北京大学
PEKING UNIVERSITY

Convolutional Neural Networks

Hao Dong

2019, Peking University

Convolutional Neural Networks

- Motivation
- Convolutional Algorithms
- Pooling Algorithms
- Hierarchical Representation Learning
- Convolutional Architectures
- Transposed Convolutional Algorithms
- Computer Vision Applications

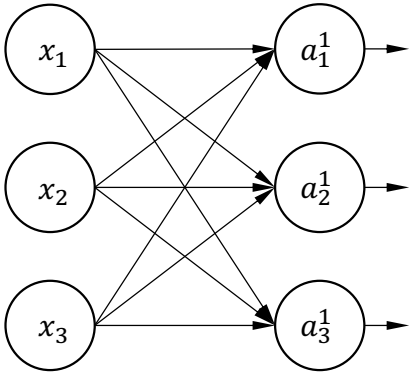
北京大学
PEKING UNIVERSITY

Motivation

Motivation

- Curse of Dimensionality

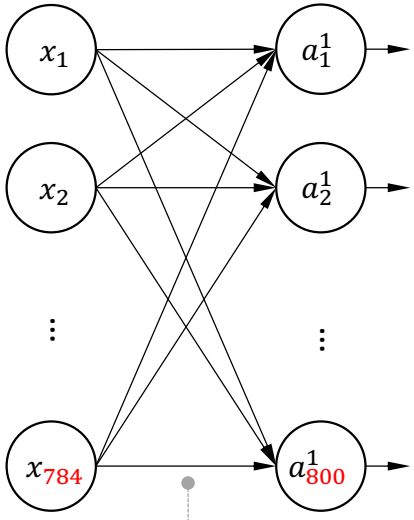
When we only have three inputs:



Fully connected layer

When we have a small image:

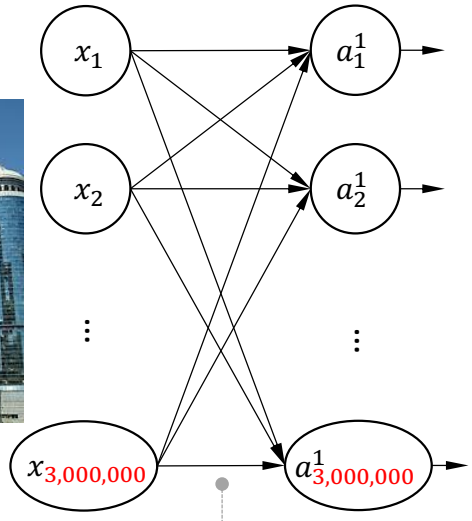
28×28×1



W is a 784×800 matrix

When we have a HD image:

$1K \times 1K \times 3$



W cannot fit into the memory …

Motivation

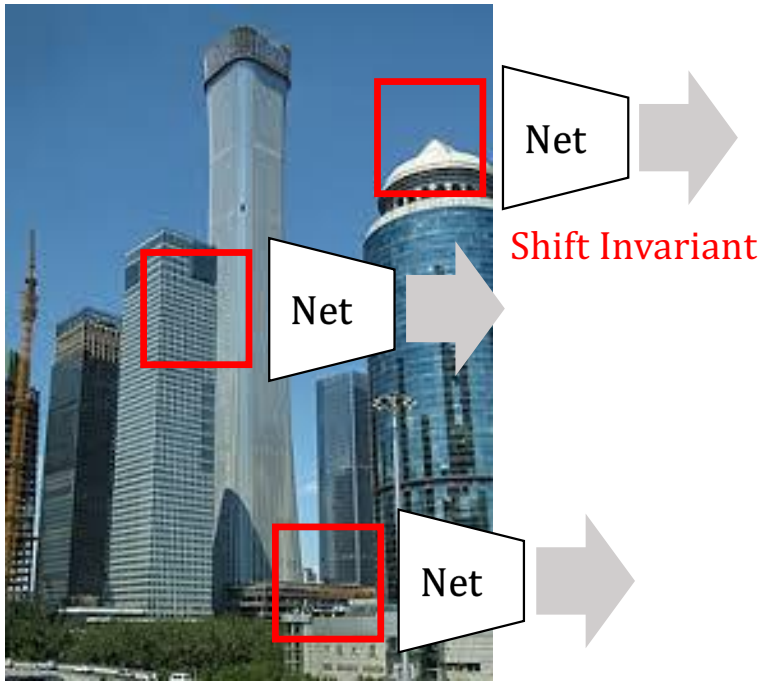
- Eye != Fully Connected Layer

- Do human analysis each pixel using different ways? NO

Shift Invariant

Motivation

- Eye != Fully Connected Layer



- Parameters shared over space
- Sparse connectivity
- Equivariant representation

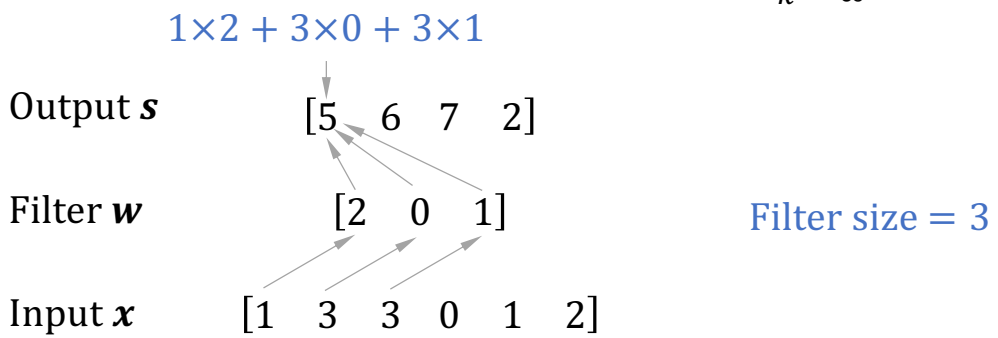
北京大学
PEKING UNIVERSITY

Convolutional Algorithms

Convolutional Algorithms

- Convolution on 1D vector

1D discrete convolution: $\boldsymbol{s}_t = (\boldsymbol{x} * \boldsymbol{w})_t = \sum_{k=-\infty}^{\infty} \boldsymbol{x}_k \boldsymbol{w}_{t-k}$



- The more similar the filter (kernel) and local path are, the higher value on the output

Convolutional Algorithms

- Edge detection on greyscale 2D images

2D discrete convolution: $\boldsymbol{s}_{r,c} = \big((\boldsymbol{x} * \boldsymbol{W})\big)_{r,c} = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} \boldsymbol{x}_{r,c}\, \boldsymbol{w}_{r-i,c-j}$

h×w×1 image

$$\boldsymbol{W} = \begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$$

Detect vertical edges

h×w×1 image

Convolutional Algorithms

- Edge detection on RGB Images

R $\boldsymbol{W}_{:,:1} = \begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$

G $\boldsymbol{W}_{:,:2} = \begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$

B $\boldsymbol{W}_{:,:3} = \begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$

Each input channel needs one filter

$\boldsymbol{W}$ is a 3×3×3 tensor

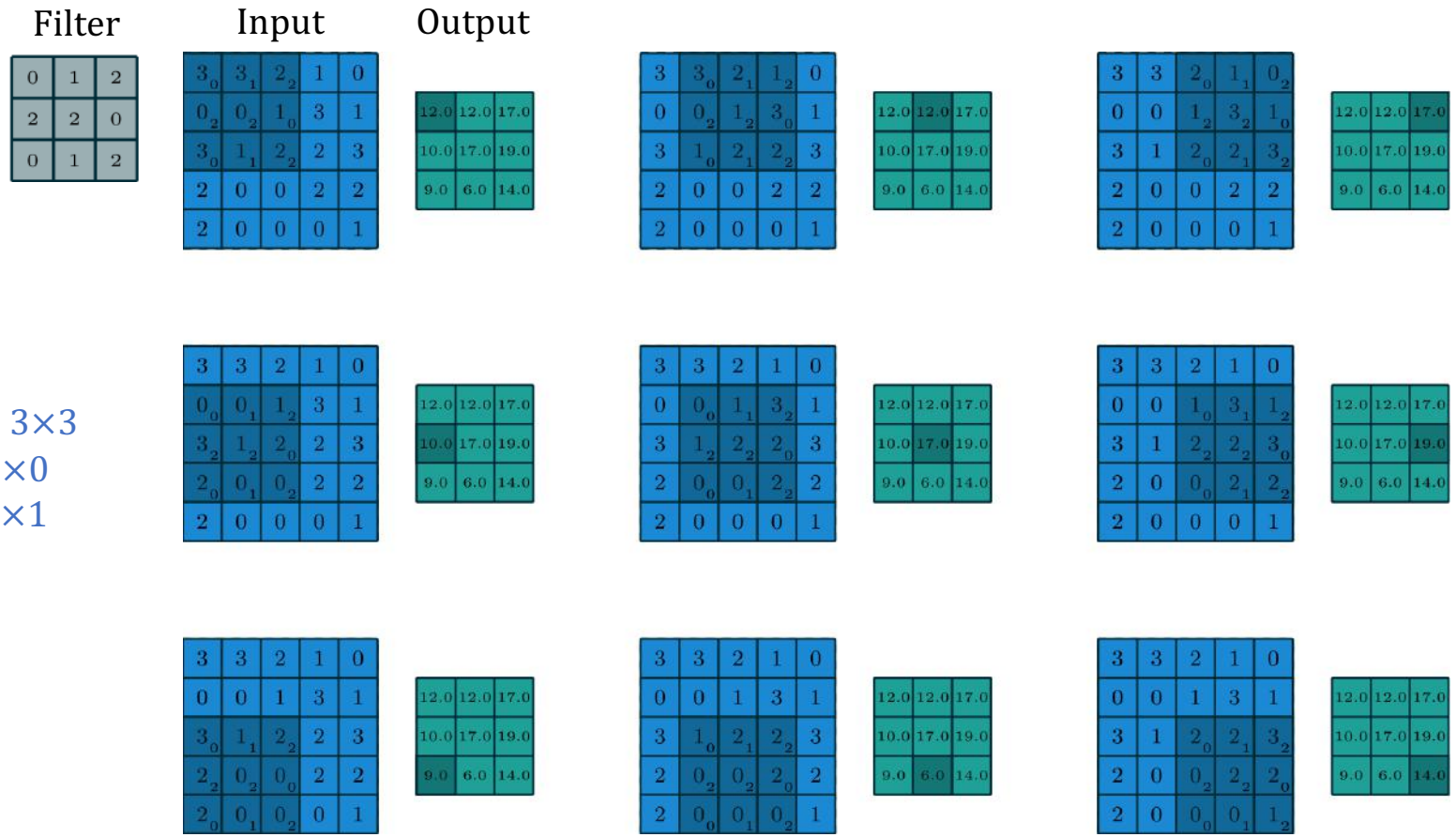
Height of filter size

Width of filter size

Depth of filter size == Number of input channels

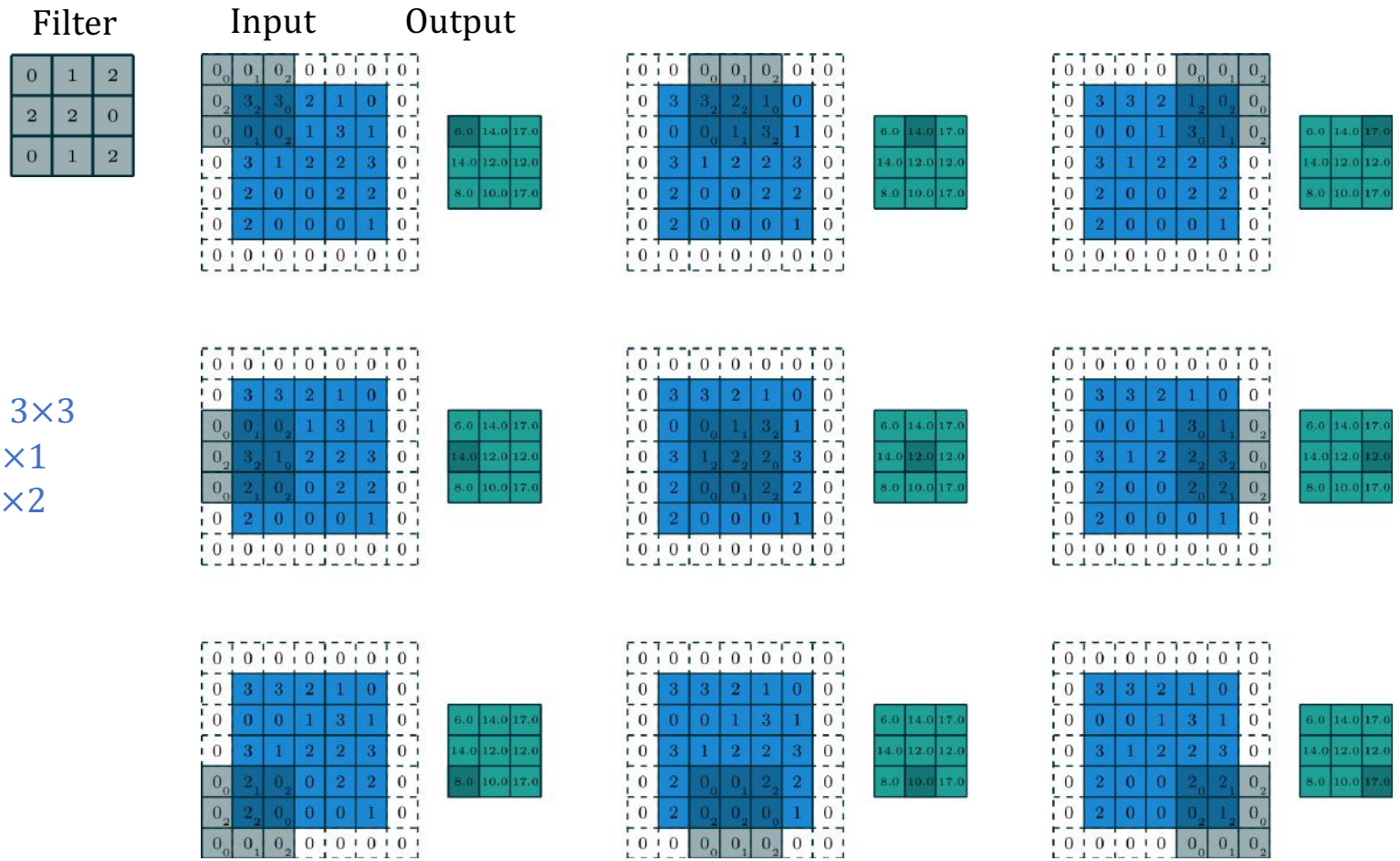
Convolutional Algorithms

- No padding, unit strides



Convolutional Algorithms

- Zero padding, non-unit strides

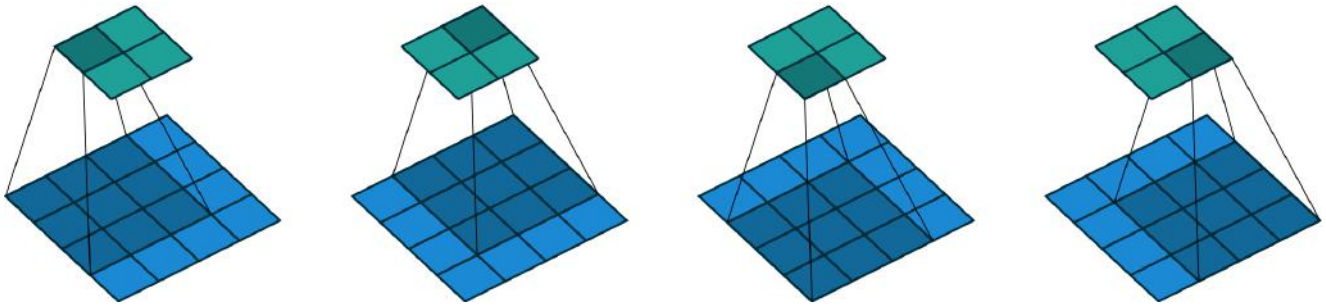


Convolutional Algorithms

- Side Views

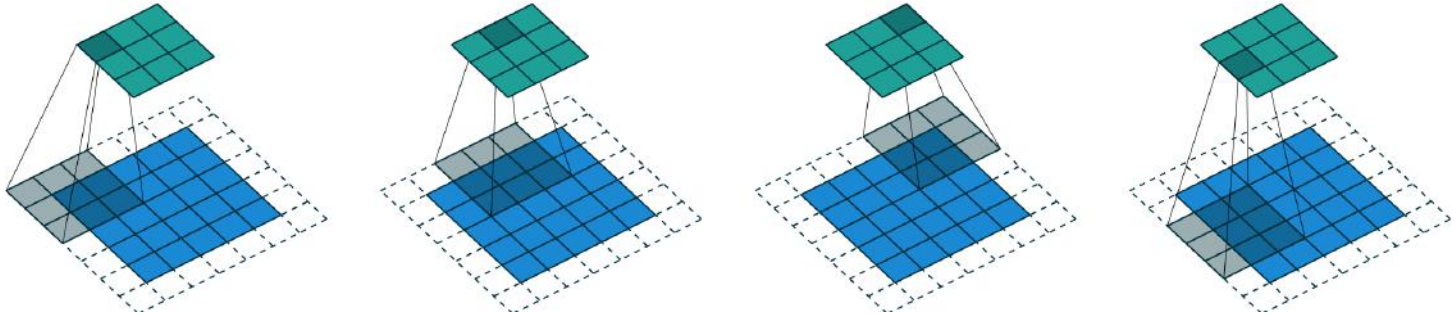
No padding and no strides

Filter size = 3×3
padding = 0×0
Strides = 1×1



Padding and strides

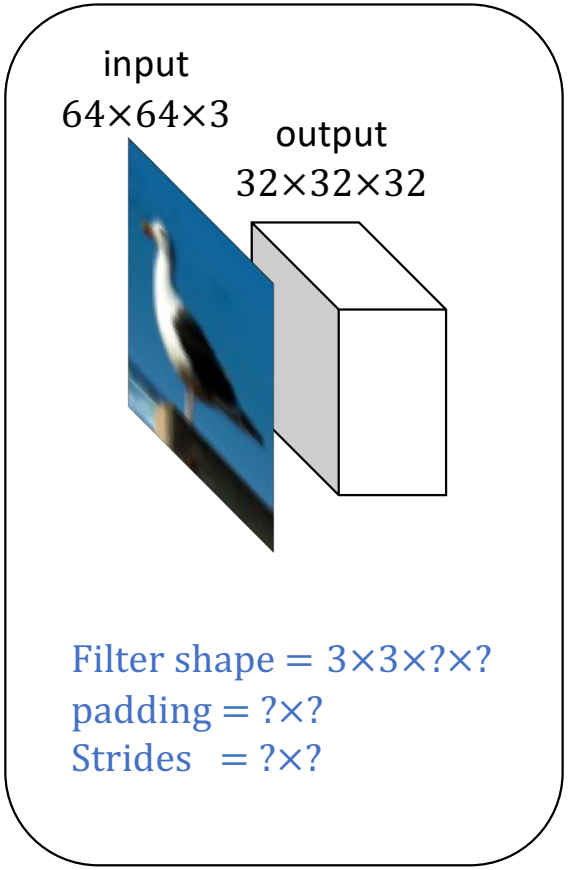
Filter size = 3×3
padding = 1×1
Strides = 2×2



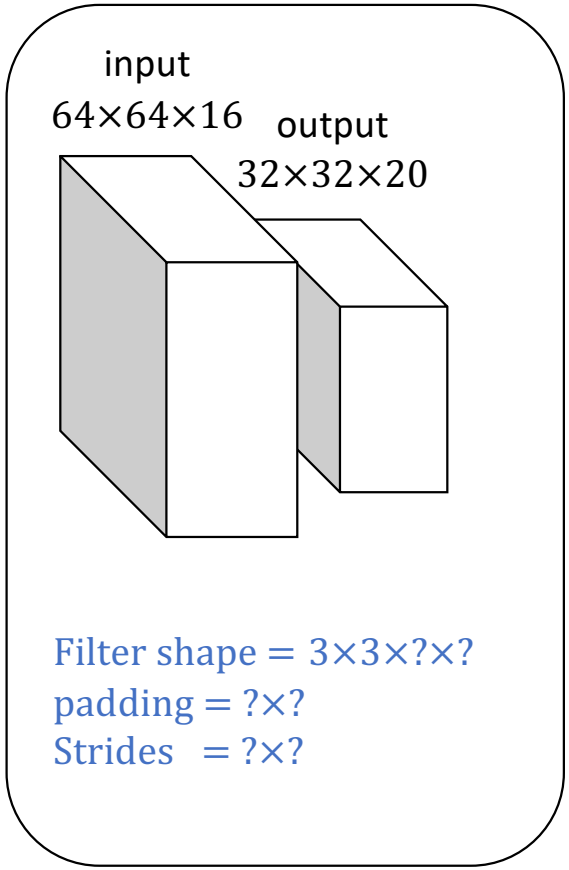
Convolutional Algorithms

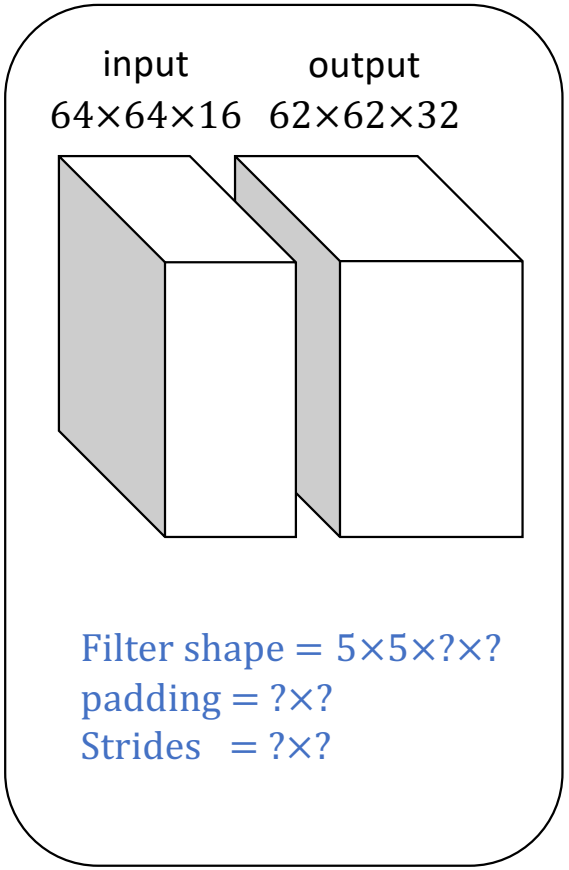
- Filter (kernel) shapes
  - To convolute RGB images, a filter have 3 channels
  - The filter shape is
    (filter_height × filter_width × input_channels × n_filters)
- Number of filters (channels)
  - One filter can have one corresponding output feature map
  - The number of output channels == The number of filters
  - The output shape is (feature_height × feature_width × n_filters)

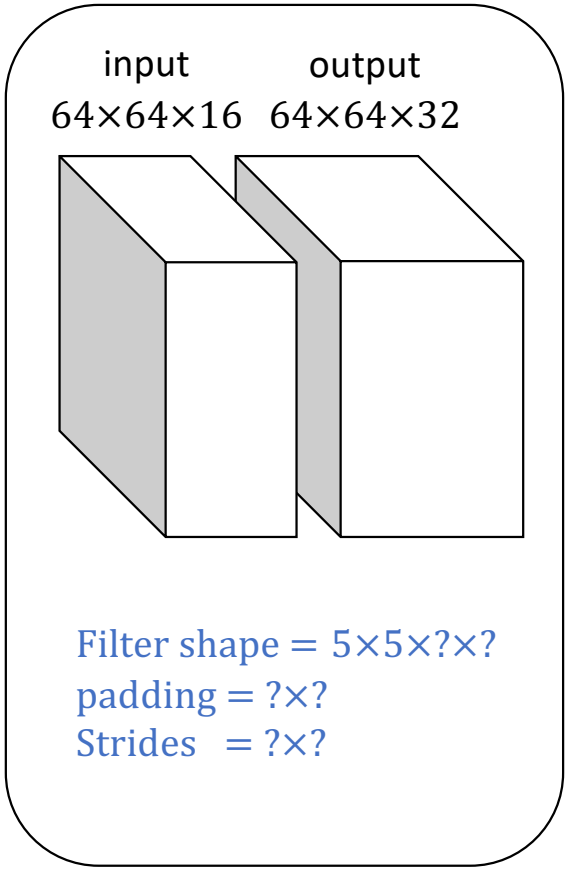
Rely on input size, padding, strides, and filter size



Convolutional Algorithms



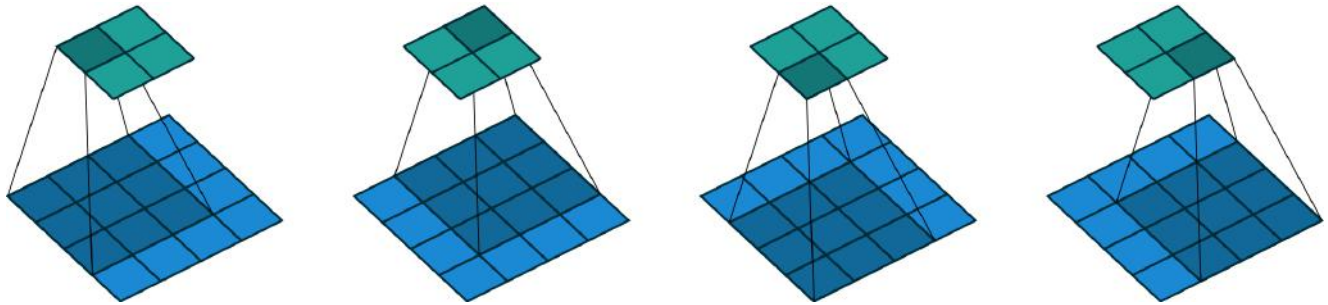




Convolutional Algorithms

- Receptive field

Receptive field $= 3\times3$



Convolutional Algorithms

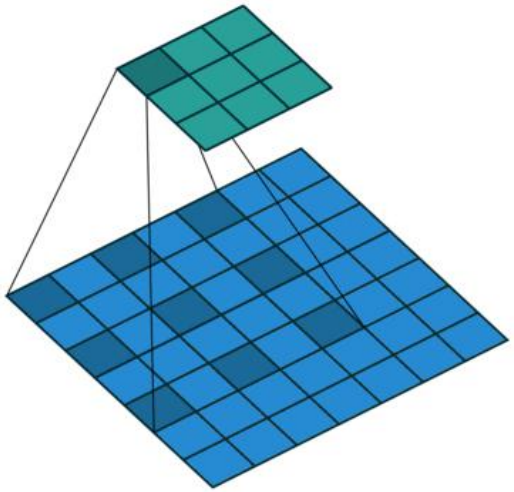
- Receptive field

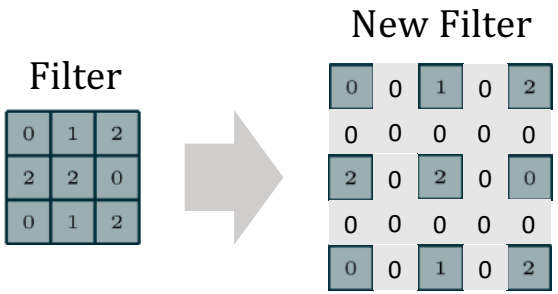
Layer #	Kernel Size	Stride	Dilation	Padding	Input Size	Output Size	Receptive Field
1	3	1	1	2	256	256	3
2	3	2	1	1	256	128	5
3	3	1	1	2	128	128	9
4	3	1	1	2	128	128	13
5	3	2	1	1	128	64	17
6	3	1	1	2	64	64	25
7	3	1	1	2	64	64	33
8	3	2	1	1	64	32	41

https://fomoro.com/research/article/receptive-field-calculator

Convolutional Algorithms

- Dilated convolution: Larger Receptive Filed





Dilation rate = 2×2

Receptive field : 3×3 → 5×5

Convolutional Algorithms

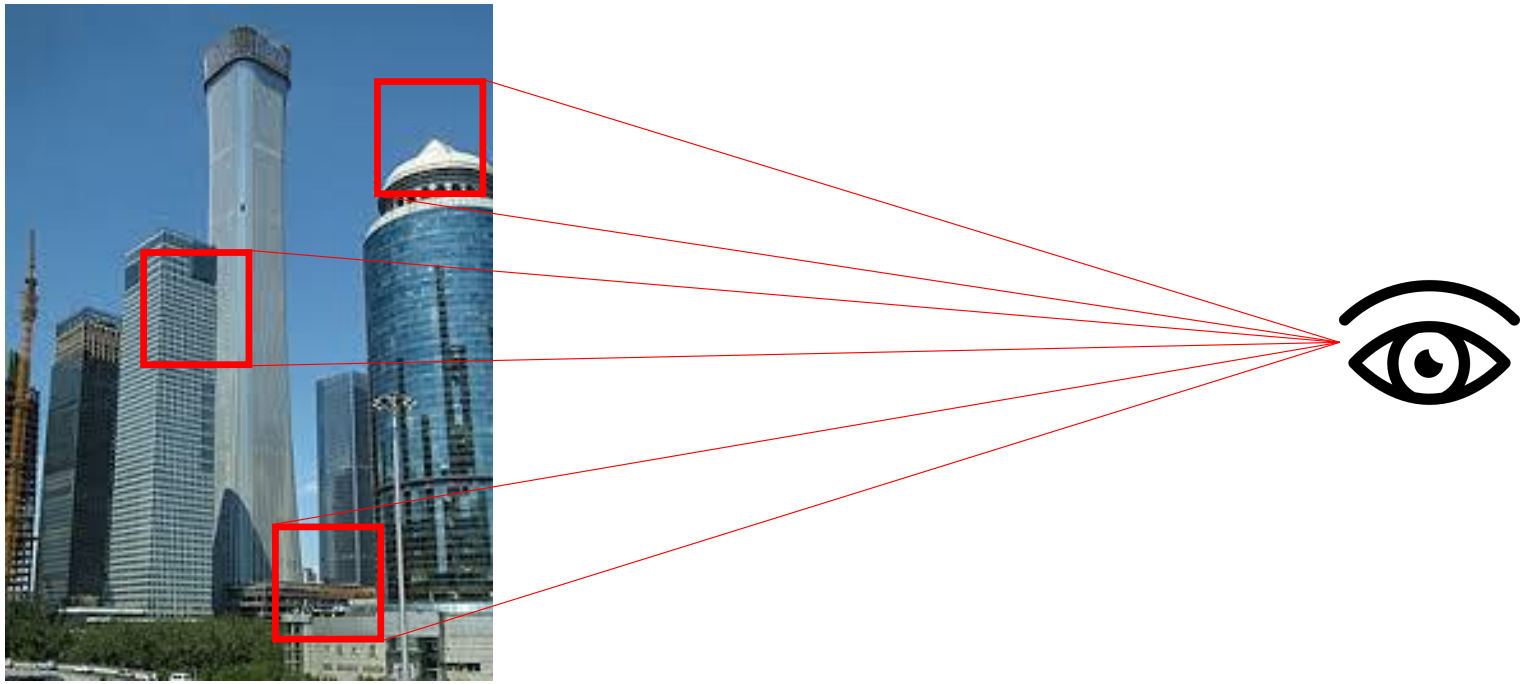
- Convolution on 3D volume
  - Filter (kernel) shapes
    - To convolute MRI images, a filter have 1 channels
    - The filter shape is (filter_depth × filter_height × filter_width × input_channels × n_filters)

  - Number of filters (channels)
    - One filter can have one corresponding output feature volume
    - The number of output channels == The number of filters
    - The output shape is (feature_depth × feature_height × feature_width × n_filters)

