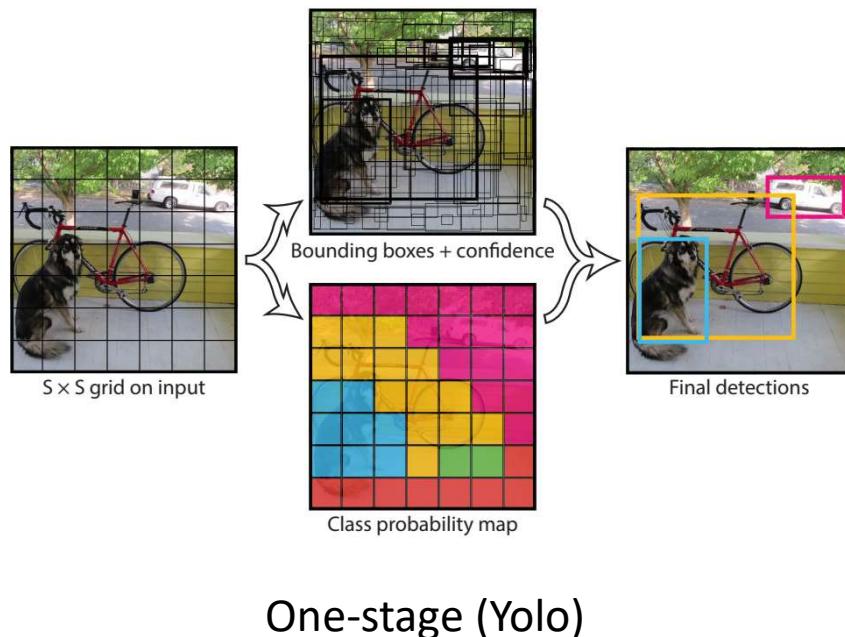
## 4.3 SSD, Yolo, RetinaNet: One-Stage Vs. Two-Stage

- Why one-stage is less accurate? **Class imbalance!**

### 1. Two-stage detector:

- 1) **Proposal (region) stage (ex: Selective Search, RPN):** Filtering out most **imbalance background (negative) samples** (only 1-2k are left) as AdaBoost
- 2) **Classification stage:** Fixed foreground-to-background ratio (1:3) or **online hard example mining (OHEM)** are used to **maintain a manageable balance between foreground and background**

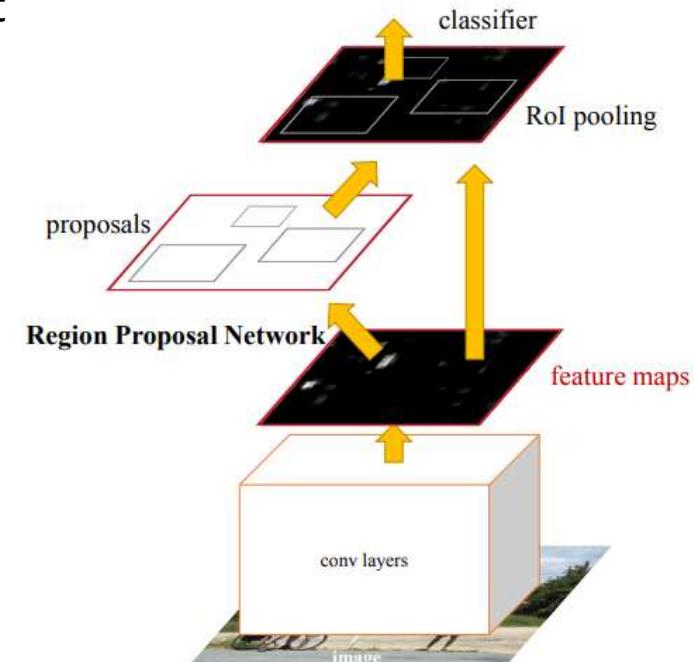
### 2. One-stage detector: Enumerating ~100k locations that **densely cover spatial positions, scales, and aspect ratios (J: Anchors??)** => inefficient



Drawbacks of class imbalance:

- 1) training is inefficient as most locations are easy negatives that contribute no useful learning signal
- 2) the easy negatives can overwhelm training and lead to degenerate models

J: Think about SVM or AdaBoost for weighting similar positive but negative samples.

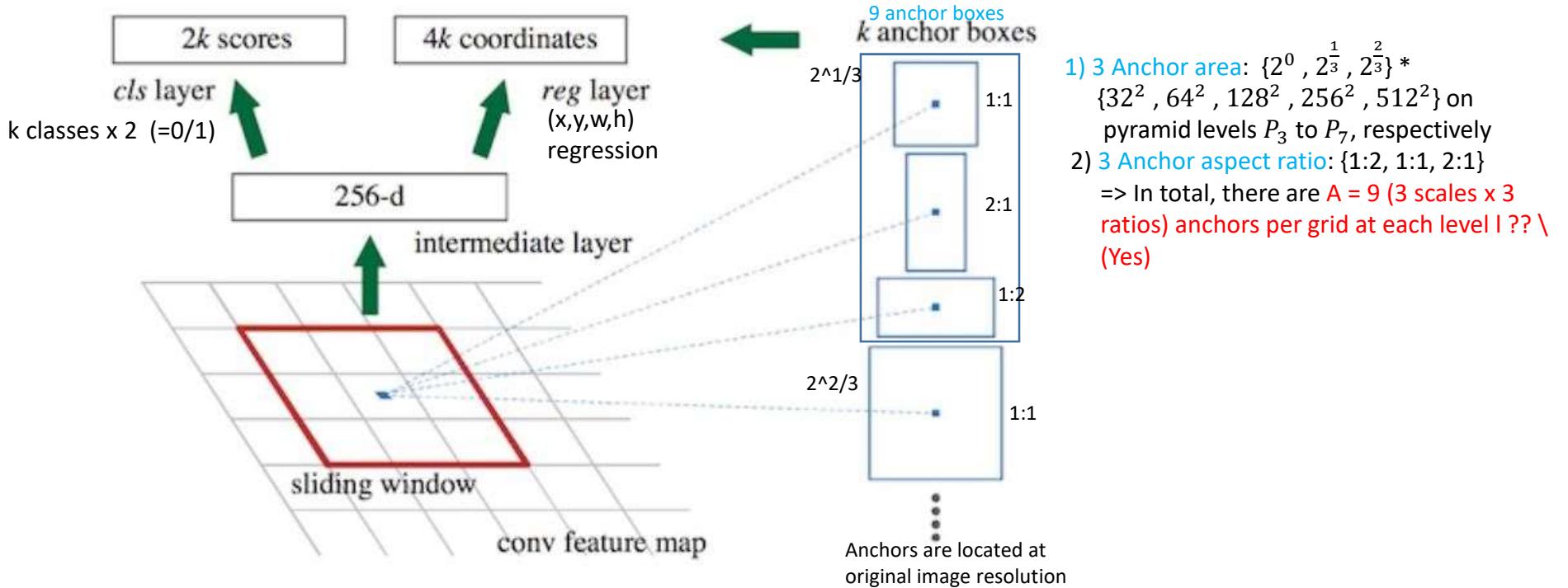


Two-stage (Faster R-CNN)

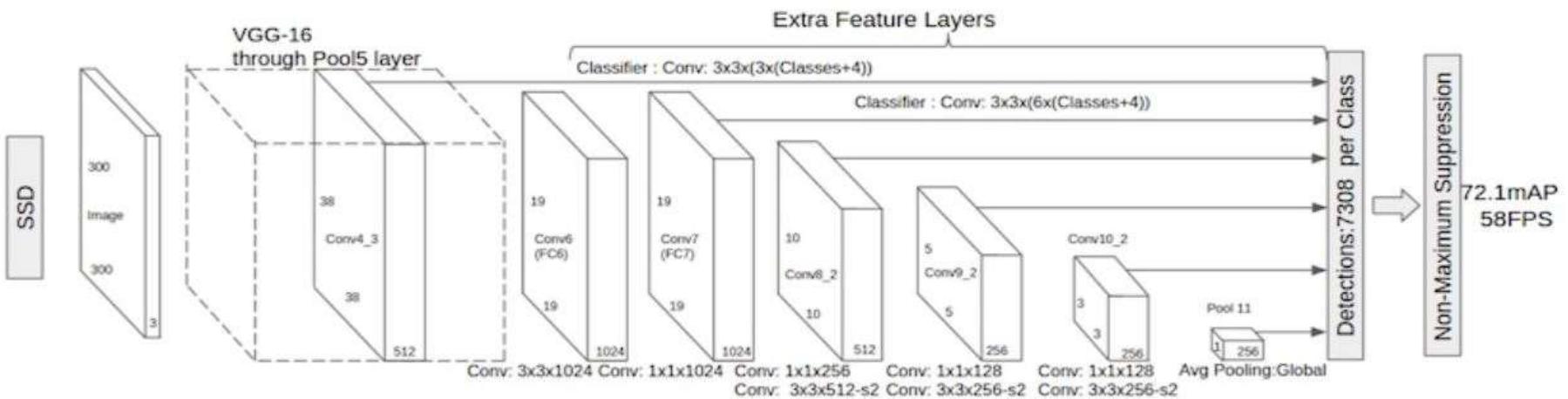
- Solution: **down-weights the loss that is assigned to well-classified examples**

## 4.3 SSD, Yolo, RetinaNet: RPN Vs. SSD

### 1) RPN: Region Proposal Network



### 2) SSD: Single Shot MultiBox Detector



## 4.3 RetinaNet

- RetinaNet is a single, unified network composed of:

1. Backbone network: Feature extraction
2. Two task-specific subnetworks BBox and Classification

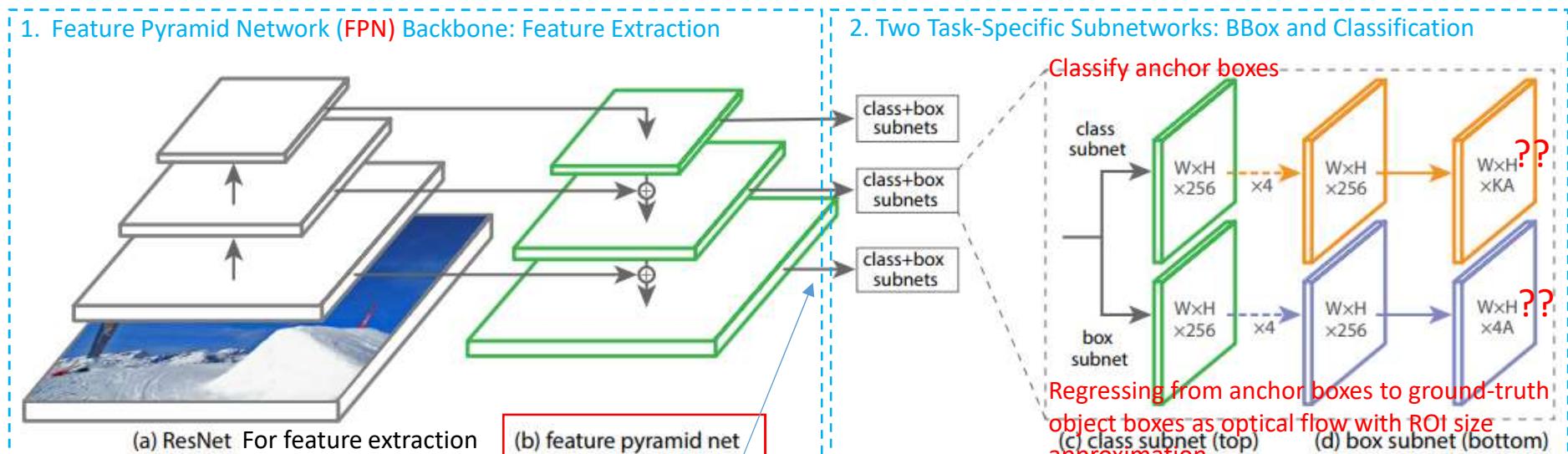


Figure 3. The one-stage **RetinaNet** network architecture uses a Feature Pyramid Network (FPN) [20] backbone on top of a feedforward ResNet architecture [16] (a) to generate a rich, multi-scale convolutional feature pyramid (b). To this backbone RetinaNet attaches two subnetworks, one for classifying anchor boxes (c) and one for regressing from anchor boxes to ground-truth object boxes (d). The network design is intentionally simple, which enables this work to focus on a novel focal loss function that eliminates the accuracy gap between our one-stage detector and state-of-the-art two-stage detectors like Faster R-CNN with FPN [20] while running at faster speeds.

