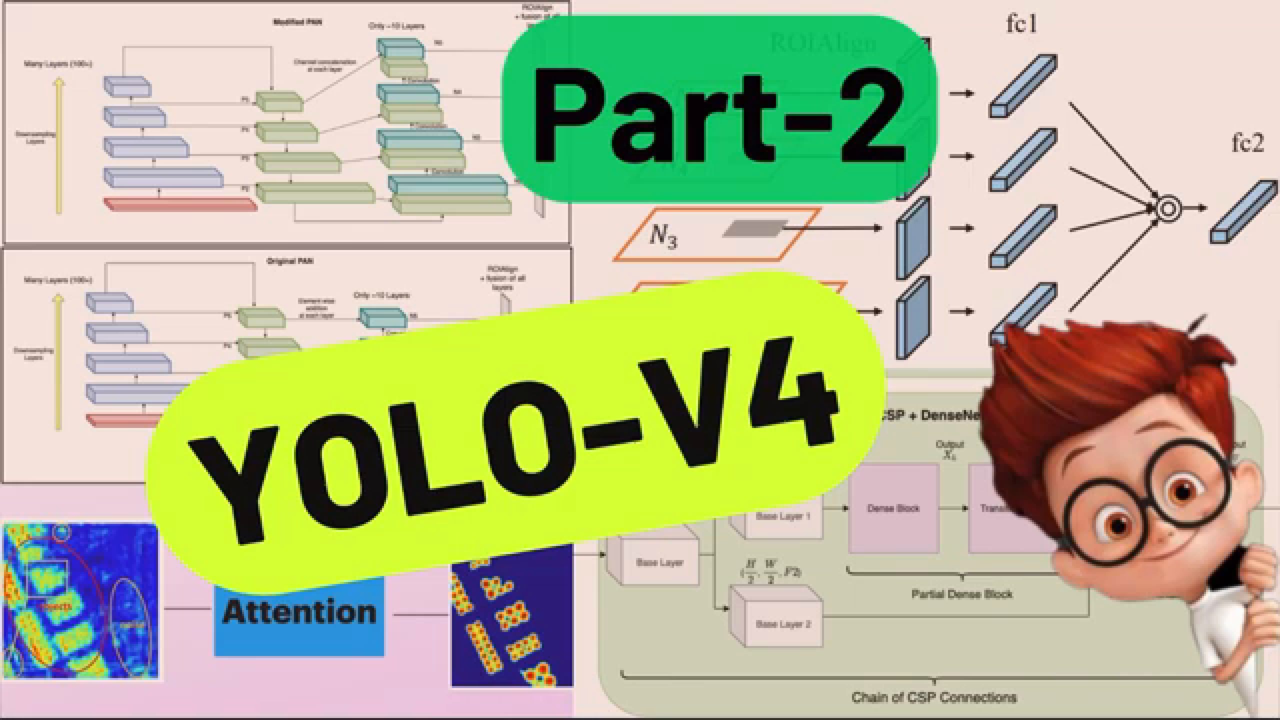
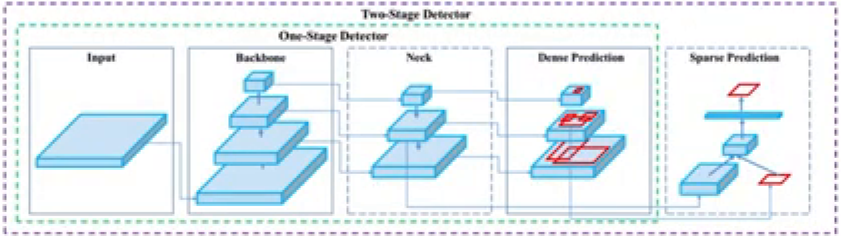

Part-2
YOLO-V4
Attention
Modified PAN
Original PAN
fc1
fc2
Dense Block
Base Layer 1
Base Layer
Base Layer 2
Partial Dense Block
Chain of CSP Connections

Components



Input: { Image, Patches, Image Pyramid, … }

Backbone: { VGG16 [68], ResNet-50 [26], ResNeXt-101 [86], Darknet53 [63], … }

Neck: { FPN [44], PANet [49], Bi-FPN [77], … }

Head:

Dense Prediction: { RPN [64], YOLO [61, 62, 63], SSD [50], RetinaNet [45], FCOS [78], … }

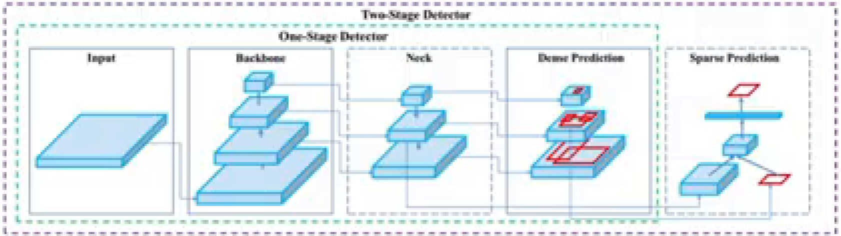
Sparse Prediction: { Faster R-CNN [64], R-FCN [9], … }

CSPDarkNet53 (Backbone) => SSP + PANet (Neck)=> YOLOv3 (head)

BoF & BoS

	Backbone	Detector
Bag of Freebies (BoF)	• CutMix • Mosaic data augmentation • DropBlock • Class label smoothing	• CIoU-loss • Cross mini-Batch Normalization • DropBlock • Mosaic data augmentation • Self-Adversarial Training • Multiple anchors for a single ground truth • Cosine annealing scheduler • Optimal hyperparameters • Random training shapes
Bag of Specials (BoS)	• Mish activation • Cross-stage partial connections (CSP) • Multi-input weighted residual connections (MiWRC)	• Mish activation • SPP-block • SAM-block • PAN path-aggregation block • DIoU-NMS

Components



Input: { Image, Patches, Image Pyramid, … }

Backbone: { VGG16 [68], ResNet-50 [26], ResNeXt-101 [86], Darknet53 [63], … }

Neck: { FPN [44], PANet [49], Bi-FPN [77], … }

Head:

- **Dense Prediction:** { RPN [64], YOLO [61, 62, 63], SSD [50], RetinaNet [45], FCOS [78], … }
- **Sparse Prediction:** { Faster R-CNN [64], R-FCN [9], … }

CSPDarkNet53 (Backbone) => SSP + PANet (Neck)=> YOLOv3 (head)

In this video..

- Backbone
  - DenseNet
  - CSPNet
  - CSPDarknet-53
- Neck
  - FPN
  - SPP
  - PAN
- Spatial Attention Module

Backbone

- **Backbone**
  - **DenseNet**
  - **CSPNet**
  - **CSPDarknet-53**

Table 1: Parameters of neural networks for image classification.

Backbone model	Input network resolution	Receptive field size	Parameters	Average size of layer output (WxHxC)	BFLOPs (512x512 network resolution)	FPS (GPU RTX 2070)
CSPResNext50	512x512	425x425	20.6 M	**1058 K**	31 (15.5 FMA)	62
CSPDarknet53	512x512	725x725	**27.6 M**	950 K	52 (26.0 FMA)	**66**
EfficientNet-B3 (ours)	512x512	**1311x1311**	12.0 M	668 K	11 (5.5 FMA)	26

Backbone

- **Backbone**
  - **DenseNet**
  - **CSPNet**
  - **CSPDarknet-53**

Table 1: Parameters of neural networks for image classification.

Backbone model	Input network resolution	Receptive field size	Parameters	Average size of layer output (WxHxC)	BFLOPs (512x512 network resolution)	FPS (GPU RTX 2070)
CSPResNext50	512x512	425x425	20.6 M	**1058 K**	31 (15.5 FMA)	62
CSPDarknet53	512x512	725x725	**27.6 M**	950 K	52 (26.0 FMA)	**66**
EfficientNet-B3 (ours)	512x512	**1311x1311**	12.0 M	668 K	11 (5.5 FMA)	26

Backbone

- **Backbone**
  - **DenseNet**
  - **CSPNet**
  - **CSPDarknet-53**

Table 1: Parameters of neural networks for image classification.

Backbone model	Input network resolution	Receptive field size	Parameters	Average size of layer output (WxHxC)	BFLOPs (512x512 network resolution)	FPS (GPU RTX 2070)
CSPResNext50	512x512	425x425	20.6 M	**1058 K**	31 (15.5 FMA)	62
CSPDarknet53	512x512	725x725	**27.6 M**	950 K	52 (26.0 FMA)	**66**
EfficientNet-B3 (ours)	512x512	**1311x1311**	12.0 M	668 K	11 (5.5 FMA)	26

Backbone

- **Backbone**
  - **DenseNet**
  - **CSPNet**
  - **CSPDarknet-53**

Table 1: Parameters of neural networks for image classification.