Pooling Algorithms

Pooling Algorithms

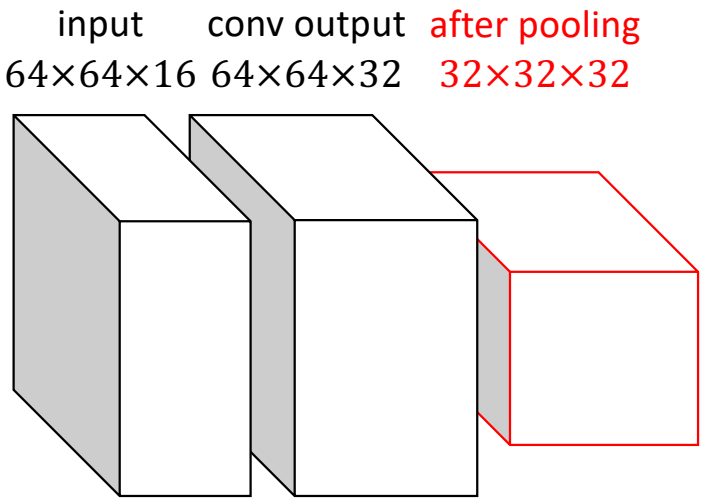
- Motivation: Shift Invariant



- Small shifts should not effect the result

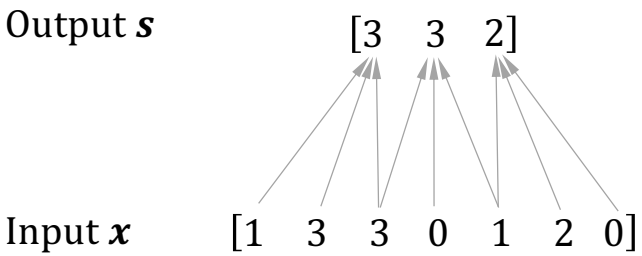
Pooling Algorithms

- Motivation: Features aggregation



Pooling Algorithms

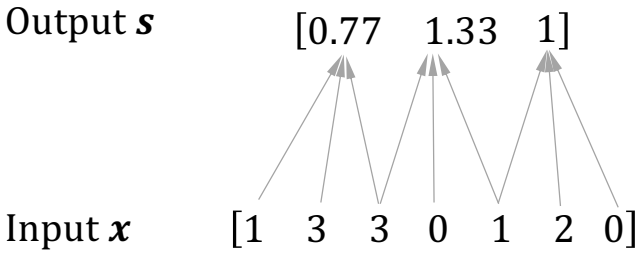
- Pooling on 1D vector
  - Max Pooling



Filter size = 3
Padding = 0
Stride = 2

  - Mean (Average) Pooling

Receptive field is increased

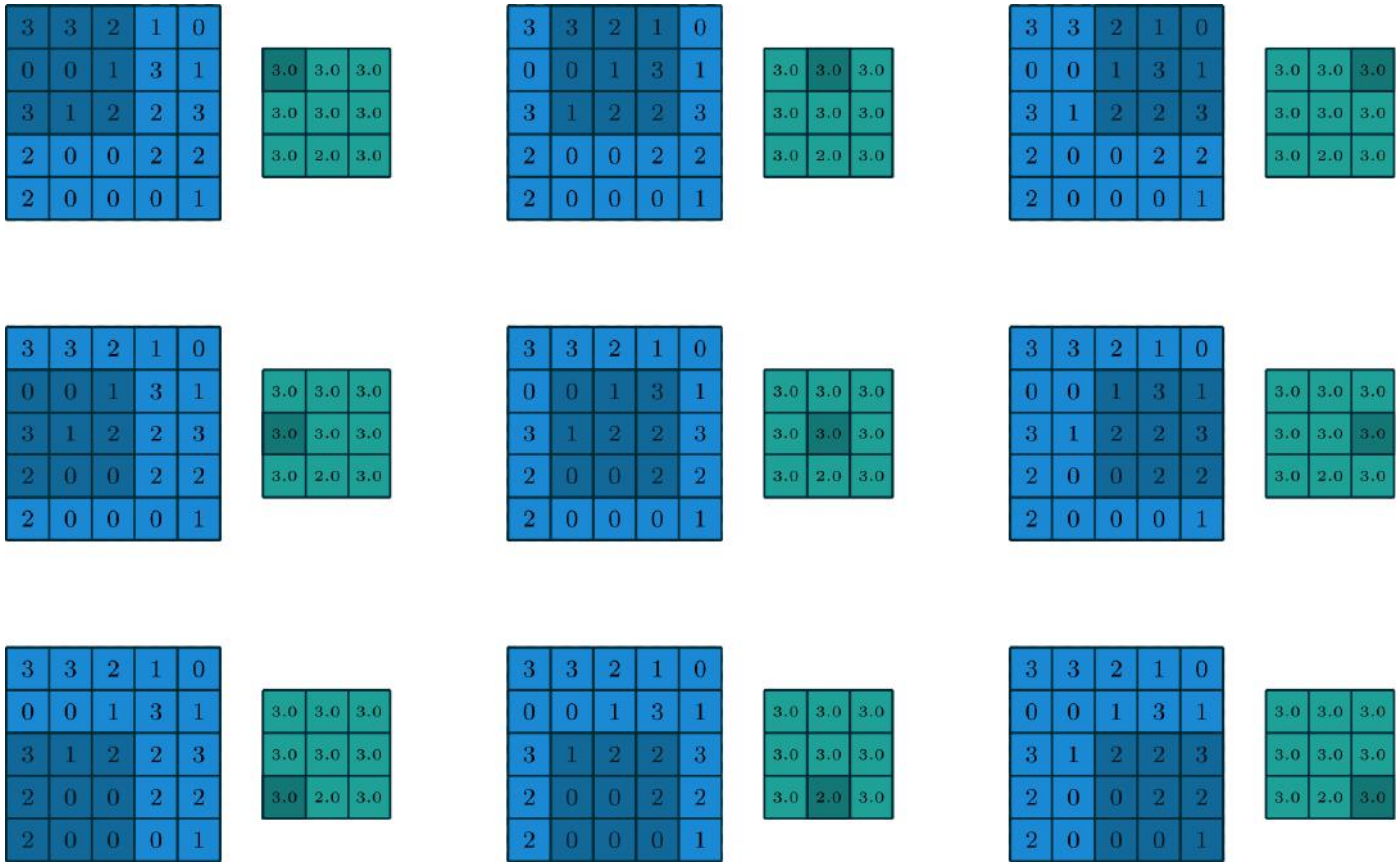


Filter size = 3
Padding = 0
Stride = 2

Pooling Algorithms

- Max Pooling on 2D matrix

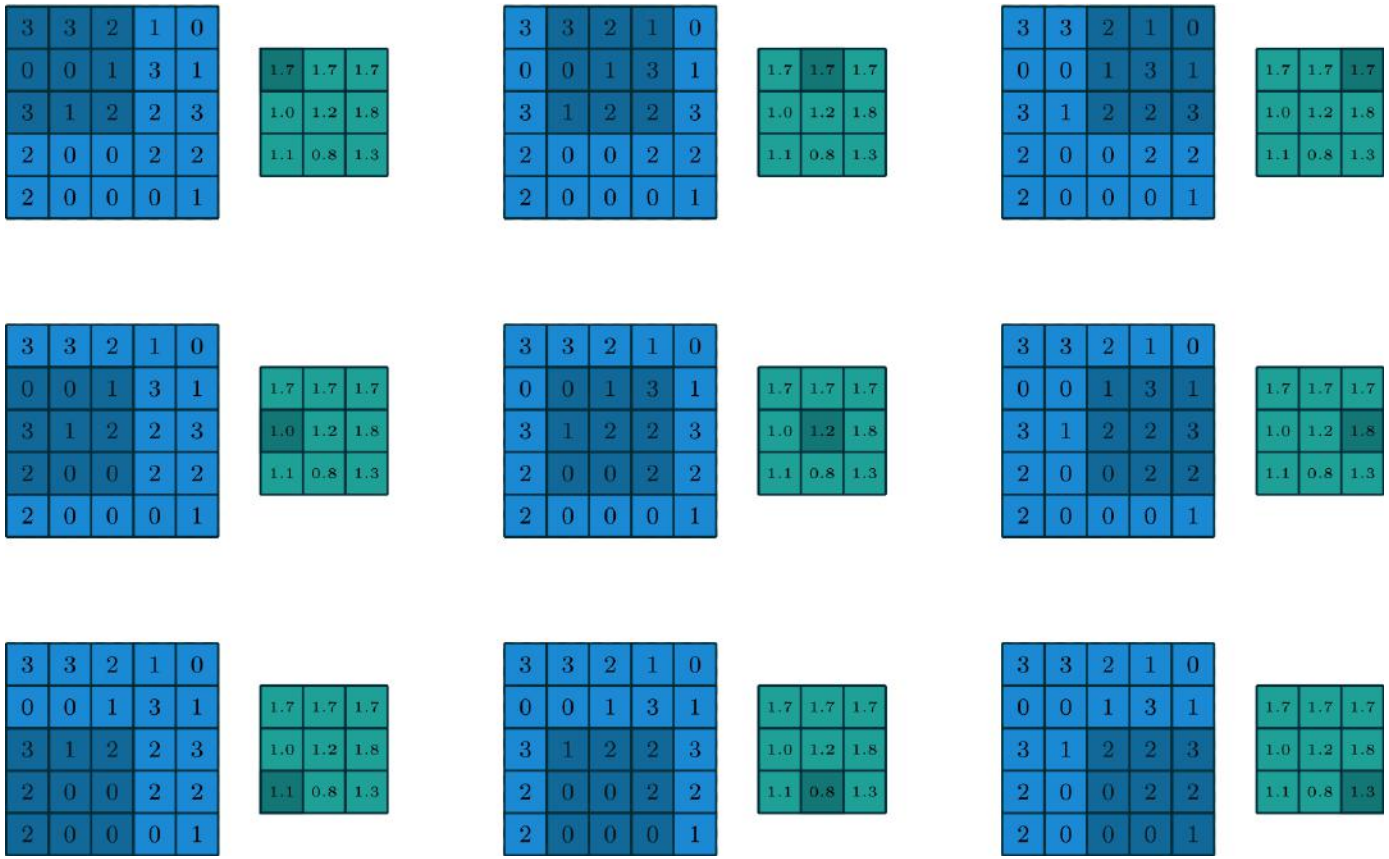
Filter size = 3×3
padding = 0×0
Strides = 1×1



Pooling Algorithms

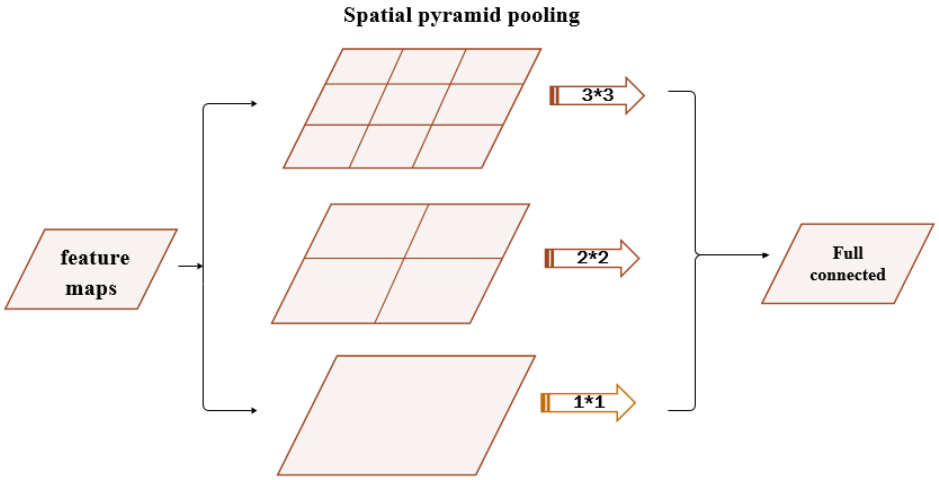
- Mean Pooling on 2D matrix

Filter size = 3×3
padding = 0×0
Strides = 1×1



Pooling Algorithms

- Spatial Pyramid Pooling



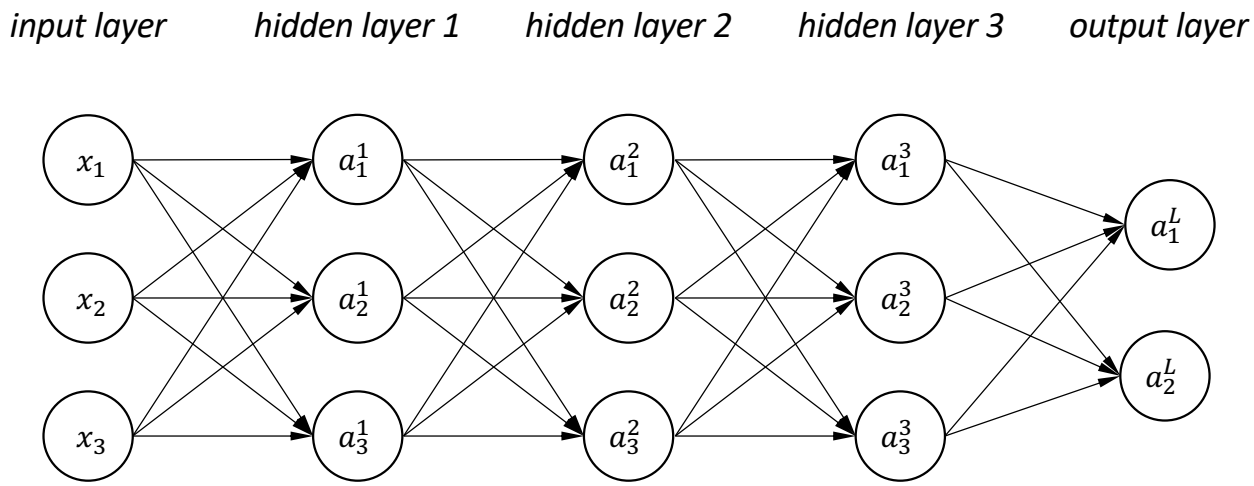
Pooling Algorithms

- Pooling on 3D volume
  - ….

Hierarchical Representation Learning

Hierarchical Representation Learning

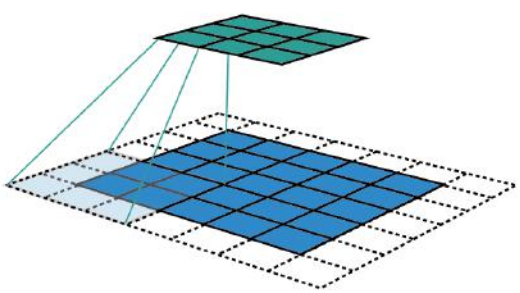
- Motivation

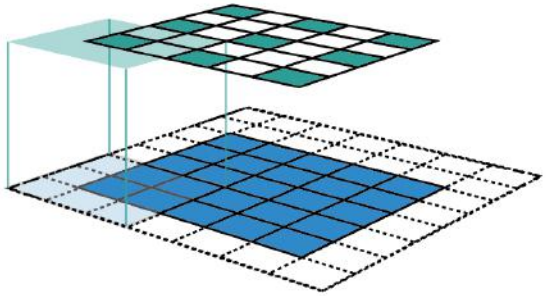


Hierarchical Representation Learning

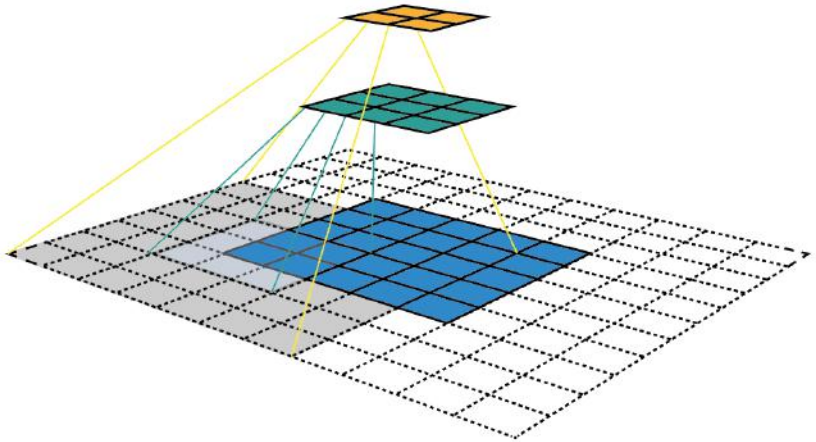
- Large receptive field

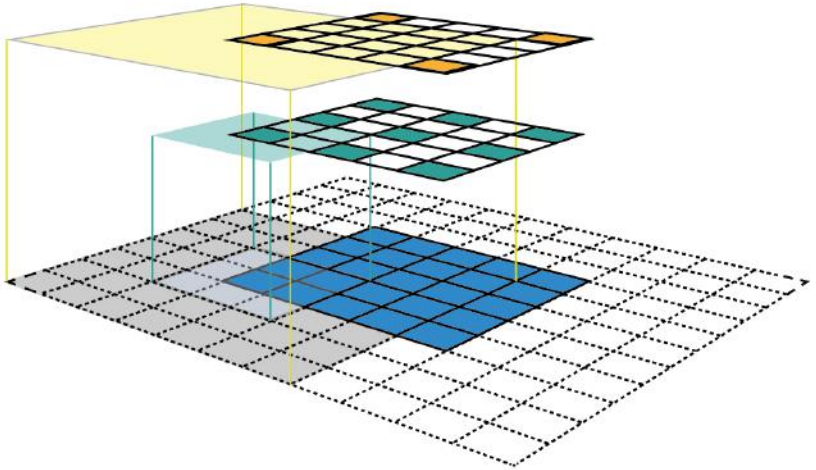
1 layer





2 layers

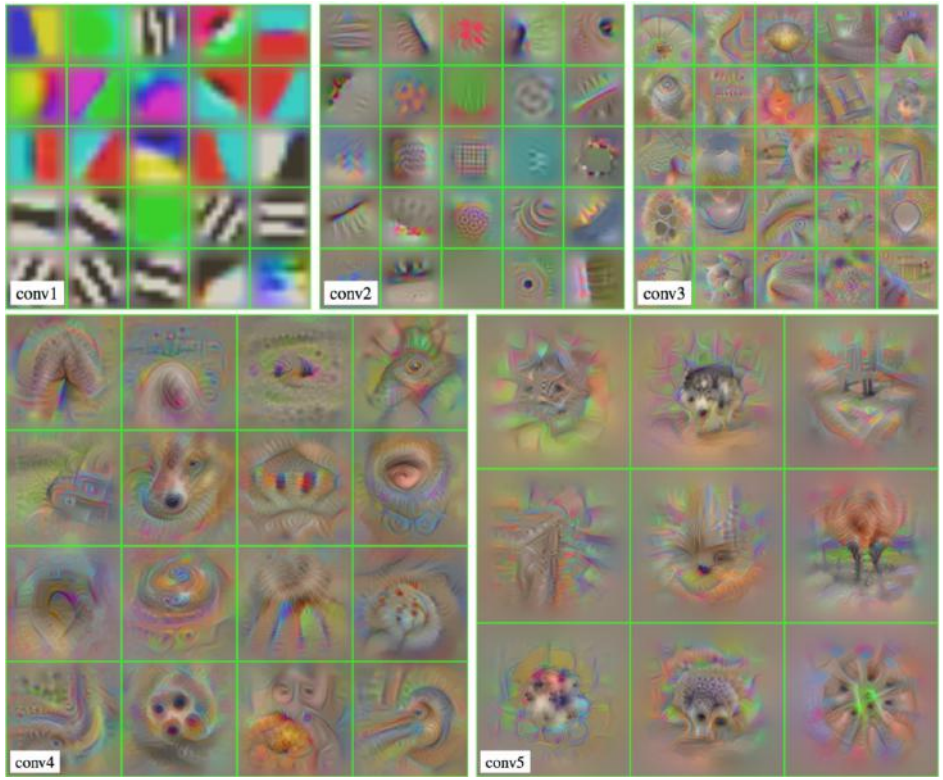




Hierarchical Representation Learning

- Visualizing the activation outputs

Activation maximization of the first filters of each convolutional layer in a well-known CNN architecture called VGG. The notations of “conv1” through “conv5” distinct hidden layers, where a larger number represents a deeper hidden layer.



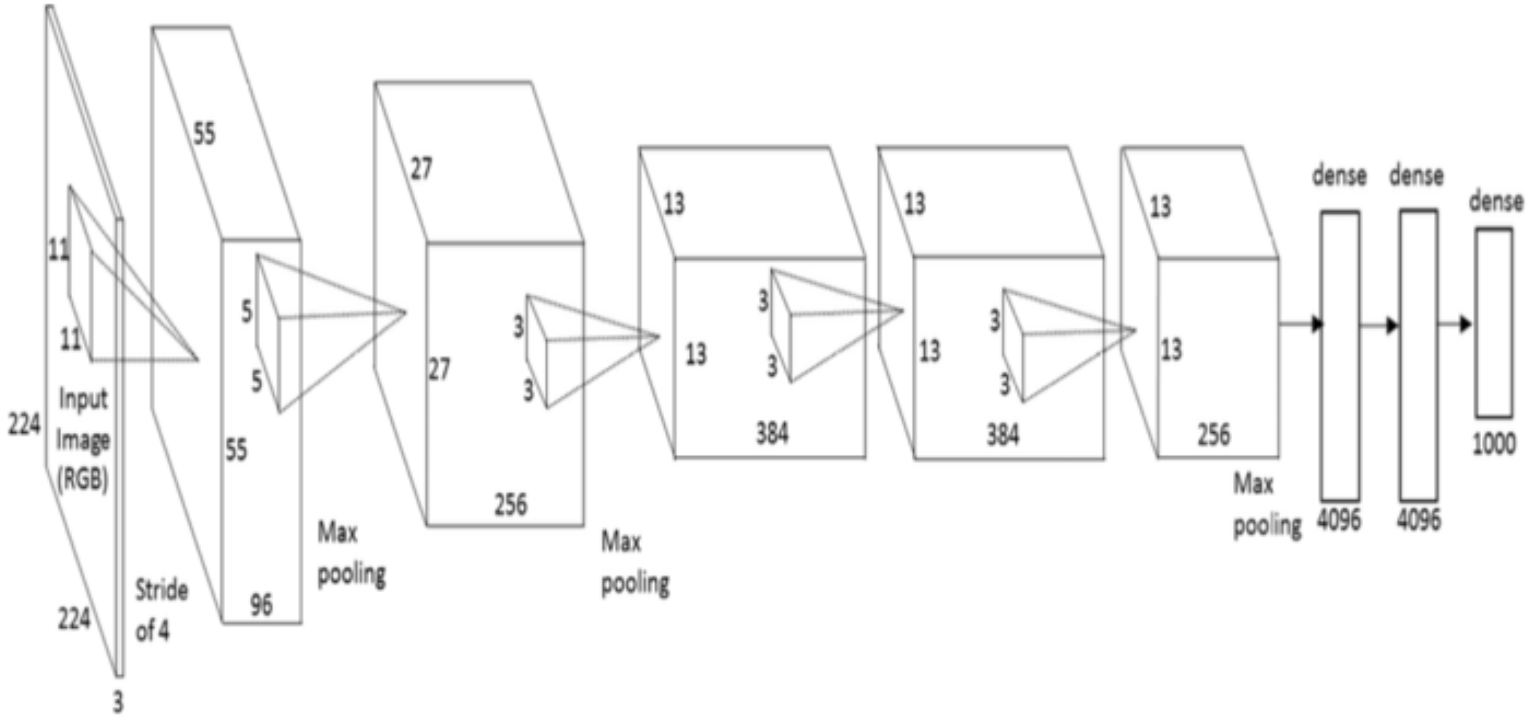
A. Mahendran and A. Vedaldi, “Visualizing deep convolutional neural networks using natural pre-images,” International Journal of Computer Vision (IJCV), vol. 120, no. 3, pp. 233–255, 2016.

北京大学
PEKING UNIVERSITY

Convolutional Architectures

Convolutional Encoding Architectures

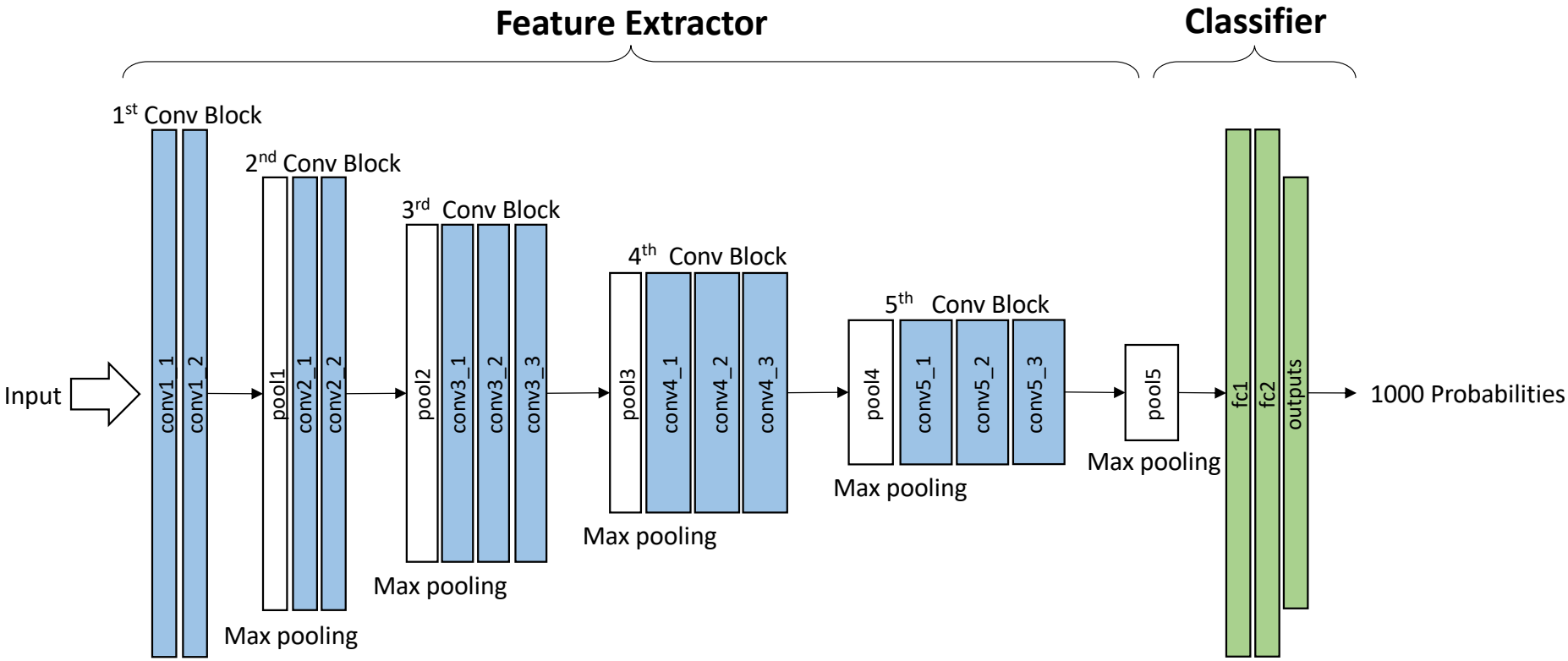
- AlexNet: make the history



In 2012, AlexNet achieves 10% performance gain on ImageNet compared to its previous methods

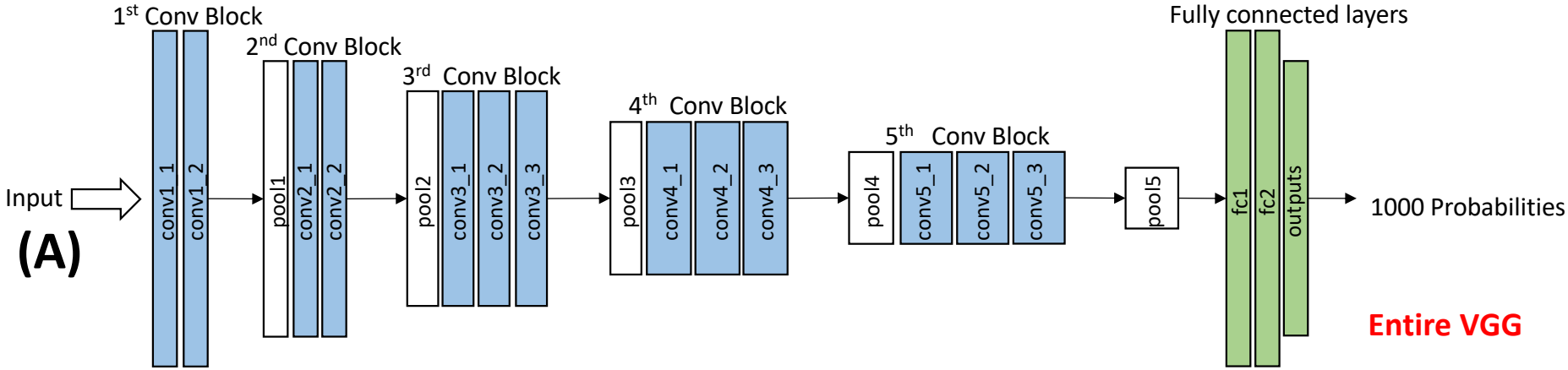
Convolutional Encoding Architectures

- VGG16



Convolutional Encoding Architectures

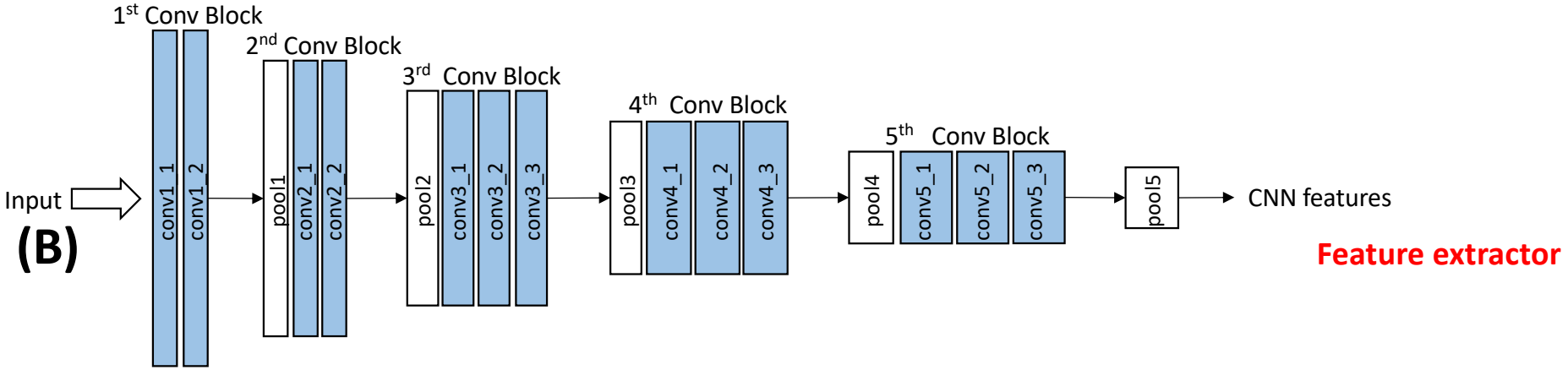
- VGG16




```
>>> # get the whole model, without pre-trained VGG parameters
>>> vgg = tl.models.vgg16()
>>> # get the whole model, restore pre-trained VGG parameters
>>> vgg = tl.models.vgg16(pretrained=True)
>>> # use for inferencing
>>> output = vgg(image, is_train=False)
>>> probs = tf.nn.softmax(output)[0].numpy()
```

Convolutional Encoding Architectures

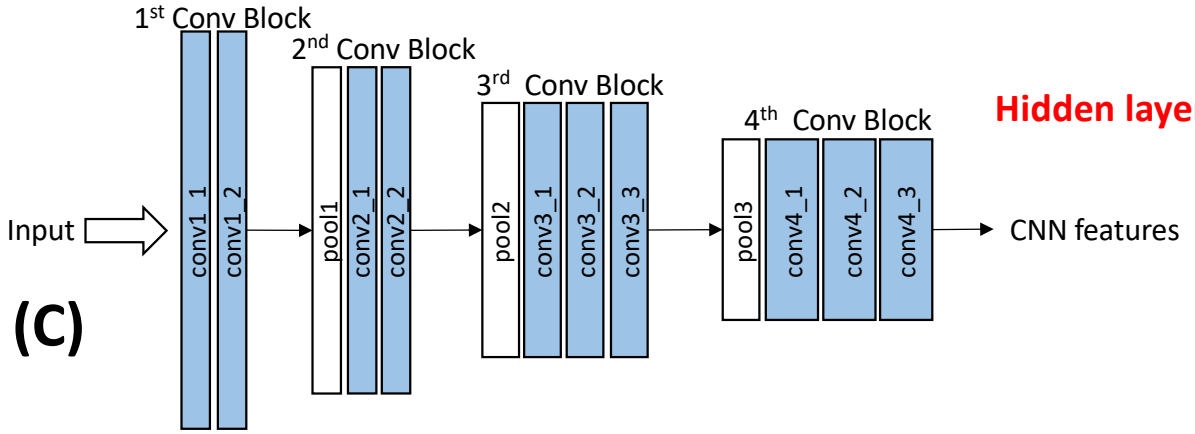
- VGG16




```
>>> vgg = tl.models.vgg16(pretrained=True, end_with='pool5', mode='static')
```