J: 1. This way cannot keep original detail and clear feature because features from upsampling will be blur

### 4.3 Multi-Task: Application – Tool Defect Detection

➤ 與中正/台大蔡孟勳教授及  
東台精機合作



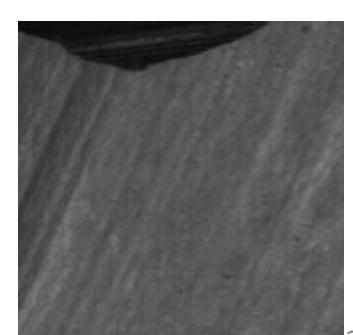
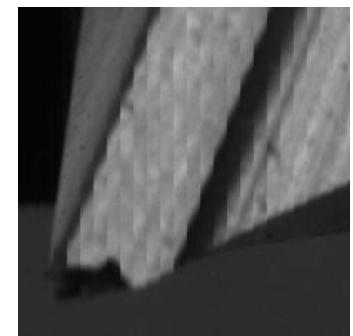
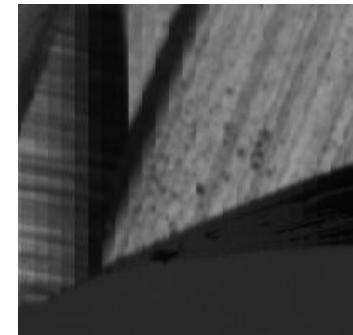
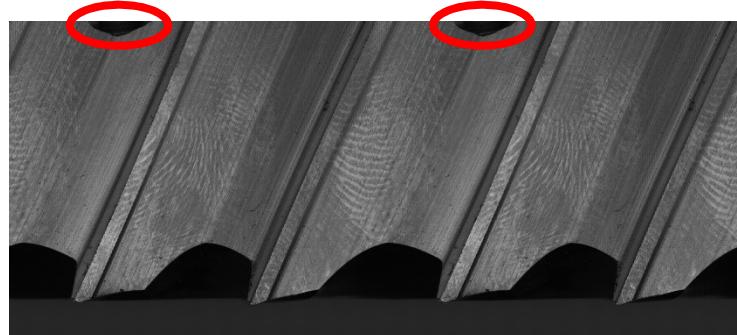
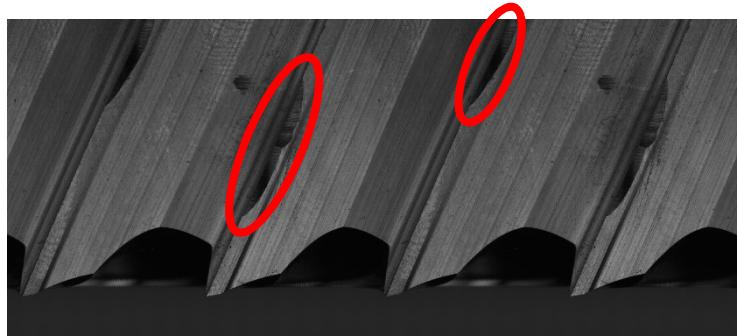
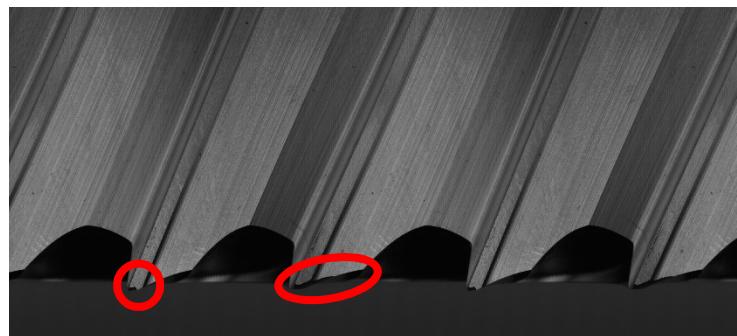
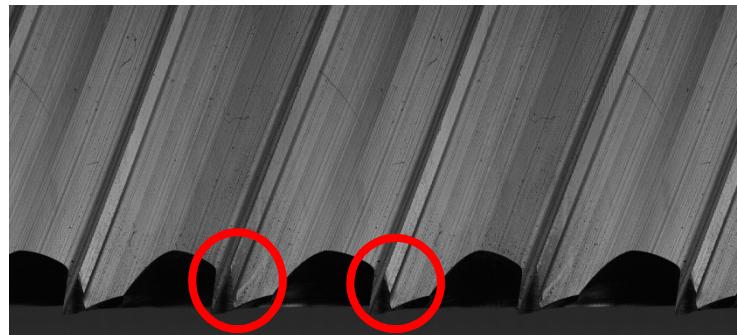
1) Tip

Defect

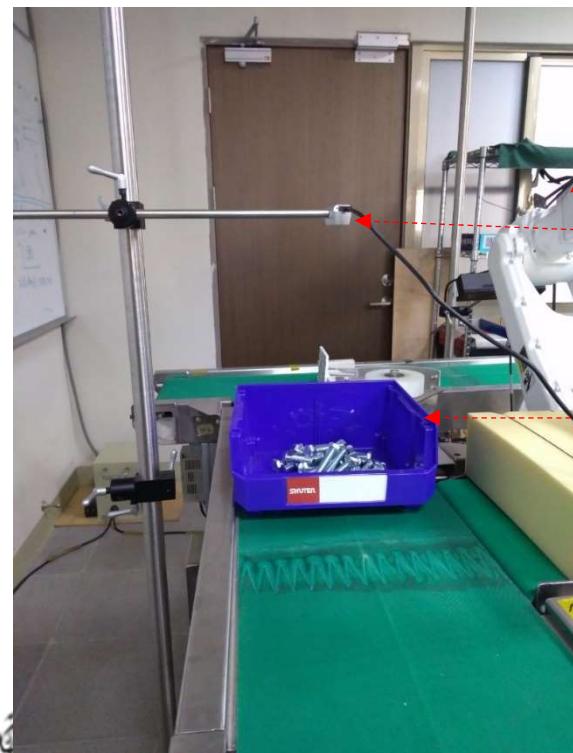
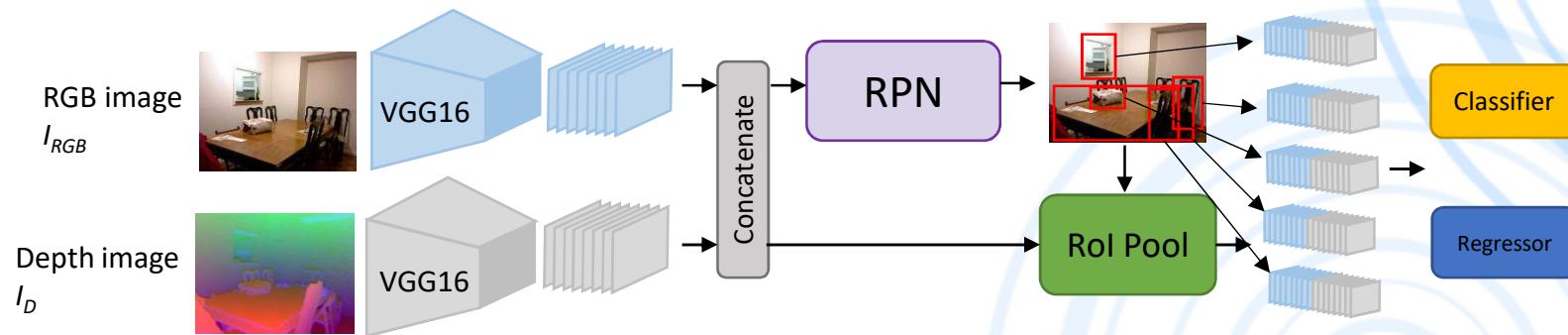
2) Flank

3) Plane

- Image Resolution: 2054 pixels \* 4581 pixels
- Patch size: 224 pixels \* 224 pixels



## 4.4 Multi-Input: Segmentation Using VGG-16 in RGB-D (1/3)



機械手臂  
RGB-D攝影機  
待夾堆疊物體



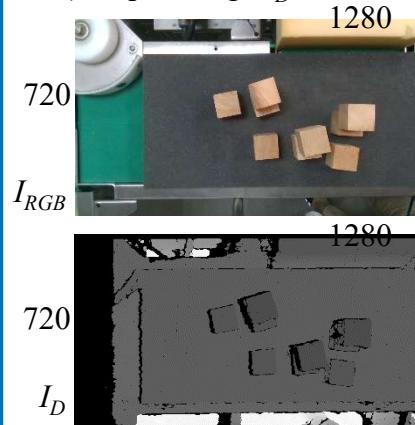
## 4.4 Multi-Input: Segmentation Using VGG-16 in RGB-D (2/3)

1. Objective: Detect and picking piled object with RGB-D images.

2. Global Framework:

### 0. Input RGB-D Image Pair

- 1) RGB image  $I_{RGB}$
- 2) Depth image  $I_D$



### 1. Detect Highest Object $B(x1, y1, x2, y2, z)$ using Faster RCNN in RGB-D Space

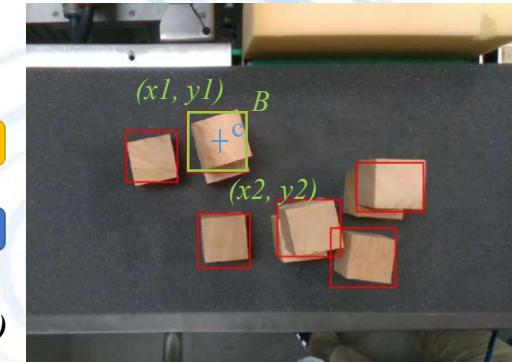
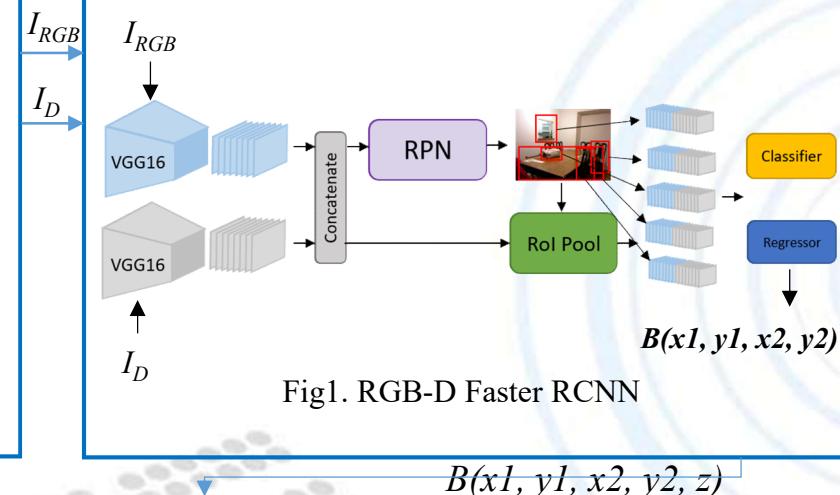
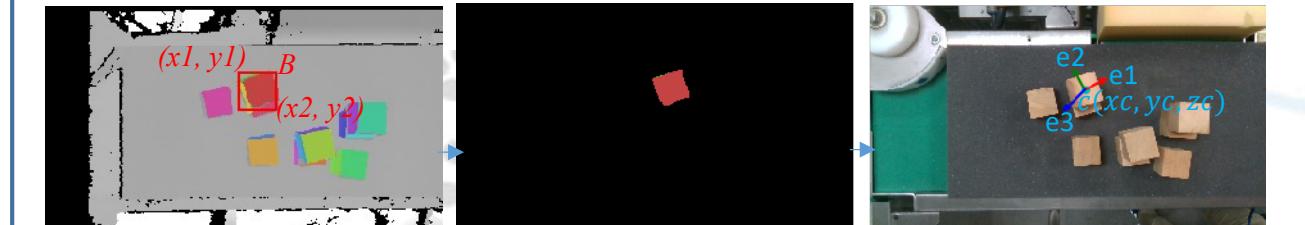


Fig1. RGB-D Faster RCNN

### 2. Facet Segmentation and Normal Direction Estimation using Mask R-CNN



Segment all facets using Mask R-CNN + mask  $B(x1, y1, x2, y2, z)$

Highest (or topes) Facet in  $B(x1, y1, x2, y2, z), I_{top}(x, y, z)$

Output after PCA (SVD):  
 1.1) Facet center point:  $c(x_c, y_c, z_c)$   
 1.2) 夾爪張開方向:  $e_2$   
 1.3) Facet normal direction:  $e_3$

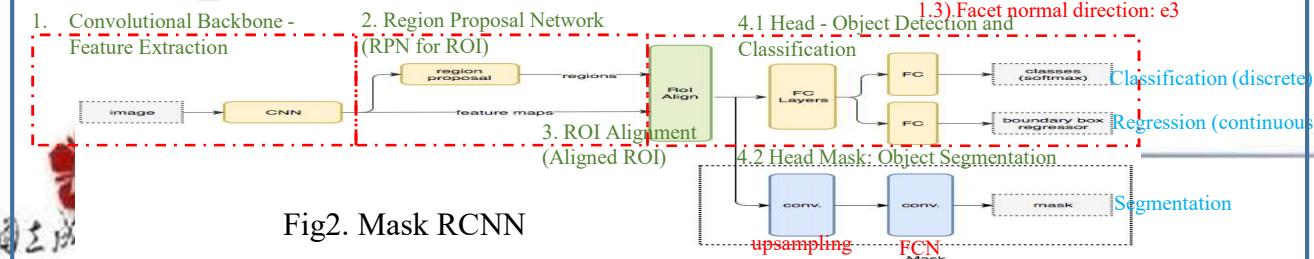
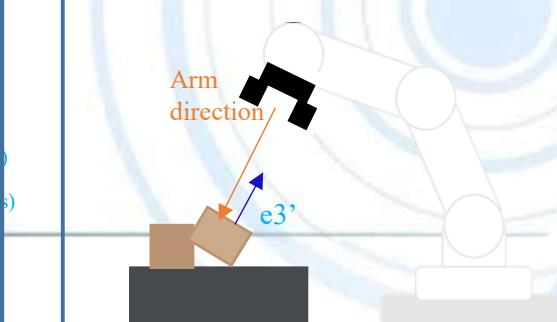


Fig2. Mask RCNN

### 3. Picking Target Object

- 1) Transform depth image coordinate  $c, e_2, e_3$  to robot arm coordinate  $c', e_2', e_3'$
- 2) Grasping object along  $e_3'$  direction



## 4.4 Multi-Task: Visual-Guided Robot Arm Using VGG-16

### - Pile of Objects (3/3) 02:42:28

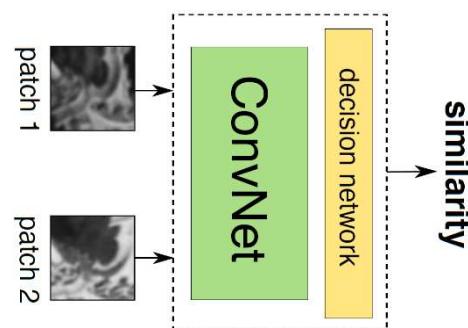


## 4.4 Multi-Task: Visual-Guided Robot Arm Using Deep Learning - Pile of Objects

堆疊螺栓辨識



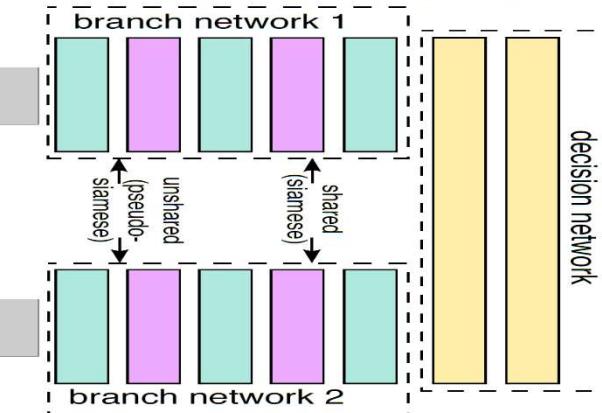
## 4.5 Multi-Input: Multi-Focus Image Fusion Using Siamese