Backbone model	Input network resolution	Receptive field size	Parameters	Average size of layer output (WxHxC)	BFLOPs (512x512 network resolution)	FPS (GPU RTX 2070)
CSPResNext50	512x512	425x425	20.6 M	**1058 K**	31 (15.5 FMA)	62
CSPDarknet53	512x512	725x725	**27.6 M**	950 K	52 (26.0 FMA)	**66**
EfficientNet-B3 (ours)	512x512	**1311x1311**	12.0 M	668 K	11 (5.5 FMA)	26

Backbone

- **Backbone**
  - **DenseNet**
  - **CSPNet**
  - **CSPDarknet-53**

Table 1: Parameters of neural networks for image classification.

Backbone model	Input network resolution	Receptive field size	Parameters	Average size of layer output (WxHxC)	BFLOPs (512x512 network resolution)	FPS (GPU RTX 2070)
CSPResNext50	512x512	425x425	20.6 M	**1058 K**	31 (15.5 FMA)	62
CSPDarknet53	512x512	725x725	**27.6 M**	950 K	52 (26.0 FMA)	**66**
EfficientNet-B3 (ours)	512x512	**1311x1311**	12.0 M	668 K	11 (5.5 FMA)	26

Backbone

- **Backbone**
  - **DenseNet**
  - **CSPNet**
  - **CSPDarknet-53**

Table 1: Parameters of neural networks for image classification.

Backbone model	Input network resolution	Receptive field size	Parameters	Average size of layer output (WxHxC)	BFLOPs (512x512 network resolution)	FPS (GPU RTX 2070)
CSPResNext50	512x512	425x425	20.6 M	**1058 K**	31 (15.5 FMA)	62
CSPDarknet53	512x512	725x725	**27.6 M**	950 K	52 (26.0 FMA)	**66**
EfficientNet-B3 (ours)	512x512	**1311x1311**	12.0 M	668 K	11 (5.5 FMA)	26

Backbone

- **Backbone**
  - **DenseNet**
  - **CSPNet**
  - **CSPDarknet-53**

Table 1: Parameters of neural networks for image classification.

Backbone model	Input network resolution	Receptive field size	Parameters	Average size of layer output (WxHxC)	BFLOPs (512x512 network resolution)	FPS (GPU RTX 2070)
CSPResNext50	512x512	425x425	20.6 M	**1058 K**	31 (15.5 FMA)	62
CSPDarknet53	512x512	725x725	**27.6 M**	950 K	52 (26.0 FMA)	**66**
EfficientNet-B3 (ours)	512x512	**1311x1311**	12.0 M	668 K	11 (5.5 FMA)	26

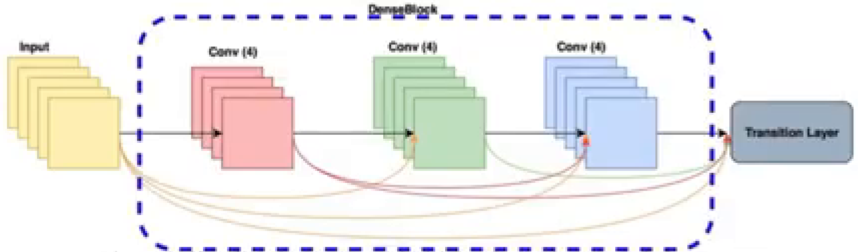
Backbone

- **Backbone**
  - **DenseNet**
  - **CSPNet**
  - **CSPDarknet-53**

Table 1: Parameters of neural networks for image classification.

Backbone model	Input network resolution	Receptive field size	Parameters	Average size of layer output (WxHxC)	BFLOPs (512x512 network resolution)	FPS (GPU RTX 2070)
CSPResNext50	512x512	425x425	20.6 M	**1058 K**	31 (15.5 FMA)	62
CSPDarknet53	512x512	725x725	**27.6 M**	950 K	52 (26.0 FMA)	**66**
EfficientNet-B3 (ours)	512x512	**1311x1311**	12.0 M	668 K	11 (5.5 FMA)	26

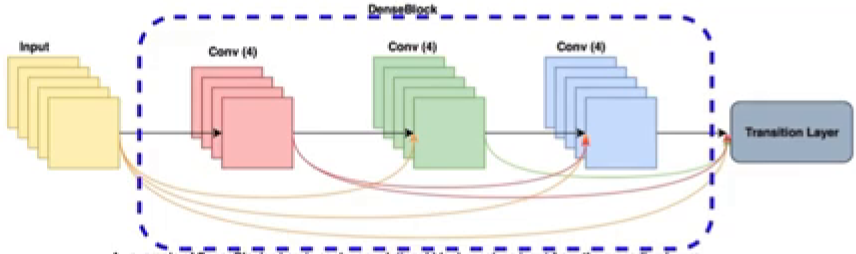
Dense Block



An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

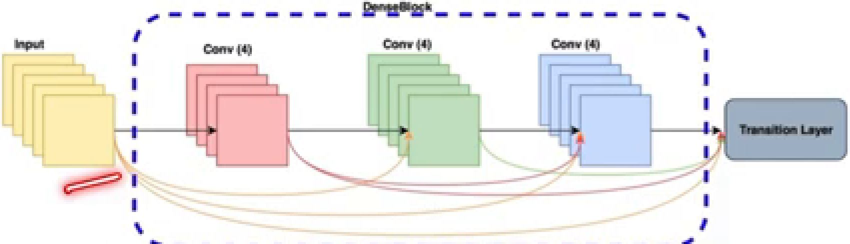
Dense Block



An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

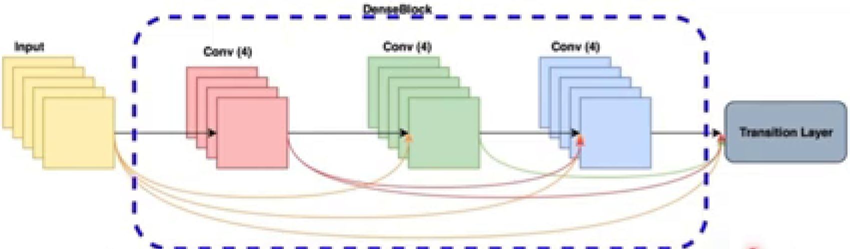
Dense Block



An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

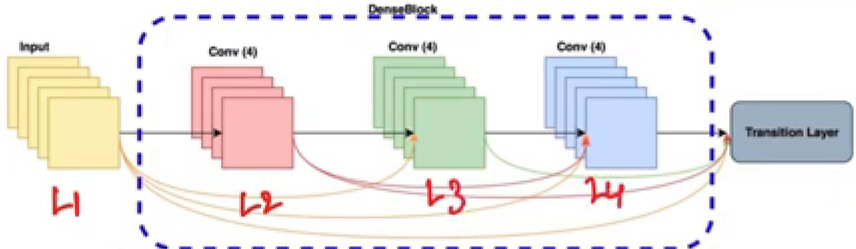
Dense Block



An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

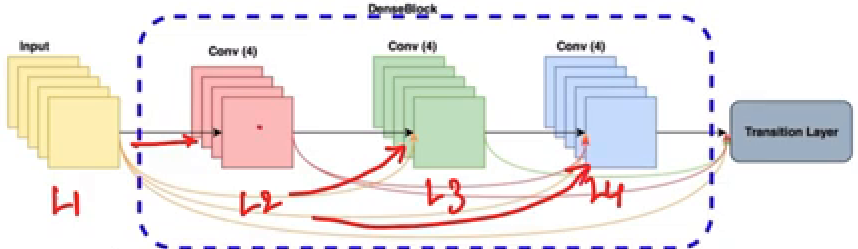
Dense Block



An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

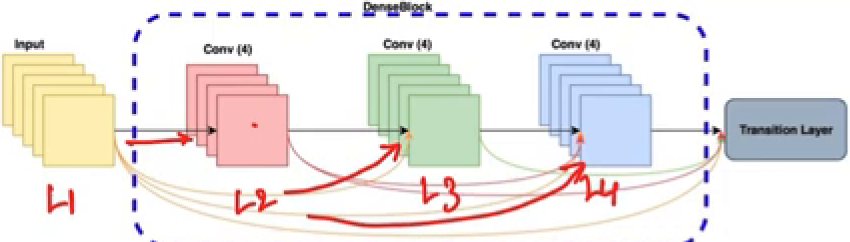
Dense Block



An example of DenseBlock wherein each convolutional block receives input from the preceding layers.
The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