Convolutional Encoding Architectures

- VGG16



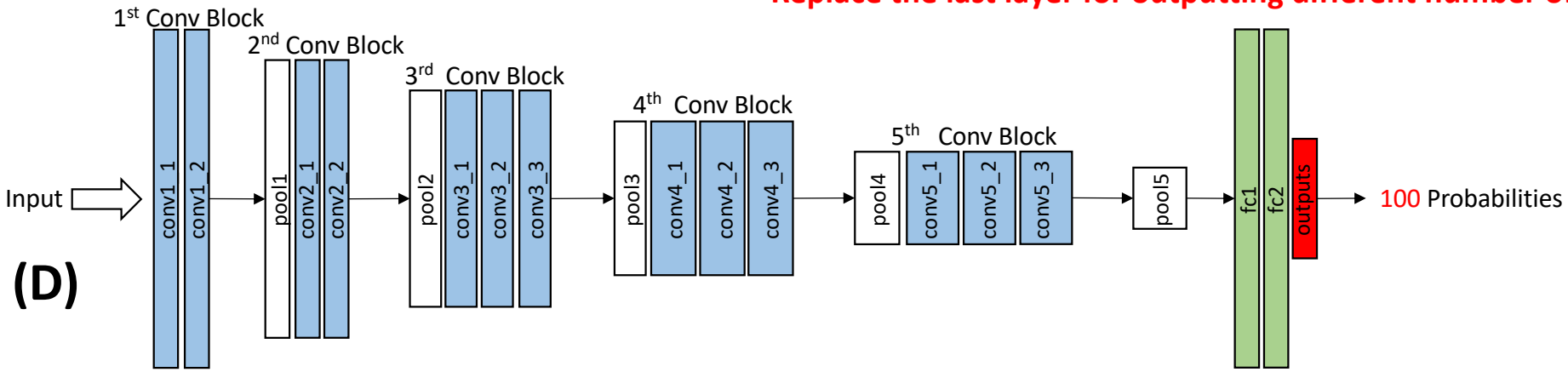
Hidden layer of feature extractor : lower level features

```
>>> vgg = tl.models.vgg16(pretrained=True, end_with='conv4_3', mode='static')
```

Convolutional Encoding Architectures

- VGG16

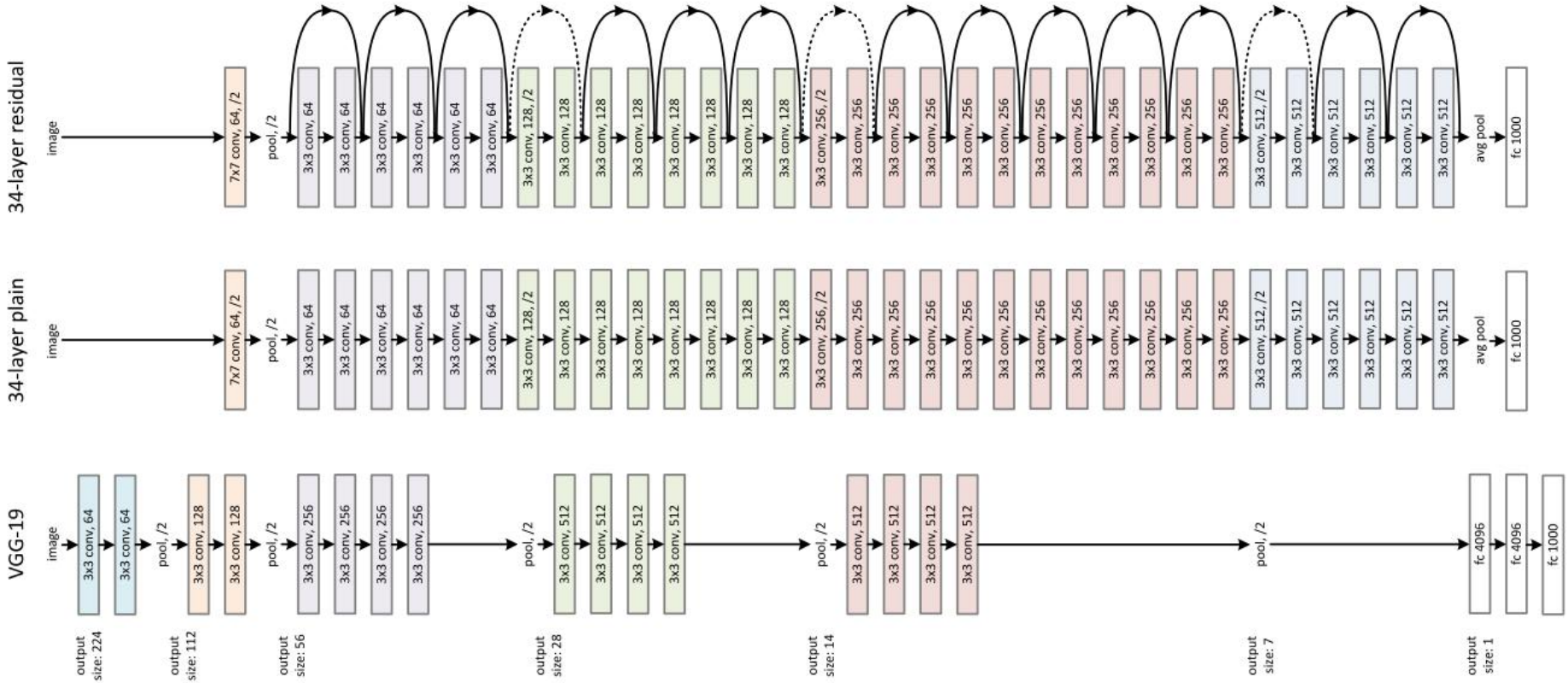
Replace the last layer for outputting different number of classes




```
>>> # get VGG without the last layer
>>> cnn = tl.models.vgg16(end_with='fc2_relu', mode='static').as_layer()
>>> # add one more layer and build a new model
>>> ni = Input([None, 224, 224, 3], name="inputs")
>>> nn = cnn(ni)
>>> nn = tl.layers.Dense(n_units=100, name='out')(nn)
>>> model = tl.models.Model(inputs=ni, outputs=nn)
>>> # train your own classifier (only update the last layer)
>>> train_weights = model.get_layer('out').trainable_weights
```

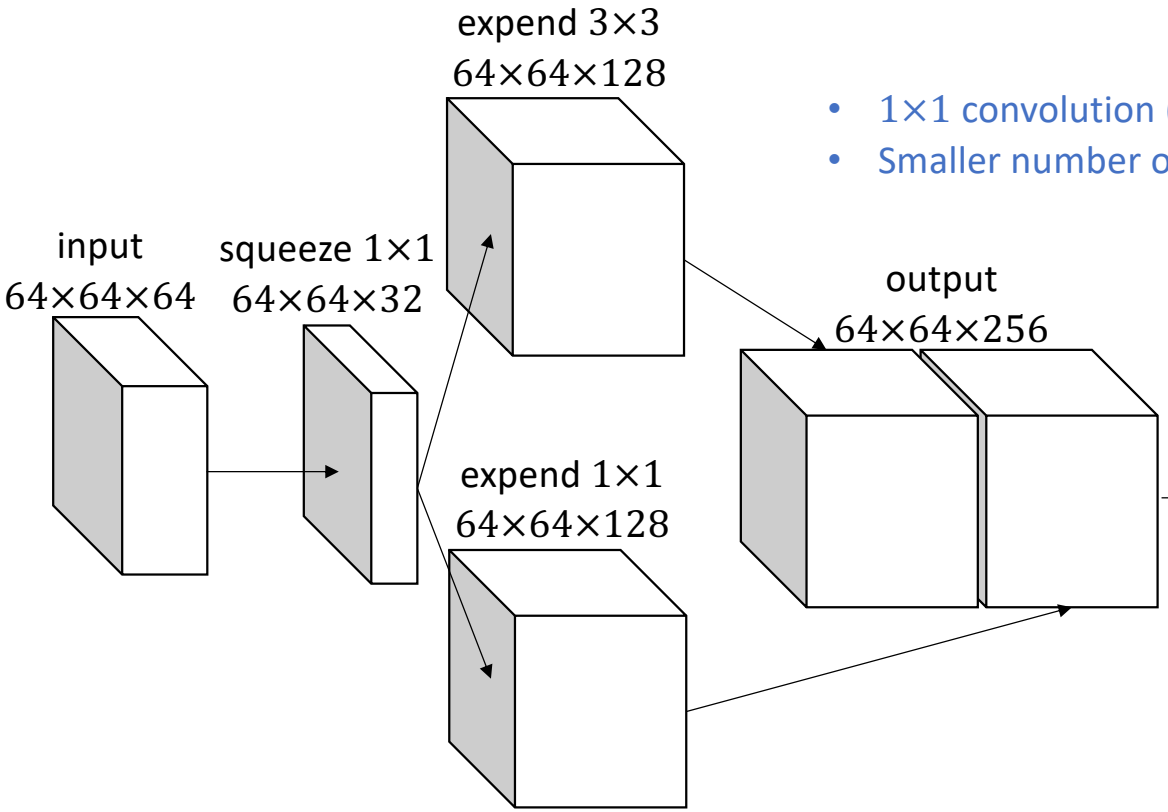
Convolutional Encoding Architectures

- ResNet



Convolutional Encoding Architectures

- SqueezeNet, MobileNet, ShuffleNet



An example of "fire" block

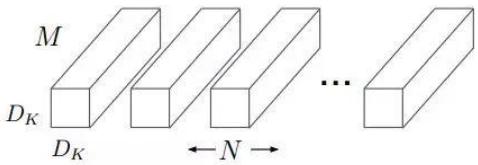
- 1×1 convolution (pointwise conv) is 9 times faster than 3×3 convolution
- Smaller number of channels == Less computation

```
import time
import numpy as np
import tensorflow as tf
import tensorlayer as tl
from tensorlayer.models.imagenet_classes import class_names

squeezenet = tl.models.SqueezeNetV1(pretrained=True)

img1 = tl.vis.read_image('data/tiger.jpeg')
img1 = tl.prepro.imresize(img1, (224, 224)) / 255
img1 = img1.astype(np.float32)[np.newaxis, ...]

start_time = time.time()
output = squeezenet(img1, is_train=False)
prob = tf.nn.softmax(output)[0].numpy()
print("  End time : %.5ss" % (time.time() - start_time))
preds = (np.argsort(prob)[::-1])[0:5]
for p in preds:
    print(class_names[p], prob[p])
```

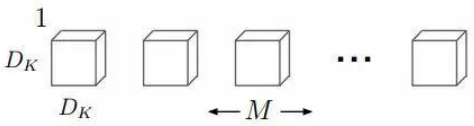
Very Fast even on CPU

Convolutional Encoding Architectures

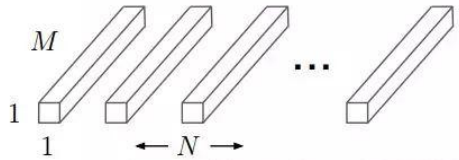
- SqueezeNet, **MobileNet**, ShuffleNet



(a) Standard Convolution Filters



(b) Depthwise Convolutional Filters



(c) 1 × 1 Convolutional Filters called Pointwise Convolution in the con-

The number of multiplications of a **standard 2D CNN** layer without strides and padding:

- $filter\_size\_height \times filter\_size\_width \times height \times width \times in\_channels \times output\_channels$

The number of multiplications of a **depthwise 2D CNN** layer without strides and padding:

- $filter\_size\_height \times filter\_size\_width \times height \times width \times in\_channels$

The number of multiplications of a **pointwise 2D CNN** layer without strides and padding:

- $height \times width \times in\_channels \times output\_channels$

Convolutional Encoding Architectures

- SqueezeNet, MobileNet, ShuffleNet

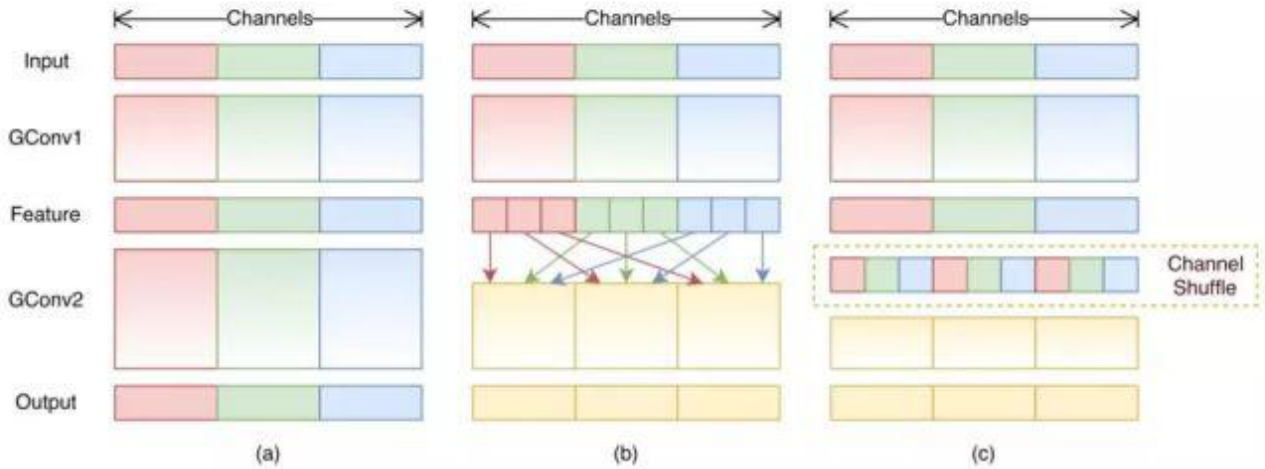


Figure 1. Channel shuffle with two stacked group convolutions. GConv stands for group convolution. a) two stacked convolution layers with the same number of groups. Each output channel only relates to the input channels within the group. No cross talk; b) input and output channels are fully related when GConv2 takes data from different groups after GConv1; c) an equivalent implementation to b) using channel shuffle.

Group Convolution: Split the channels into several "ground" and perform standard convolution using different CNN layers.

Shuffle layer: Merge the information of different group by shuffling the channels.

Convolutional Encoding Architectures

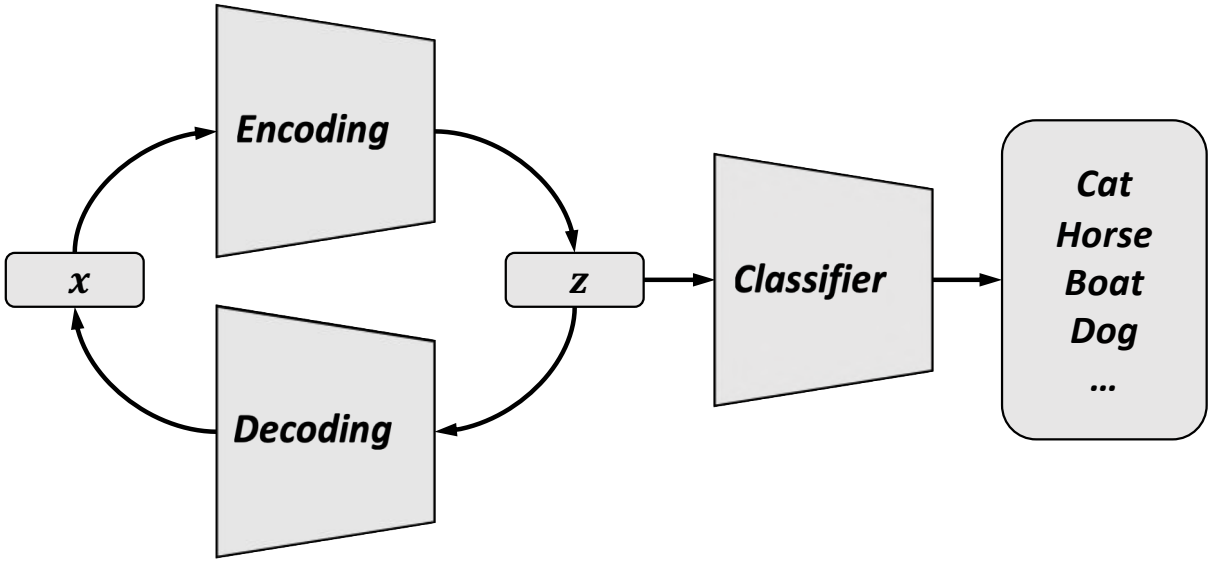
- SqueezeNet++
- MobileNetV2
- MobileNetV3
- ShuffleNetV2
- ...

Transposed Convolutional Algorithms

Transposed Convolutional Algorithms

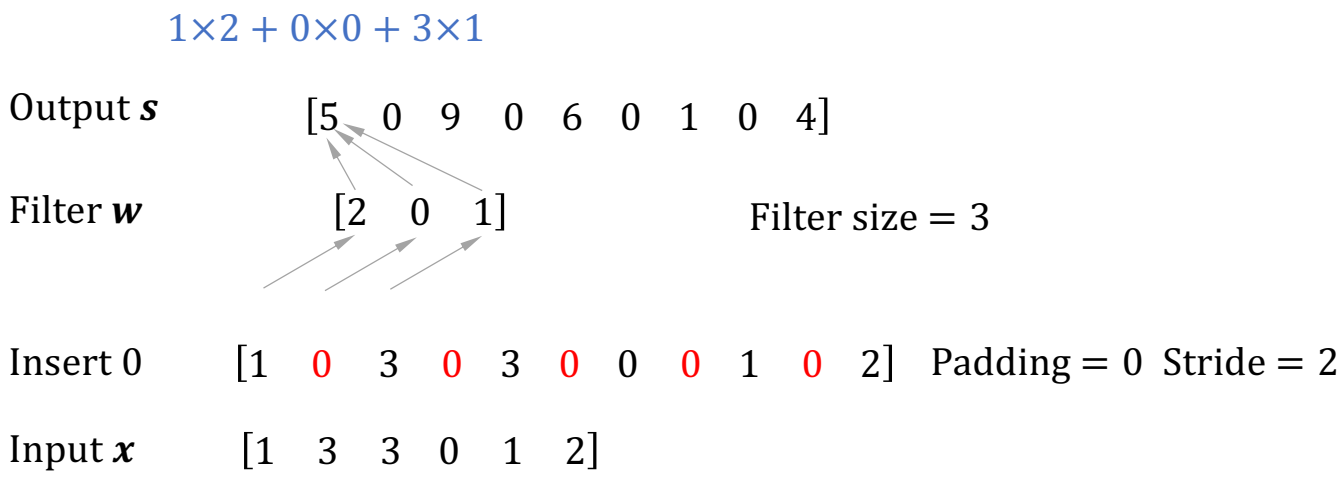
- Motivation

Convolution and pooling can only keep or reduce the size of the feature maps



Transposed Convolutional Algorithms

- Transposed convolution on 1D vector

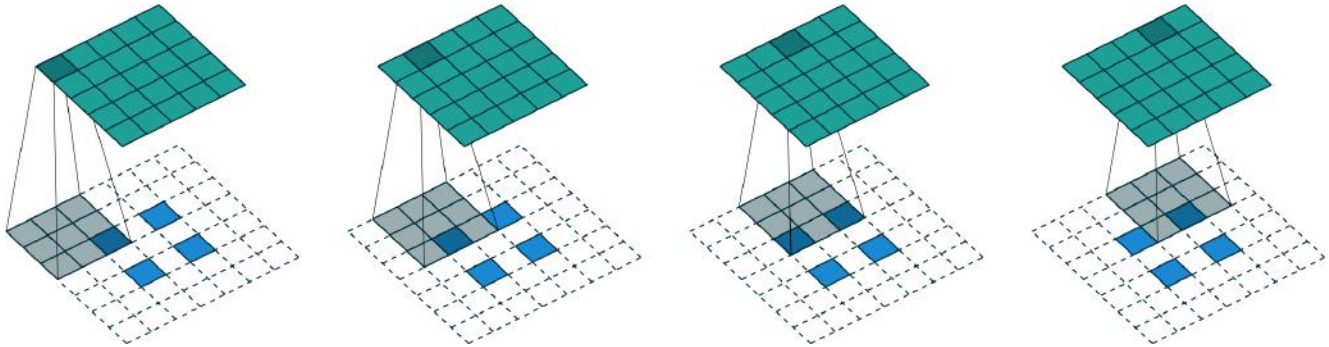


Transposed Convolutional Algorithms

- Inserting zeros between inputs

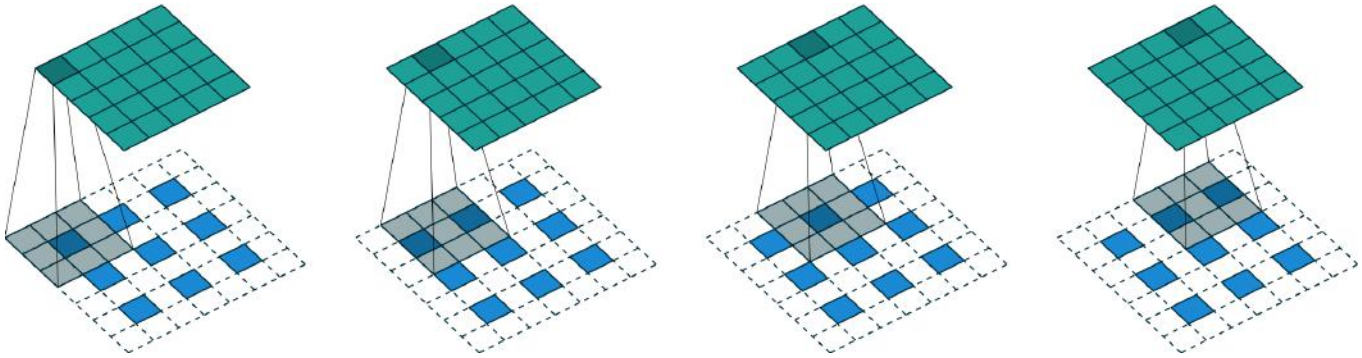
Padding , strides and transposed

Padding = 2×2
Strides = 2×2



Padding, strides and transposed

Padding = 1×1
Strides = 2×2



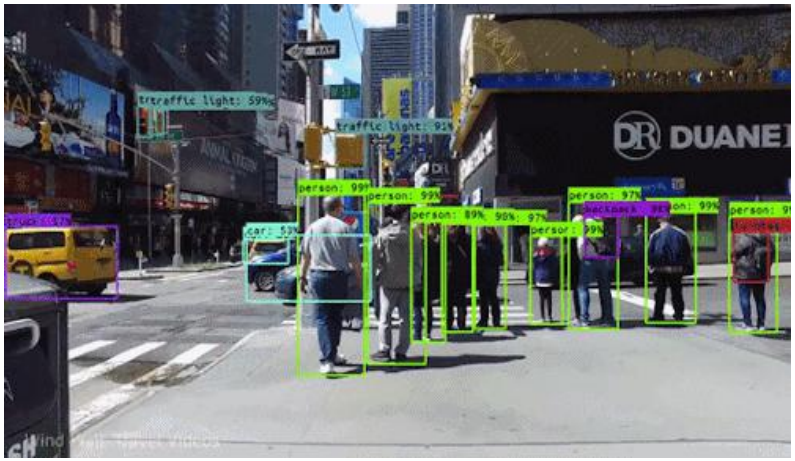
Transposed Convolutional Algorithms

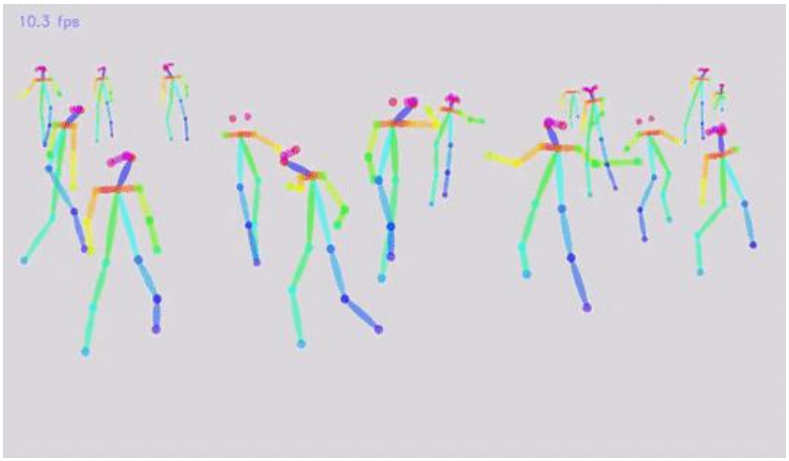
- Resize convolution
- Subpixel convolution
- …

北京大学
PEKING UNIVERSITY

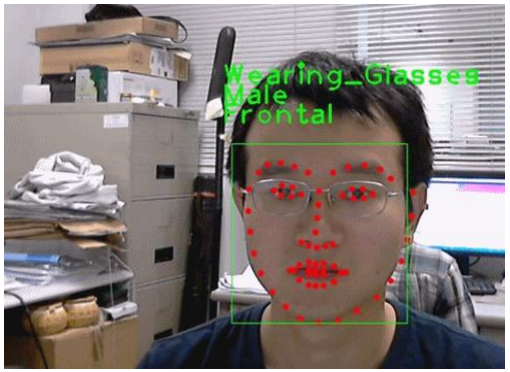
Computer Vision Applications

Computer Vision Applications

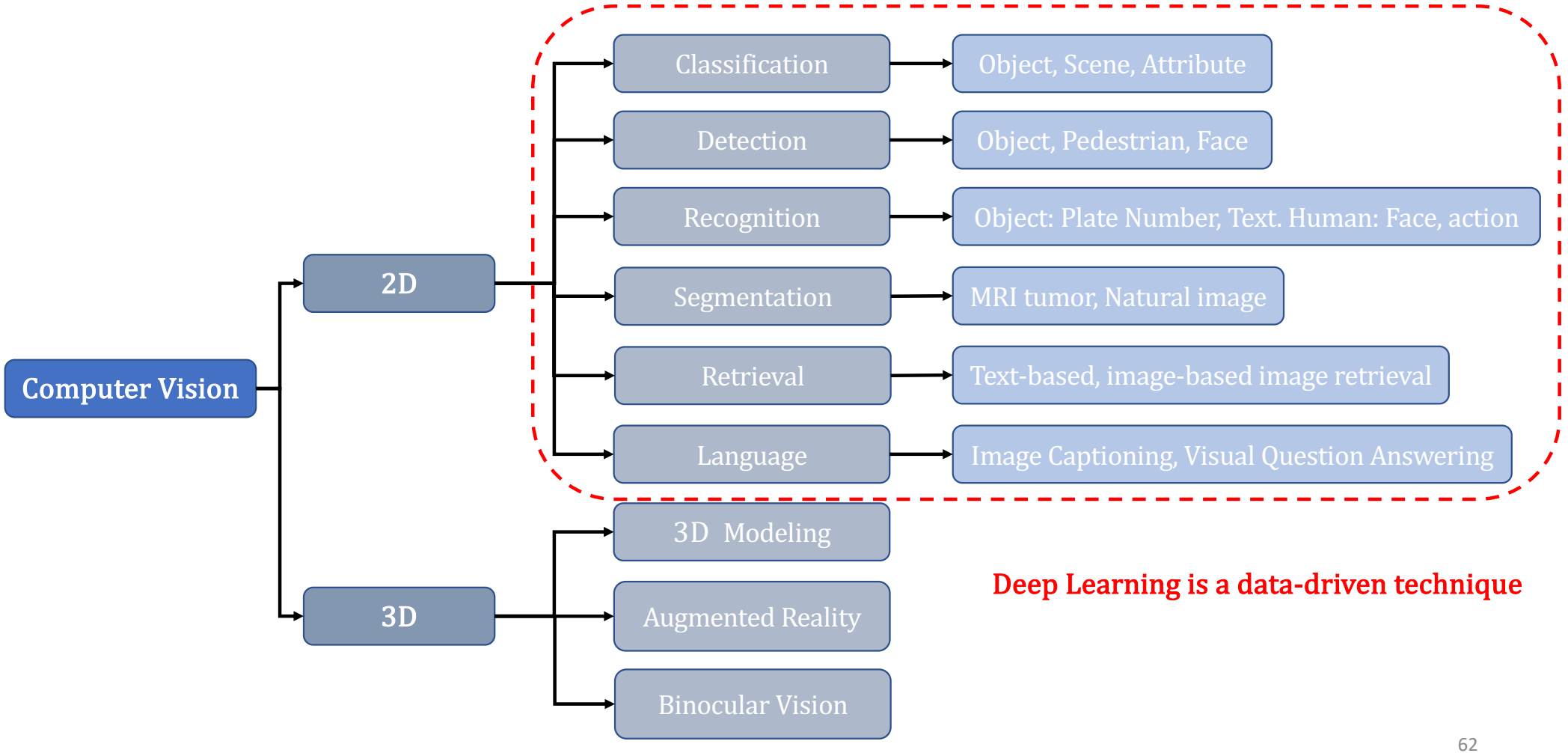




RGB



Computer Vision Applications



Deep Learning is a data-driven technique

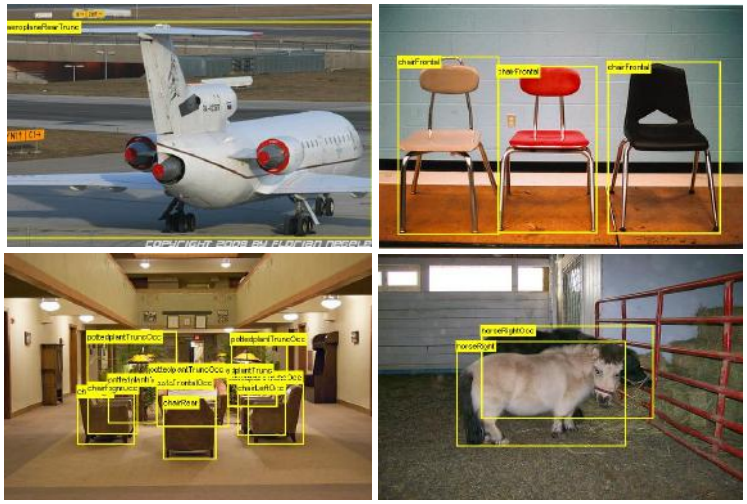
Computer Vision Applications

- Object Detection: Dataset

Oxford Pascal VOC

- VOC 2007 and 2012

- 20 Classes：
  - Person
  - Animal: Bird, Cat, Cow, Dog, Horse, Sheep
  - Vehicle: Aeroplane, Bicycle, Boat, Bus, Car, Motorbike, Train
  - Indoor: Bottle, Chair, Dining Table, Potted Plant, Sofa, TV/Monitor



- VOC 2012: 11,530 images (train/val). 27,450 bounding boxes

http://host.robots.ox.ac.uk/pascal/VOC/voc2012/

Computer Vision Applications

- Object Detection: Dataset

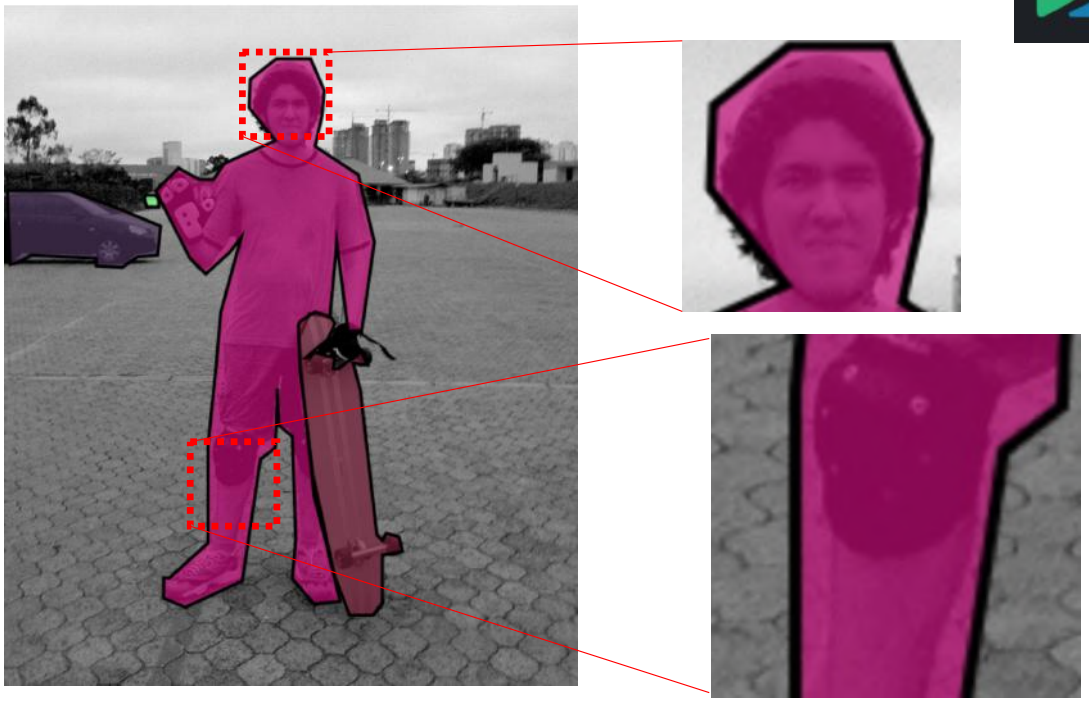
Microsoft COCO

- 80 classes
- COCO 2014, 82,783 training images, 40,504 validating images, and 40,775 testing images
- Apart from object detection, it also have the annotations of:
  - Image captioning
  - Pose estimation
  - Image segmentation

http://cocodataset.org/#explore

Computer Vision Applications

- Object Detection: Dataset

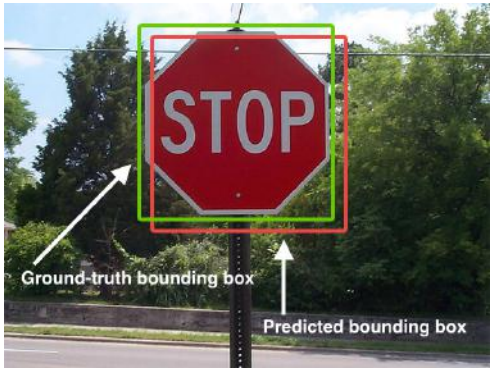


Sometime the dataset is “bad”

Computer Vision Applications

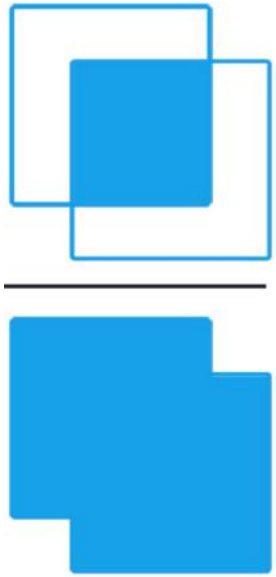
- Object Detection: Evaluation

Intersection over Union (IoU)



$$IoU = \frac{Area\ of\ Intersection}{Area\ of\ Union}$$

=



- IoU is a scalar between 0 ~ 1
- If IoU==1, the predicted and GT boxes are fully matched.
- If IoU==0, the predicted and GT boxes are fully mismatched.
- If IoU>threshold, object detected.