A: FG Clear, BG Blur



B: FG Blur, BG Clear

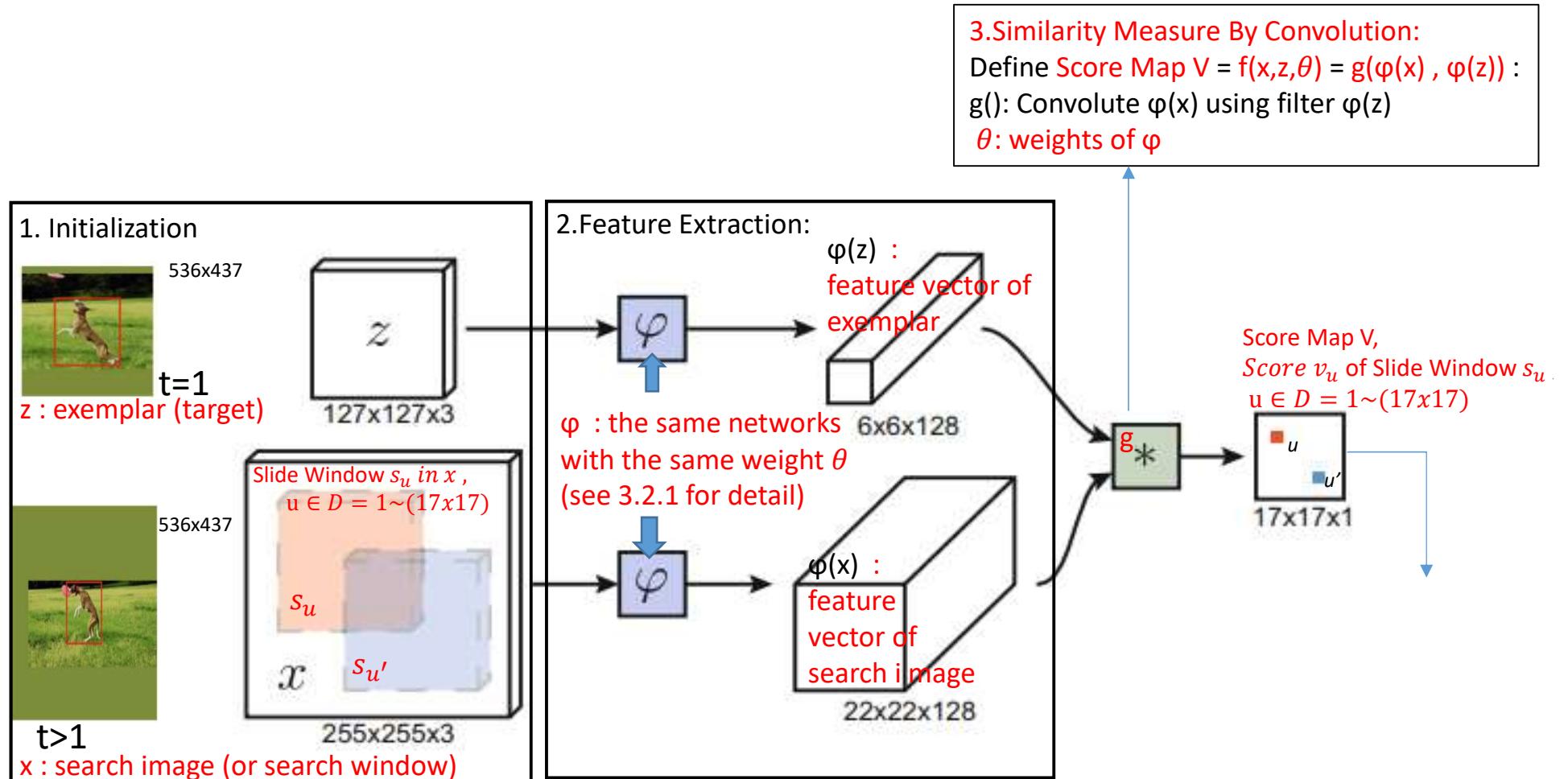


A+B: Fusion FG Clear, BG Clear



of Computer Science and Engineering

## 4.5 Multi-Input: Tracking / Stereo Using Siamese



# 5.1 Deep Learning: 3. R-NN/LSTM

Which one do I want to be?

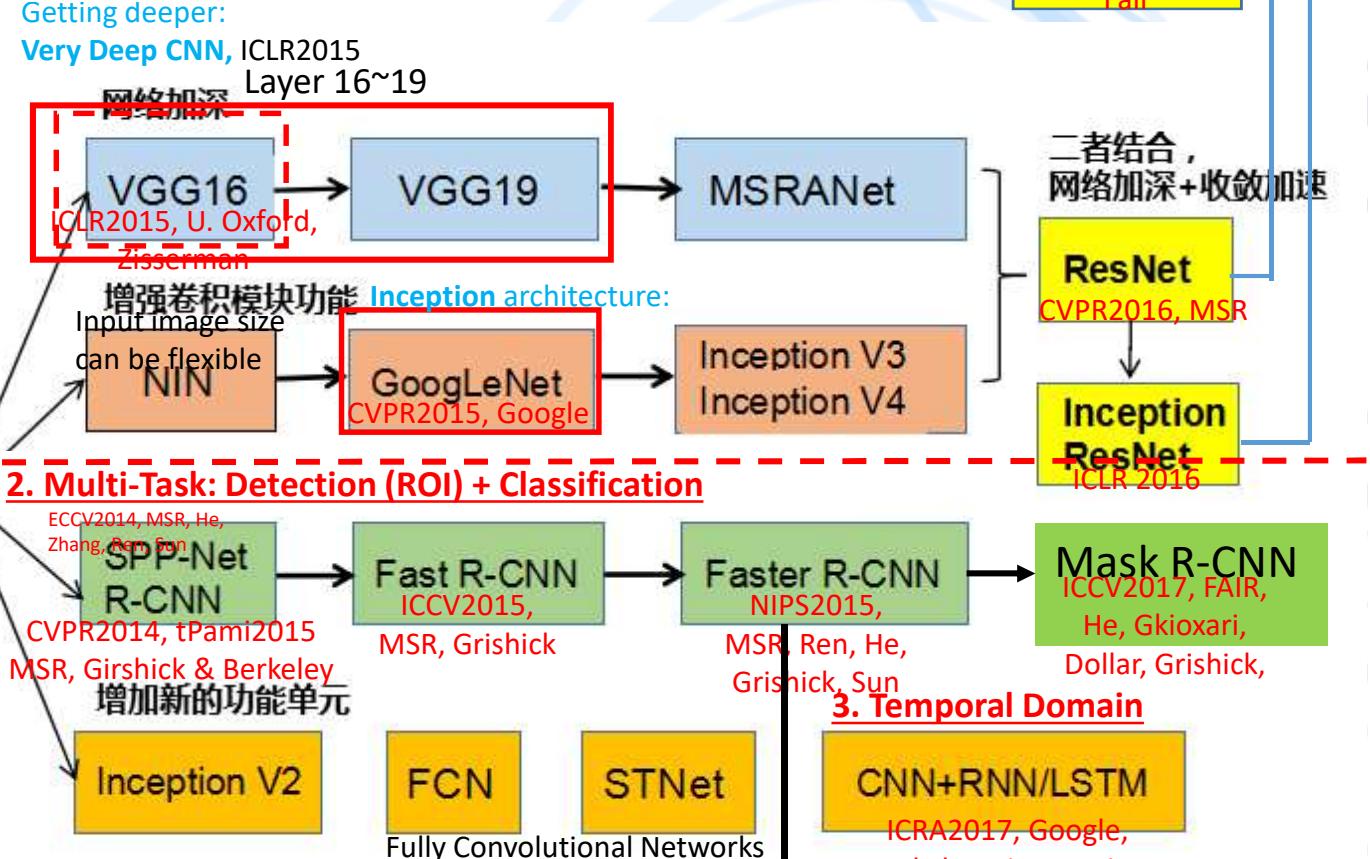
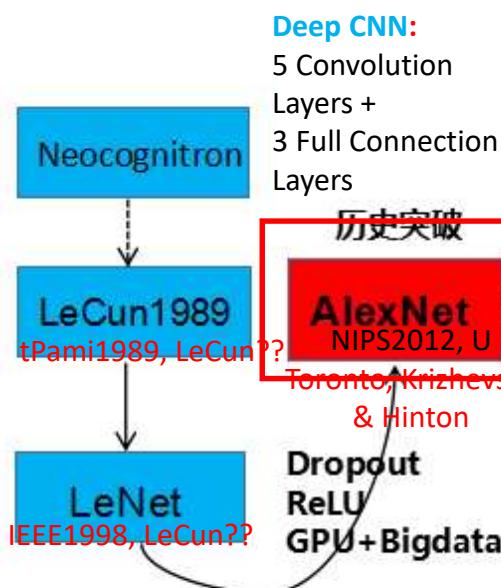
1. Basic CNN Creation
2. Modified CNN
3. CNN Application: Data collection, organization and analysis / benchmark

## 1. Basic CNN

Getting deeper:

Very Deep CNN, ICLR2015

Layer 16~19



tPAMI: IEEE Transactions on Pattern Analysis and Machine Intelligence

CVPR: Conference on Computer Vision and Pattern Recognition

NIPS: Conference on Neural Information Processing Systems

ICLR: International Conference on Learning Representation

NIN: Network in Network

R-CNN: Region-based Convolutional Network method

SPP-Net: Spatial Pyramid Pooling networks

## 4. GAN, AutoEncoder:

Generative Adversarial Network Vs. PCA

## 5. Reinforcement Learning

Unsupervised

DenseNet  
CVPR2017

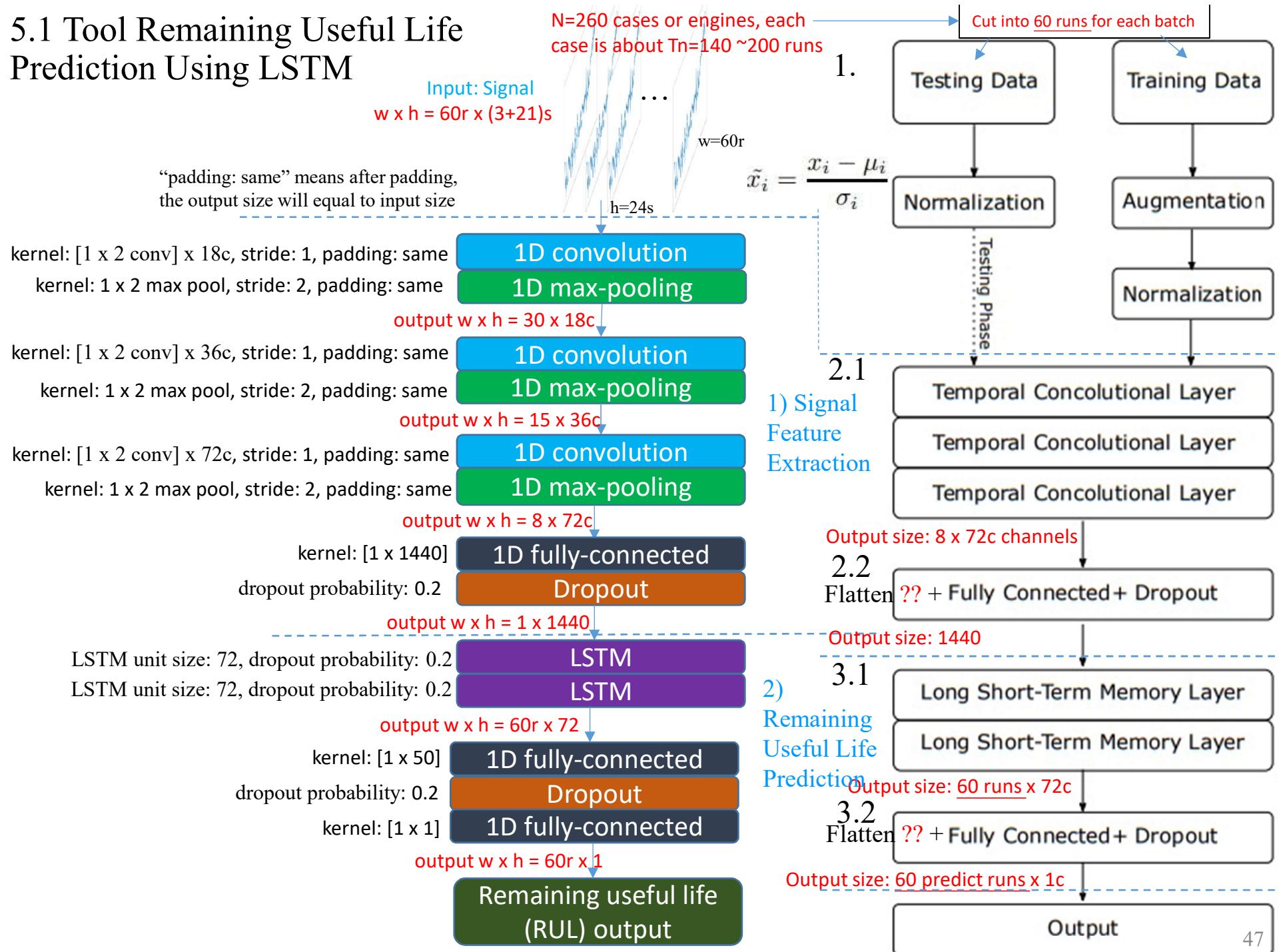
ResNeXt  
CVPR2017,  
Fair

二者结合，  
网络加深+收敛加速  
ResNet  
CVPR2016, MSR  
Inception ResNet  
ICLR 2016

CNN+RNN/LSTM  
ICRA2017, Google,  
Berkeley, Finn, Levine

1. CVPR2016, Girshick  
YOLO V1 V2 V3  
Retina

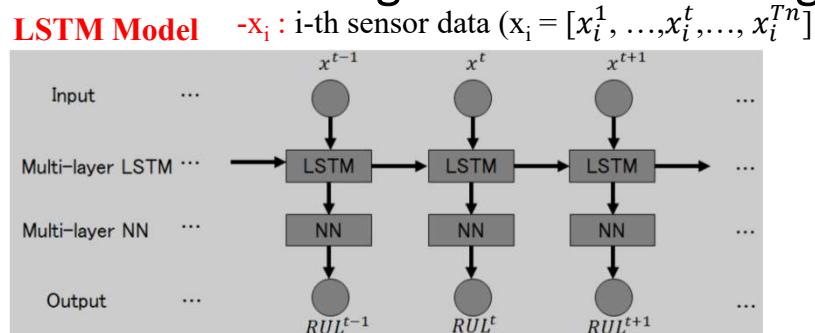
## 5.1 Tool Remaining Useful Life Prediction Using LSTM



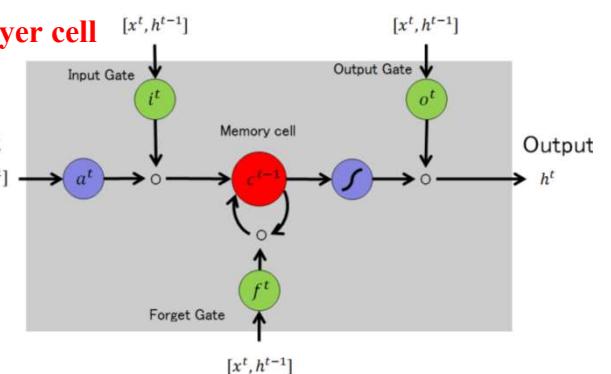
## 5.1 LSTM: Tool Life Signal Prediction Using **LSTM**

02:12.31 **LSTM Layer**

### LSTM Model



LSTM:  
Long Short-Term Memory



# 6.0 Deep Learning: 4. GAN, AutoEncoder

Which one do I want to be?