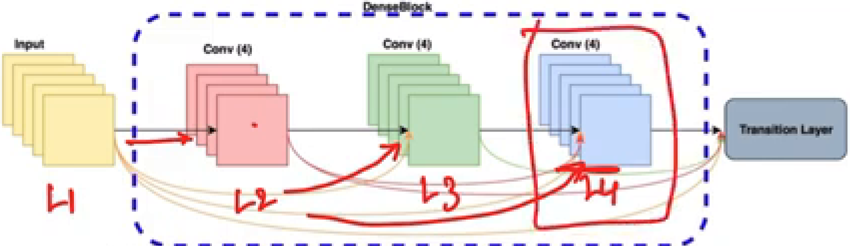
Dense Block



An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

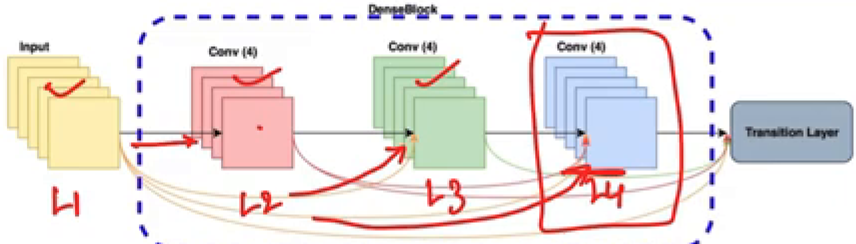
Dense Block



An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$
\begin{aligned}
x_1 &= w_1 * x_0 \\
x_2 &= w_2 * [x_0, x_1] \\
&\vdots \\
x_k &= w_k * [x_0, x_1, \ldots, x_{k-1}]
\end{aligned}
$$

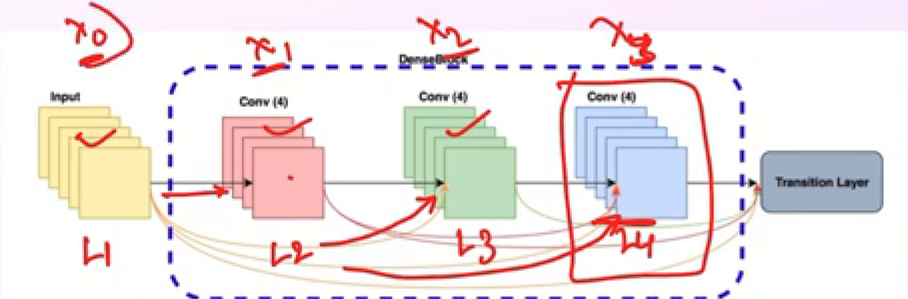
Dense Block



An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

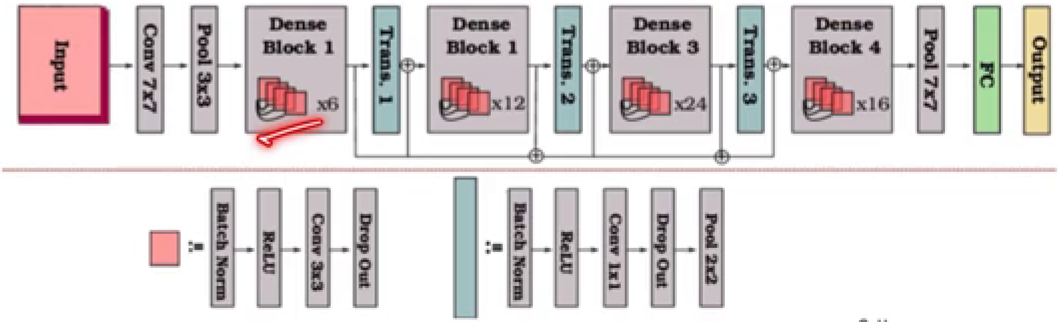
Dense Block



An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

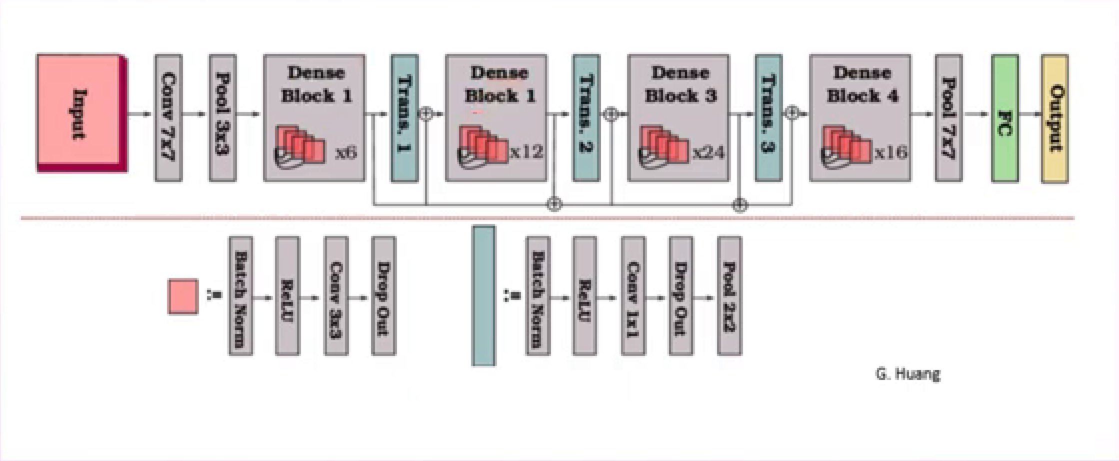
$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

DenseNet



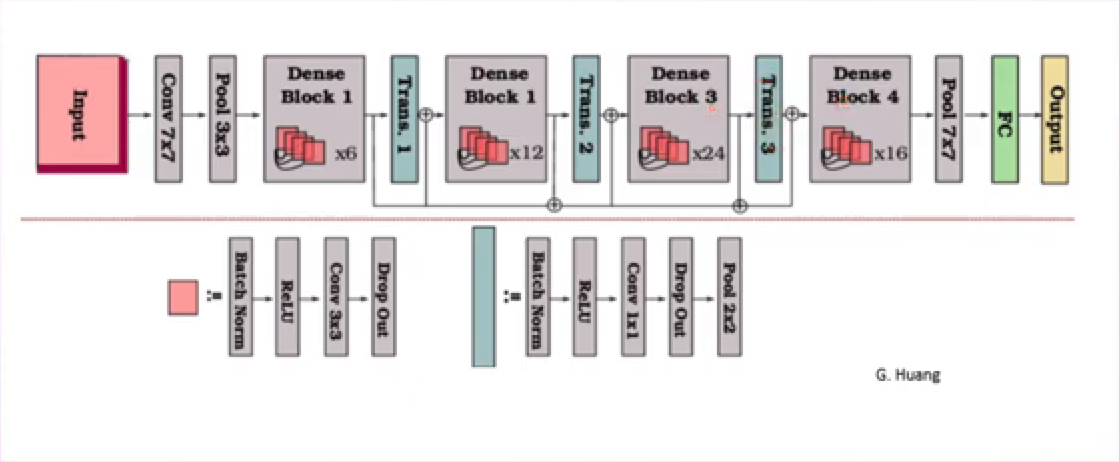
G. Huang

DenseNet

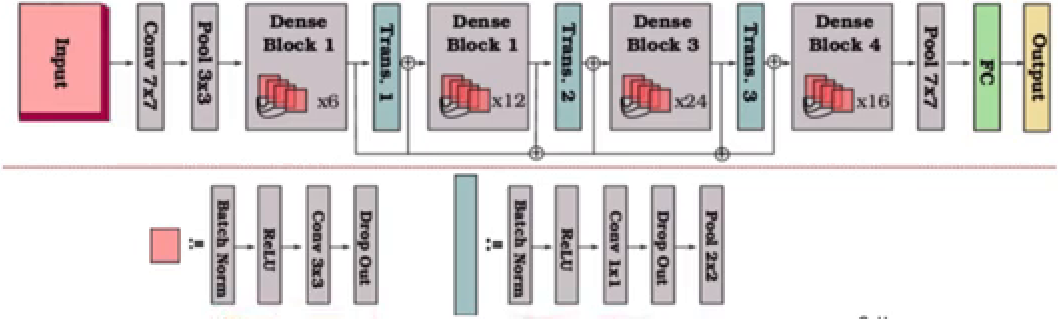


Input
Conv 7x7
Pool 3x3
Dense Block 1
x6
Trans. 1
Dense Block 1
x12
Trans. 2
Dense Block 3
x24
Trans. 3
Dense Block 4
x16
Pool 7x7
FC
Output
=:
Batch Norm
ReLU
Conv 3x3
Drop Out
=:
Batch Norm
ReLU
Conv 1x1
Drop Out
Pool 2x2
G. Huang