Computer Vision Applications

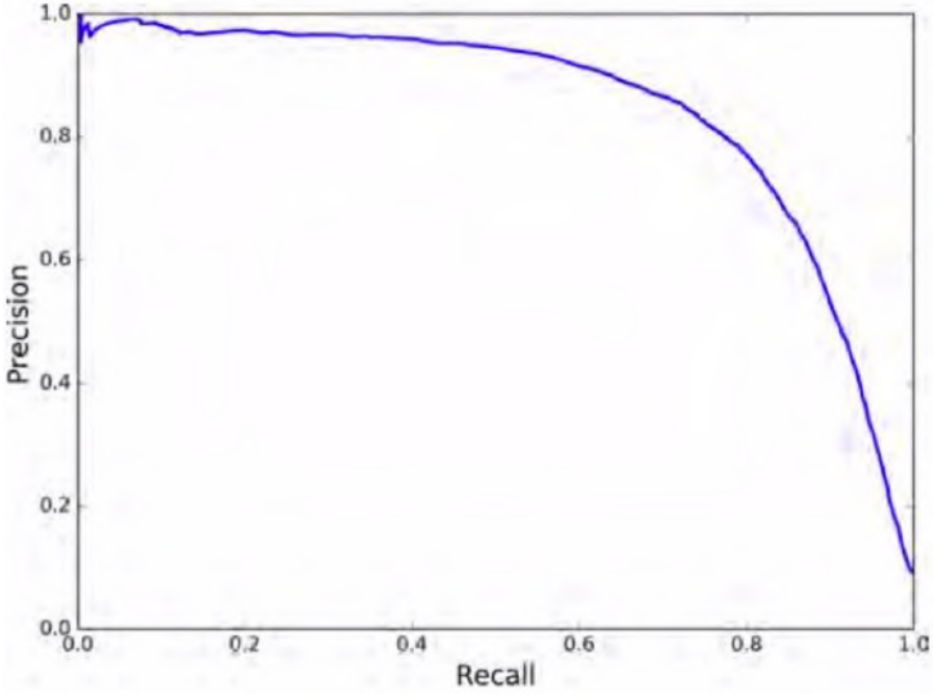
- Object Detection: Evaluation

Average Precision (AP)

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

- The area under P-R curve is the Average Precision (AP)



Computer Vision Applications

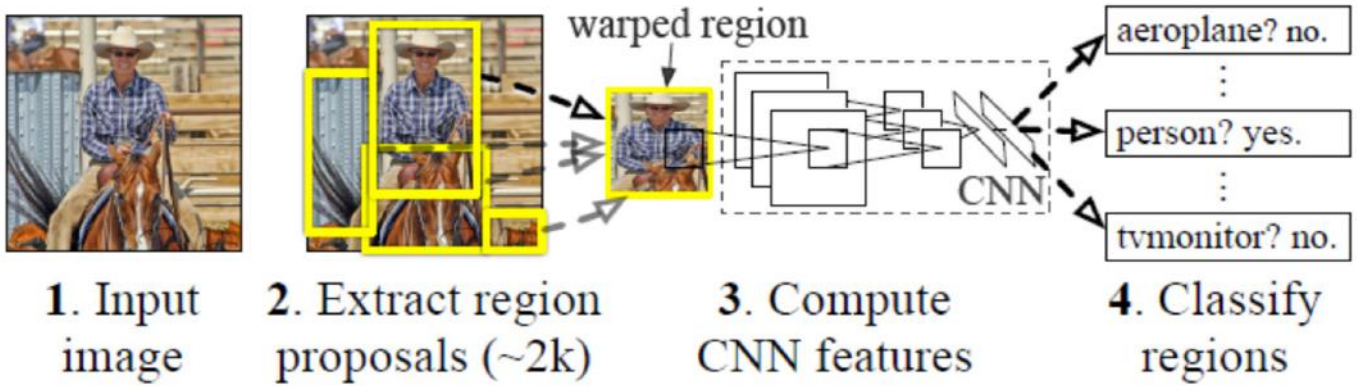
- Object Detection: Evaluation

- The mAP is the averaged AP for each class
- To further evaluate the algorithm, we would compute AP using different settings, such as the IoU threshold and the size of object.

```
Average Precision (AP):
  AP                  % AP at IoU=.50:.05:.95 (primary challenge metric)
  AP^IoU=.50          % AP at IoU=.50 (PASCAL VOC metric)
  AP^IoU=.75          % AP at IoU=.75 (strict metric)
AP Across Scales:
  AP^small            % AP for small objects: area < 32^2
  AP^medium           % AP for medium objects: 32^2 < area < 96^2
  AP^large            % AP for large objects: area > 96^2
Average Recall (AR):
  AR^max=1            % AR given 1 detection per image
  AR^max=10           % AR given 10 detections per image
  AR^max=100          % AR given 100 detections per image
```

Computer Vision Applications

- Object Detection: R-CNN

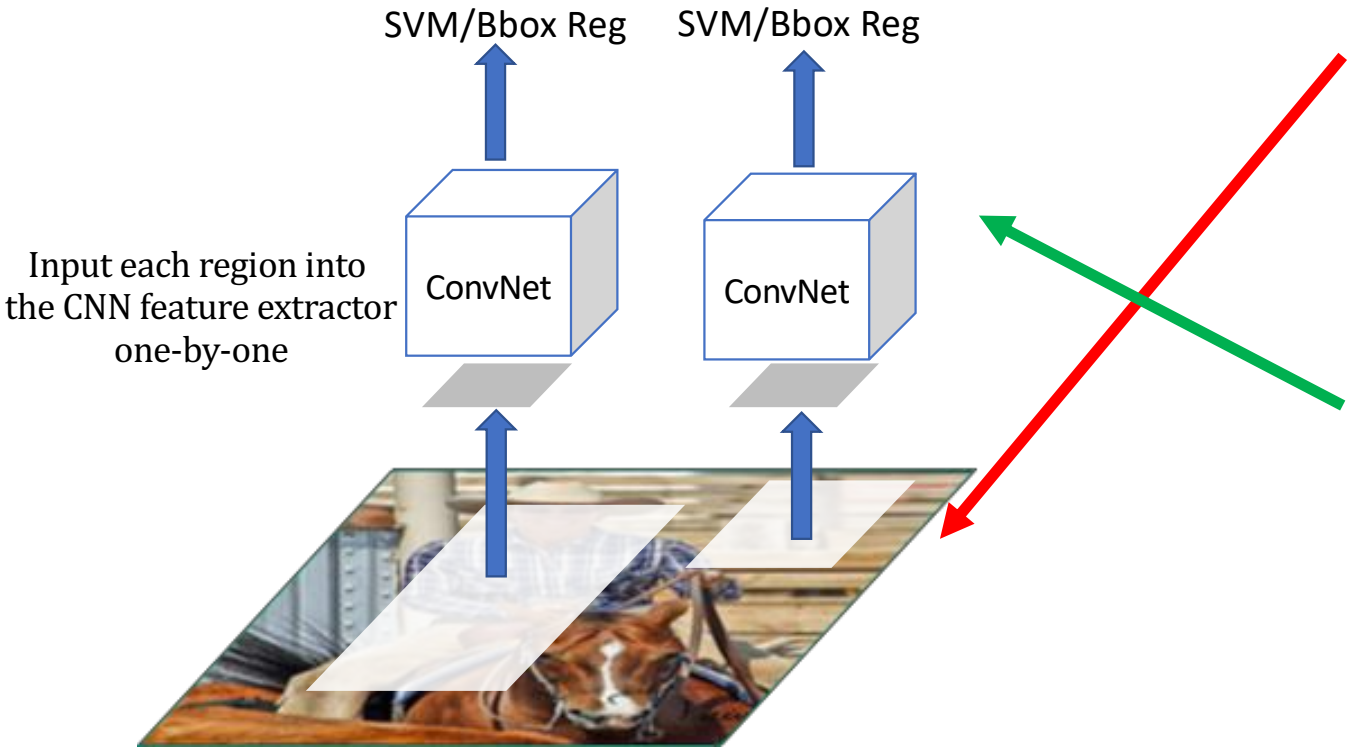


- Step 1: Use Selective Search algorithm to obtain 2,000 proposal regions from an image.
- Step 2: Resize all regions to a given fixed size.
- Step 3: Feed each regions to the VGG to extract image features, and classify the image via the features.
- Step 4: Obtain the bounding box location via regression.

Rich feature hierarchies for accurate object detection and semantic segmentation. *Girshick, Ross etal. CVPR. 2014.*

Computer Vision Applications

- Object Detection: R-CNN

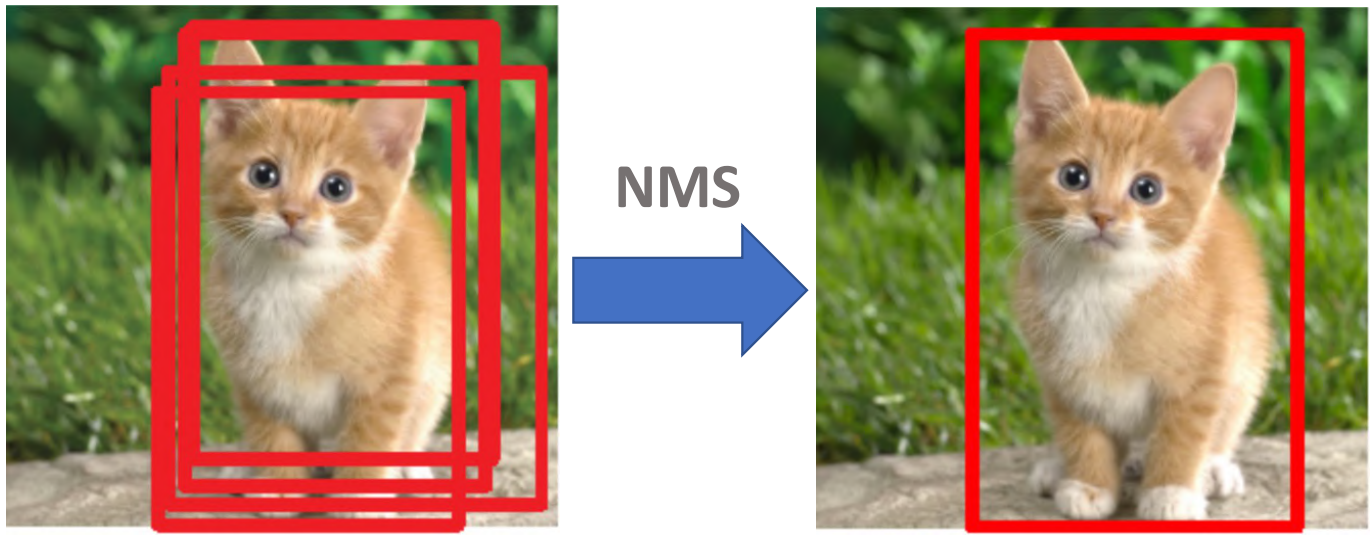


- Step 1: Use Selective Search algorithm to obtain 2,000 proposal regions from an image.
- Step 2: Resize all regions to a given fixed size.
- Step 3: Feed each region to the VGG to extract image features, and classify the image via the features.
- Step 4: Obtain the bounding box location via regression.

Rich feature hierarchies for accurate object detection and semantic segmentation. *Girshick, Ross etal. CVPR. 2014.*

Computer Vision Applications

- Object Detection: Non-Maximum Suppression, NMS



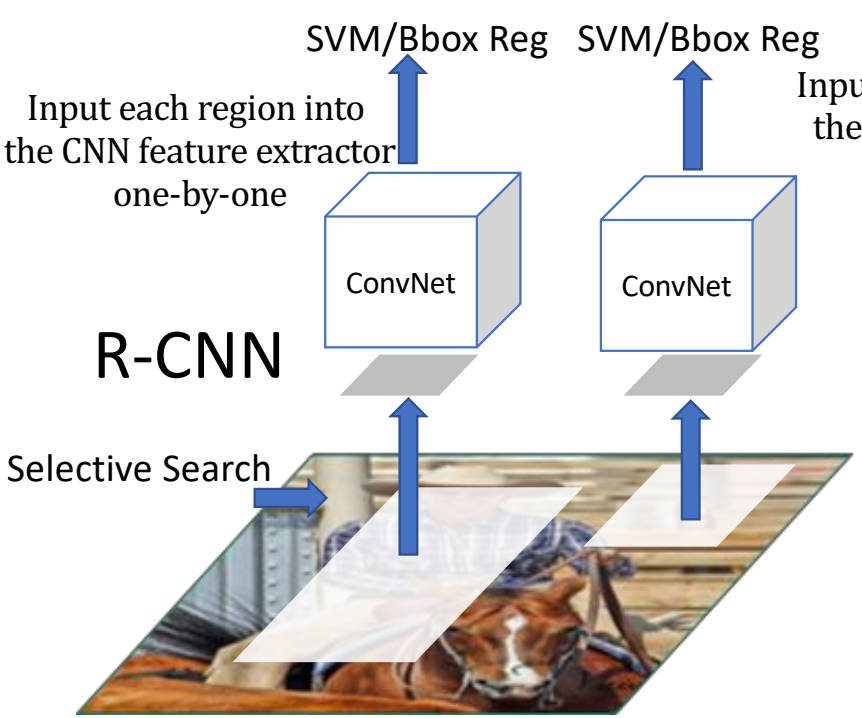
An object can have many potential output bounding boxes, NMS helps to remove the overlap boxes and keep the best one.

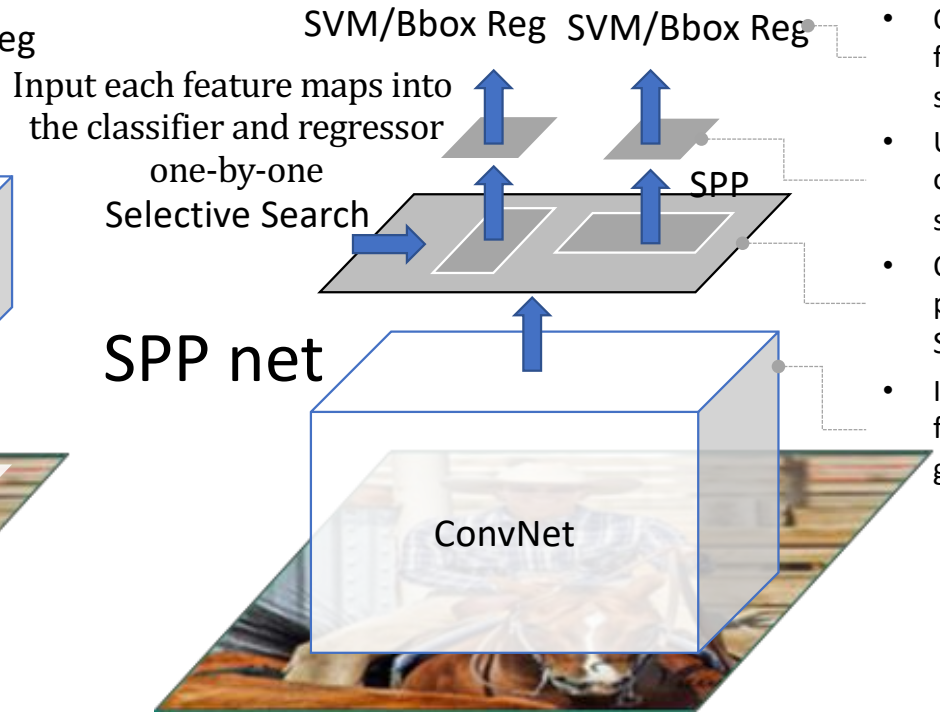
Computer Vision Applications

- Object Detection: SPP Net
  - R-CNN Limitations:
    - 1. Selective Search is slow.
    - 2. Resizing the proposal region leads to scale change of width and height, which affect the classification accuracy.
    - 3. Feed each region into the VGG individually, which is very slow.
    - 4. Not end-to-end training.
  - SPP Net Contribution:
    - 1. Solve the 2^{nd} and 3^{rd} limitations of R-CNN.
    - 2. Propose Spatial Pyramid Pooling, SPP, which resize the features into the same size.
    - 3. Feed entire image into CNN to obtain global features, and obtain the features of each region on the global features.

Computer Vision Applications

- Object Detection: SPP Net



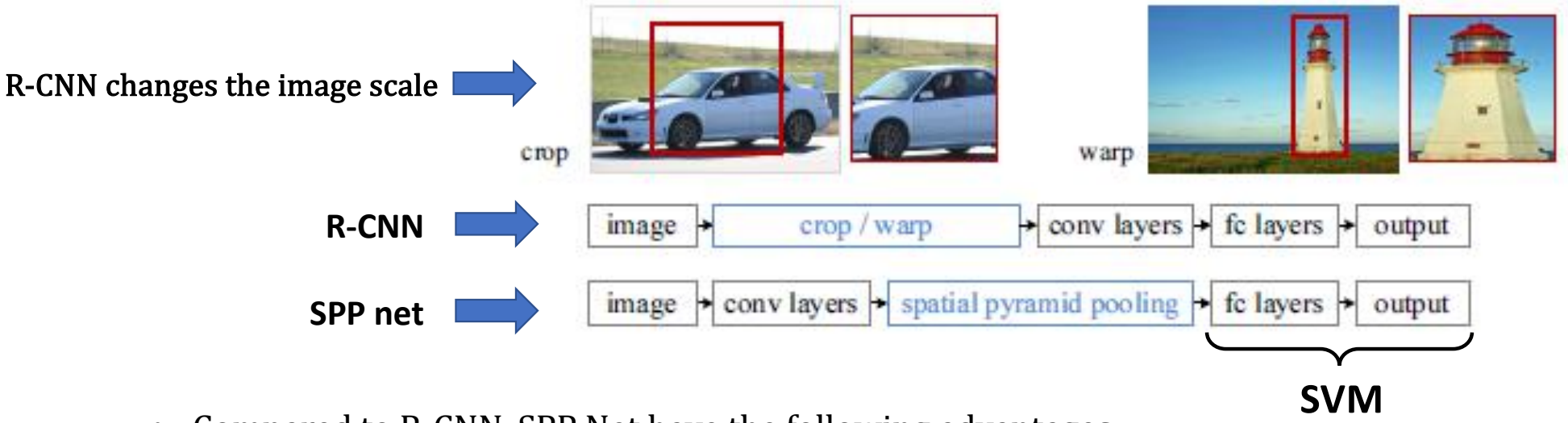


- Obtain the class labesl and bboxes from the feature maps with the same size
- Use SPP to resize the feature maps of all proposal regions into the same size
- Obtain the feature maps of all proposal regions based on Selective Search.
- Input the entire image into the feature extractor to obtain the global feature maps.

Spatial pyramid pooling in deep convolutional networks for visual recognition. *Kaiming He etal. ECCV. 2014.*

Computer Vision Applications

- Object Detection: SPP Net



- Compared to R-CNN, SPP Net have the following advantages.
  - Use global features instead of feeding images into VGG one-by-one.
  - Do not change the image scale.

Spatial pyramid pooling in deep convolutional networks for visual recognition. *Kaiming He etal. ECCV. 2014.* （图片来自）

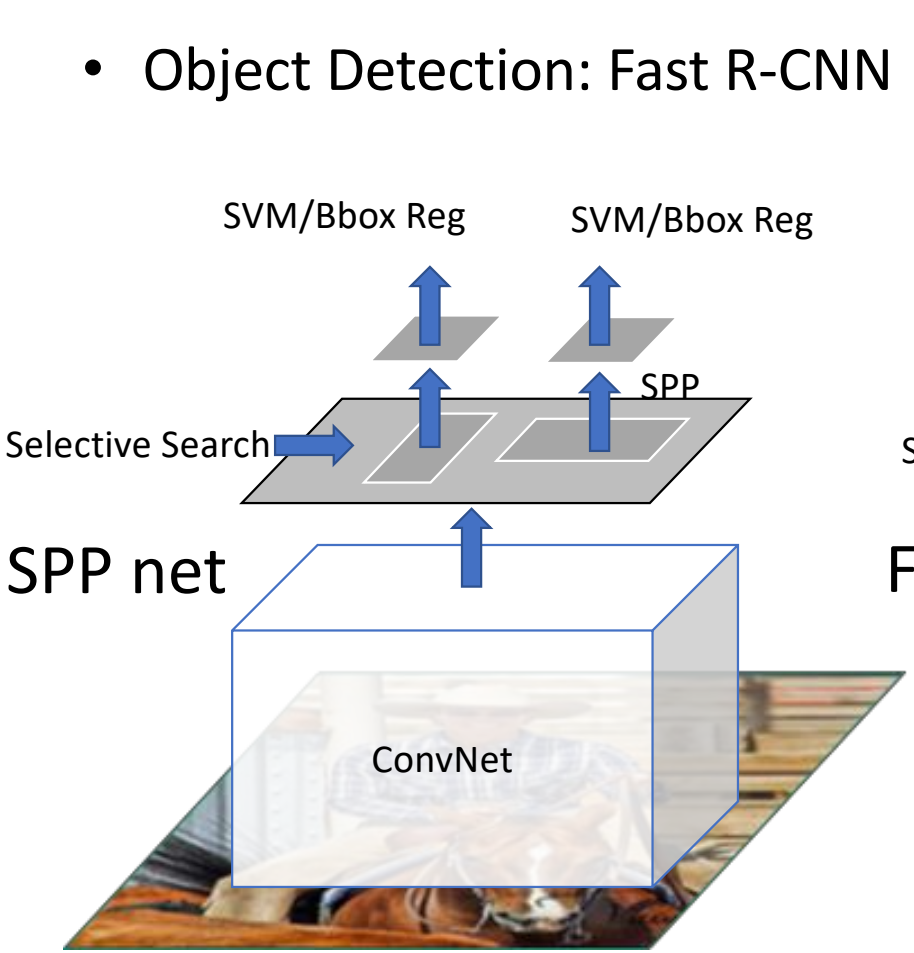
Computer Vision Applications

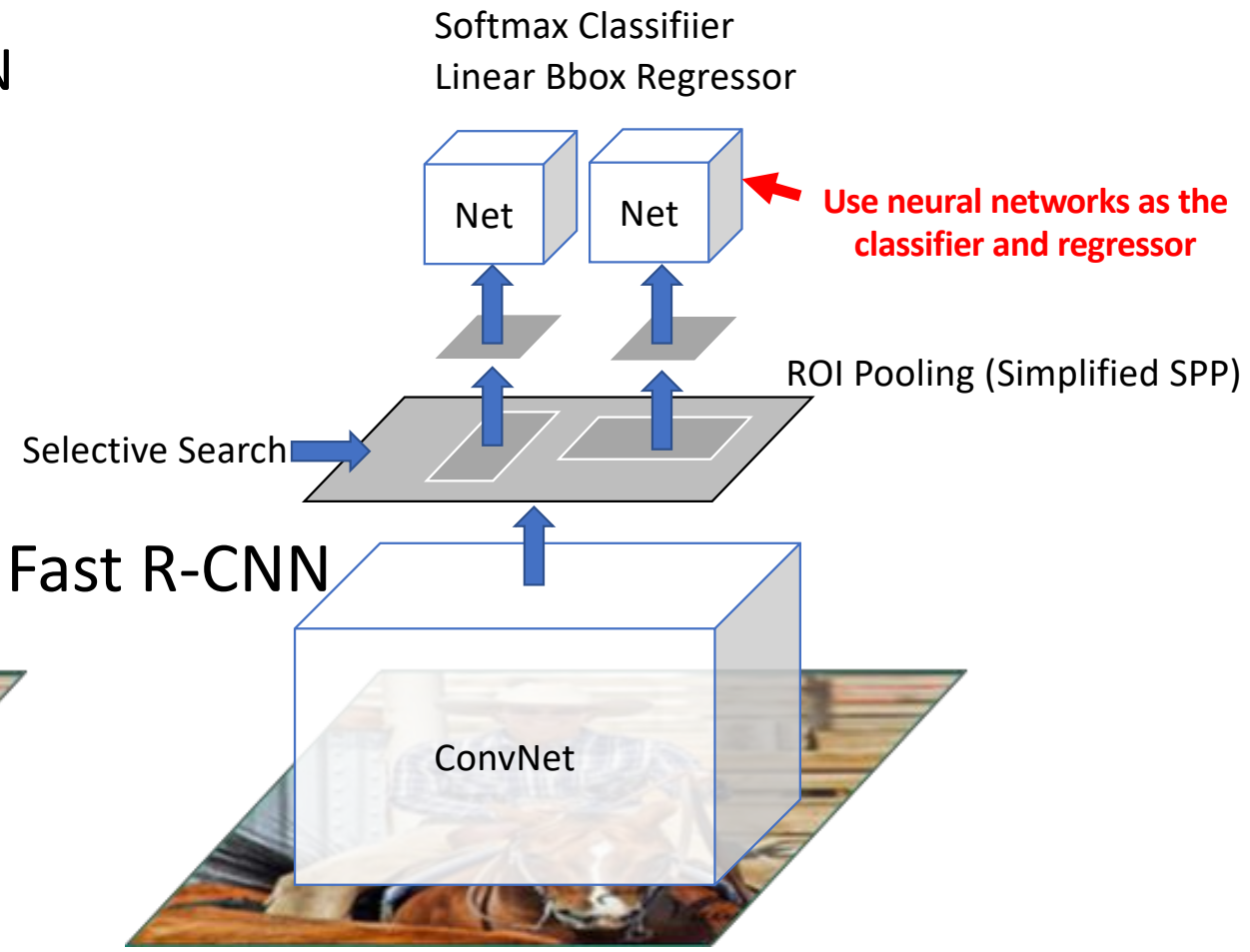
- Object Detection: Fast R-CNN
  - SPP Net Limitations:
    - 1. Still use Selective Search, which is very slow.
    - 2. Still not end-to-end training.
  - Fast R-CNN Contributions:
    - 1. Propose ROI (Region of Interest) Pooling layer, which is a simplified Spatial Pyramid Pooling that uses one pooling size only.
    - 2. End-to-end training, the classifier and bbox regressor are trained together with the CNN feature extractor.

Fast R-CNN. *Girshick, Ross etal. ICCV. 2015.*

Computer Vision Applications

- Object Detection: Fast R-CNN





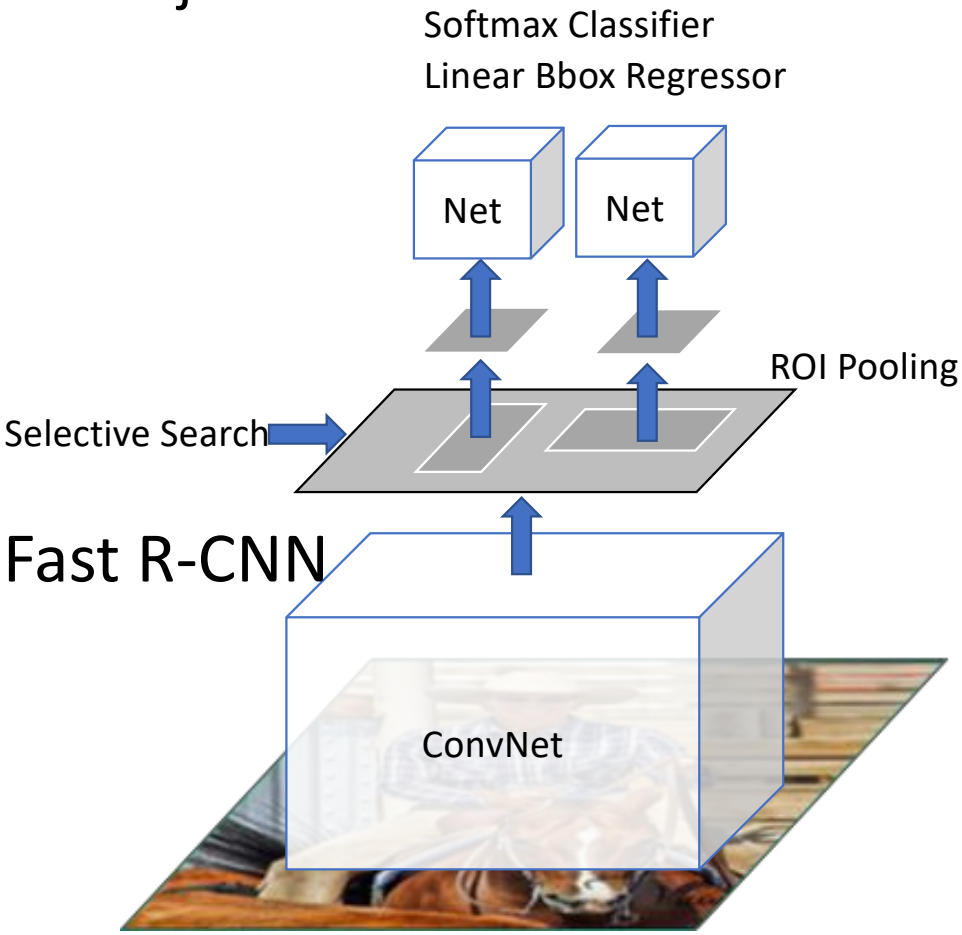
Fast R-CNN. *Girshick, Ross etal. ICCV. 2015.*

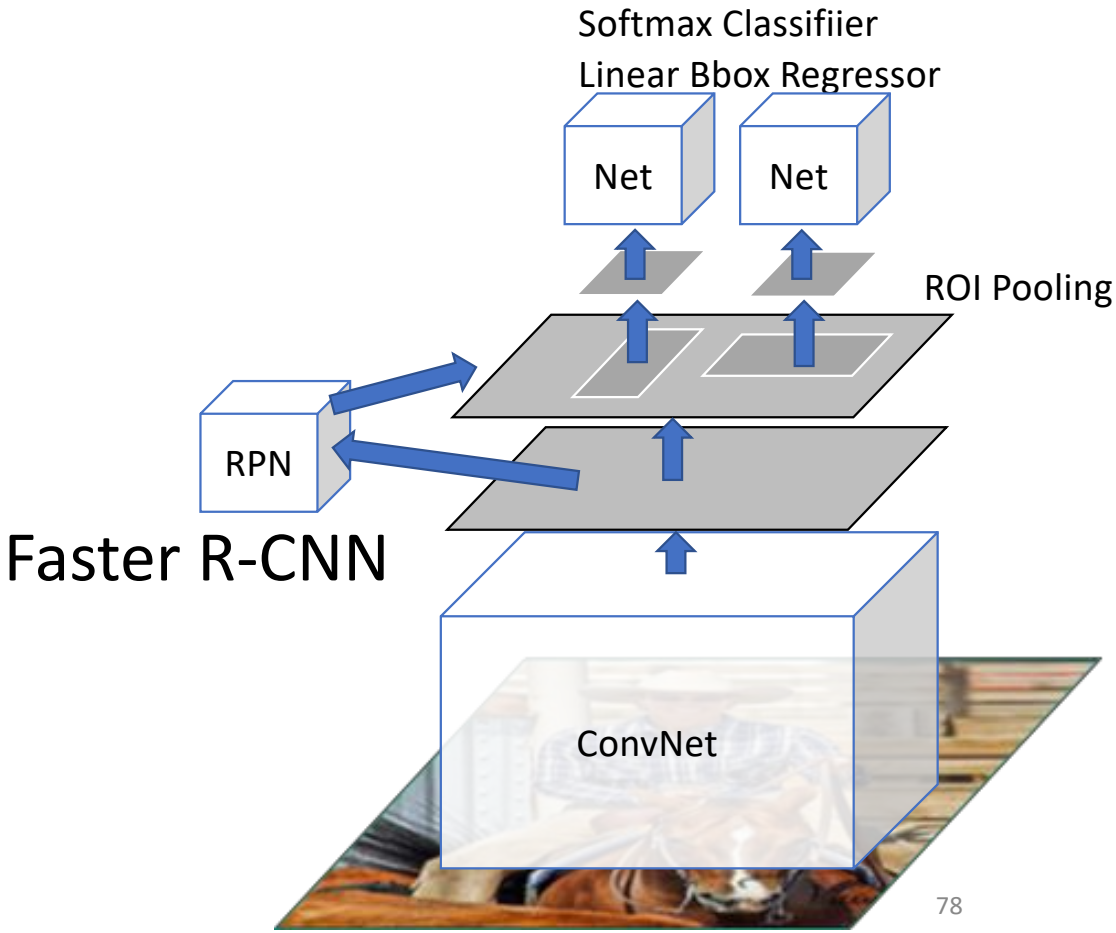
Computer Vision Applications

- Object Detection: Faster R-CNN
  - Fast R-CNN Limitation:
    - 1. Still use Selective Search, which is very slow.
  - Faster R-CNN Contribution:
    - 1. Use Region Proposal Networks (RPN) to replace Selective Search, enable neural networks to search the proposal regions, which is very faster.
    - 2. Achieve end-to-end training, accuracy increased.