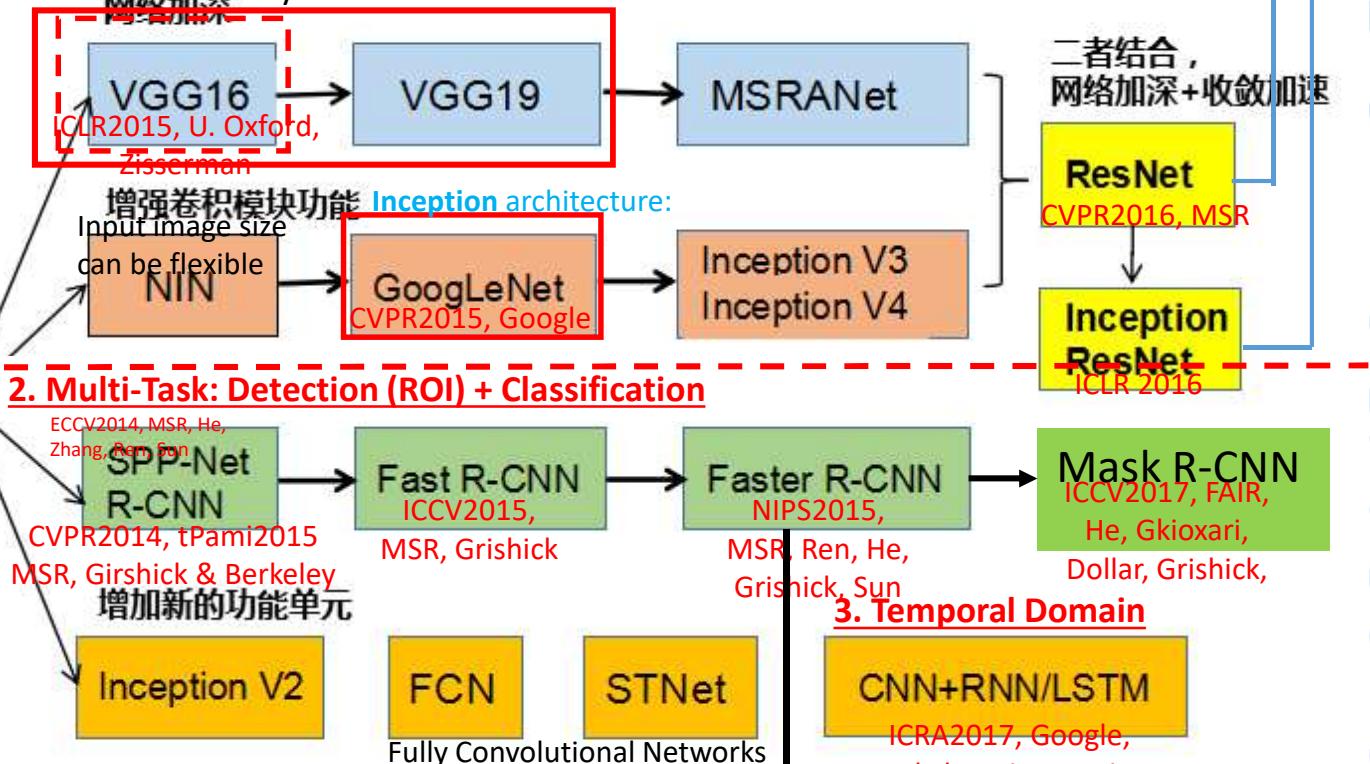
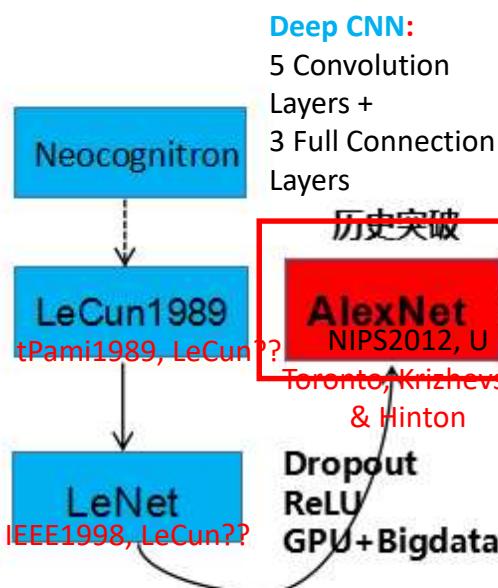
1. Basic CNN Creation
2. Modified CNN
3. CNN Application: Data collection, organization and analysis / benchmark

## 1. Basic CNN

Getting deeper:

Very Deep CNN, ICLR2015

Layer 16~19



tPAMI: IEEE Transactions on Pattern Analysis and Machine Intelligence

CVPR: Conference on Computer Vision and Pattern Recognition

NIPS: Conference on Neural Information Processing Systems

ICLR: International Conference on Learning Representation

NIN: Network in Network

R-CNN: Region-based Convolutional Network method

SPP-Net: Spatial Pyramid Pooling networks

## 4. GAN, AutoEncoder:

Generative Adversarial Network Vs. PCA

## 5. Reinforcement Learning

Unsupervised

# 6.0 GAN (Generative Adversarial Networks) Vs. PCA: PCA (生成對抗網路)

PCA

- Face Recognition: Input/output gray values

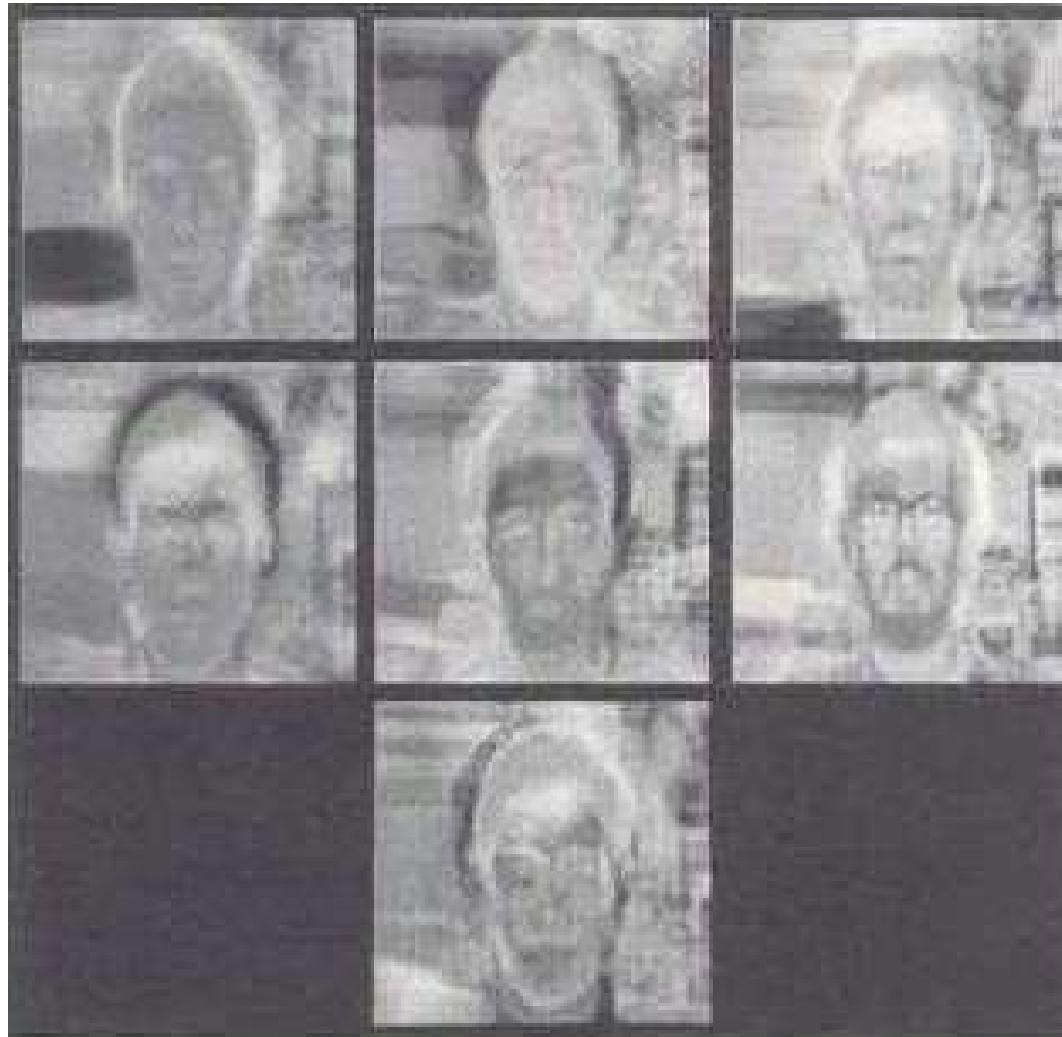


PCA

Jenn-Jier James Lien

## 6.0 GAN (Generative Adversarial Networks) Vs. PCA: PCA

- ❑ PCA – Eigenspace encoding: Eigenvectors



PCA

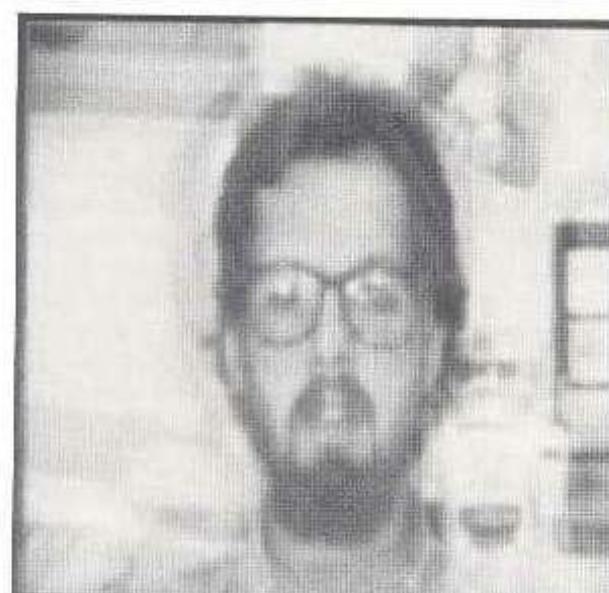
Jenn-Jier James Lien

# 6.0 GAN (Generative Adversarial Networks) Vs. PCA: PCA

## □ PCA – Eigenspace decoding: Reconstruction



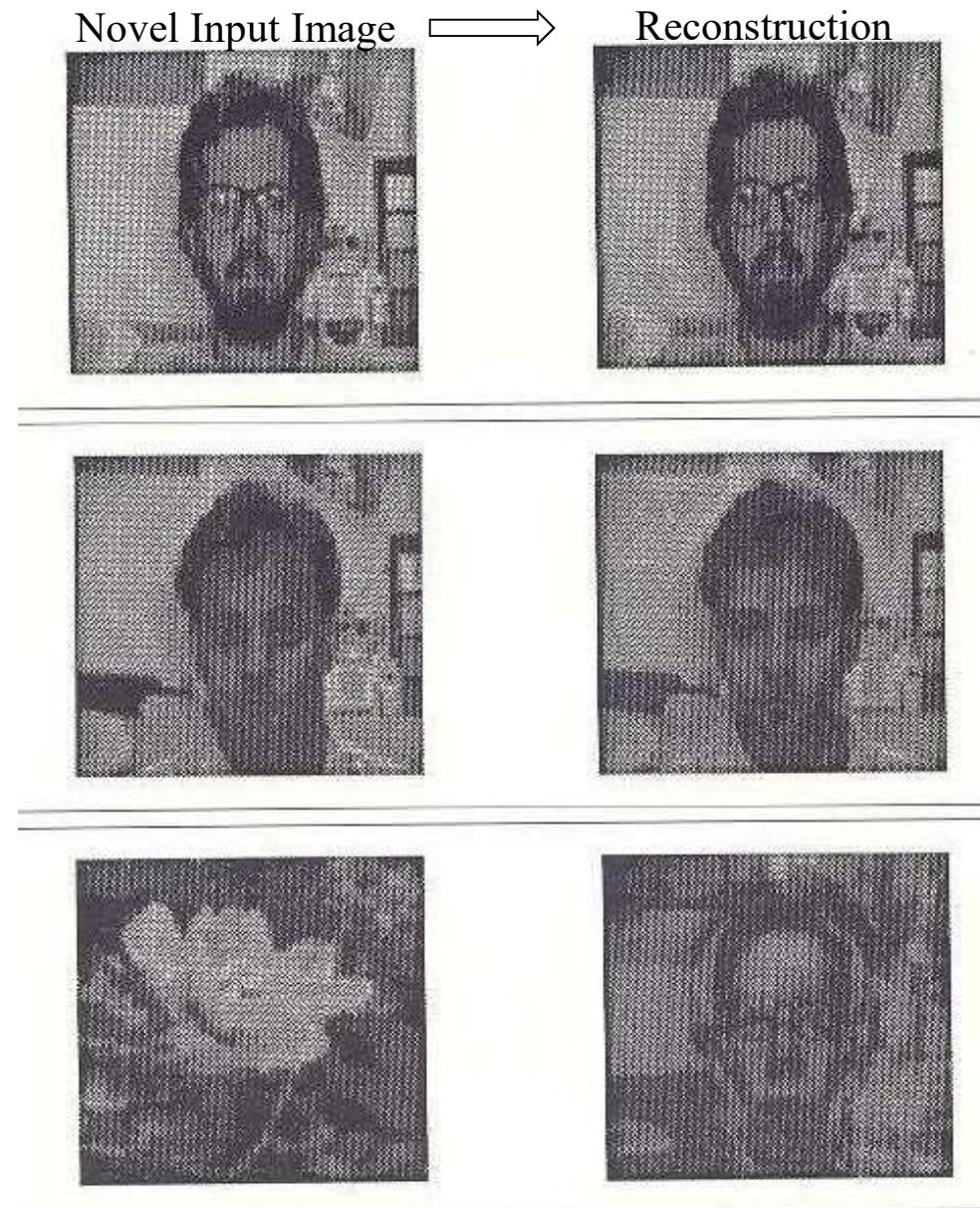
- Decoding:  
Reconstruction  
but Lost details