DenseNet

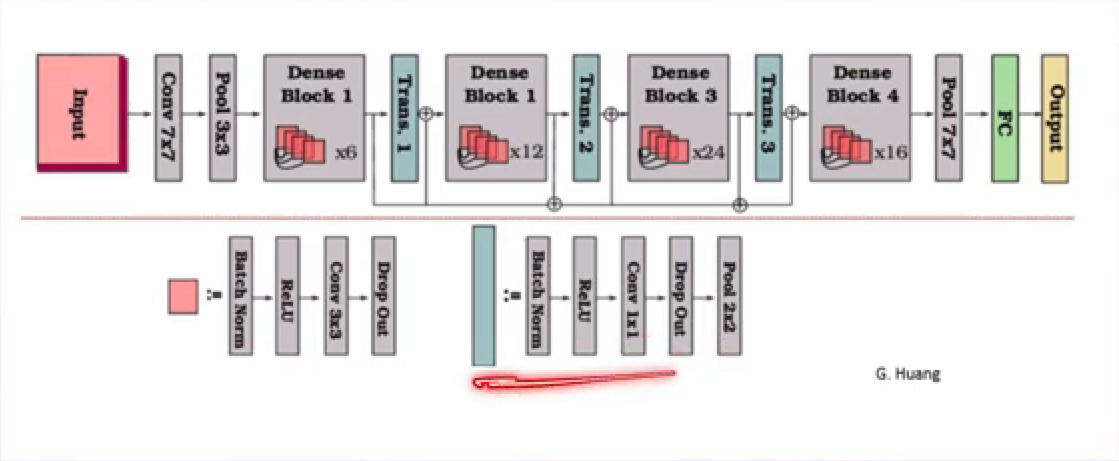

Input
Conv 7x7
Pool 3x3
Dense Block 1
x6
Trans. 1
Dense Block 1
x12
Trans. 2
Dense Block 3
x24
Trans. 3
Dense Block 4
x16
Pool 7x7
FC
Output
:=
Batch Norm
ReLU
Conv 3x3
Drop Out
:=
Batch Norm
ReLU
Conv 1x1
Drop Out
Pool 2x2
G. Huang

DenseNet

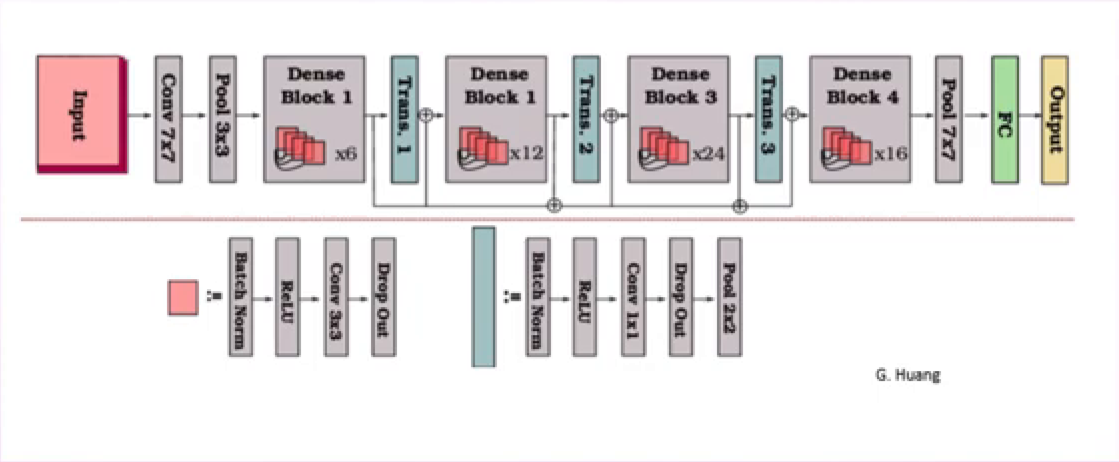


G. Huang

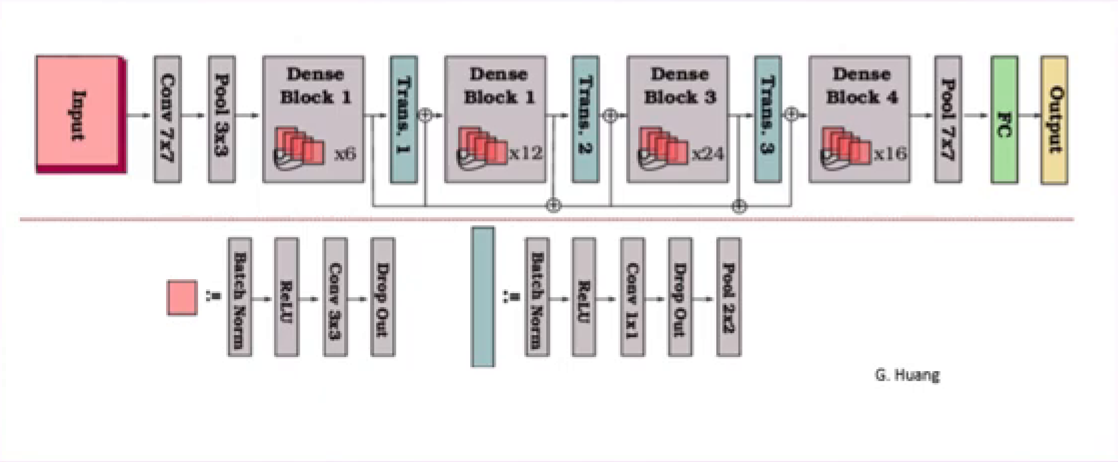
DenseNet


Input
Conv 7x7
Pool 3x3
Dense Block 1
x6
Trans. 1
Dense Block 1
x12
Trans. 2
Dense Block 3
x24
Trans. 3
Dense Block 4
x16
Pool 7x7
FC
Output
:=
Batch Norm
ReLU
Conv 3x3
Drop Out
:=
Batch Norm
ReLU
Conv 1x1
Drop Out
Pool 2x2
G. Huang

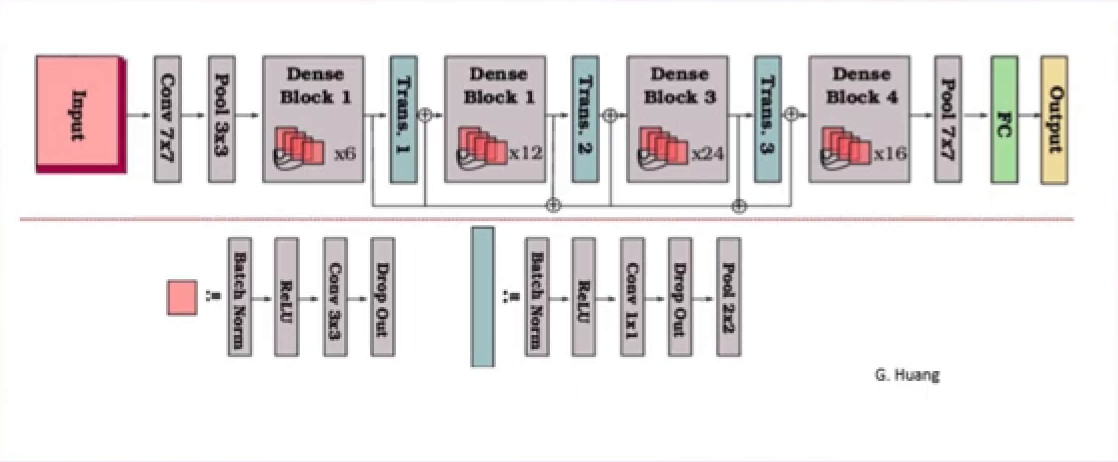
DenseNet


Input
Conv 7x7
Pool 3x3
Dense Block 1
x6
Trans. 1
Dense Block 1
x12
Trans. 2
Dense Block 3
x24
Trans. 3
Dense Block 4
x16
Pool 7x7
FC
Output
:=
Batch Norm
ReLU
Conv 3x3
Drop Out
:=
Batch Norm
ReLU
Conv 1x1
Drop Out
Pool 2x2
G. Huang

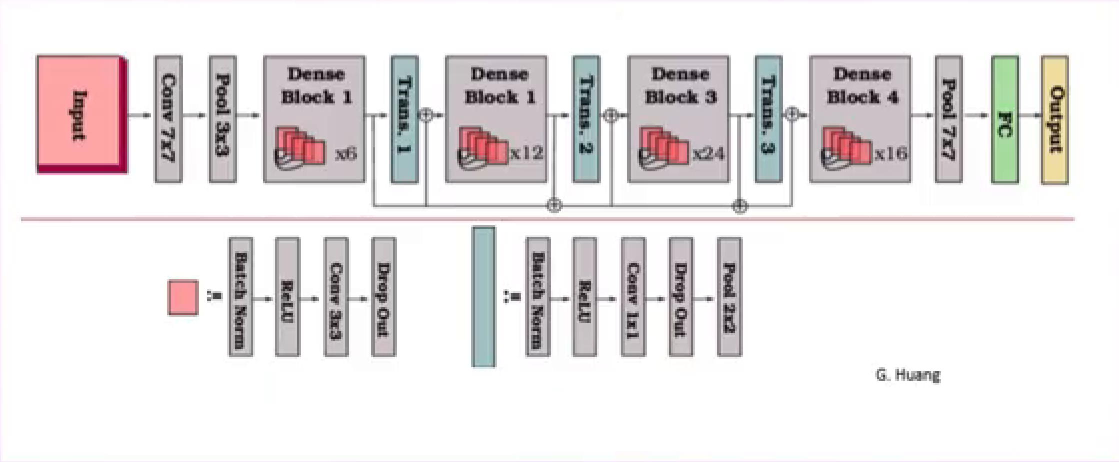
DenseNet


Input
Conv 7x7
Pool 3x3
Dense Block 1
x6
Trans. 1
Dense Block 1
x12
Trans. 2
Dense Block 3
x24
Trans. 3
Dense Block 4
x16
Pool 7x7
FC
Output
:=
Batch Norm
ReLU
Conv 3x3
Drop Out
:=
Batch Norm
ReLU
Conv 1x1
Drop Out
Pool 2x2
G. Huang