Computer Vision Applications

- Object Detection: Faster R-CNN





Faster R-CNN. *Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun. NIPS. 2015.*

Computer Vision Applications

- Object Detection: Timeline of Proposal-based Methods

	R-CNN	SPP net	Fast R-CNN	Faster R-CNN
Region Proposal	Selective Search	Selective Search	Selective Search	Region Proposal Network
Feature Extraction	Deep Network	Deep Network	Deep Network	Deep Network
Classification & Regression	SVM	SVM	Deep Network	Deep Network

Computer Vision Applications

北京大学 PEKING UNIVERSITY

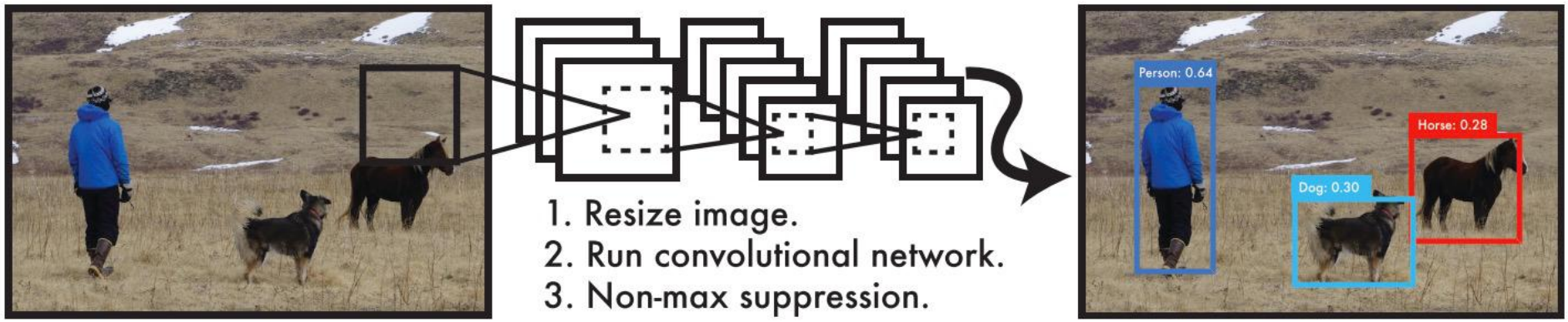
- Object Detection: Rethinking
  - The above algorithms all need region proposal, but it is necessary?
  - To know the locations of different objects in an image, the human would not manually select many proposal regions and classify them one-by-one, instead, human can find all objects in an image by just one glance.

YOLO

You Only Look Once

Computer Vision Applications

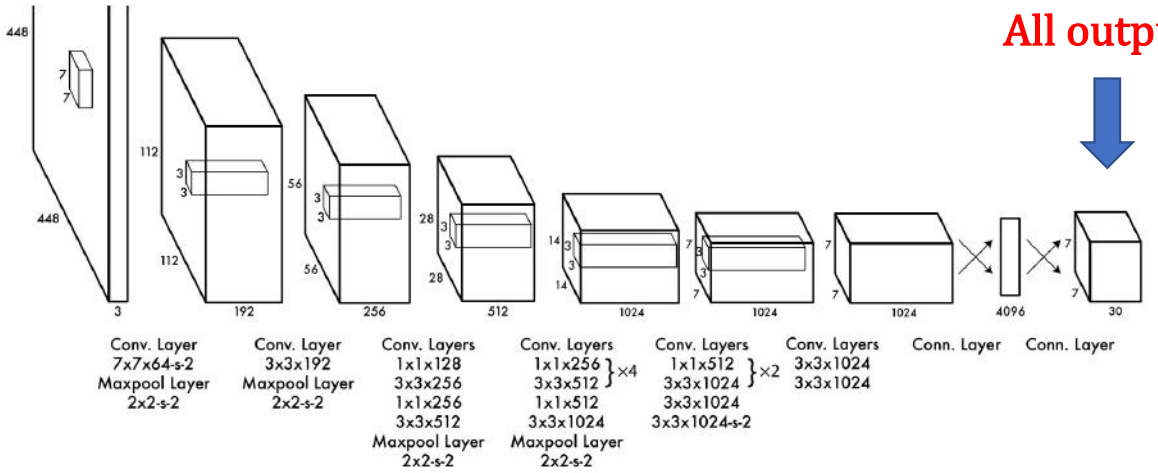
- Object Detection: YOLO
  - Object detection problem → Regression problem.
  - No Region Proposal, a fully convolutional networks output the class labels and location information directly.
  - Very fast.



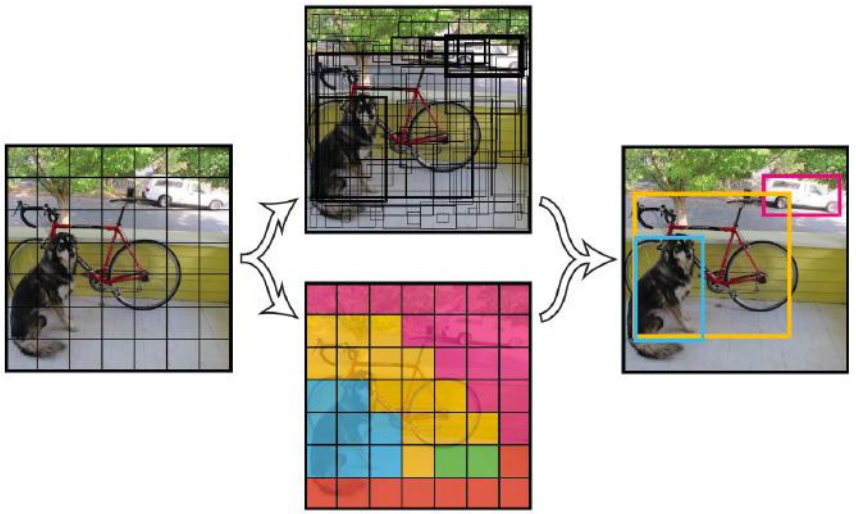
Computer Vision Applications

- Object Detection: YOLO

An image is separated into a 7x7 grid, each cell in the grid predicts the class label, object width and height, and object confidence of the corresponding image location.



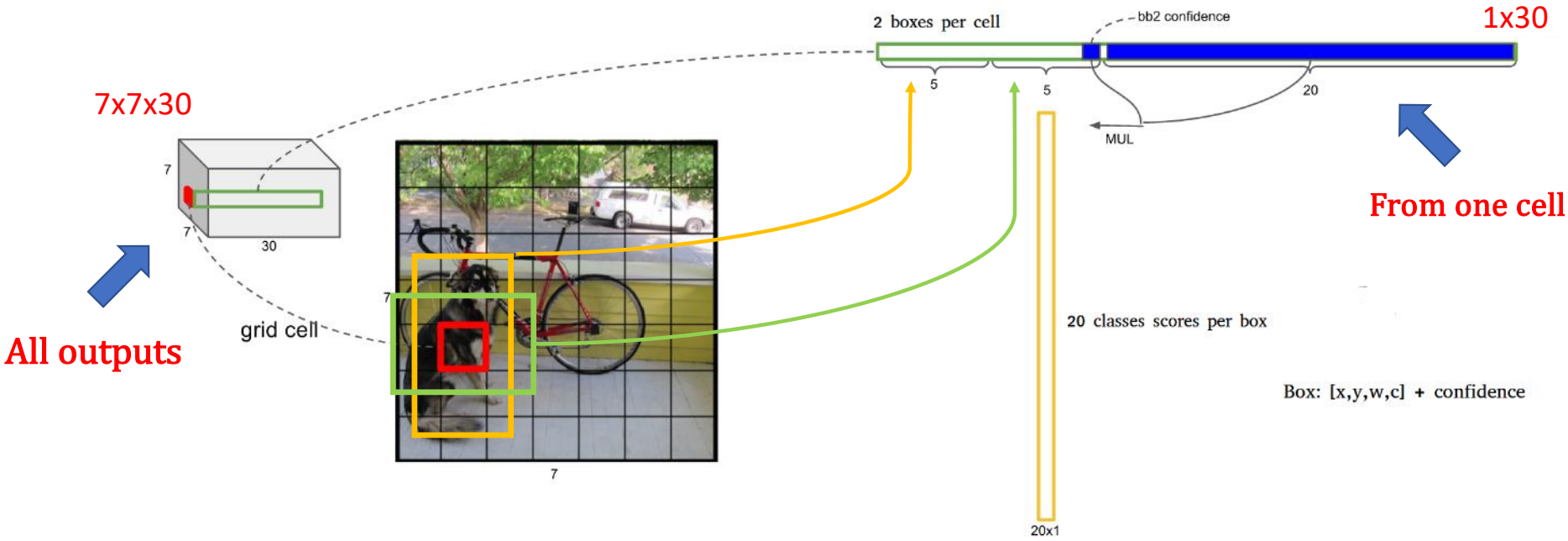
All outputs



Computer Vision Applications

- Object Detection: YOLO

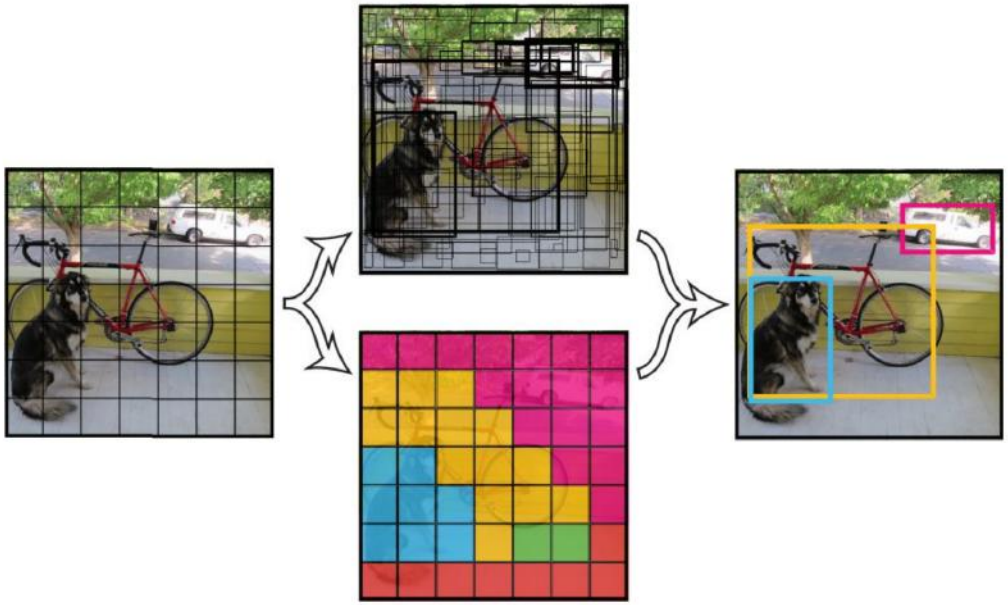
  - Each cell has 30 outputs, 10 for the information of 2 bboxes (x, y, w, h, confidence), 20 for 20 class probabilities of VOC dataset.
  - Therefore, each cell has 2 potential bboxes, but can only have one kind of object.



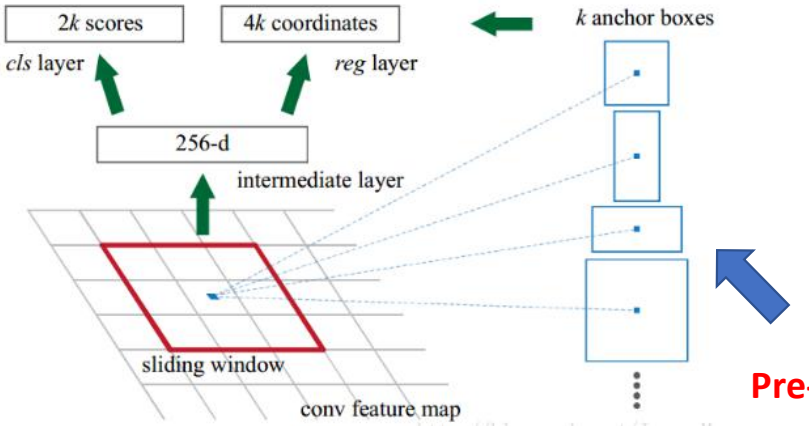
Computer Vision Applications

- Object Detection: YOLO

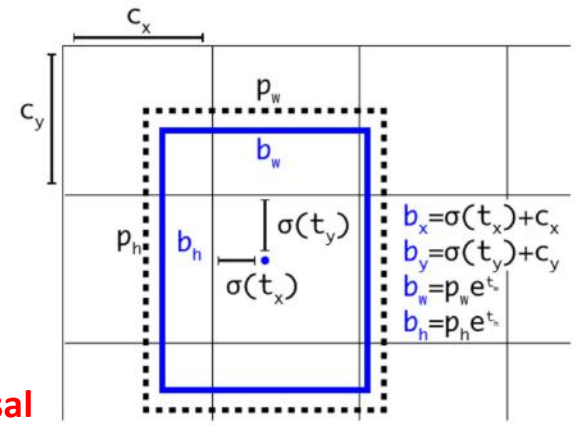
Remove the bboxes with low confidence and perform NMS to obtain the final result.



- Object Detection: YOLO2
  - YOLO Limitation:
    - Difficult to detect small objects.
  - YOLO2：
    - Pre-define multiple proposal regions for each cell i.e., pre-defined anchor.

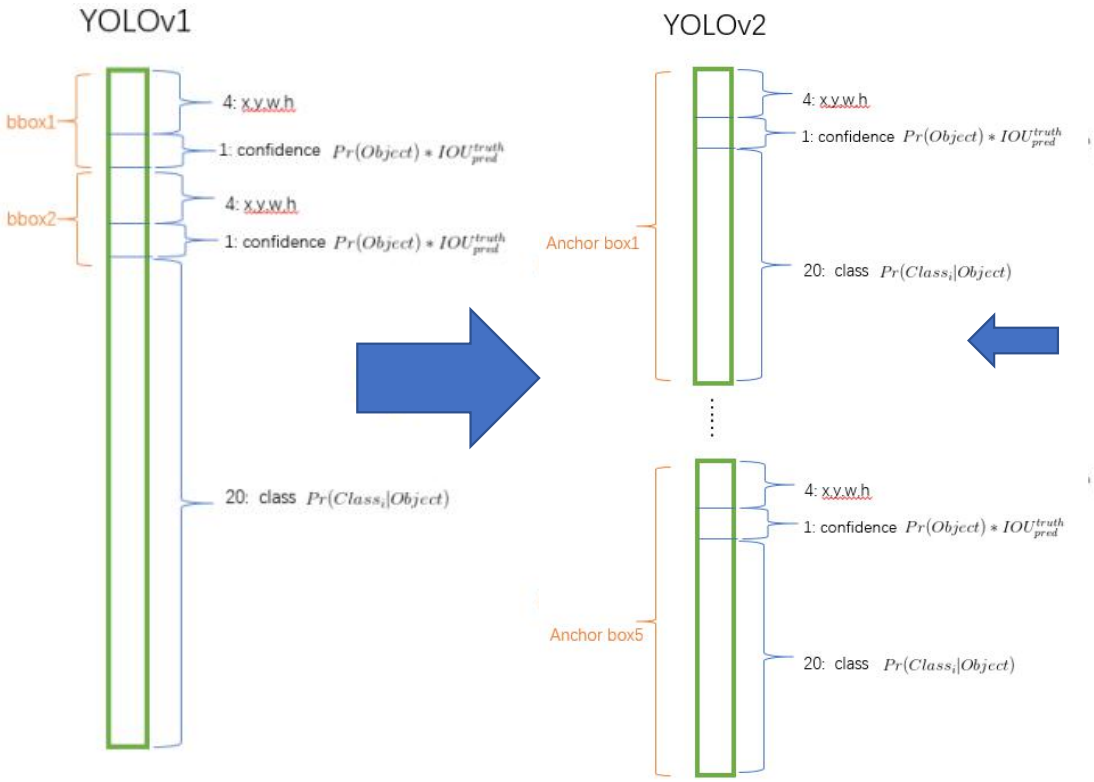


Pre-defined Region Proposal



Computer Vision Applications

- Object Detection: YOLO2

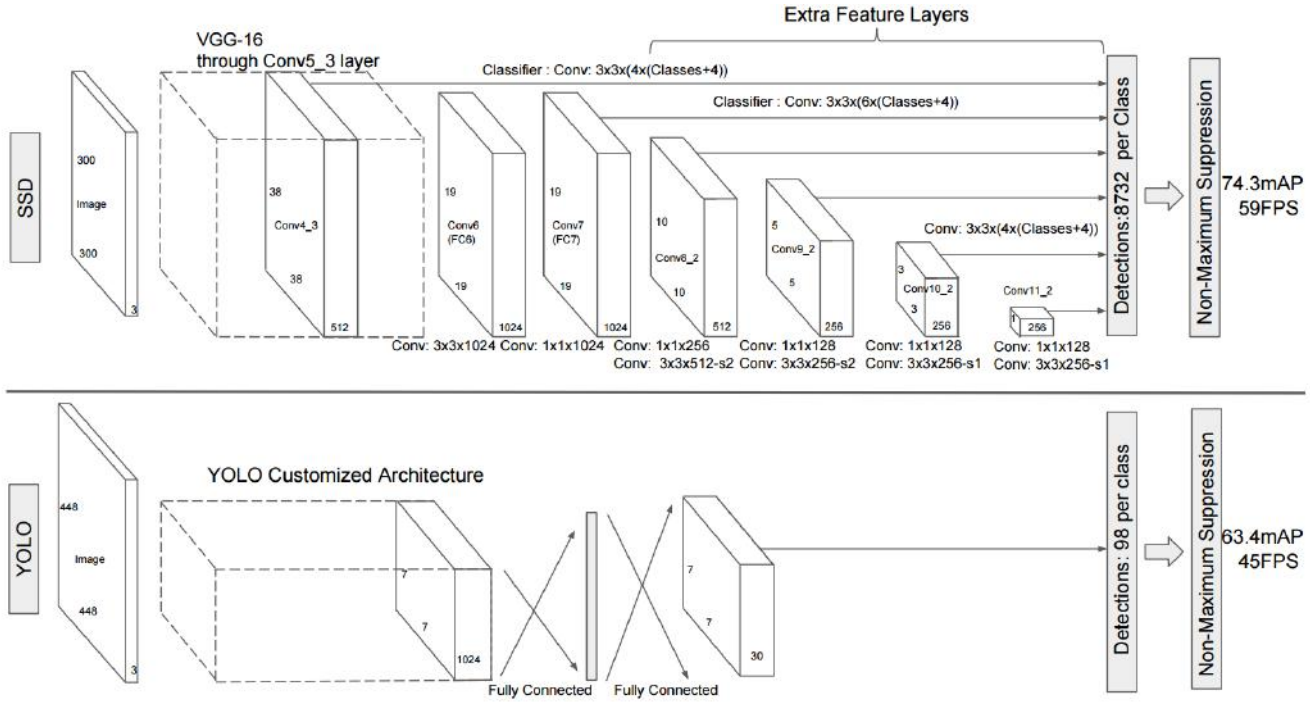


- Increase the grid resolution from 7x7 to 13x13.
- Pre-define 6 anchors for each cell, YOLO2 can predict 13x13x6=1014 potential bboxes, while YOLO1 only have 7x7x2=98 bboxes.
- Each cell has multiple class probabilities vectors i.e., each cell can detect different types of object. (each cell in YOLO1 can only be one class)

Computer Vision Applications

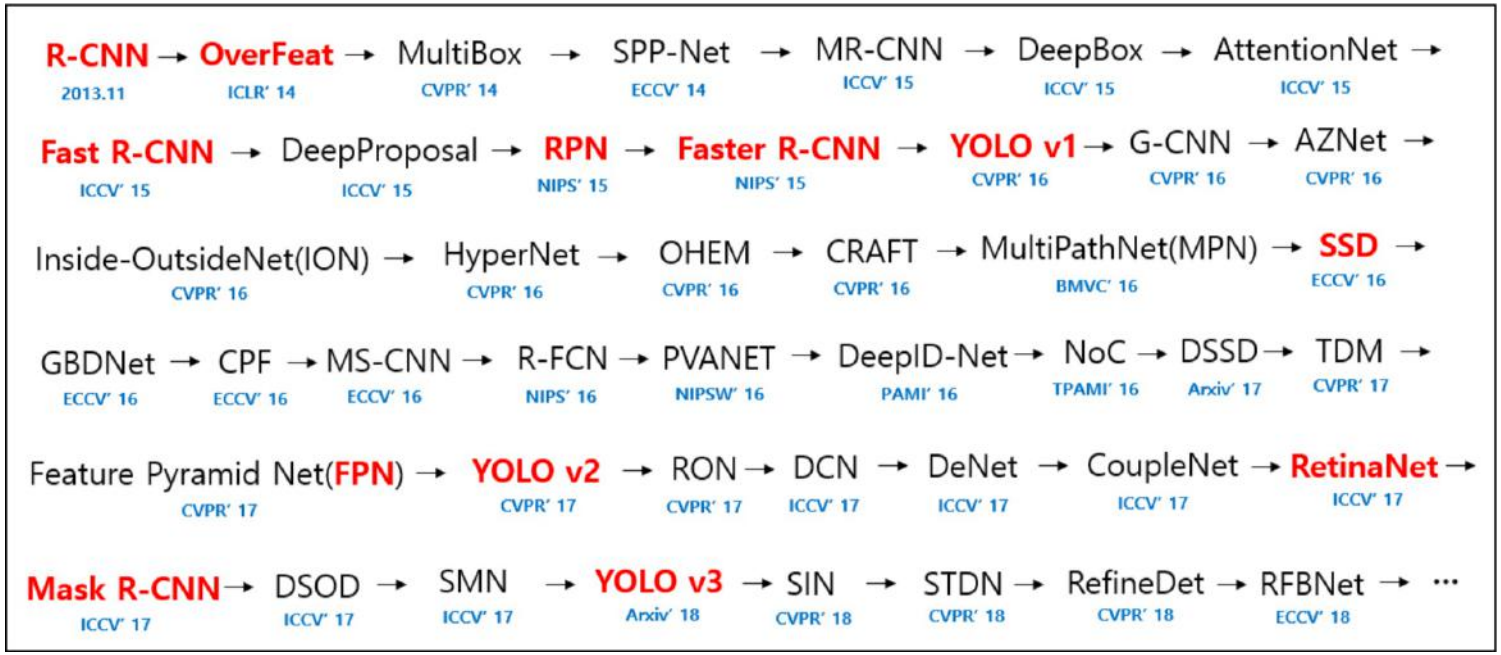
- Object Detection: SSD

SSD is another “Only Look Once” algorithm from Google.



Computer Vision Applications

- Object Detection: More and more …

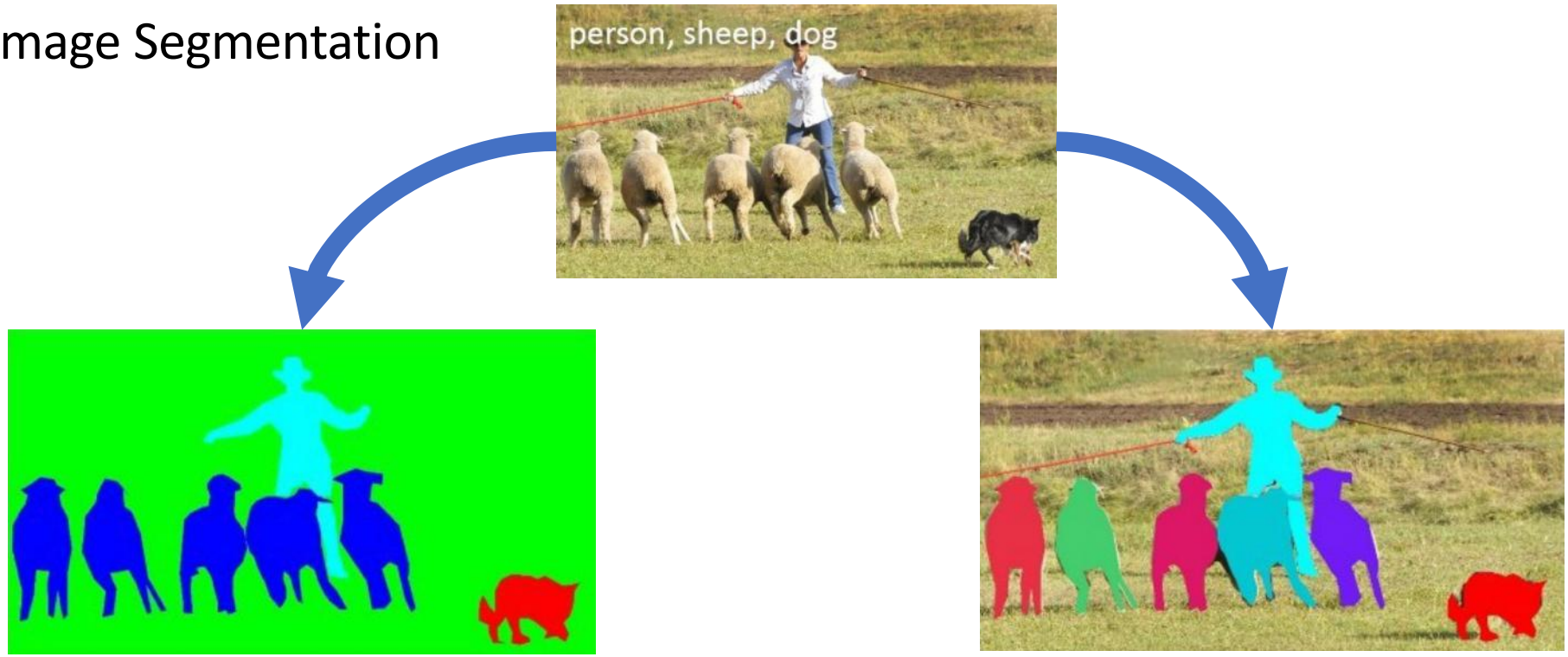


Challenge：

- Speed
- Accuracy
- Weakly-supervised
- 3D detection
- Small object
- Object overlapping
- Multi-task learning
- …..

Computer Vision Applications

- Image Segmentation



Semantic Segmentation

Instance Segmentation

Computer Vision Applications

- Image Segmentation: Pixel-wise Classification

Input

segmented

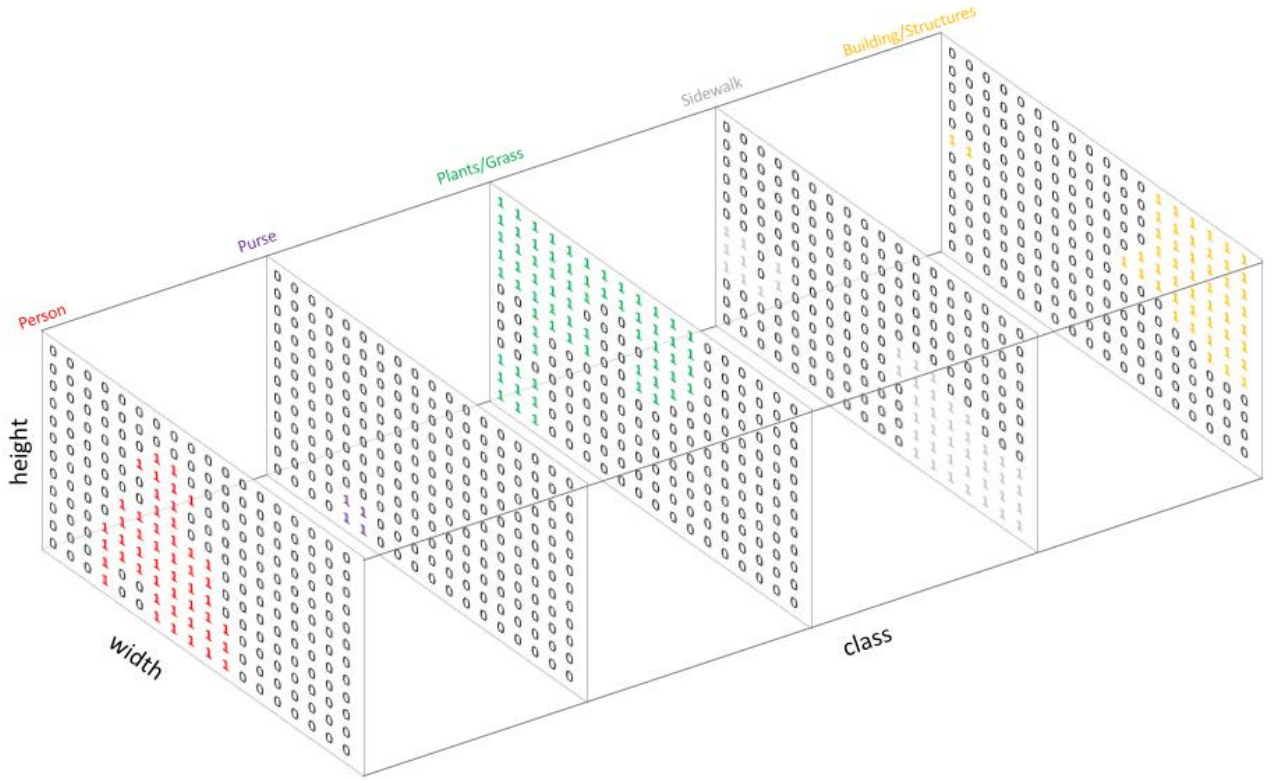
1: Person
2: Purse
3: Plants/Grass
4: Sidewalk
5: Building/Structures

3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5
3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5
3	3	3	3	3	3	1	1	3	3	3	3	5	5	5	5	5	5
3	3	3	3	3	1	1	1	1	3	3	3	5	5	5	5	5	5
3	3	3	3	3	3	1	1	3	3	3	5	5	5	5	5	5	5
5	5	3	3	3	3	1	1	3	3	5	5	5	5	5	5	5	5
4	4	3	4	1	1	1	1	1	1	4	4	4	5	5	5	5	5
4	4	3	4	1	1	1	1	1	1	4	4	4	4	4	5	5	5
4	4	4	1	1	1	1	1	1	1	1	4	4	4	4	4	4	4
3	3	3	1	1	1	1	1	1	1	1	4	4	4	4	4	4	4
3	3	3	1	2	2	1	1	1	1	1	4	4	4	4	4	4	4
3	3	3	1	2	2	1	1	1	1	1	4	4	4	4	4	4	4

Semantic Labels

Computer Vision Applications

- Image Segmentation: Pixel-wise Classification

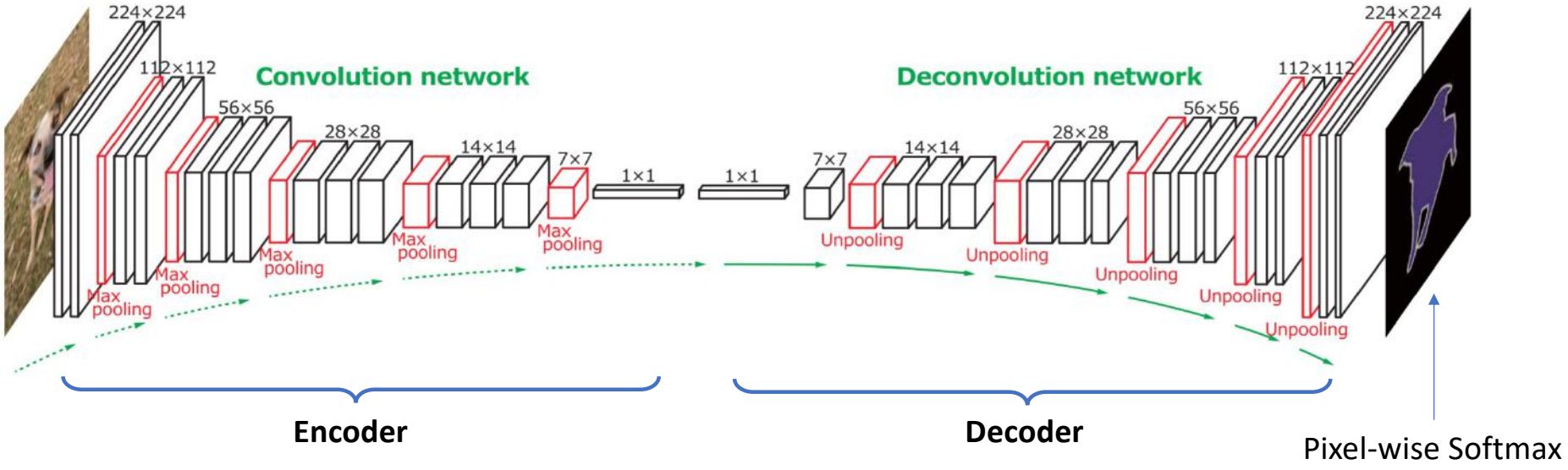


Computer Vision Applications

- Image Segmentation: Fully Convolutional Networks, FCN

FCN uses convolution and pooling only, which allows to input images with arbitrary size.