- Decoding:  
Recovery

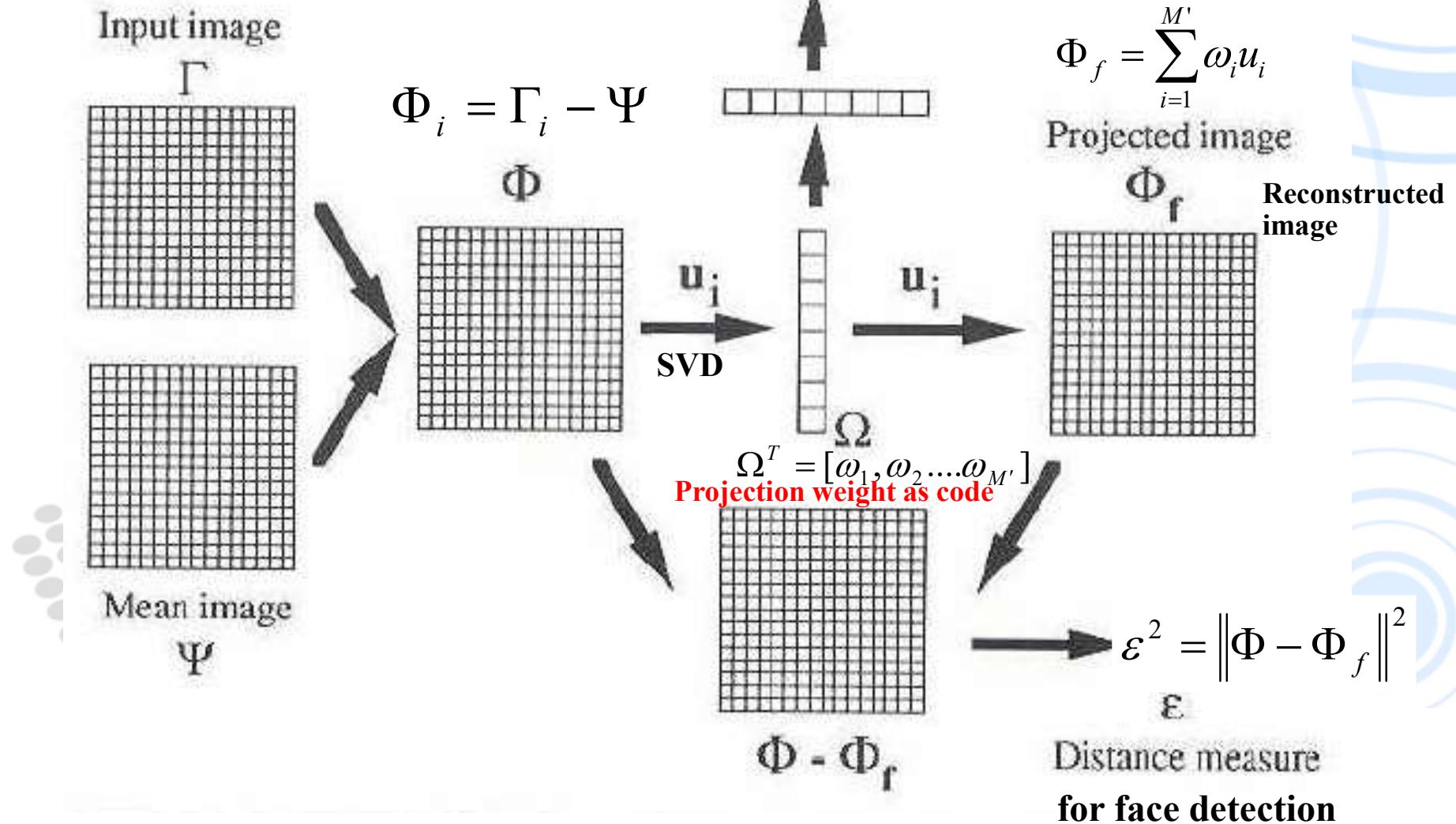
# 6.0 GAN (Generative Adversarial Networks) Vs. PCA: PCA

## □ PCA – Eigenspace decoding: Reconstruction



# 6.0 GAN (Generative Adversarial Networks) Vs. PCA: PCA for face recognition

- PCA: (1) Encoder/Analysis and (2) Decoder/Synthesis/Generator



## 6.1 GAN Vs. PCA: Auto-encoder and Decoder

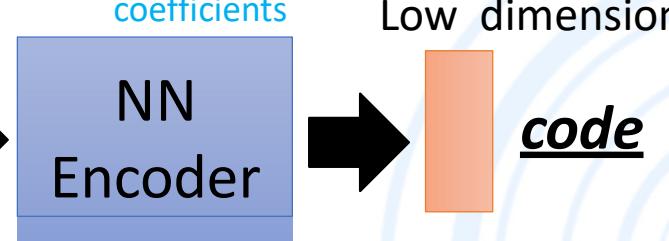
- GAN: (1) Encoder/Analysis and (2) Decoder/Synthesis/Generator

High dimension to low dimension

(1) Analysis:  
Computer  
Vision

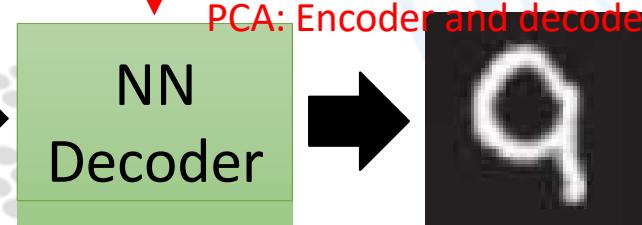


$28 \times 28 = 784$



Compact  
representation of  
the input object

(2) Synthesis /  
Generator:  
ComputerGra  
phics



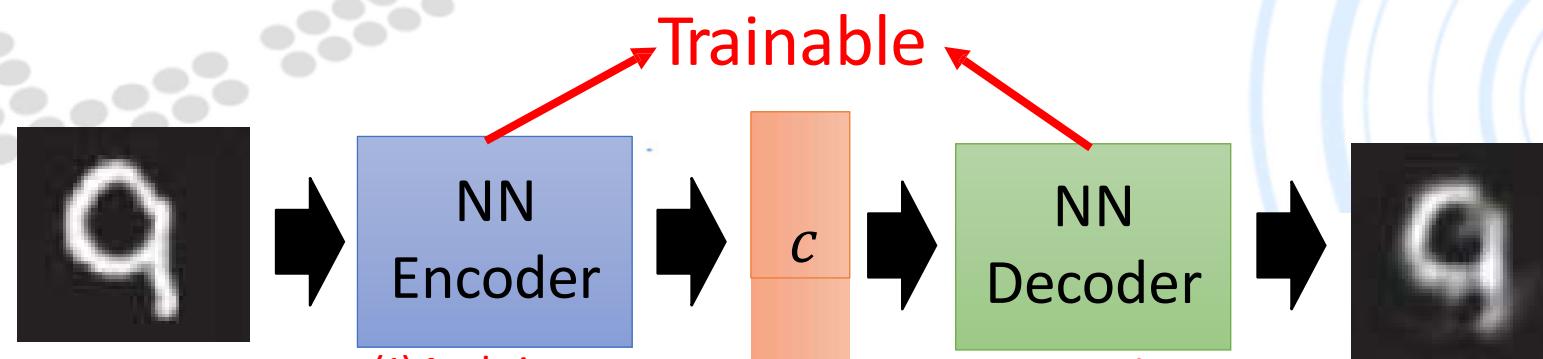
Can reconstruct  
the original object

J: Trainable Encoder/Decoder. A set of basis functions as  
1) PCA (linear) eigenvector,  
2) Filter: Edge filter, wavelets ( $\cos, \sin$ )...  
3) NN (non-linear)

J: Trainable Encoder/Decoder:  
As PCA eigenvectors or filter  
coefficients

J: Code: This low dimensional vector is as projection weights  
or feature extraction result

GAN: NN para. Learn together but different  
Eigenvectors. That is, Encoder coeff. And Decoder  
coeff are different  
PCA: Encoder and decoder have the same para.



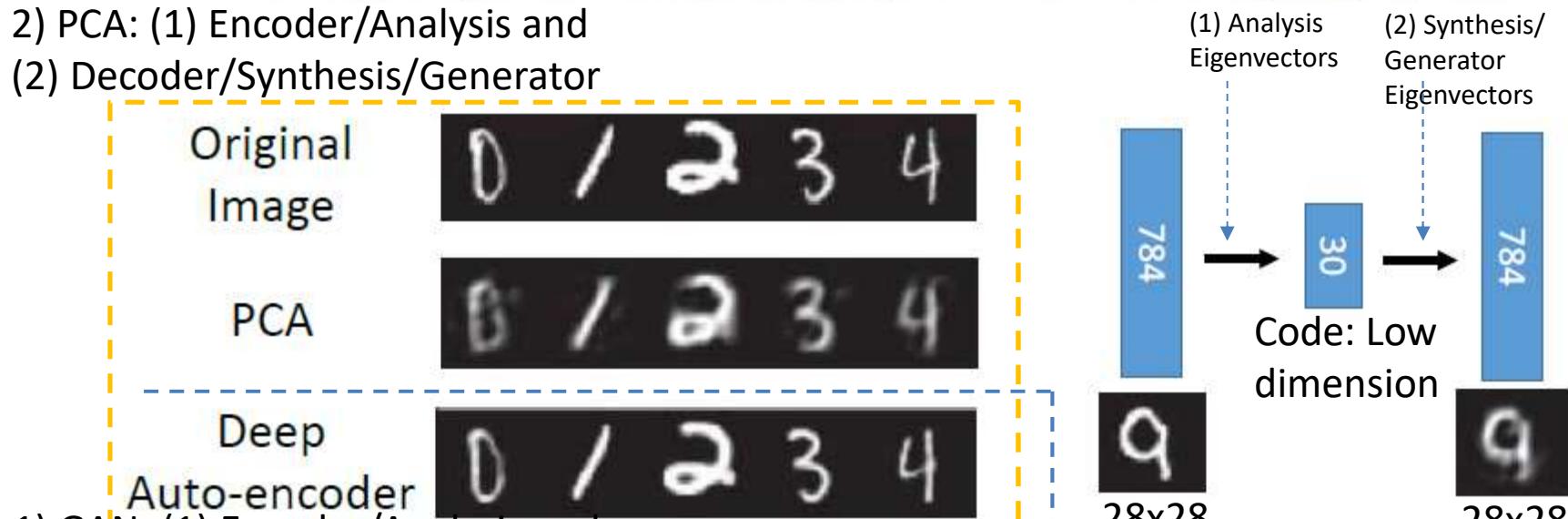
Need to keep processing /  
improving....

# 6.1 GAN (Generative Adversarial Networks) Vs. PCA

Jenny: difference of the result of PCA and Deep auto-encoder

Reference: Hinton, Geoffrey E., and Ruslan R. Salakhutdinov. "Reducing the dimensionality of data with neural networks." *Science* 313.5786 (2006): 504-507

- 2) PCA: (1) Encoder/Analysis and  
(2) Decoder/Synthesis/Generator



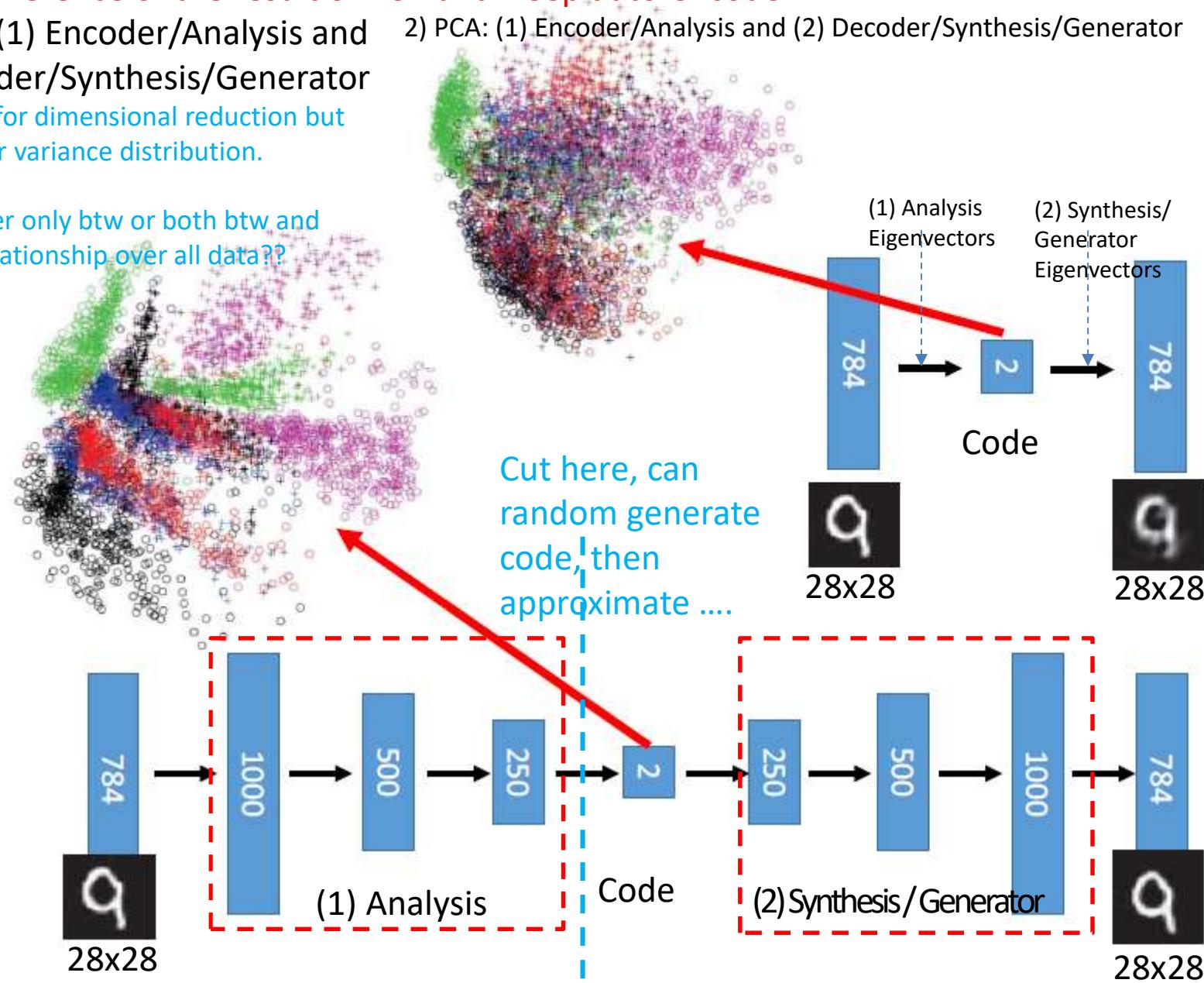
# 6.1 GAN (Generative Adversarial Networks) Vs. PCA

Jenny: difference of the result of PCA and Deep auto-encoder

- 1) GAN: (1) Encoder/Analysis and  
(2) Decoder/Synthesis/Generator

Not only for dimensional reduction but  
has better variance distribution.