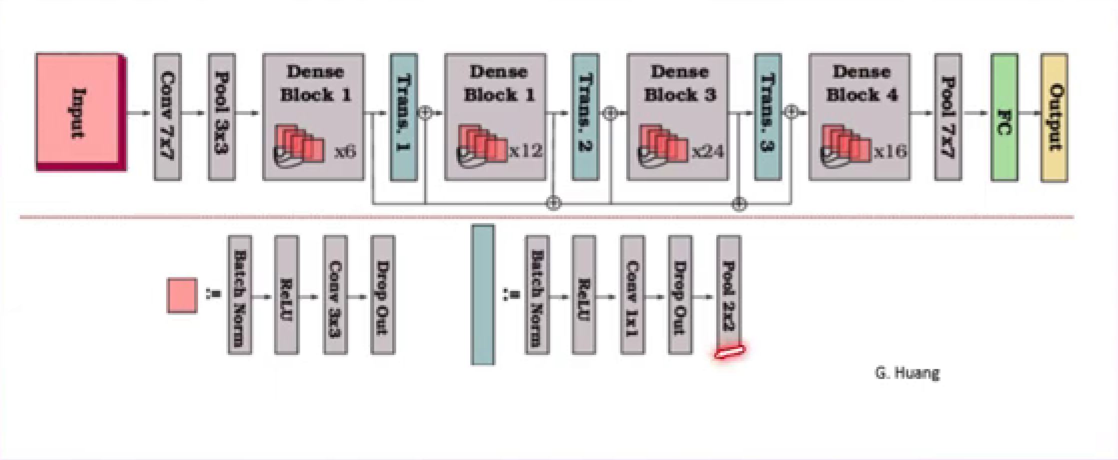
DenseNet


Input
Conv 7x7
Pool 3x3
Dense Block 1
x6
Trans. 1
Dense Block 1
x12
Trans. 2
Dense Block 3
x24
Trans. 3
Dense Block 4
x16
Pool 7x7
FC
Output
:=
Batch Norm
ReLU
Conv 3x3
Drop Out
:=
Batch Norm
ReLU
Conv 1x1
Drop Out
Pool 2x2
G. Huang

DenseNet

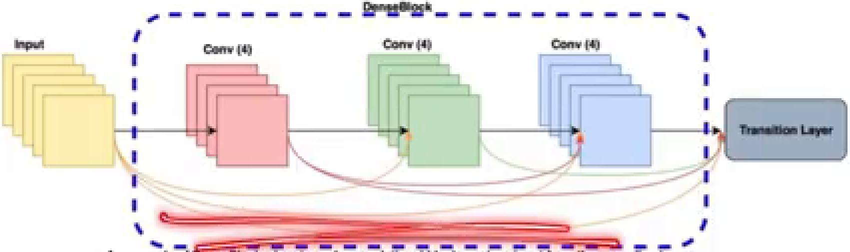

Input
Conv 7x7
Pool 3x3
Dense Block 1
x6
Trans. 1
Dense Block 1
x12
Trans. 2
Dense Block 3
x24
Trans. 3
Dense Block 4
x16
Pool 7x7
FC
Output
:=
Batch Norm
ReLU
Conv 3x3
Drop Out
:=
Batch Norm
ReLU
Conv 1x1
Drop Out
Pool 2x2
G. Huang

DenseNet



Input
Conv 7x7
Pool 3x3
Dense Block 1
x6
Trans. 1
Dense Block 1
x12
Trans. 2
Dense Block 3
x24
Trans. 3
Dense Block 4
x16
Pool 7x7
FC
Output
:=
Batch Norm
ReLU
Conv 3x3
Drop Out
:=
Batch Norm
ReLU
Conv 1x1
Drop Out
Pool 2x2
G. Huang

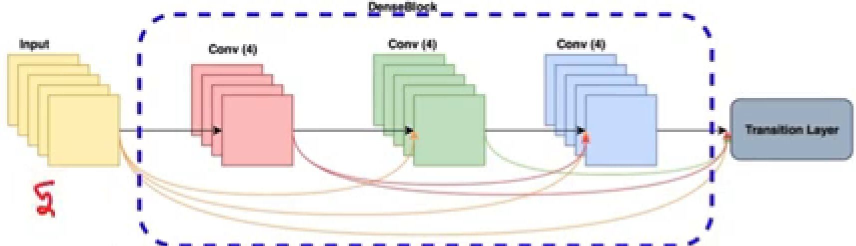
Dense Block



An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$
\begin{aligned}
x_1 &= w_1 * x_0 \\
x_2 &= w_2 * [x_0, x_1] \\
&\vdots \\
x_k &= w_k * [x_0, x_1, \ldots, x_{k-1}]
\end{aligned}
$$

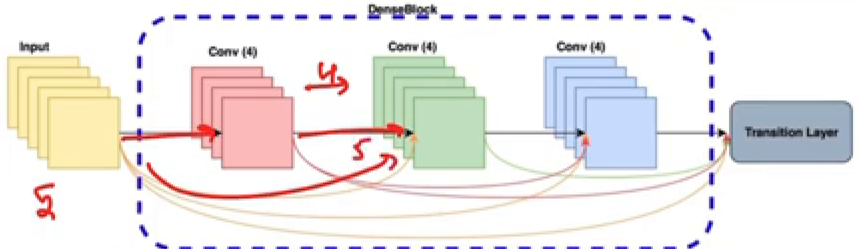
Dense Block



An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

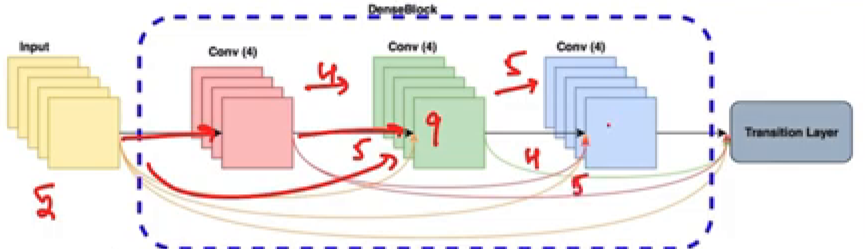
Dense Block



An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

Dense Block

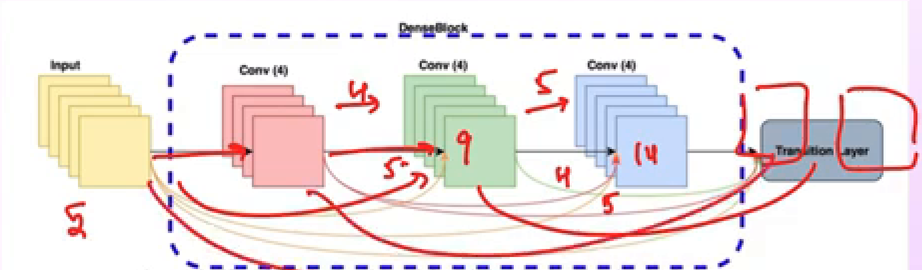


An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

9

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

Dense Block

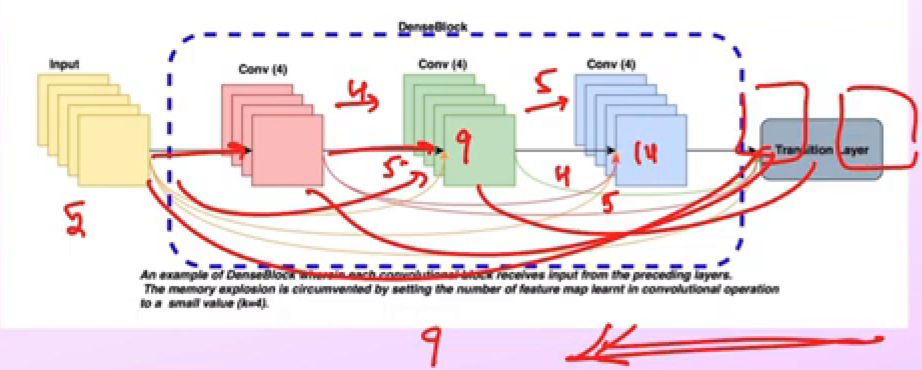


An example of DenseBlock wherein each convolutional block receives input from the preceding layers.
The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