- Encoder uses pooling, strided convolution for “downsampling”.
- Decoder uses transposed convolution for “upsampling”.



Fully Convolutional Networks for Semantic Segmentation. *Long, Shelhamer, and Darrel. CVPR. 2015.*

Learning Deconvolution Network for Semantic Segmentation. *Noh et al. ICCV. 2015*. (Image is from here)

Computer Vision Applications

- Image Segmentation: Skip-connection

In the encoding process, as the number of layers increased, we can have higher level features, but lost the low-level features.

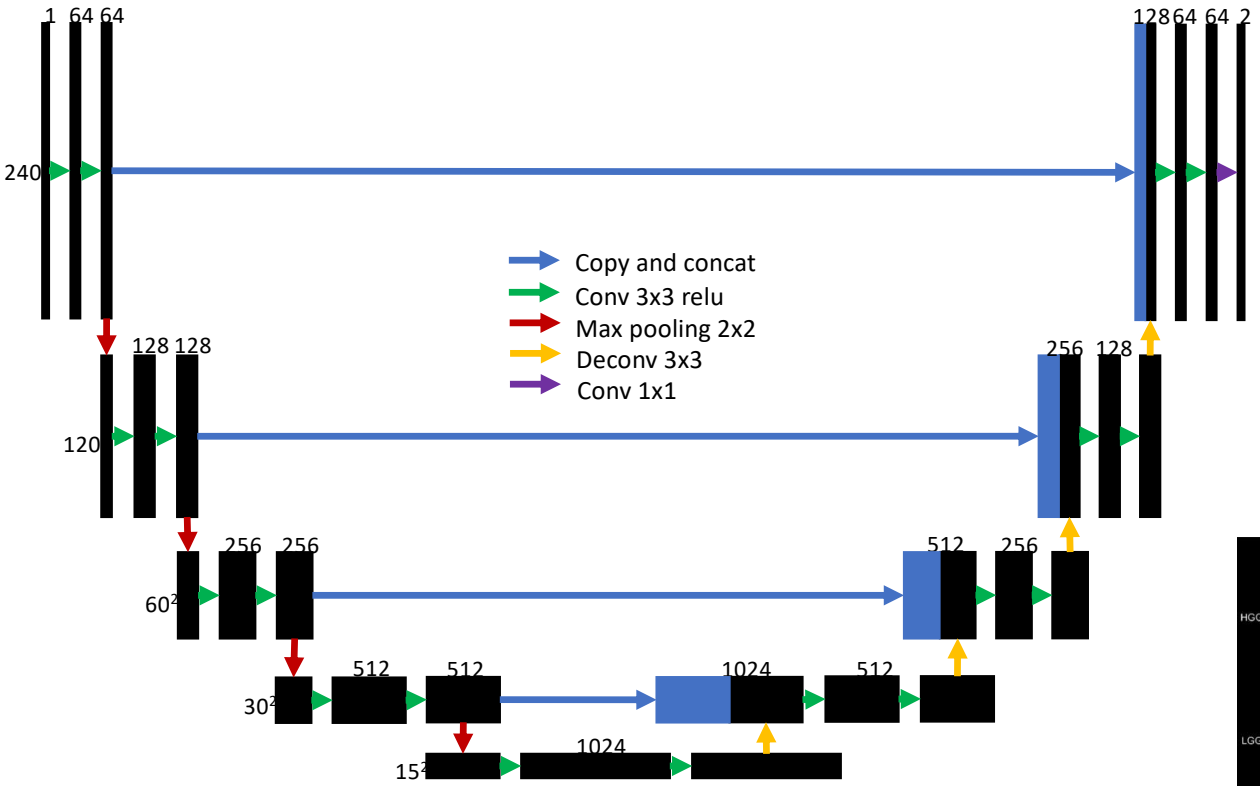
Ground truth target

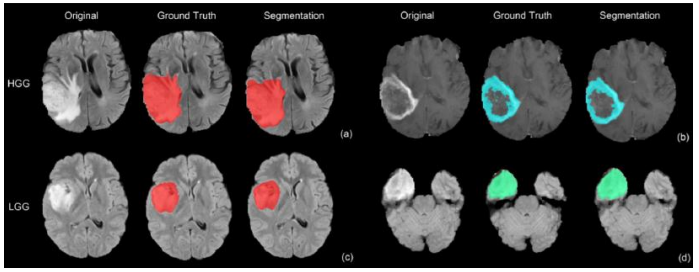
Predicted segmentation

Lost low-level details

Computer Vision Applications

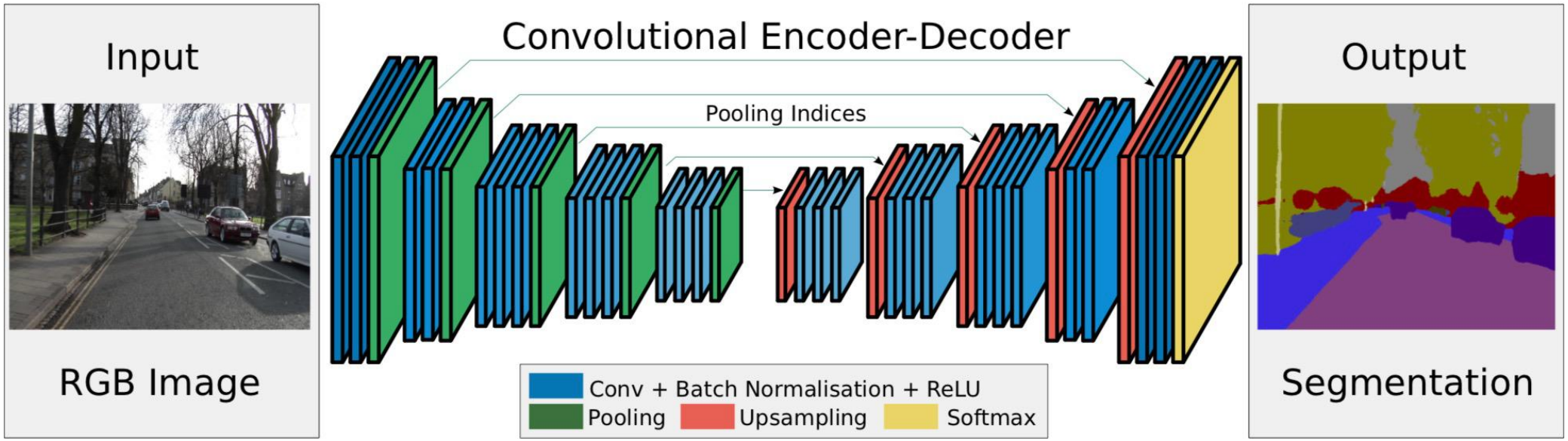
- Image Segmentation: Skip-connection





Computer Vision Applications

- Image Segmentation: Skip-connection

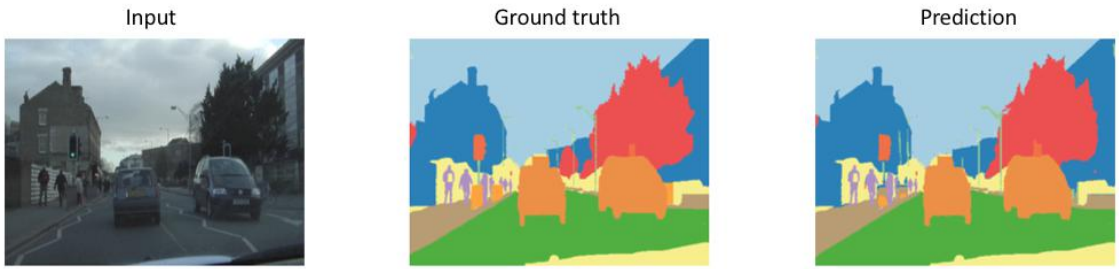


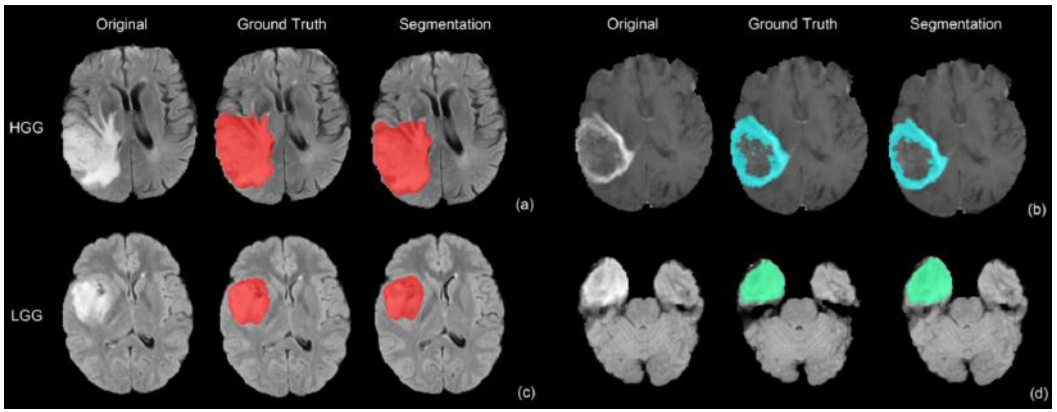
SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *Vijay, Badrinarayanan etal. PAMI. 2015*

Computer Vision Applications

- Image Segmentation: Skip-connection

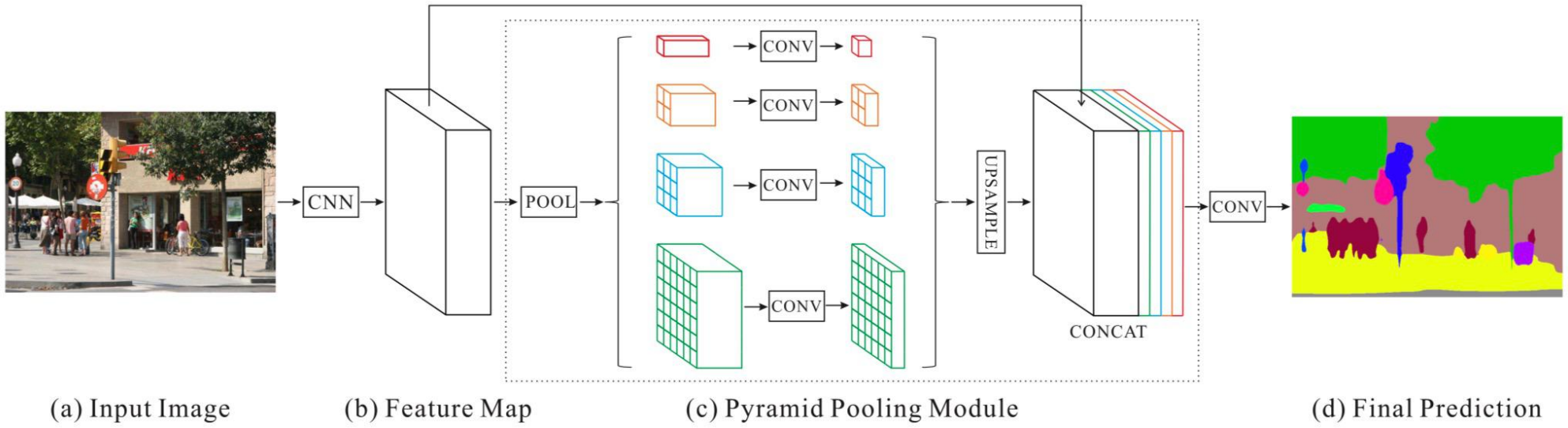
  Skip-connection sends low-level features to the decoding process, which improves the performance for segmenting the detailed pattern.





Computer Vision Applications

- Image Segmentation: PSPnet (Pyramid Scene Parsing Network)



PSPnet: Pyramid Scene Parsing Network. *Hengshuang. Zhao, Jianping. Shi etal. CVPR. 2017.*

北京大学 PEKING UNIVERSITY

- Image Segmentation

Pixel-wise cross entropy:

- Considering each pixel as an individual label for classification
- Drawback: objects will larger area have larger weights on the loss, which lead to low performance for segmenting small objects.

Dice coefficient:

- Address the imbalance problem of pixel-wise cross entropy.
- $Dice = \frac{2|A \cap B|}{|A|+|B|}$, where A and B are two vectors with values of 0 and1,
  - $|A \cap B|$ is the intersection area.
  - If A and B are fully overlapped, Dice=1.
  - If A and B are fully separated, Dice=0,

$$Dice = \frac{2|A \cap B|}{|A| + |B|} = \frac{2TP}{2TP + FP + FN} = \frac{2|A \cdot B|}{|A|^2 + |B|^2}$$

when values are all 0 and 1

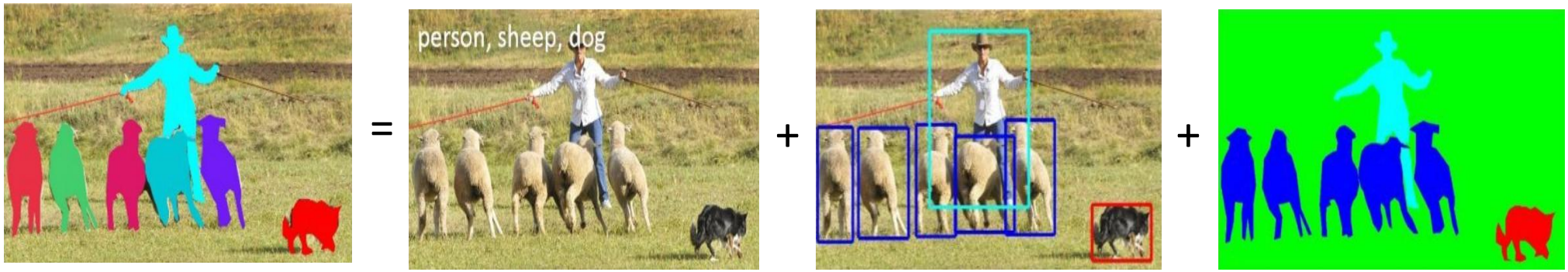
Computer Vision Applications

- Image Segmentation
  - DeepLab series
  - Discriminative Feature Network, DFN. CVPR 2018
  - ExFuse. ECCV 2018
  - …

Computer Vision Applications

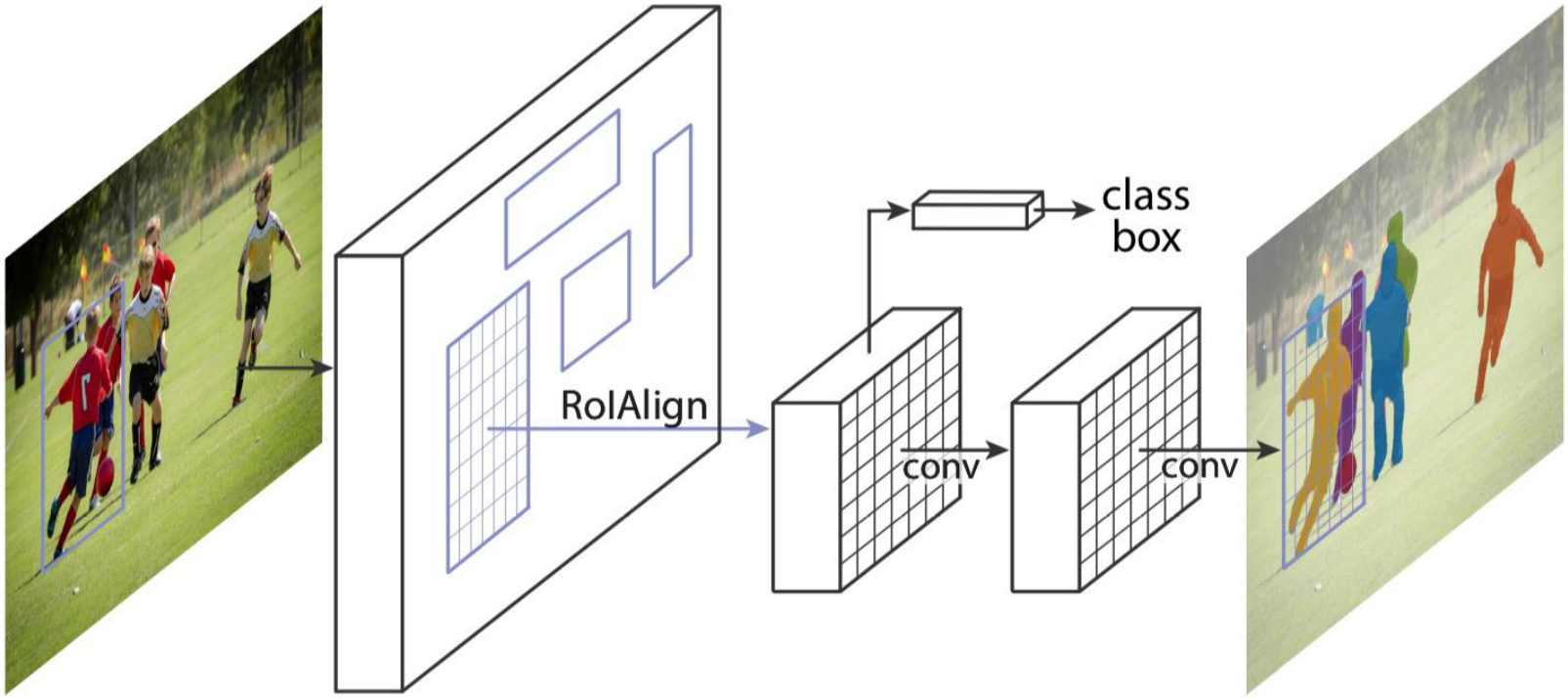
- Image Segmentation: Instance Segmentation

Instance Segmentation = Object Classification + Object Detection + Semantic Segmentation



Computer Vision Applications

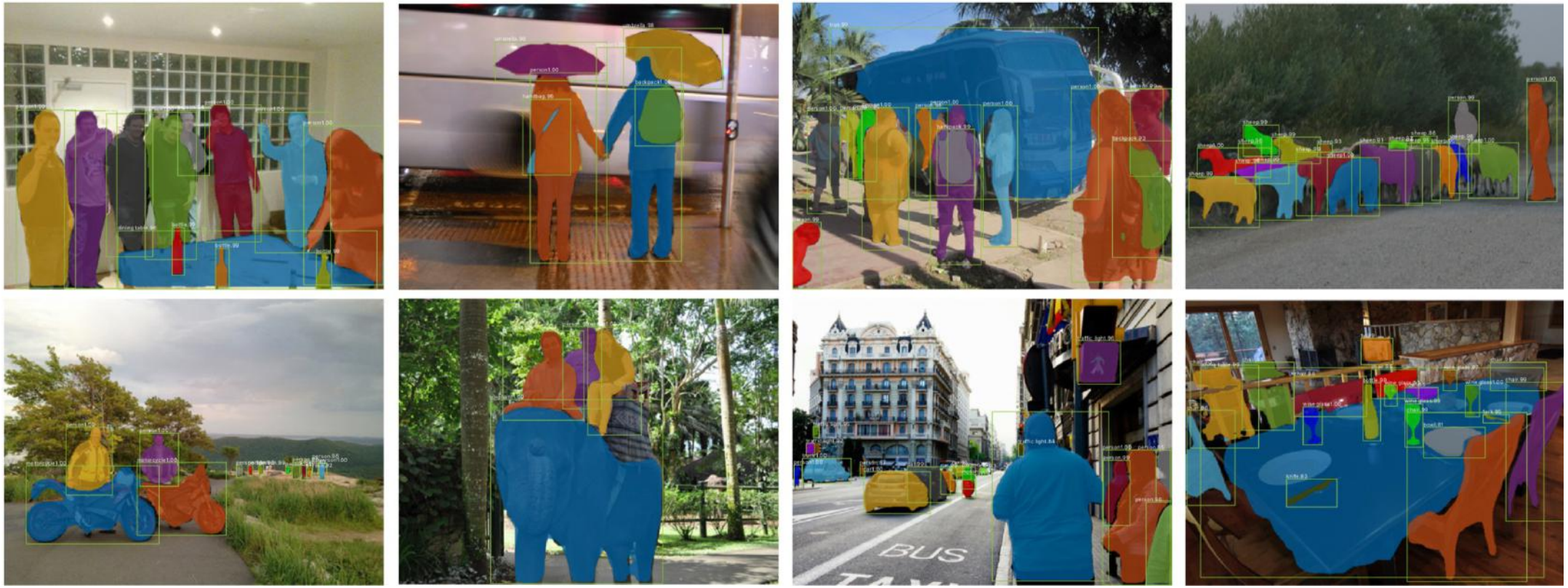
- Image Segmentation: Instance Segmentation



Mask R-CNN. *Long, Kaiming He, Georgia Gkioxari etal. ICCV. 2017.*

Computer Vision Applications

- Image Segmentation: Instance Segmentation



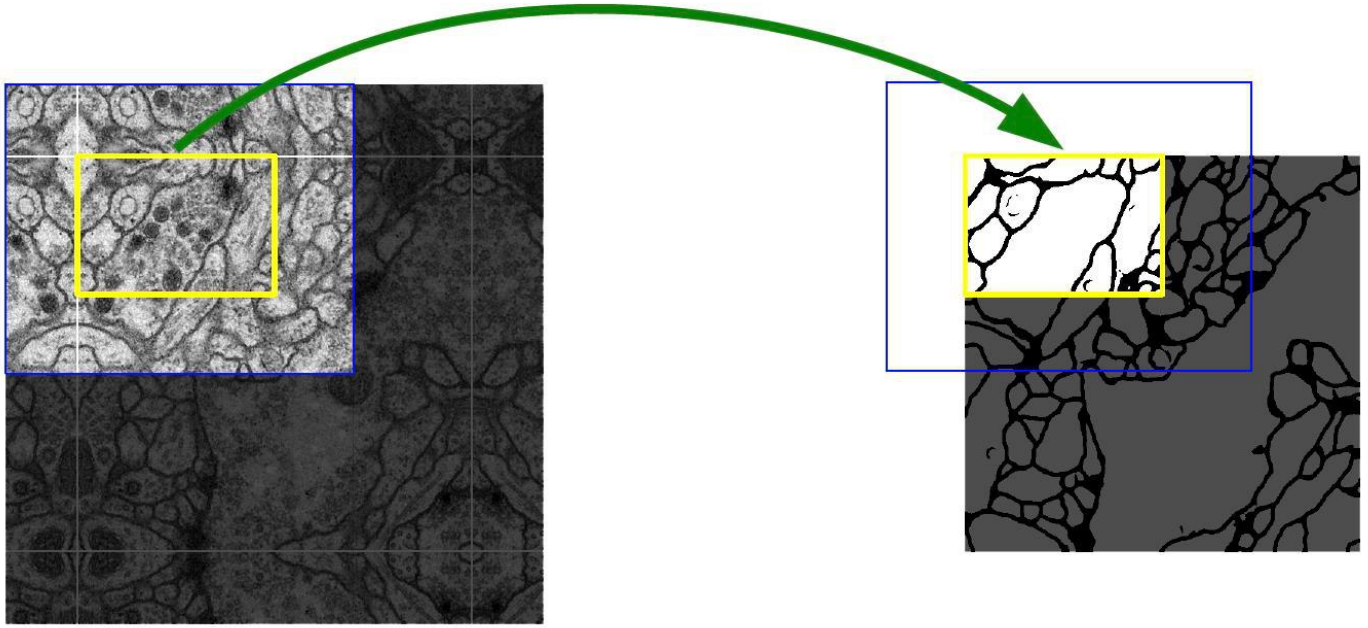
Mask R-CNN. *Long, Kaiming He, Georgia Gkioxari etal. ICCV. 2017.*

Computer Vision Applications

- Image Segmentation: Tricks

Mirror Padding

- Avoid to lost information on the boundary.

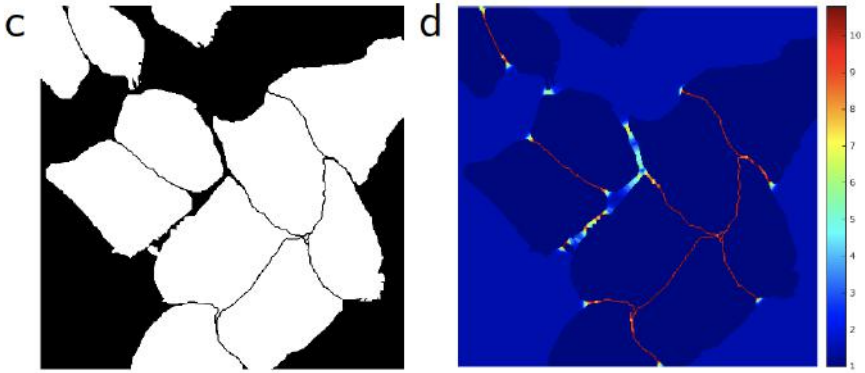


Computer Vision Applications

- Image Segmentation: Tricks

Loss weighting：

- Edge weighting: increase the weight of edge, which make the loss more sensitive to the edges.
- Balance weighting: increase the weight of small objects according to their size.



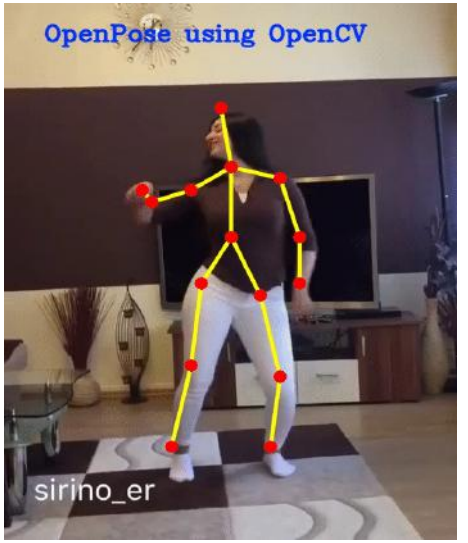
Increase the weight of edges

Increase the weight of small objects e.g., eyes

Computer Vision Applications

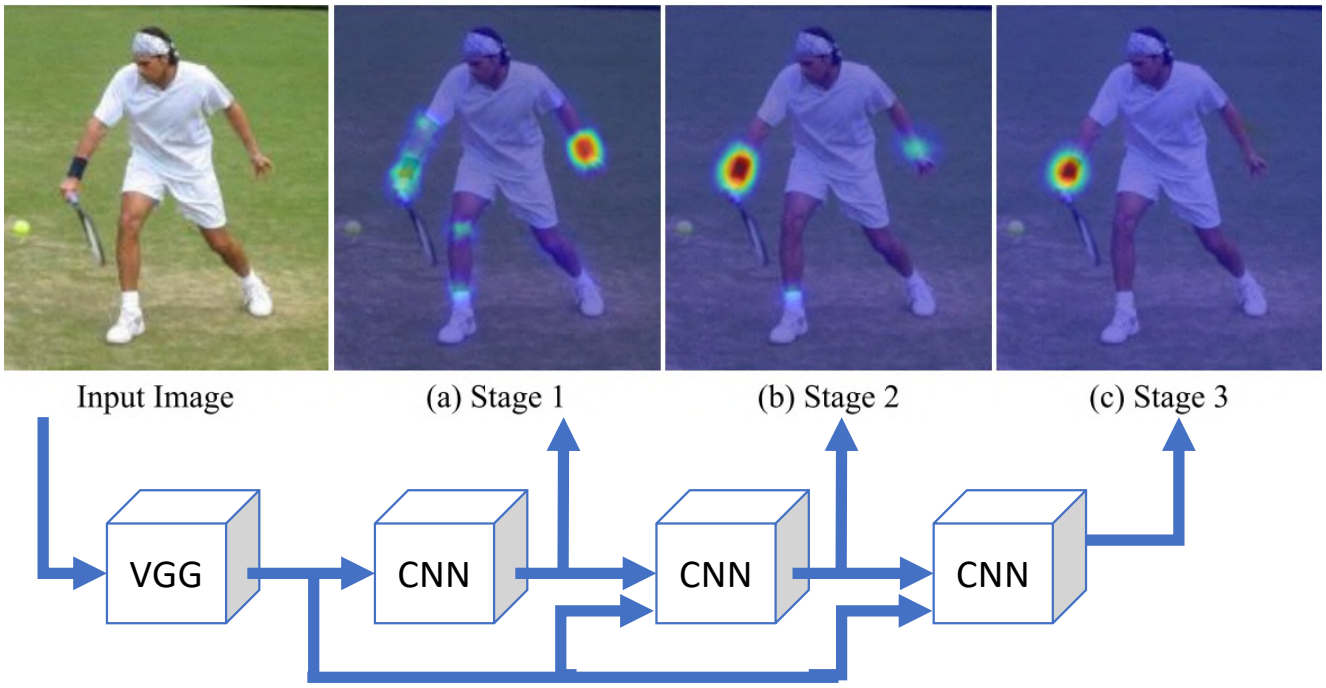
- Pose Estimation

	Pipeline	Advantage	Disadvantage
Top-down approach	Detect each person and then estimate their pose one-by-one.	If object detection works well, the accuracy is high.	If object detection fails to detect the person, we cannot estimate the pose. Inferencing time relate to the number of people.
Bottom-up approach	Detect all keypoints and then assign the keypoints to different person.	Fixed inferencing time.	Difficult to assign the keypoints to different person.