J: Consider only btw or both btw and  
within relationship over all data???



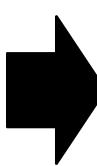
# 6.1 Basic Idea of GAN: Adversarial - Basic Idea of GAN: Adversarial - Relationship btw Generator and **Discriminator**

- GAN: 2 Models. Example

2) **Generator model as a butterfly:** changing to fit the brown-color leaf background in order not to be found.

3) **Discriminator model as a bird:** who wants to eat butterfly. Getting smarter to find Generator model target.

Improving better and better **till cannot find the Generator Model butterfly.**



Becomes brown



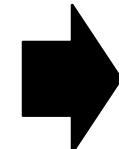
veins

(2) Generator

Butterflies are  
not brown,  
easy to be found

Butterflies becomes brown  
color as background but don't  
have veins,  
Still easy to be found

Where is the butterfly?  
Cannot be found



(3) Discriminator

國立成功大學

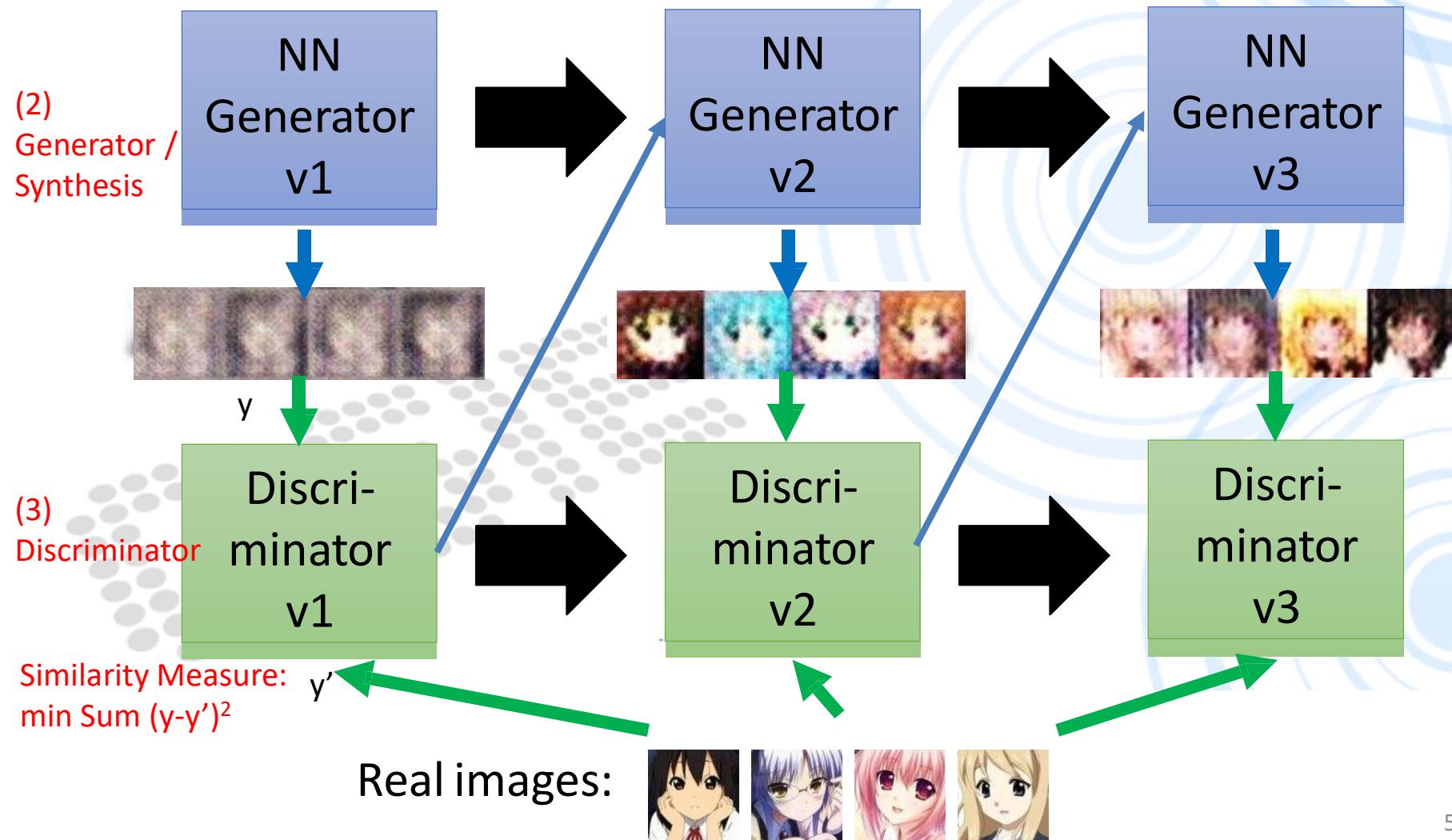


Department of Computer Science and  
Robotics Lab

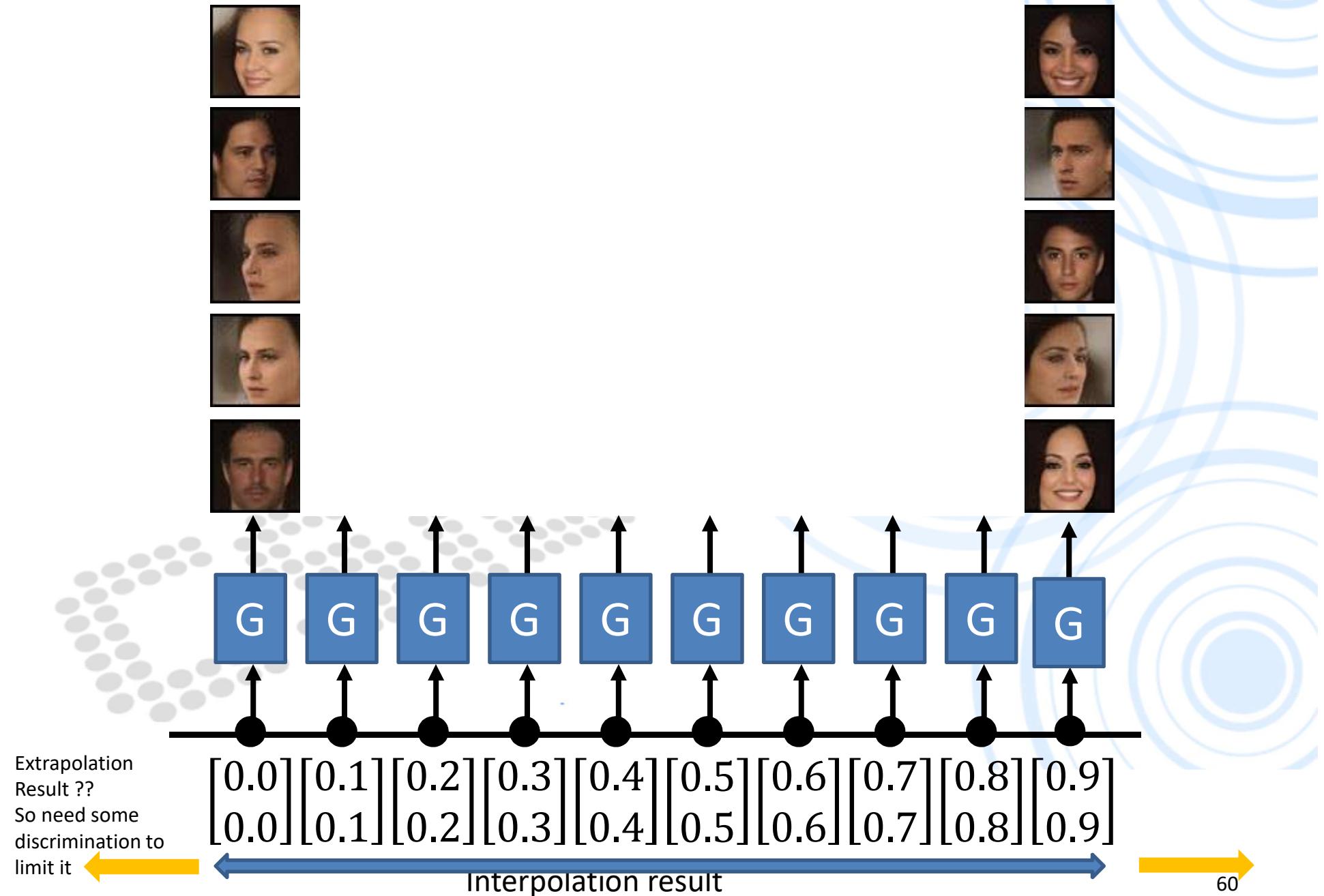
## 6.1 Basic Idea of GAN: Adversarial

This is where the term "**“adversarial”** comes from.

You can explain the process in different ways.....

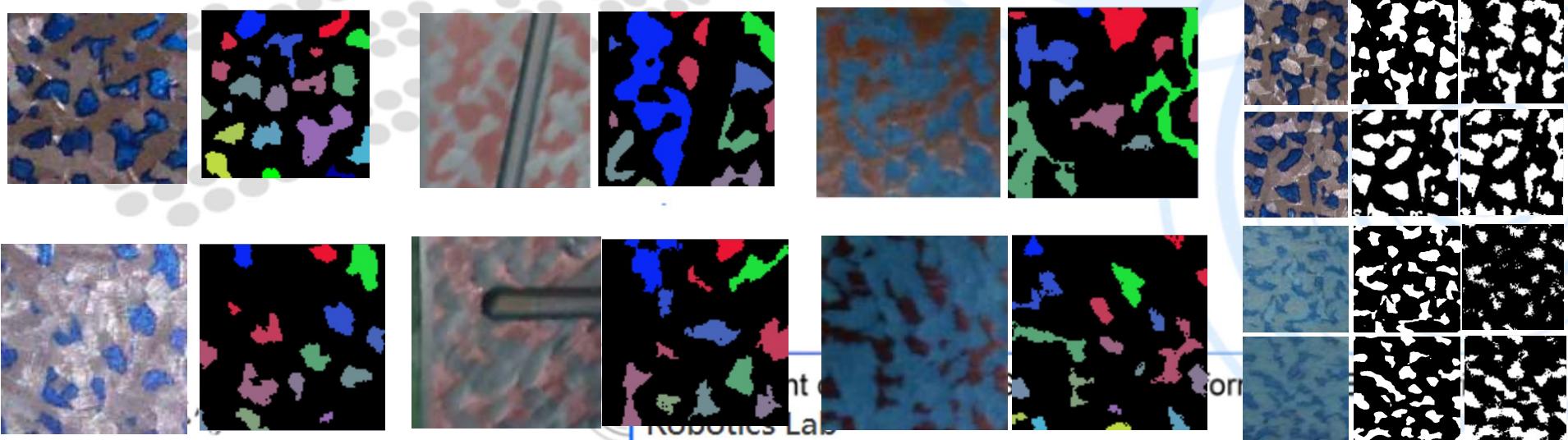
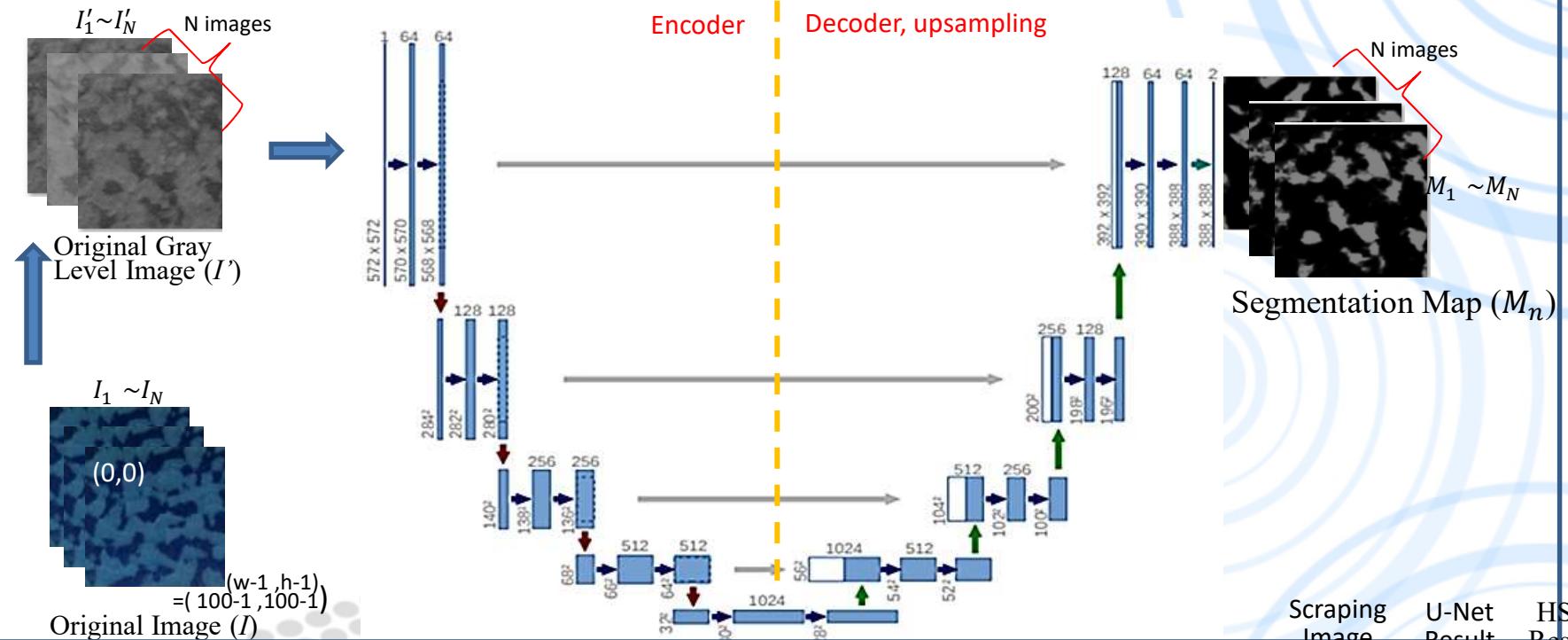