9

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

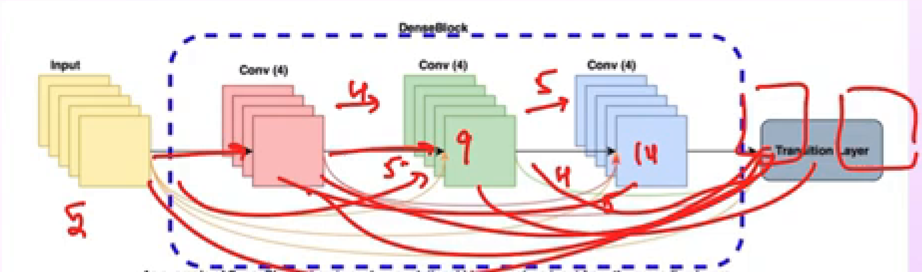
Dense Block



An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

Dense Block

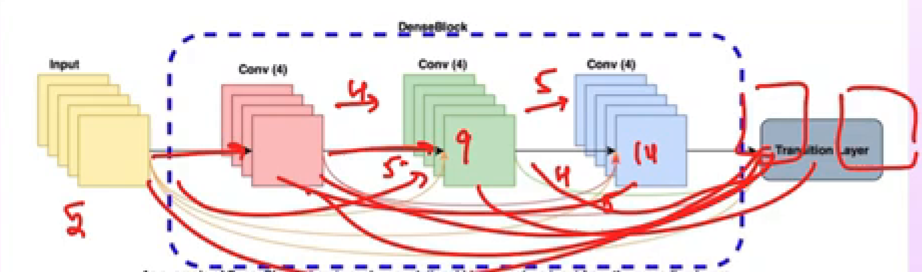


An example of DenseBlock wherein each convolutional block receives input from the preceding layers. The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

9

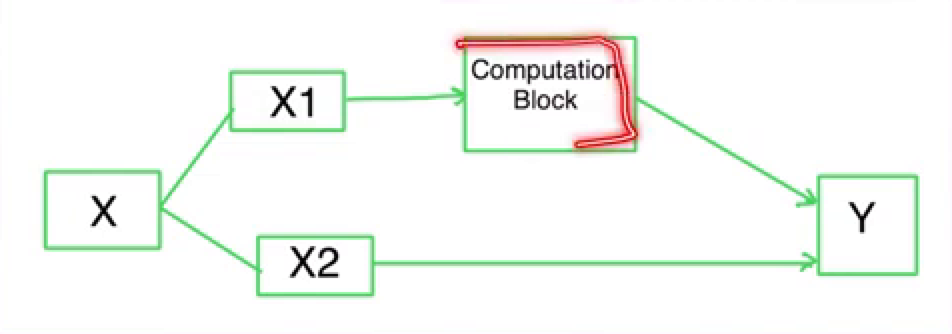
Dense Block



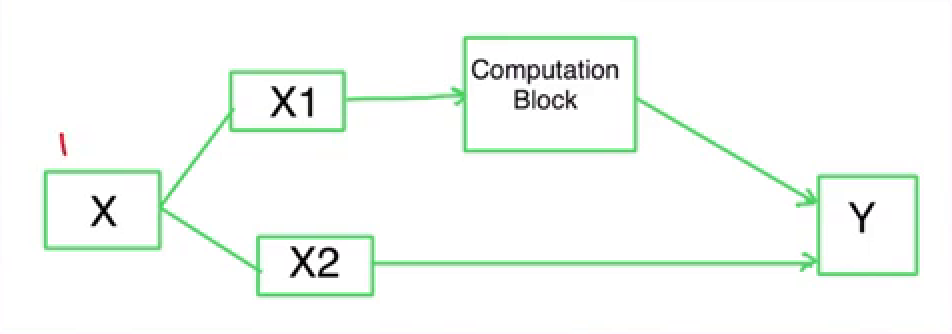
An example of DenseBlock wherein each convolutional block receives input from the preceding layers.
The memory explosion is circumvented by setting the number of feature map learnt in convolutional operation to a small value (k=4).

$$x_1 = w_1 * x_0$$
$$x_2 = w_2 * [x_0, x_1]$$
$$\vdots$$
$$x_k = w_k * [x_0, x_1, \ldots, x_{k-1}]$$

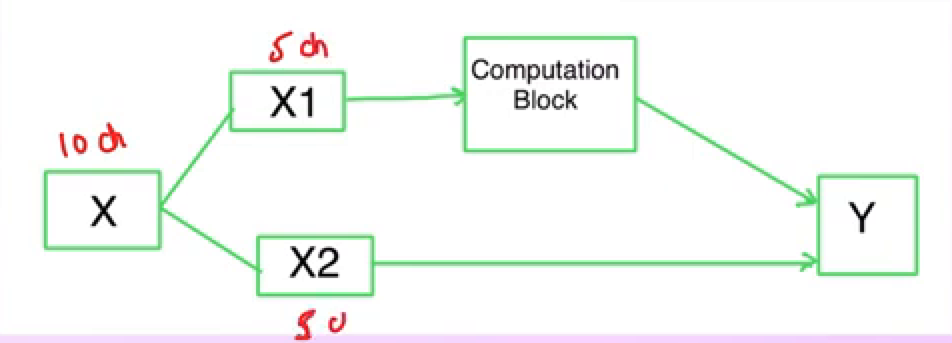
Cross Stage Partial Network (CSPNet)



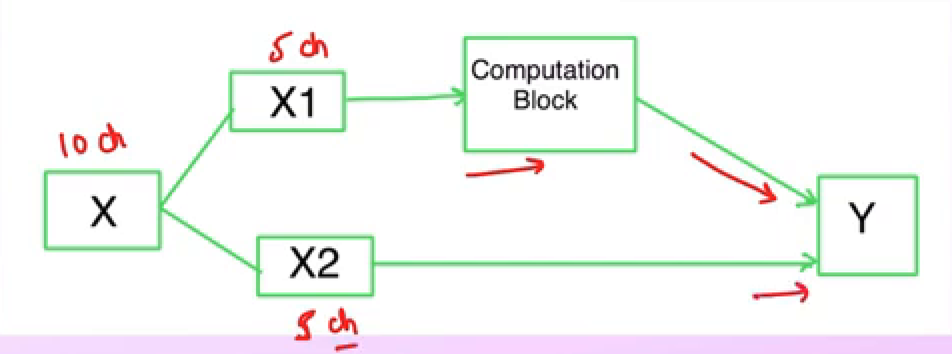
Cross Stage Partial Network (CSPNet)



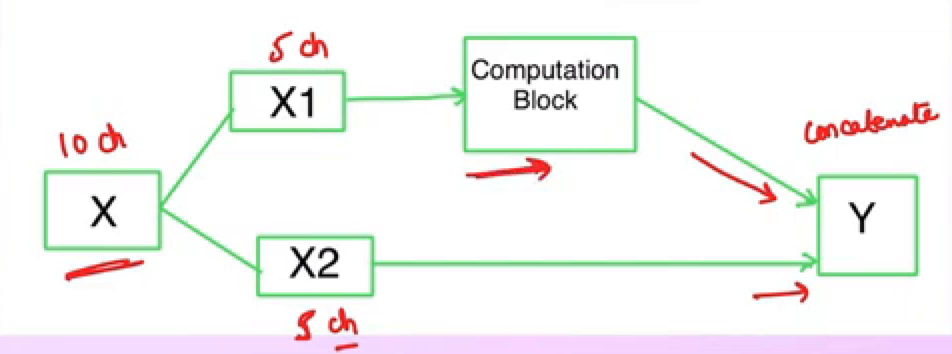
Cross Stage Partial Network (CSPNet)



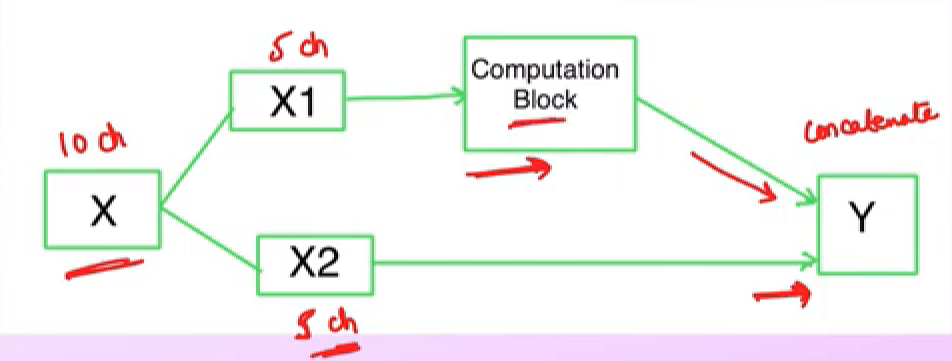
Cross Stage Partial Network (CSPNet)