Computer Vision Applications

- Pose Estimation: Convolutional Pose Machine, CPM



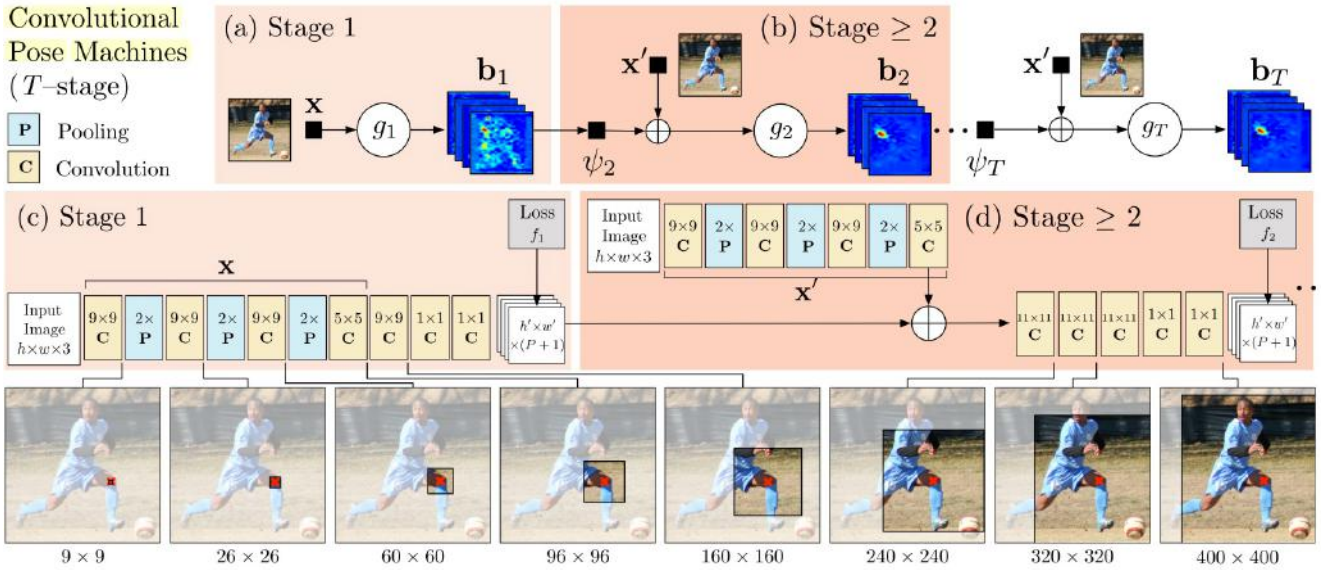
Top-down

- Step 1: Use object detection to find the bbox of person.
- Step 2: Feed the person image into VGG to obtain the image features.
- Step 3: Feed the image features to a stage-1 CNN to obtain the keypoint heatmaps.
- Step 4: Feed the keypoint heatmaps and image feature to the next CNN to obtain better keypoint estimation.

Convolutional Pose Machines. *Shih-En Wei, Varun Ramakrishna etal. CVPR. 2016.*

Computer Vision Applications

- Pose Estimation: Convolutional Pose Machine, CPM

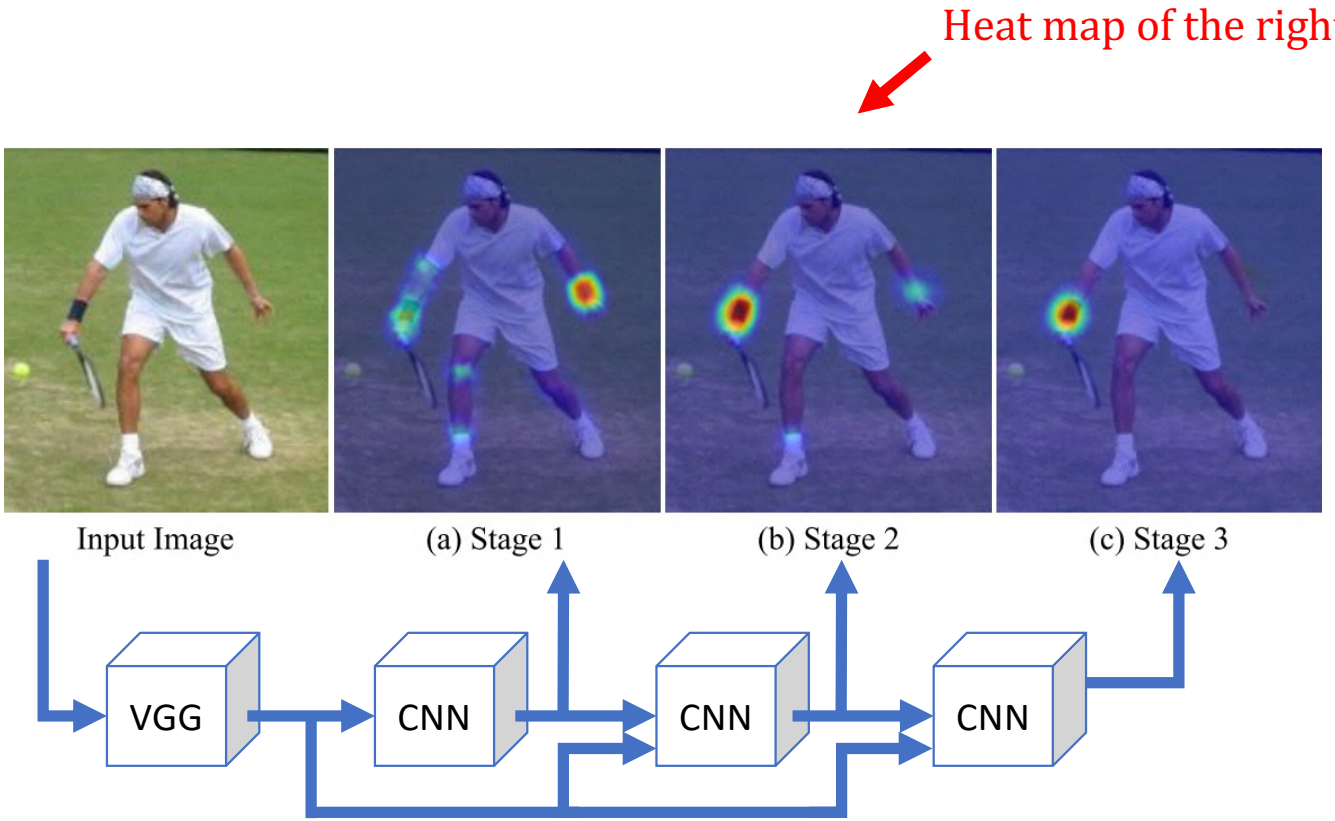


- Advantages of multiple stages:
  - Larger receptive field
  - Fine-tune the keypoint estimation

Convolutional Pose Machines. *Shih-En Wei, Varun Ramakrishna etal. CVPR. 2016.*

Computer Vision Applications

- Pose Estimation: Convolutional Pose Machine, CPM



- In this example, the stage 1 fails to estimate the correct location of the right hand, but as using more stages, the model successfully estimates the correct right hand location.

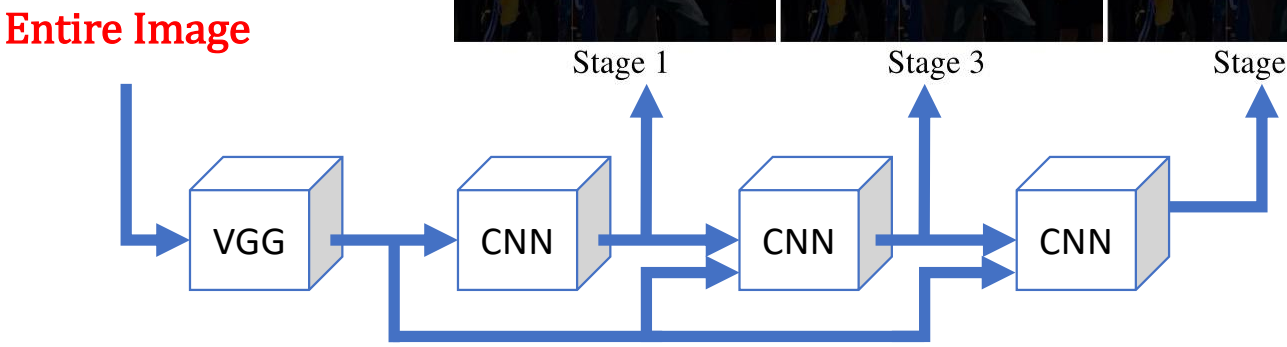
Convolutional Pose Machines. *Shih-En Wei, Varun Ramakrishna etal. CVPR. 2016.*

Computer Vision Applications

- Pose Estimation: OpenPose

OpenPose = CPM + Bottom-up

- Keypoint heatmaps
- "Connection" heatmaps

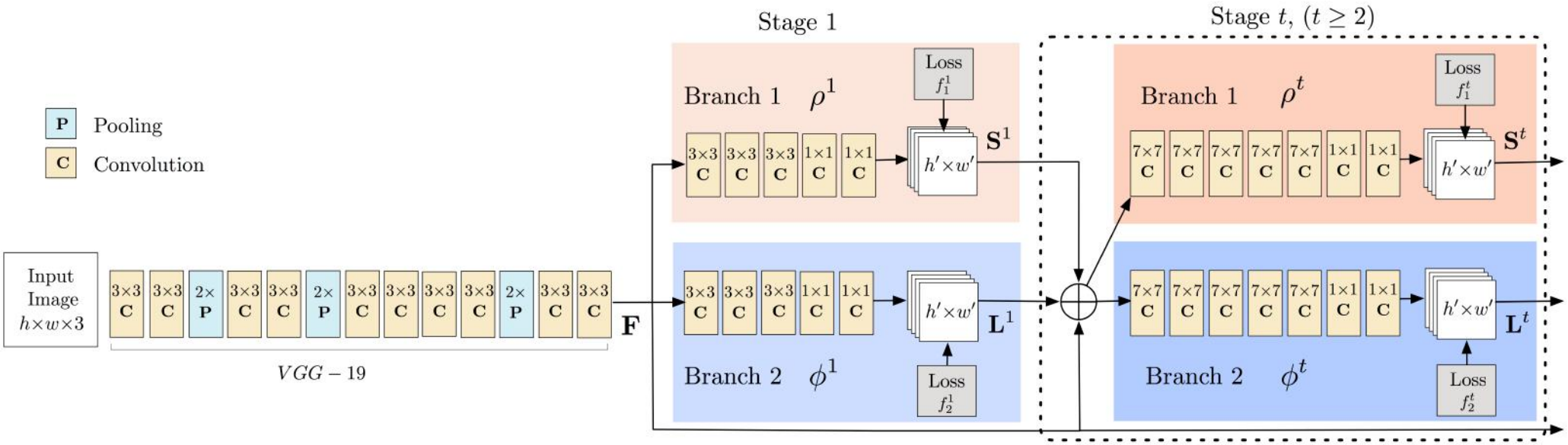


Realtime Multi-Person 2D Pose Estimation using Part Affinity Field. *Cao Zhe, Tomas Simon etal. CVPR. 2017.*

Computer Vision Applications

- Pose Estimation: OpenPose

OpenPose = CPM + Bottom-up



Realtime Multi-Person 2D Pose Estimation using Part Affinity Field. *Cao Zhe, Tomas Simon etal. CVPR. 2017.*

Computer Vision Applications

- Pose Estimation: Pose Proposal Networks, PPN

  PPN = YOLO + OpenPose very fast

- Disadvantage of OpenPose: Parse the heatmaps pixel-wisely to find all keypoints and connections. The speed bottleneck is on CPU rather than GPU.

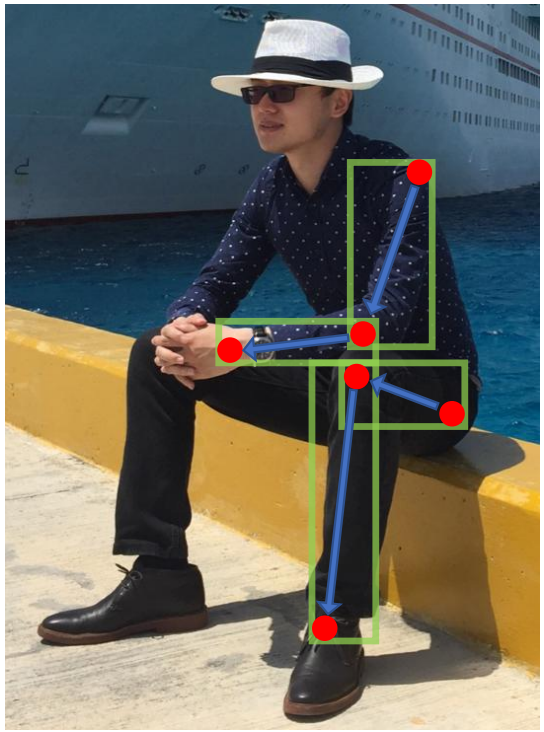
- PPN considers the pose estimation problem as an object detection problem, without requiring pixel-wisely parsing.

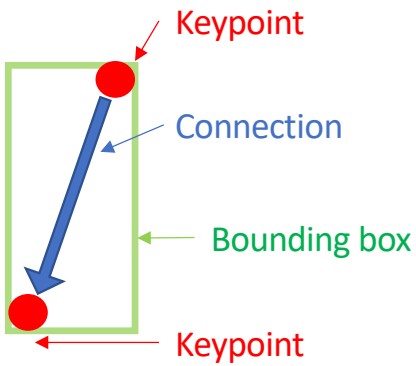
Pose Proposal Networks. *Taiki Sekii. ECCV. 2018.*

Computer Vision Applications

- Pose Estimation: PPN

PPN = YOLO + OpenPose very fast



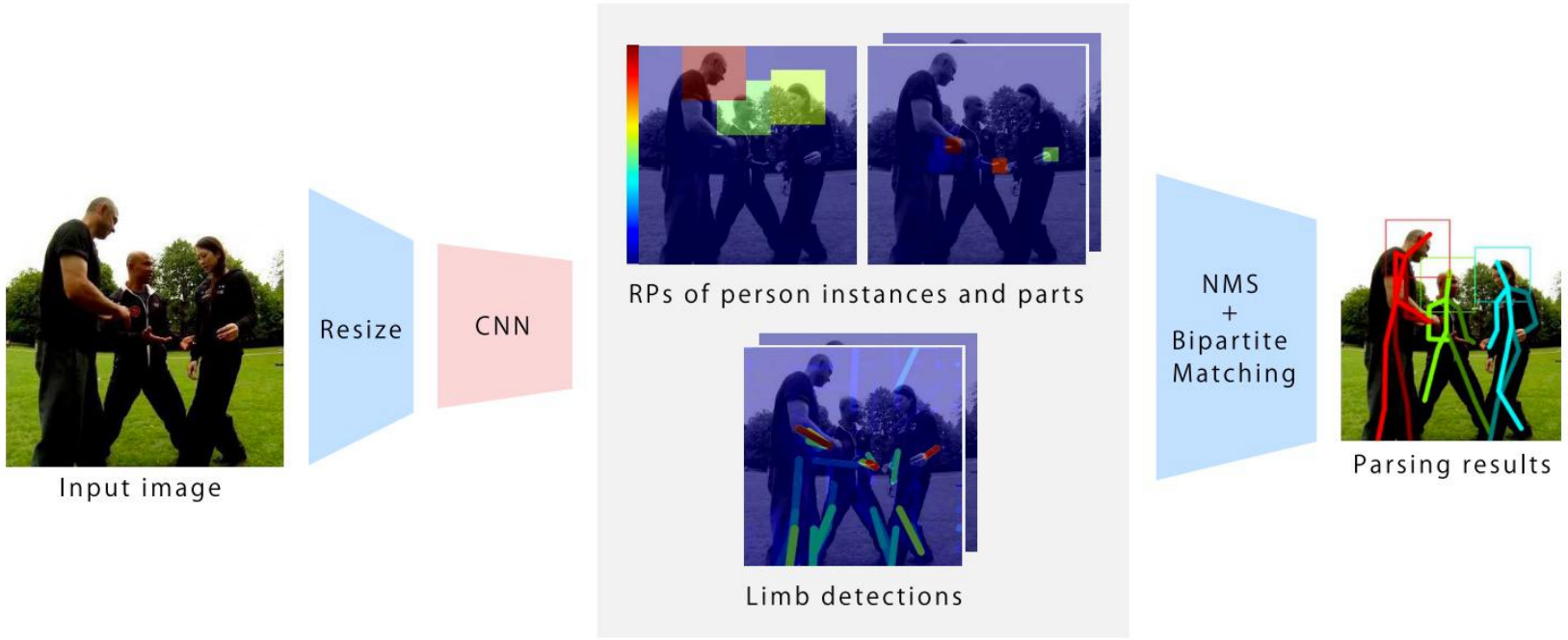


- One body connection has two keypoints.
- OpenPose estimates the red points and blue connections, while PPN estimates the green bounding boxes.
- PPN uses a greedy algorithm to connect all bounding boxes to form a person pose.

Pose Proposal Networks. *Taiki Sekii. ECCV. 2018.*

Computer Vision Applications

- Pose Estimation: PPN



Pose Proposal Networks. *Taiki Sekii. ECCV. 2018.*

Computer Vision Applications

- Pose Estimation: PPN Limitations
  - Poor performance for the crowd.
  - Poor performance for datasets that the person has a large size range.

Pose Proposal Networks. *Taiki Sekii. ECCV. 2018.*

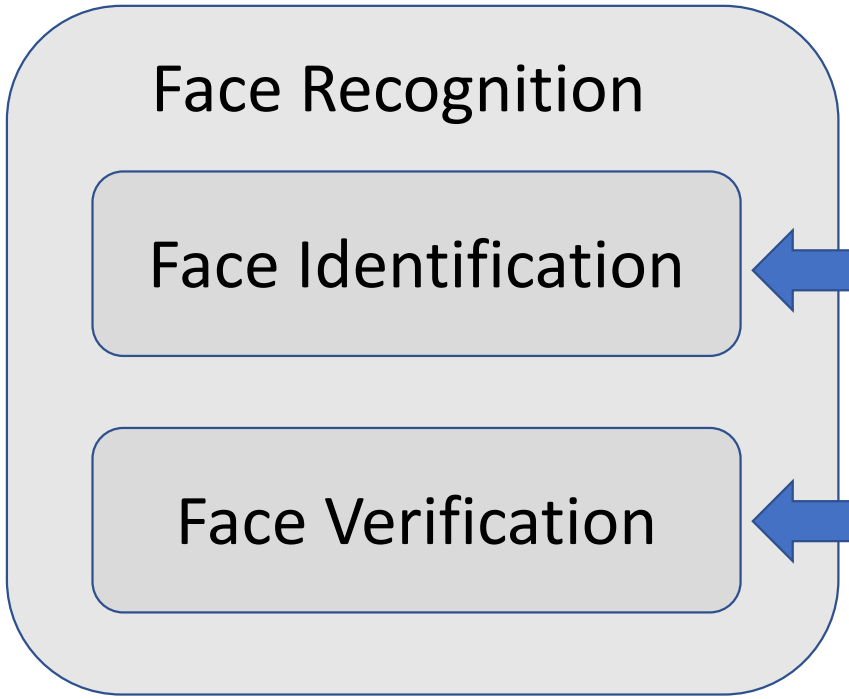
Computer Vision Applications

- Face Recognition: Problem Definition

- Step 1: Detect the face location on the image.
- Step 2: Align the face to the center of the image.
- Step 3: Perform face recognition on the image.

Computer Vision Applications

- Face Recognition: Problem Definition

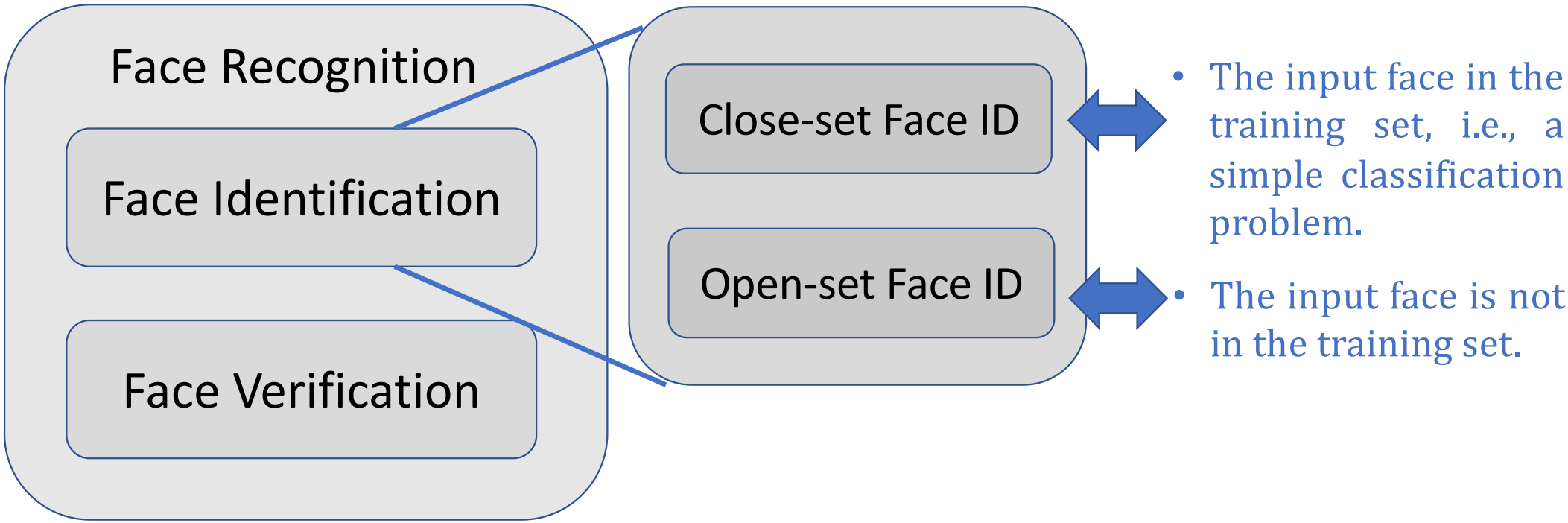


Two different tasks

- Given an image, classify the face ID.
- Given two images, classify whether they are the same person.

Computer Vision Applications

- Face Recognition



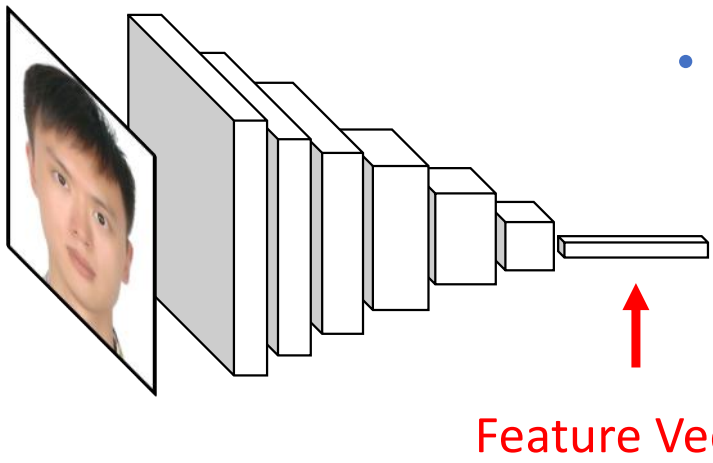
Computer Vision Applications

- Face Recognition

- The most commonly used method is Open-set Face ID:
  - The model is fixed, we cannot retrained the model.
  - When adding new person, a single image is used as the reference.

Computer Vision Applications

- Face Recognition

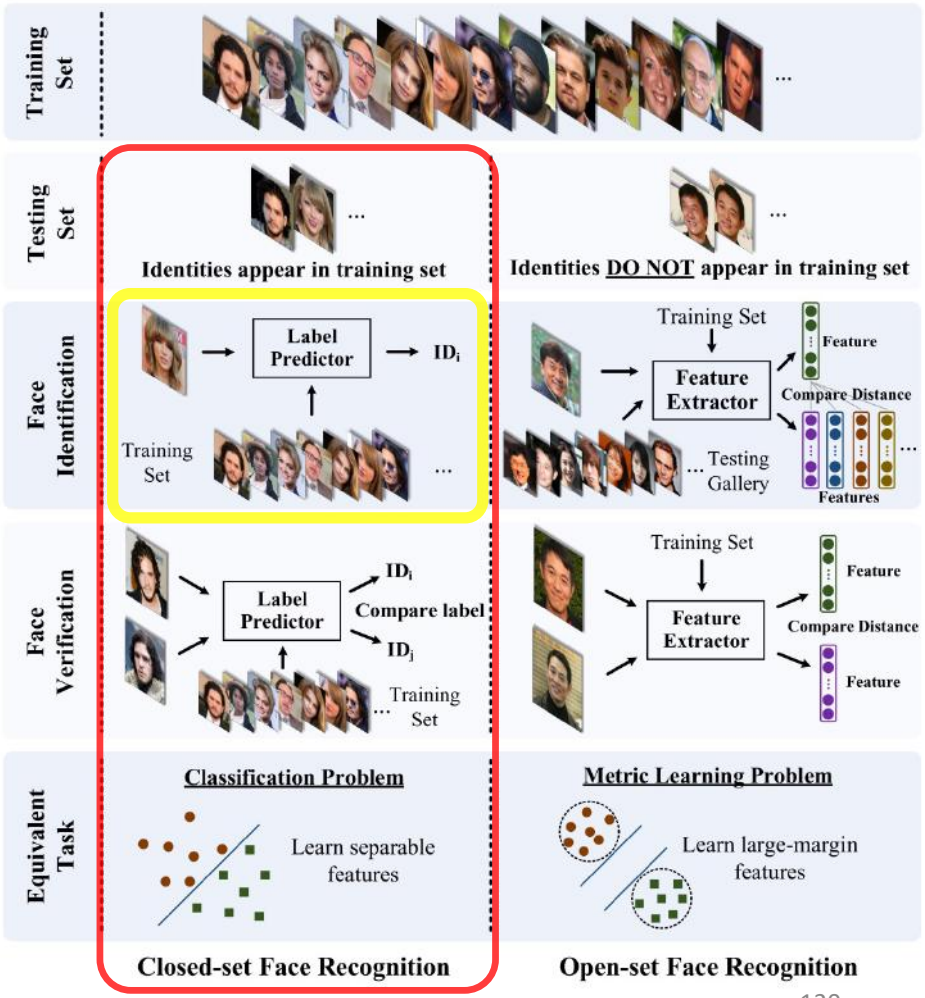


- Open-set Face ID :
  - Pretrained an image encoder to extract discriminative features from the images.
  - Different people's face have very different feature vectors. While the feature vectors of the same person are similar even the images have different lighting conditions.
  - Find the face ID by comparing the feature vectors in the dataset.

Computer Vision Applications

- Face Recognition

- **Close-set Face Identification:** A simple classification problem, similar with MNIST. Given an image, output the class label.
- **Close-set Face Verification:** Also a simple classification problem. Given two images, output two class labels and compare whether the labels are equal.
- **Open-set Face Identification:** A feature extraction problem, it first extracts all feature vectors of the images in the dataset, then when input an image, find the ID that has the best match feature vector.
- **Open-set Face Verification:** Also a feature extraction problem, it extracts the feature vectors of two images and computes the similarity score. If the score higher than a threshold, these two images are the same person.

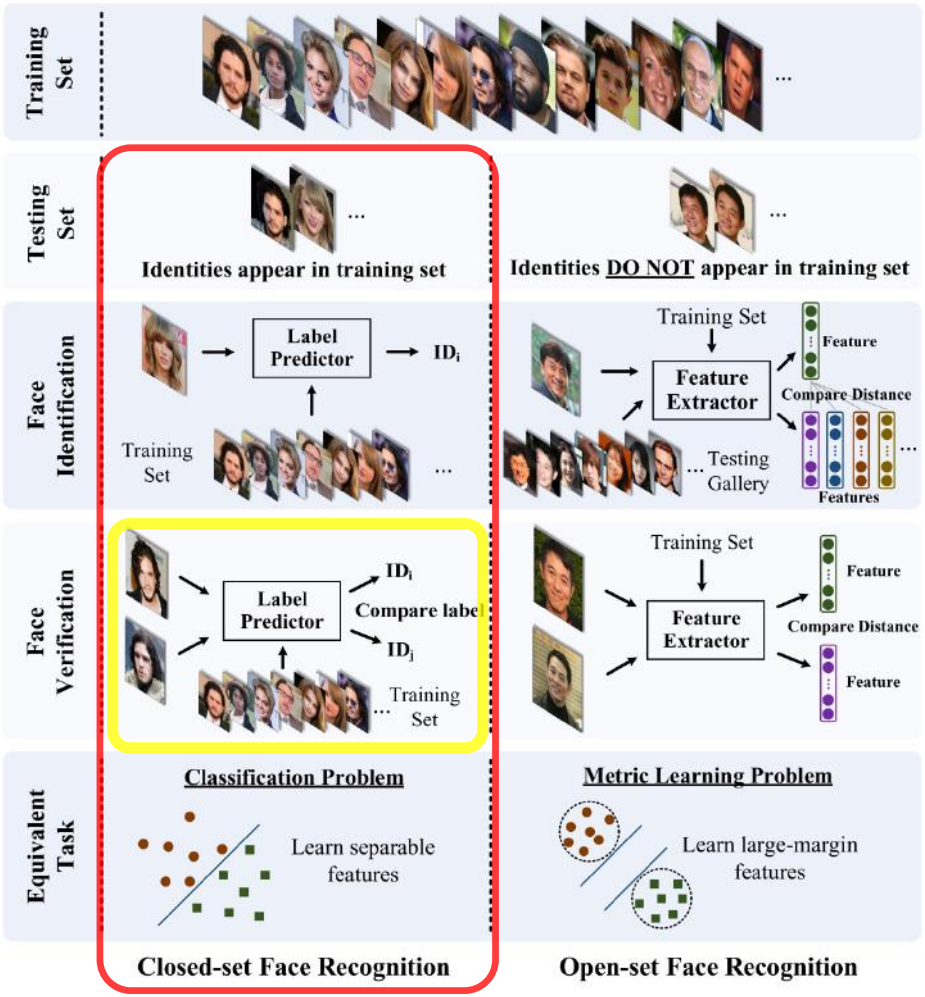


SphereFace: Deep Hypersphere Embedding for Face Recognition. *Weiyang Liu etal. CVPR. 2017.*

Computer Vision Applications

- Face Recognition

- **Close-set Face Identification:** A simple classification problem, similar with MNIST. Given an image, output the class label.
- **Close-set Face Verification:** Also a simple classification problem. Given two images, output two class labels and compare whether the labels are equal.
- **Open-set Face Identification:** A feature extraction problem, it first extracts all feature vectors of the images in the dataset, then when input an image, find the ID that has the best match feature vector.
- **Open-set Face Verification:** Also a feature extraction problem, it extracts the feature vectors of two images and computes the similarity score. If the score higher than a threshold, these two images are the same person.