## 6.1 GAN: Interpolation Vs. Extrapolation



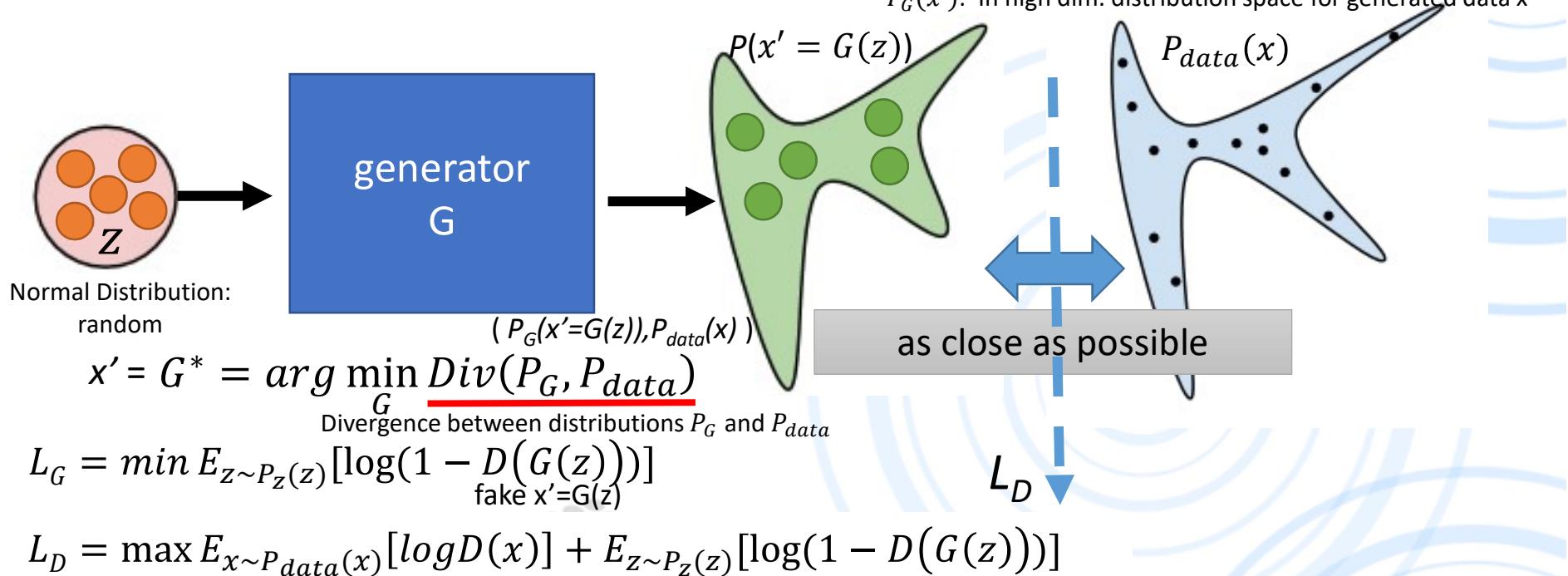
## 6.2 Application 1 - Scraping Segmentation Using U-Net

### Scraping Segmentation Using U-Net (Vs. AutoEncoder)



# 6.2 GAN: Augmented Data

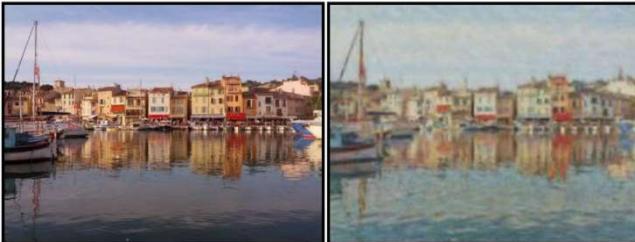
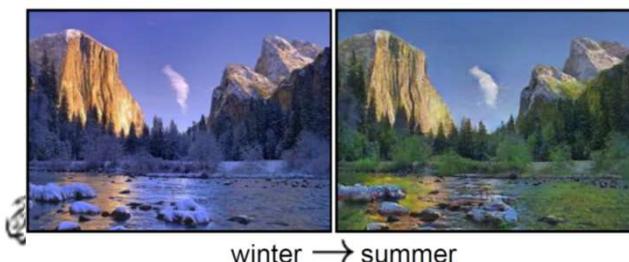
GAN: Generative Adversarial Networks



1) GAN:



2) GAN: Deblurring



en



# 7.0 Deep Learning: 5. Reinforcement Learning

Which one do I want to be?

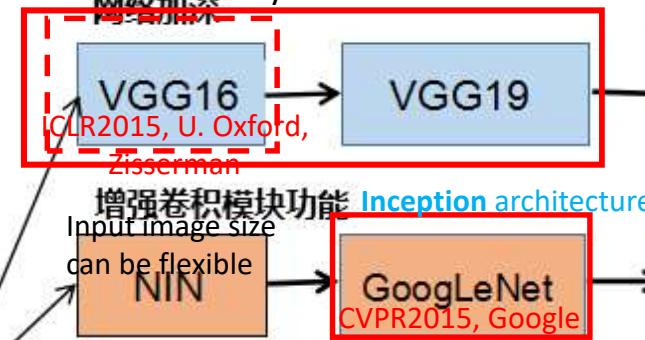
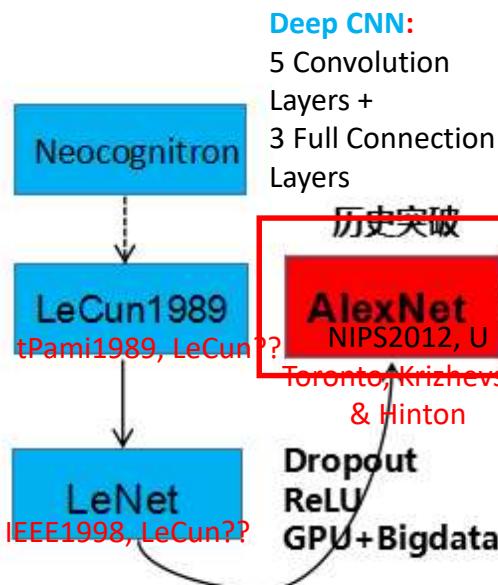
1. Basic CNN Creation
2. Modified CNN
3. CNN Application: Data collection, organization and analysis / benchmark

## 1. Basic CNN

Getting deeper:

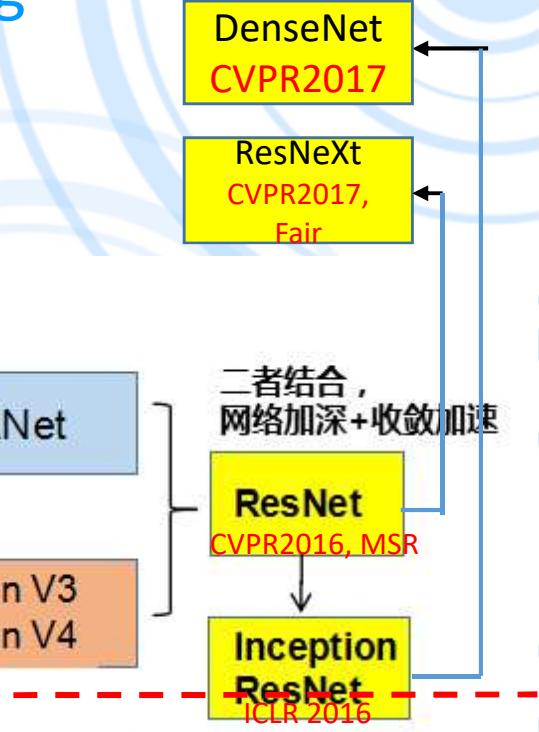
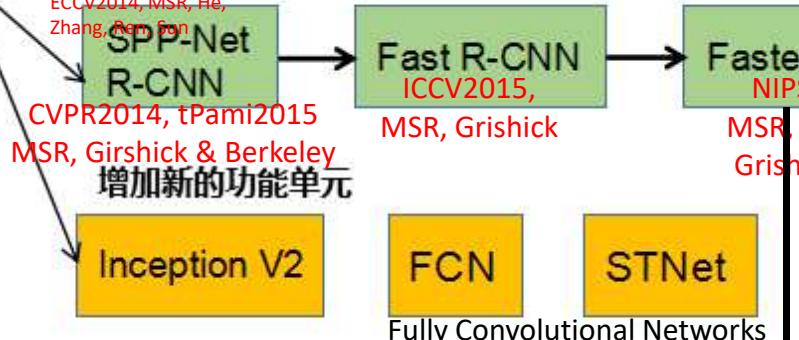
Very Deep CNN, ICLR2015

Layer 16~19



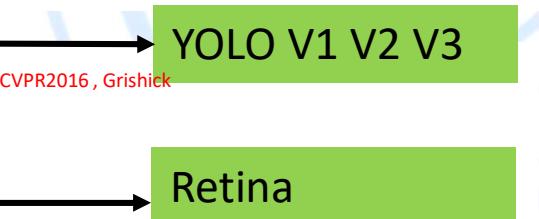
## 2. Multi-Task: Detection (ROI) + Classification

ECCV2014, MSR, He, Zhang, Ren, Sun  
增加新的功能单元 (Add new functional units)



## 3. Temporal Domain

CNN+RNN/LSTM (ICRA2017, Google, Berkeley, Finn, Levine)



## 4. GAN, AutoEncoder:

Generative Adversarial Network Vs. PCA

## 5. Reinforcement Learning

Unsupervised

tPAMI: IEEE Transactions on Pattern Analysis and Machine Intelligence

CVPR: Conference on Computer Vision and Pattern Recognition

NIPS: Conference on Neural Information Processing Systems

ICLR: International Conference on Learning Representation

NIN: Network in Network

R-CNN: Region-based Convolutional Network method

SPP-Net: Spatial Pyramid Pooling networks

# 7.0 Reinforcement: Neural network learning category

## 1) Supervised Learning: PCA, LDA

- 紿定 input 和 label (target) 的學習方式。

## 2) Unsupervised Learning: VQ

- 只給定 input，學習 input data 間的關係，可用來分群。

## 3) Semi-Supervised Learning

- 紿定少部分資料有 label，大部分資料沒 label，使沒 label 的資料得到對應的 label。

## 4) Reinforcement Learning: Self-Supervised

- 在學習過程中設立 reward (with environment)，藉由 reward 機制自我學習，

- need policy (brain of agent) as rule for agent to learn。



## 7.0.1 Reinforcement Learning

(One Episode, 一局)

MDP (Markov Decision Process)

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

$\times_0$  joint determined

Learning to play Go

T: termination

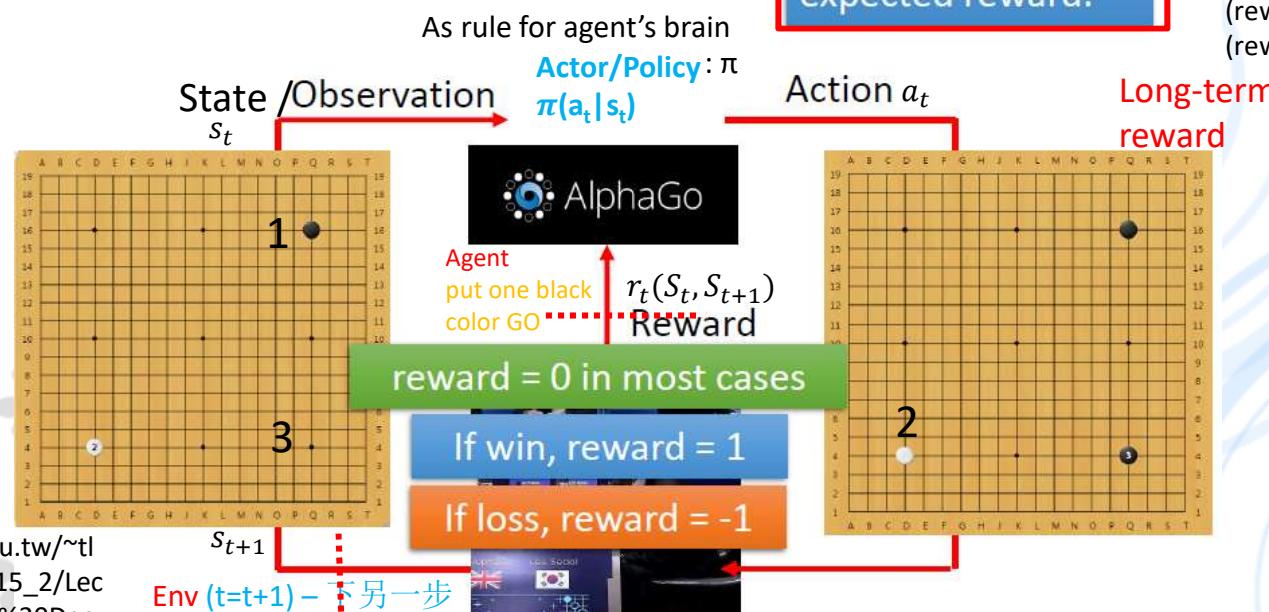
Agent learns to take actions to maximize expected reward.

Reward:

- 1) Delay reward as GO (long-term reward)
- 2) Immediate reward

-Sometimes, reward can't be received immediately (long-term reward).