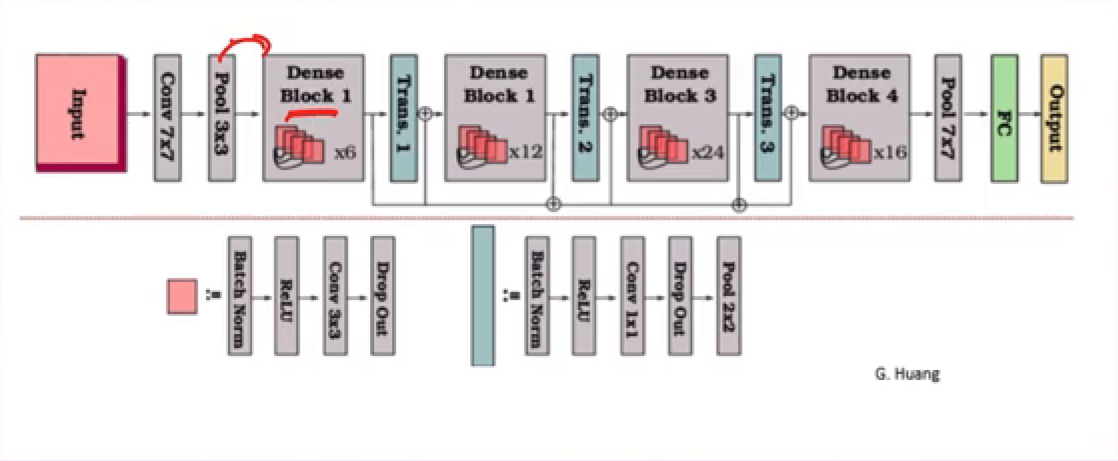
Cross Stage Partial Network (CSPNet)



Cross Stage Partial Network (CSPNet)

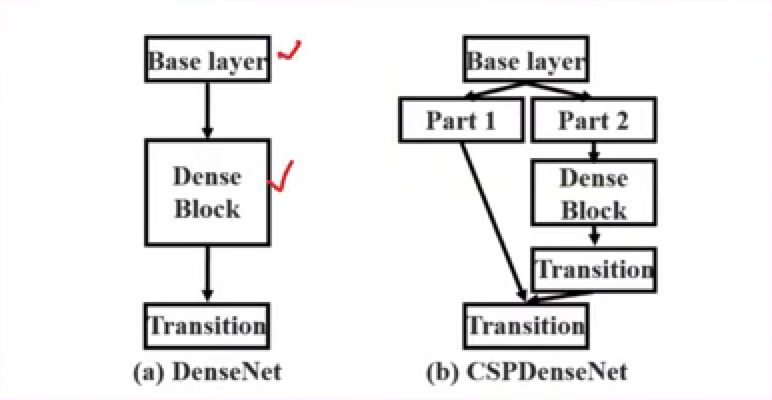


DenseNet

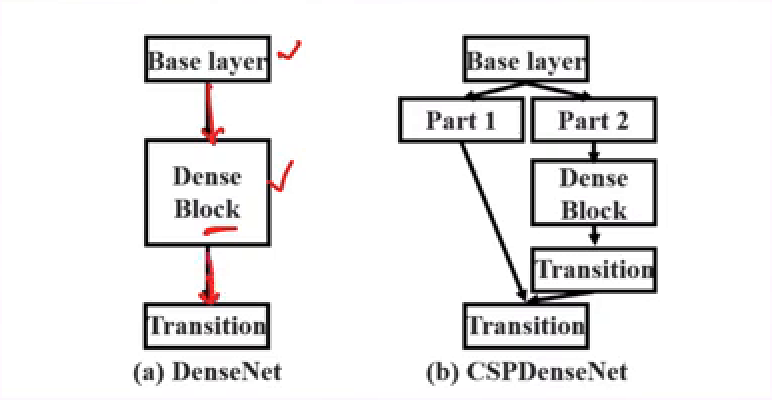


Input
Conv 7x7
Pool 3x3
Dense Block 1
x6
Trans. 1
Dense Block 1
x12
Trans. 2
Dense Block 3
x24
Trans. 3
Dense Block 4
x16
Pool 7x7
FC
Output
:=
Batch Norm
ReLU
Conv 3x3
Drop Out
:=
Batch Norm
ReLU
Conv 1x1
Drop Out
Pool 2x2
G. Huang

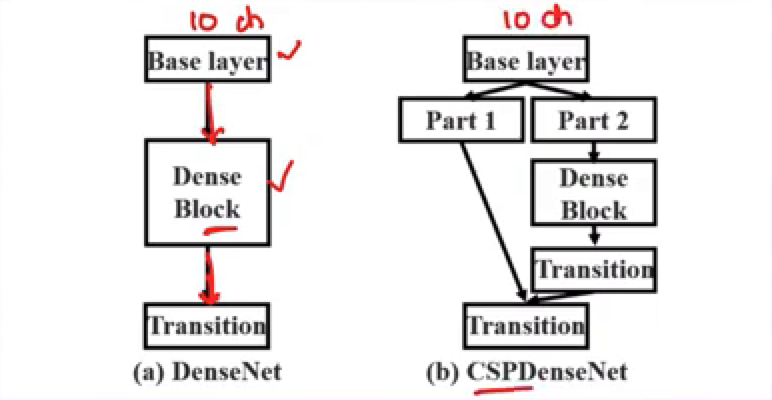
Cross Stage Partial Network (CSPNet)



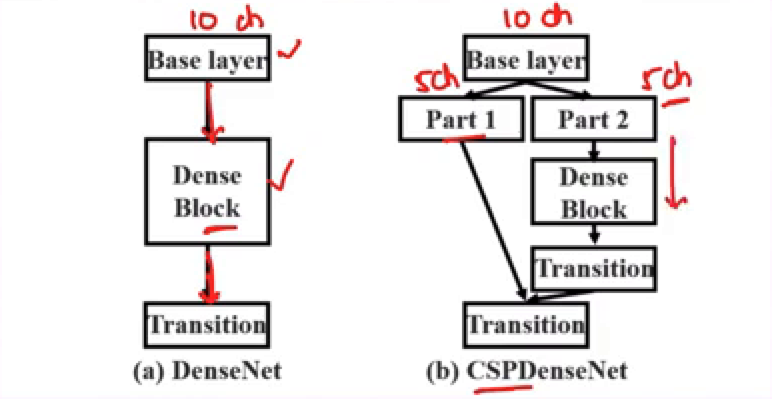
Cross Stage Partial Network (CSPNet)



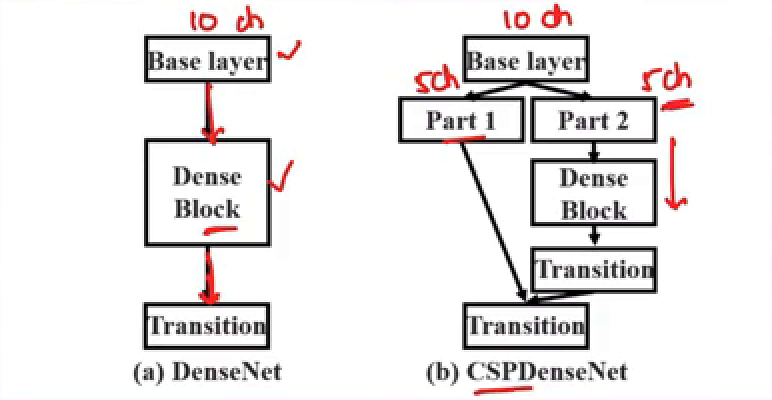
Cross Stage Partial Network (CSPNet)



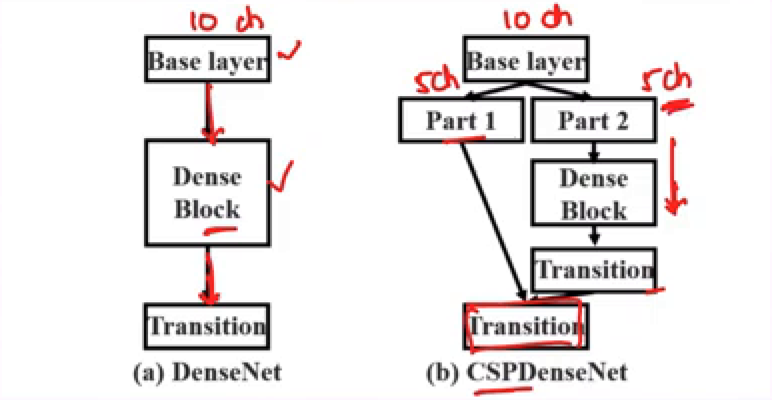
Cross Stage Partial Network (CSPNet)