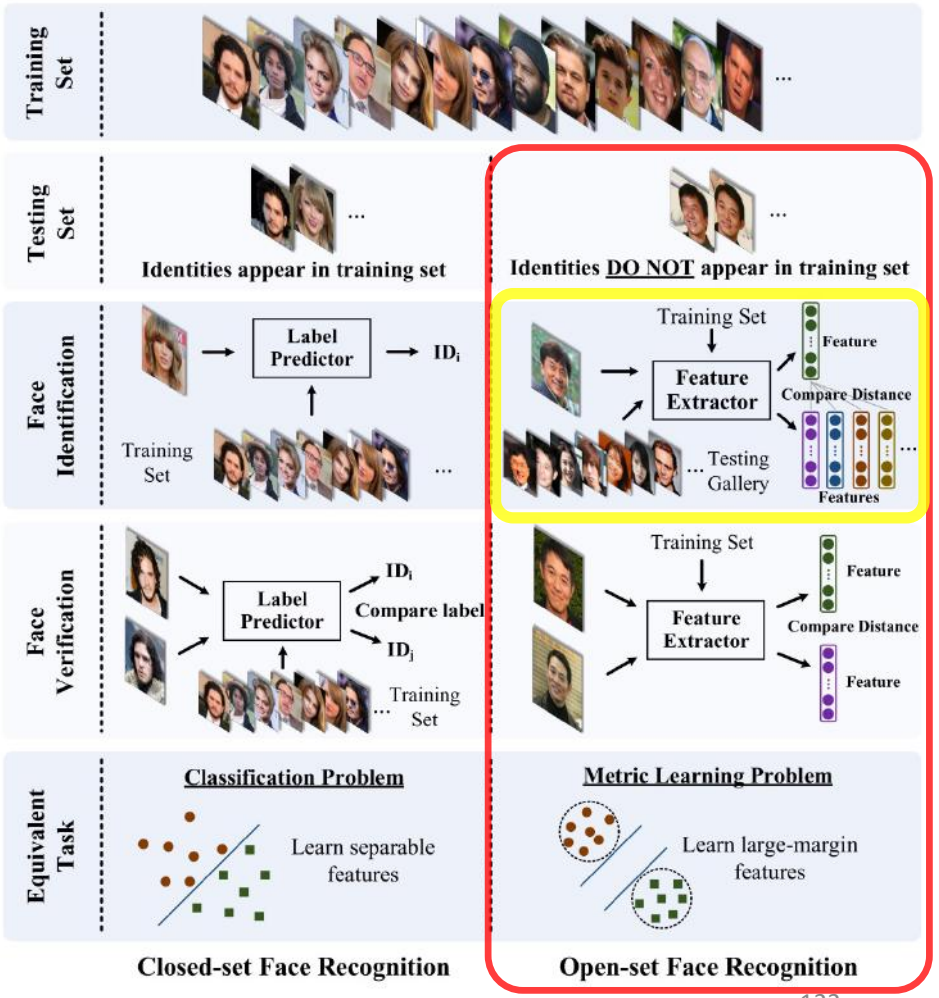


SphereFace: Deep Hypersphere Embedding for Face Recognition. *Weiyang Liu etal. CVPR. 2017.*

Computer Vision Applications

- Face Recognition

- **Close-set Face Identification:** A simple classification problem, similar with MNIST. Given an image, output the class label.
- **Close-set Face Verification:** Also a simple classification problem. Given two images, output two class labels and compare whether the labels are equal.
- **Open-set Face Identification:** A feature extraction problem, it first extracts all feature vectors of the images in the dataset, then when input an image, find the ID that has the best match feature vector.
- **Open-set Face Verification:** Also a feature extraction problem, it extracts the feature vectors of two images and computes the similarity score. If the score higher than a threshold, these two images are the same person.

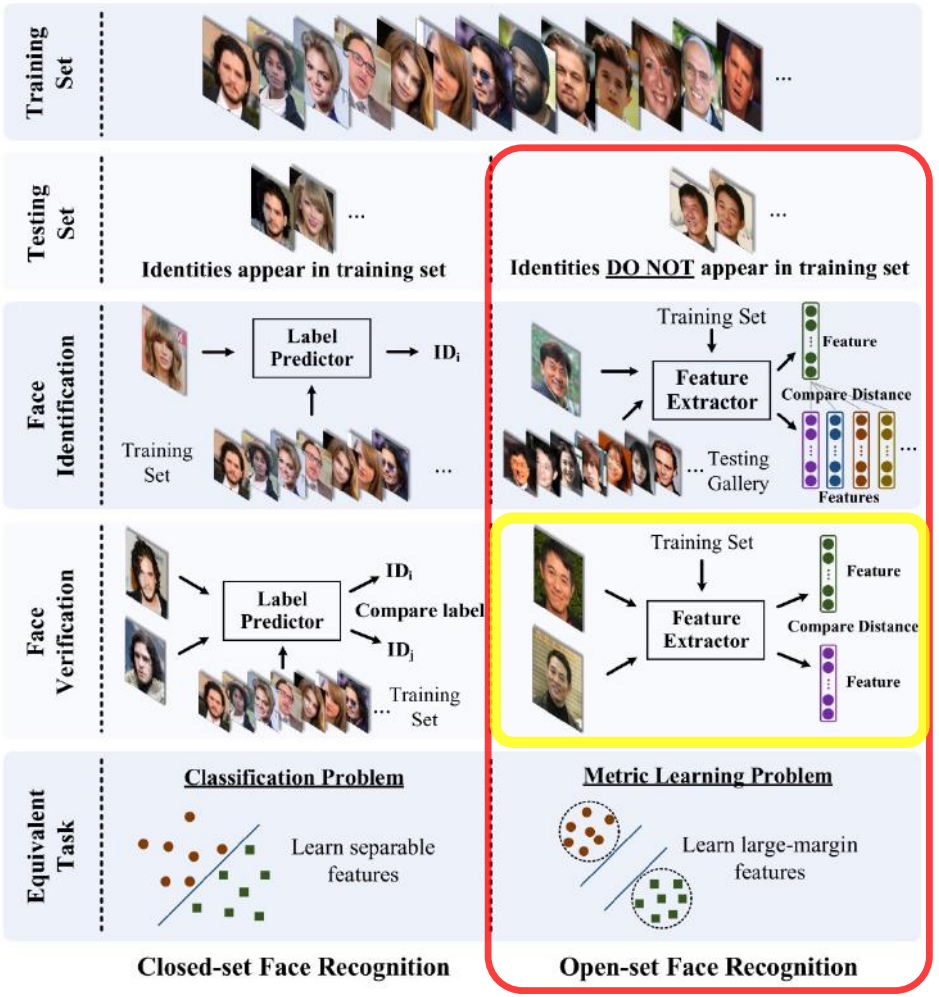


SphereFace: Deep Hypersphere Embedding for Face Recognition. *Weiyang Liu etal. CVPR. 2017.*

Computer Vision Applications

- Face Recognition

- **Close-set Face Identification:** A simple classification problem, similar with MNIST. Given an image, output the class label.
- **Close-set Face Verification:** Also a simple classification problem. Given two images, output two class labels and compare whether the labels are equal.
- **Open-set Face Identification:** A feature extraction problem, it first extracts all feature vectors of the images in the dataset, then when input an image, find the ID that has the best match feature vector.
- **Open-set Face Verification:** Also a feature extraction problem, it extracts the feature vectors of two images and computes the similarity score. If the score higher than a threshold, these two images are the same person.

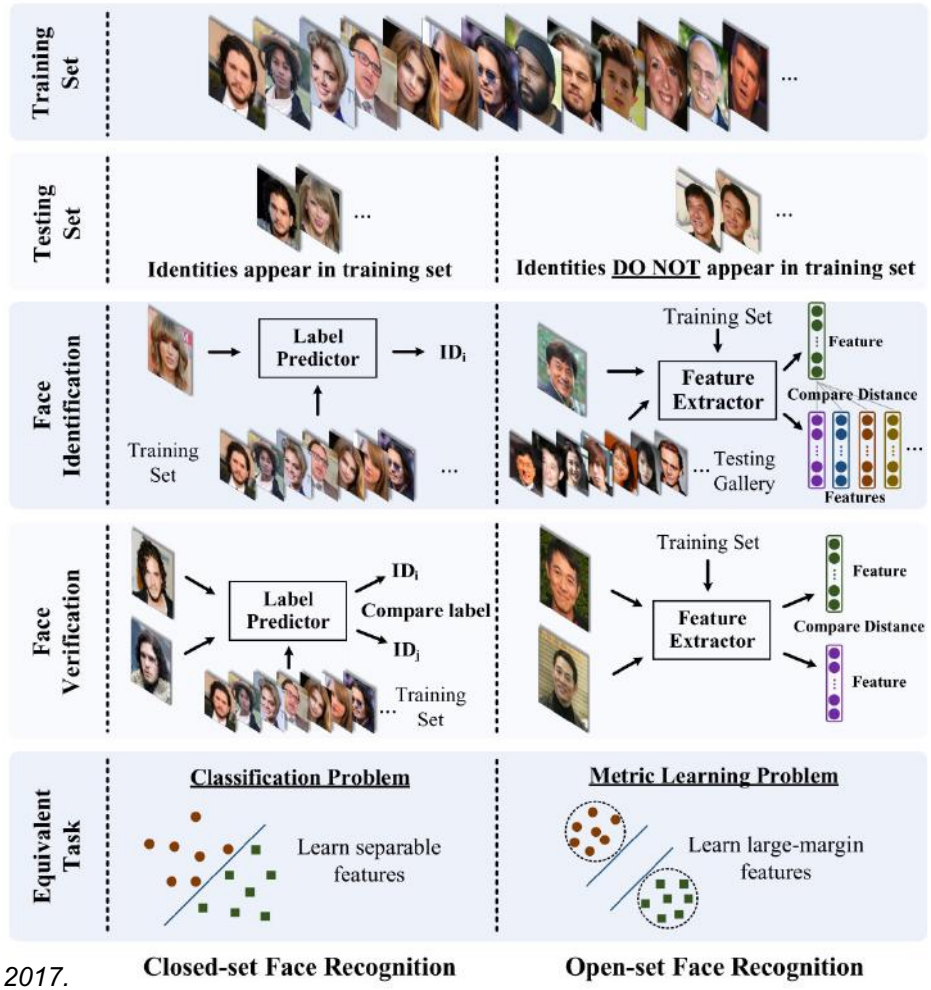


SphereFace: Deep Hypersphere Embedding for Face Recognition. *Weiyang Liu etal. CVPR. 2017.*

Computer Vision Applications

- Face Recognition

- **Close-set Face Identification:** A simple classification problem, similar with MNIST. Given an image, output the class label.
- **Close-set Face Verification:** Also a simple classification problem. Given two images, output two class labels and compare whether the labels are equal.
- **Open-set Face Identification:** A feature extraction problem, it first extracts all feature vectors of the images in the dataset, then when input an image, find the ID that has the best match feature vector.
- **Open-set Face Verification:** Also a feature extraction problem, it extracts the feature vectors of two images and computes the similarity score. If the score higher than a threshold, these two images are the same person.

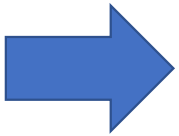


SphereFace: Deep Hypersphere Embedding for Face Recognition. *Weiyang Liu etal. CVPR. 2017.*

Computer Vision Applications

- Face Recognition

Design new network architecture
Design new loss function



Learn discriminative feature space by supervised learning.

Computer Vision Applications

- Face Recognition: Dataset

Image size of 112x112, 112x96, and 96x96 are commonly used in academic.
Larger image size usually has better accuracy but larger computation cost.

MS1M-refine-v2 112x112

MS1M-refine-v2 112x96

MS1M-refine-v2 96x96

Microsoft 1 Million Face Dataset

Computer Vision Applications

- Face Recognition

Always looking for better loss function … … …

… L-Softmax → SphereFace → ArcFace …

ICML 2016 CVPR 2017 CVPR 2018

ArcFace: Additive Angular Margin Loss for Deep Face Recognition. *Jiankang Deng etal. CVPR. 2018.*

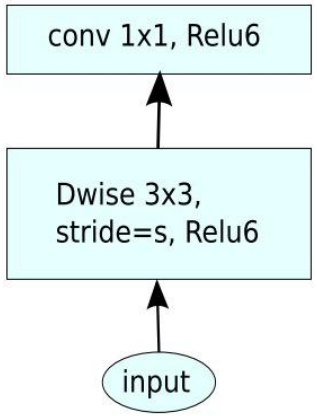
SphereFace: Deep Hypersphere Embedding for Face Recognition. *Weiyang Liu etal. CVPR. 2017.*

Computer Vision Applications

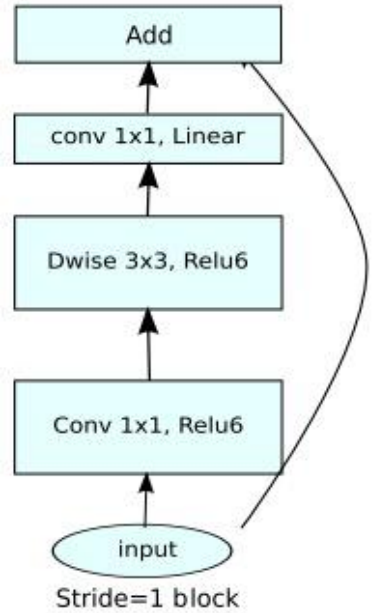
- Face Recognition: MobileFaceNet = ArcFace + MobileNetV2

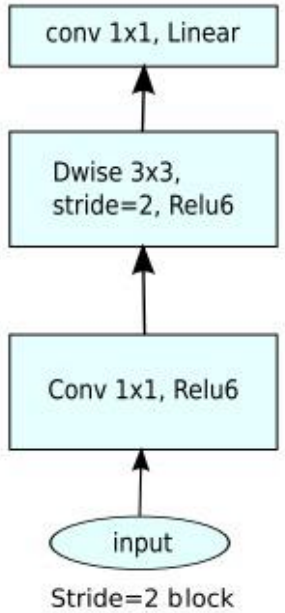
MOBILEFACENET ARCHITECTURE FOR FEATURE EMBEDDING

Input	Operator	t	c	n	s
$112^2 \times 3$	conv3x3	-	64	1	2
$56^2 \times 64$	depthwise conv3x3	-	64	1	1
$56^2 \times 64$	bottleneck	2	64	5	2
$28^2 \times 64$	bottleneck	4	128	1	2
$14^2 \times 128$	bottleneck	2	128	6	1
$14^2 \times 128$	bottleneck	4	128	1	2
$7^2 \times 128$	bottleneck	2	128	2	1
$7^2 \times 128$	conv1x1	-	512	1	1
$7^2 \times 512$	linear GDConv7x7	-	512	1	1
$1^2 \times 512$	linear conv1x1	-	128	1	1



MobileNetV1





MobileNetV2

t: expansion factor c: num of channels n: num of repeat s: stride

Model size < 4MB， CPU runtime < 24ms

Computer Vision Applications

- More and more …

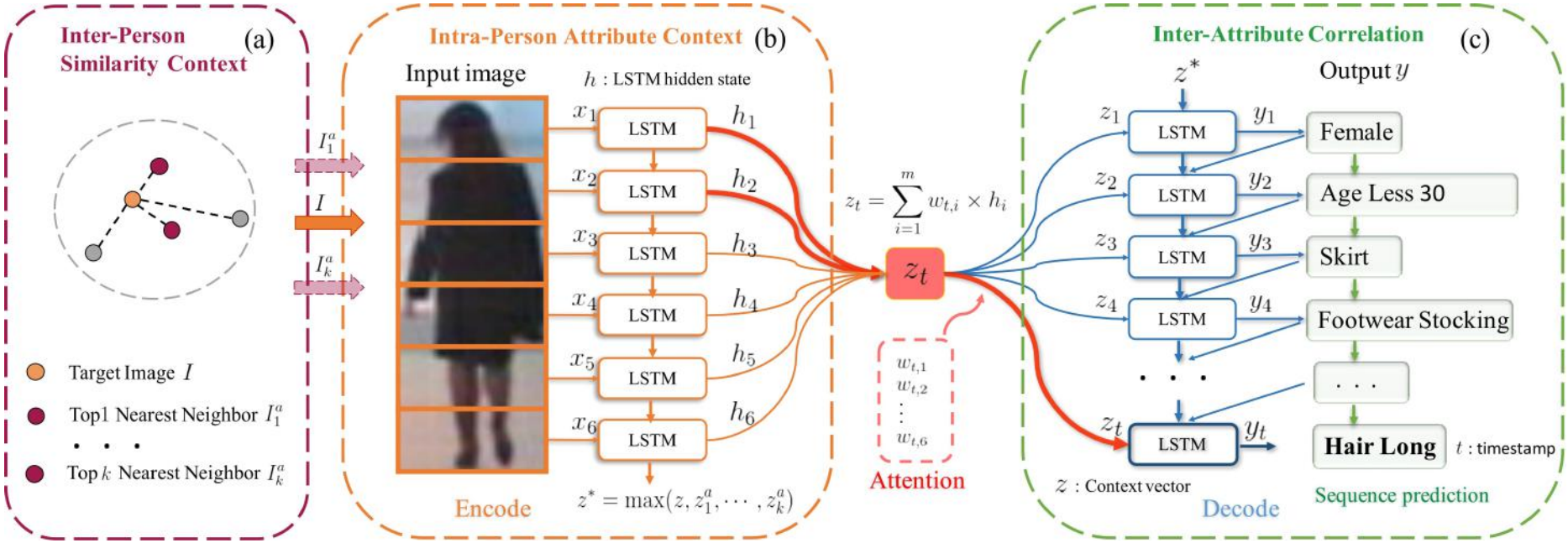
Person Re-identification（ReID）

- Track person in different cameras

Computer Vision Applications

- More and more …

Person Attribute Classification

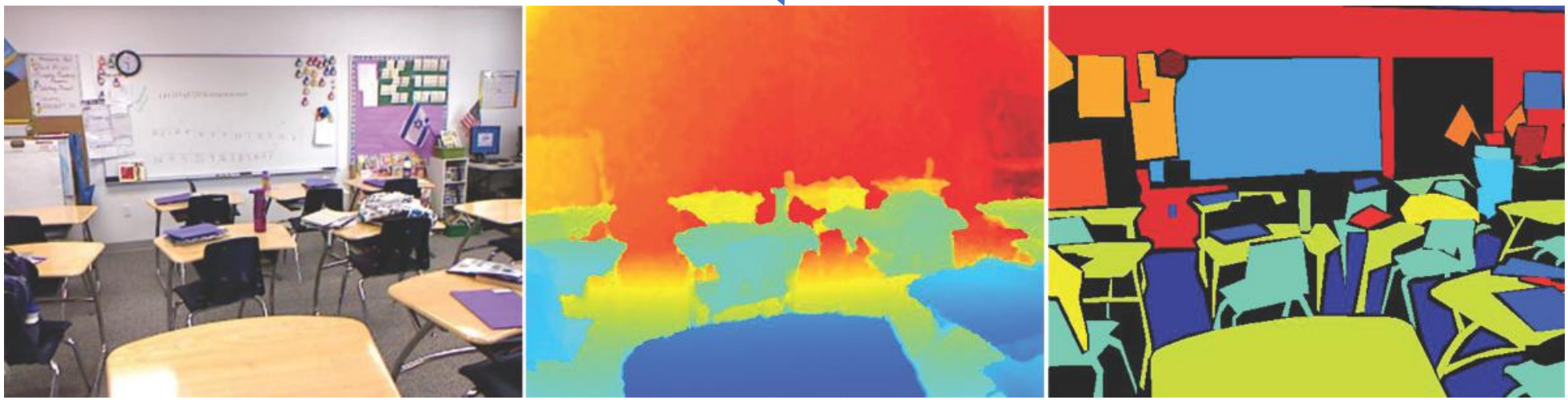


Attribute Recognition by Joint Recurrent Learning of Context and Correlation. *Jingya Wang etal. ICCV. 2017.*

Computer Vision Applications

- More and more …

Depth Estimation



NYU Depth Dataset V2

Computer Vision Applications

- More and more …

Style Transfer

Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization. *X. Huang, S. Belongie. ICCV 2017.*

Computer Vision Applications

- More and more …

Super-resolution

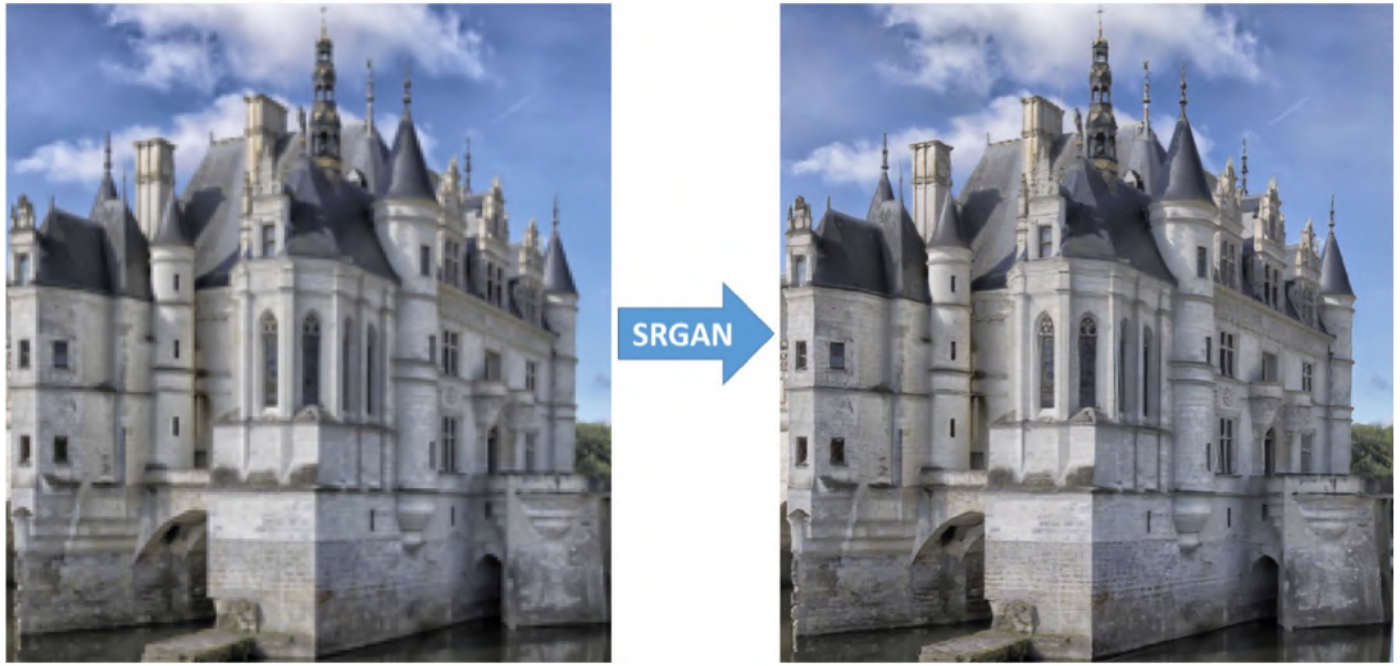


Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. *C. Ledig, L. Theis et al. CVPR 2017*.

Computer Vision Applications

- More and more …

Image-to-image Translation

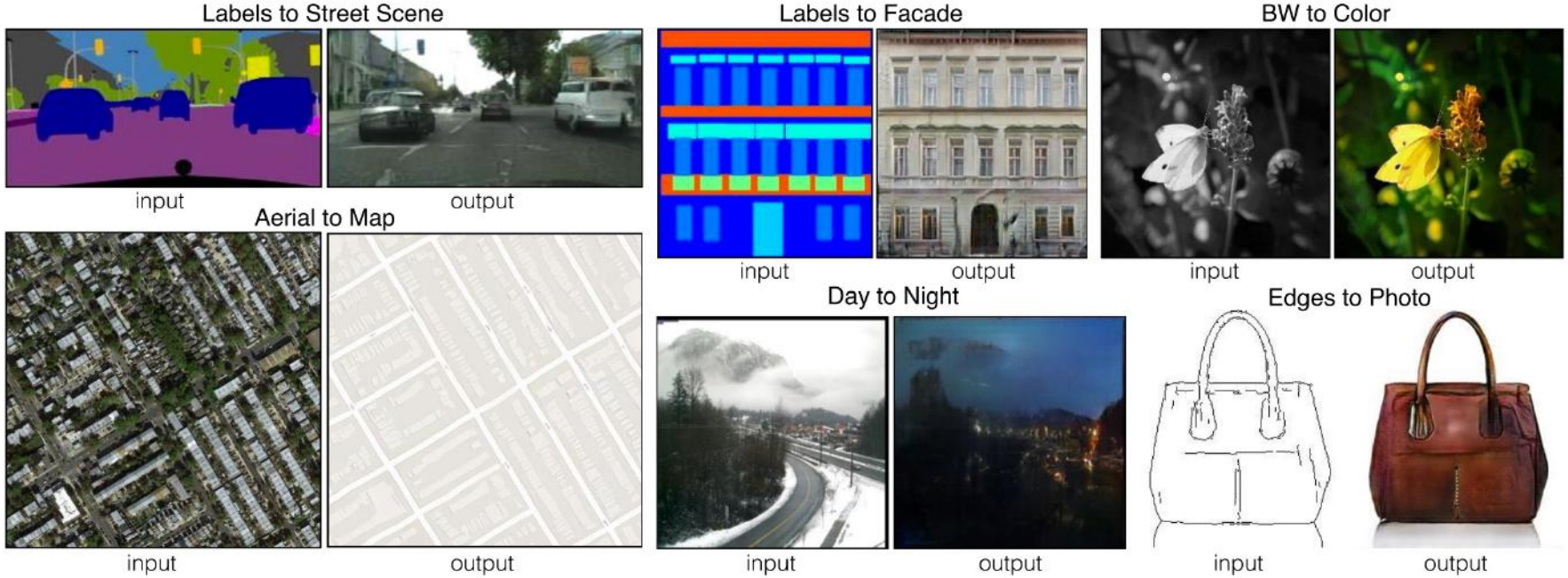
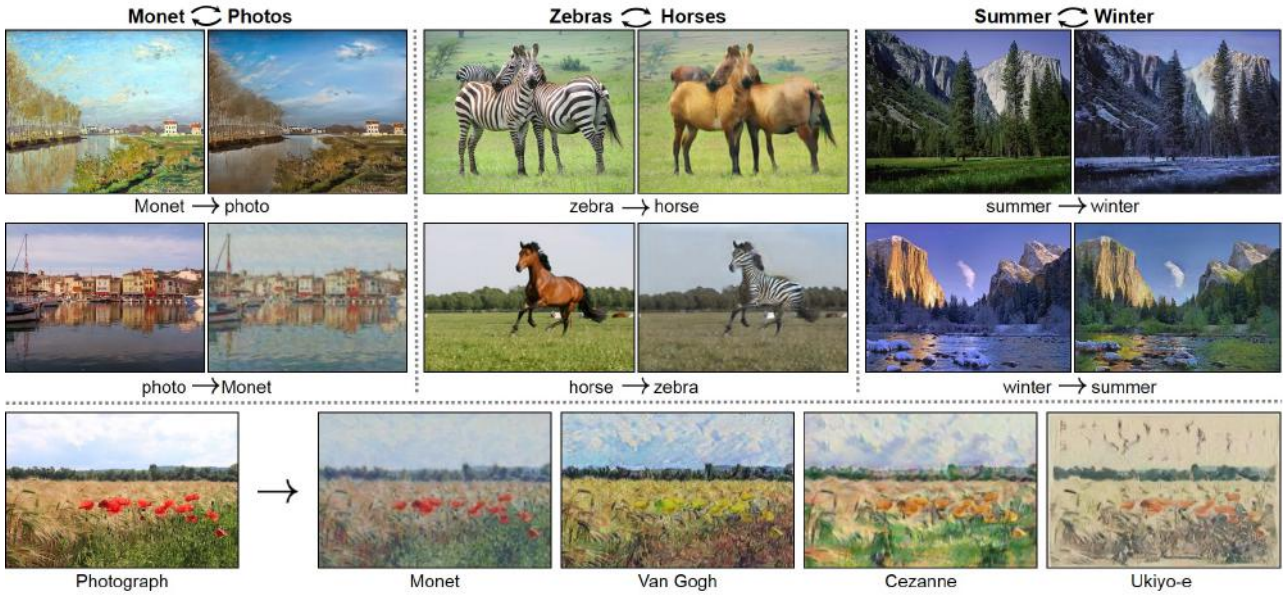


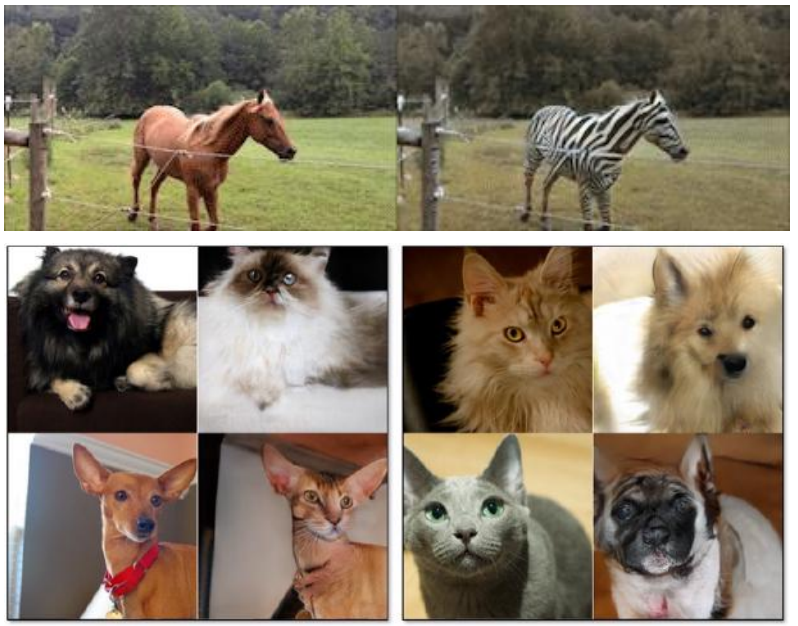
Image-to-Image Translation with Conditional Adversarial Networks. *P. Isola, J. Zhu et al. CVPR 2017.*

Computer Vision Applications

- More and more …

Unsupervised/Unpaired Image-to-image Translation



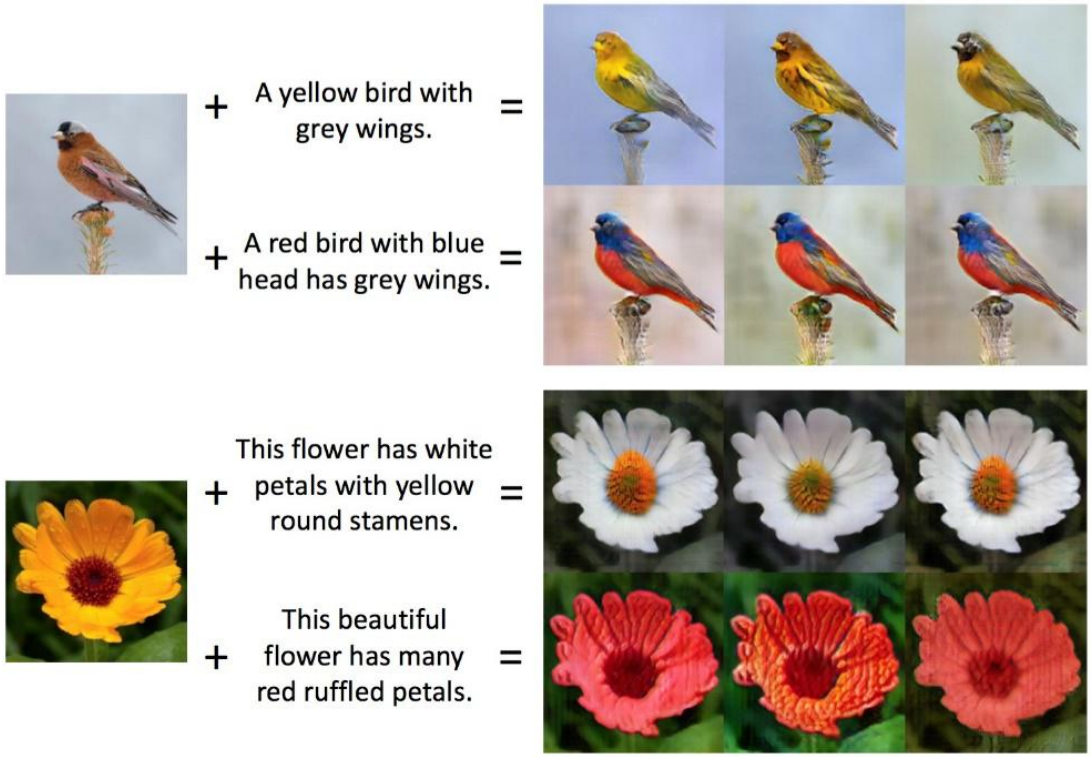


Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. *J. Zhu, T. Park et al. ICCV 2017.*

Computer Vision Applications

- More and more …

Semantic Image Synthesis



Semantic Image Synthesis via Adversarial Learning. H. Dong, S. Yu et al. ICCV 2017.

Computer Vision Applications

- More and more …

We will introduce the generative works in the GAN lesson.

北京大学
PEKING UNIVERSITY

Summary

Convolutional Neural Networks

- Motivation
  - Curse of dimensionality, parameter sharing …
- Convolutional Algorithms
  - Convolution, channel, receptive field, dilated convolution …
- Pooling Algorithms
  - Max pooling, mean pooling, pyramid pooling …
- Hierarchical Representation Learning
  - Large receptive field, feature representation …
- Convolutional Architectures
  - VGG, ResNet, MobileNet, SqueezeNet …
- Transposed Convolutional Algorithms
  - Decoding, common transposed convolution …
- Computer Vision Applications
  - Detection, segmentation, recognition, pose estimation …

- Exercise 1:
  - Use TensorLayer to classify the CIFAR10 dataset
- Exercise 2: (Optional)
  - Choice an application and implement it

北京大学
PEKING UNIVERSITY

Questions?