-The reward is delay, such as, until game is over, then we can understand which one is the winner (reward=1) or loser (reward=0 or -1)



截自：

[http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS\\_2015\\_2/Lecture/Introduction%20to%20Deep%20Reinforcement%20Learning.pdf](http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/Introduction%20to%20Deep%20Reinforcement%20Learning.pdf)

Env. - Put the other white color GO  
Now, there are 2 white and 2 black

對打:  
Agent AlphaGo Action (t) - 下一步 black 3 GO  
Env - white 下下一步 4

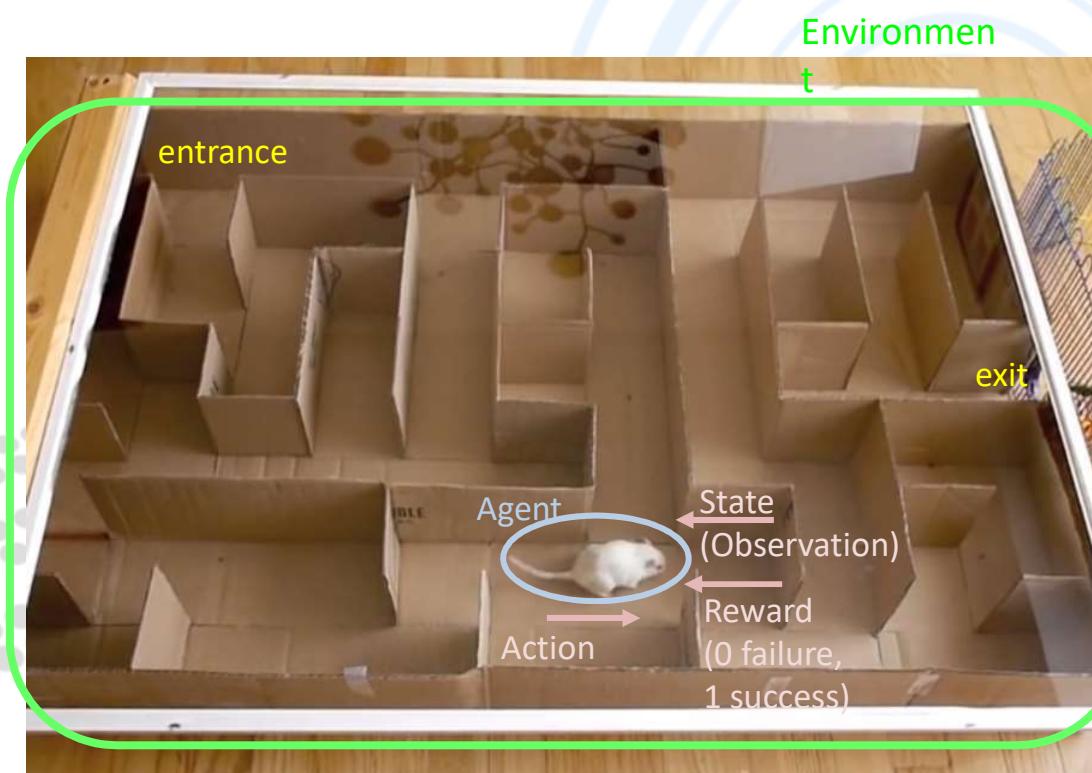


## 7.0.2 Reinforcement Learning

補充

### Reinforcement Learning – Example 1

If we define  
walk toward exit,  
**reward is 1,**  
**Otherwise reward is 0**  
the reward is delay



## 7.0.3 Reinforcement Learning

Observation: current frame  
Action : left, right, fire  
Reward : Score  
Episode : 一局遊戲  
Termination: ....

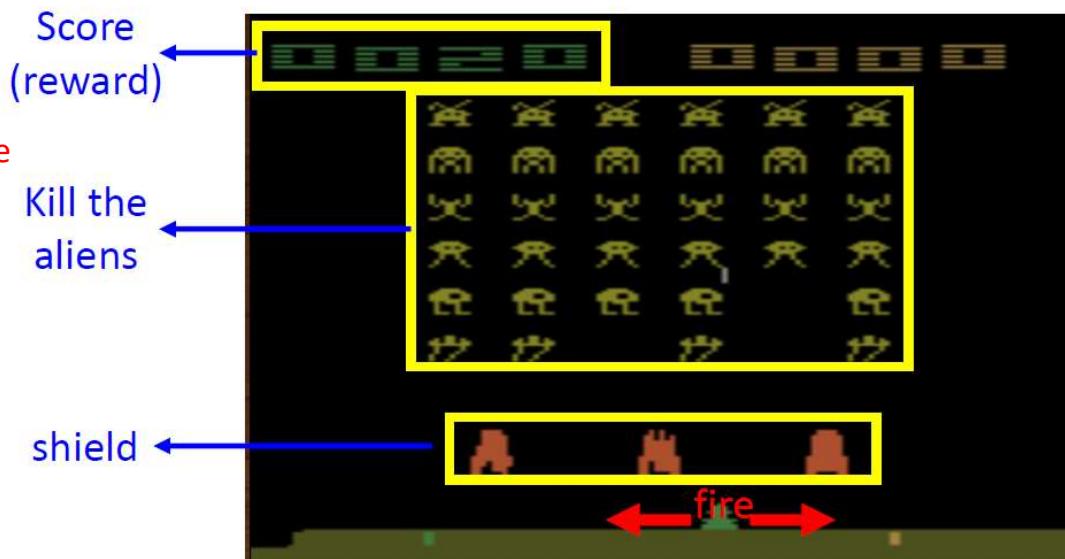
截自：

[http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS\\_2015\\_2/Lecture/Introduction%20to%20Deep%20Reinforcement%20Learning.pdf](http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/Introduction%20to%20Deep%20Reinforcement%20Learning.pdf)



- Space invader

Termination: all the aliens are killed, or your spaceship is destroyed.



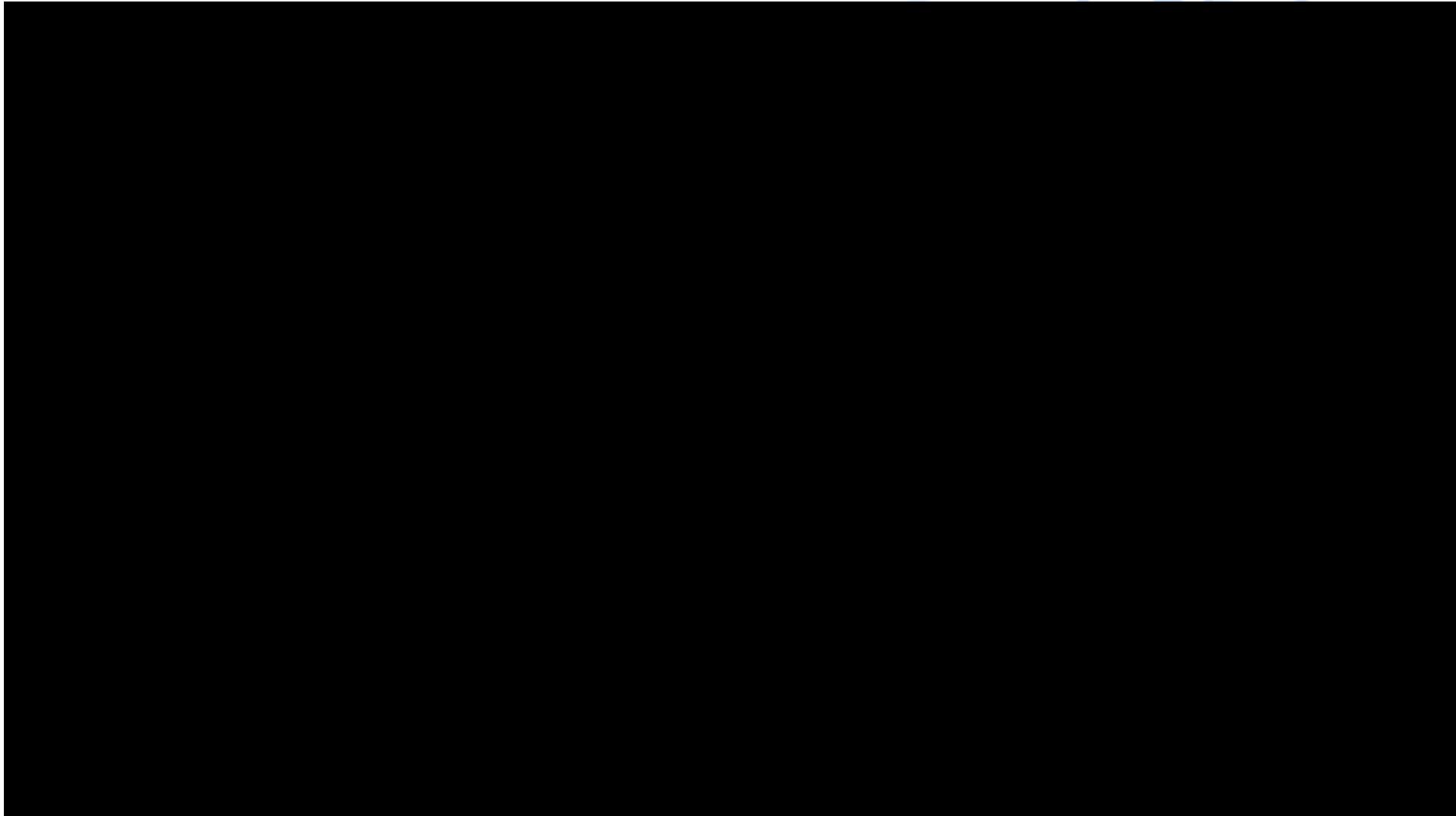
Agent learns to take actions to maximize expected reward.

Long-term reward

If agent doesn't maximize the long-term reward, the agent can only take fire action

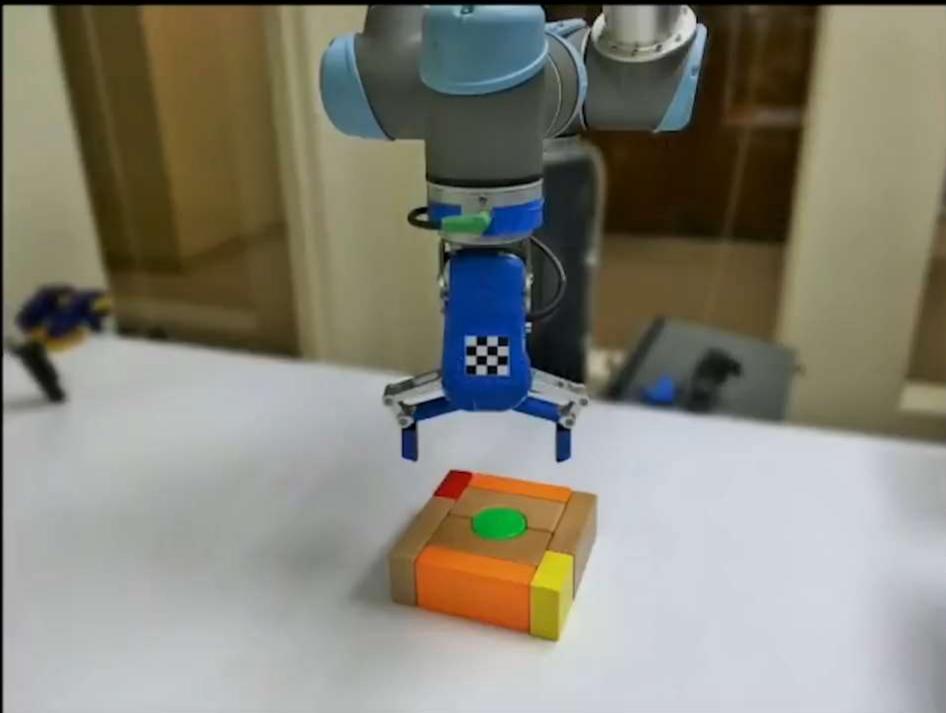
Agent need long term planning

## 7.1 Reinforcement Learning: Robot Arm ~~Grasp or Push~~, and Place – Detection, Classification, and ~~Angle Estimation or Push~~-(1/2) 03:22.44

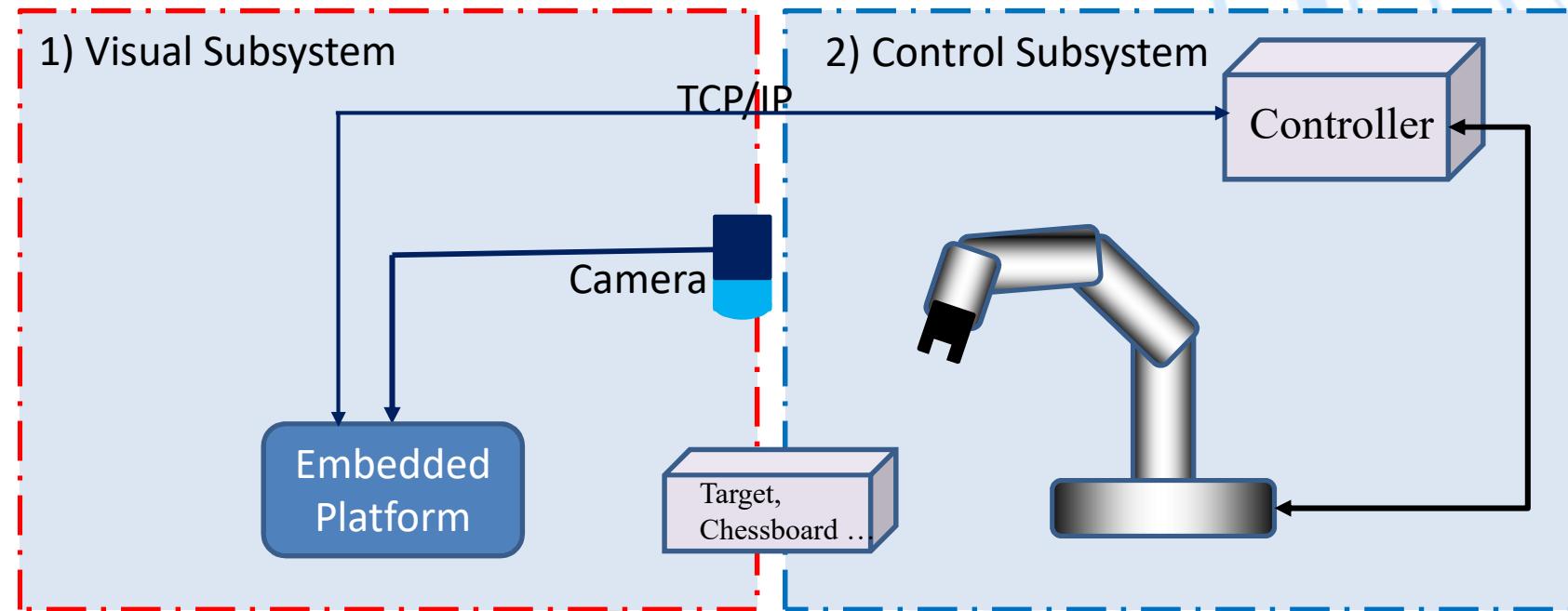


## 7.1 Reinforcement Learning: Robot Arm **Grasp or Push**, and Place – Detection, Classification, and **Angle Estimation or Push** (2/2) 02:30.83

What makes grasping hard?



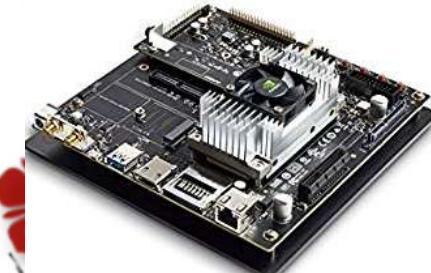
# 8.0 Embedded Deep Learning by Nvidia Jetson TX2



1.1) Camera



1.2) Jetson TX2



國立成功大學



Robotics Lab

2.1) Controller



2.2) Robot Arm