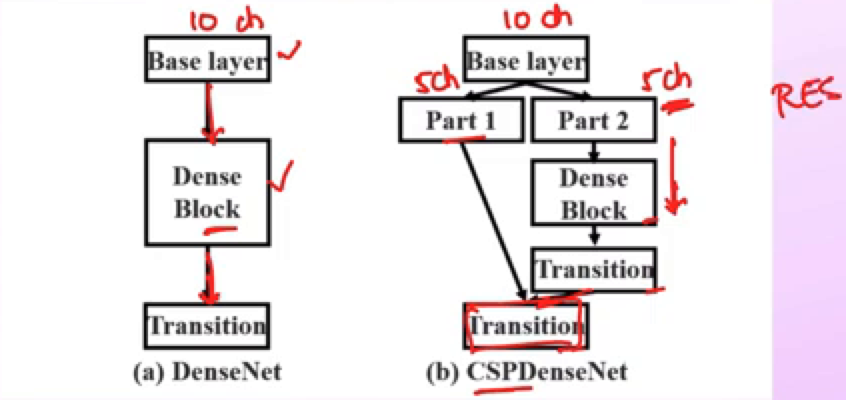
Cross Stage Partial Network (CSPNet)



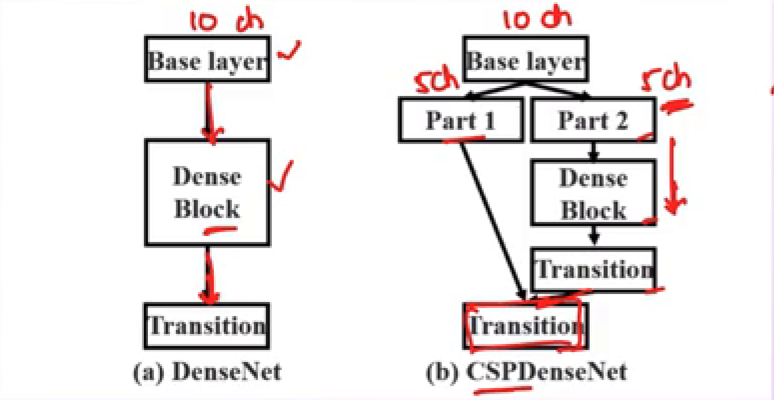
Cross Stage Partial Network (CSPNet)



Cross Stage Partial Network (CSPNet)

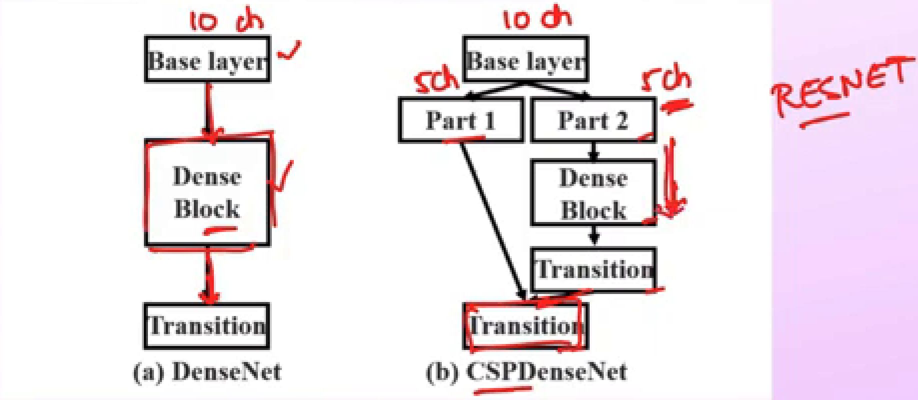


Cross Stage Partial Network (CSPNet)

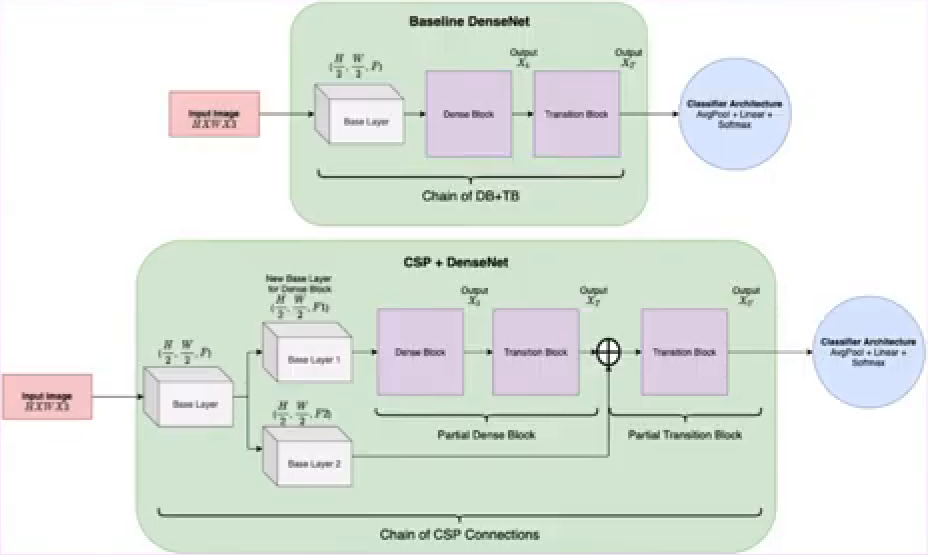


RESNET

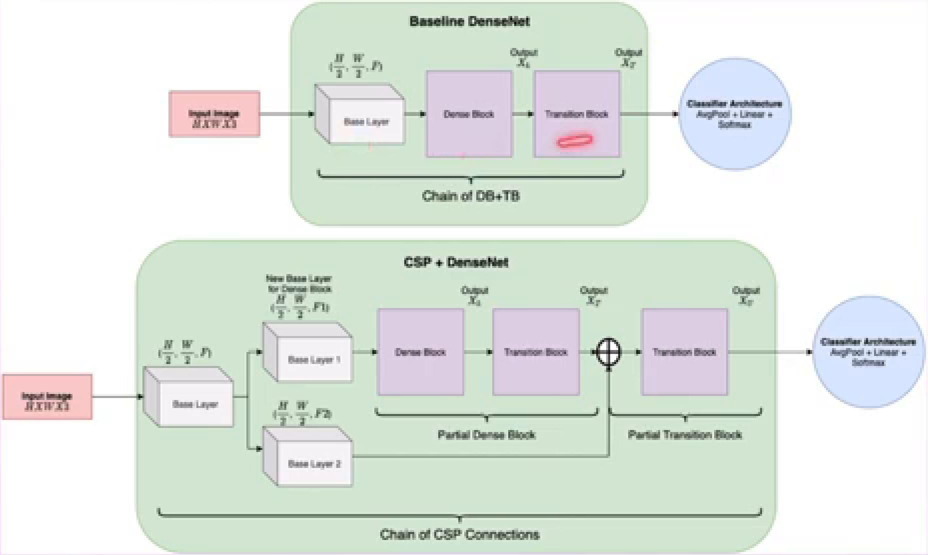
Cross Stage Partial Network (CSPNet)



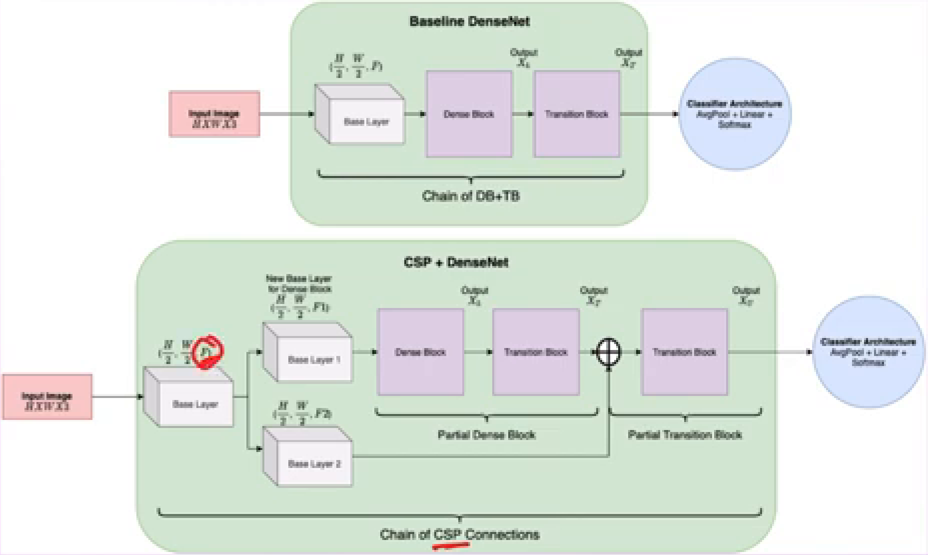
Cross Stage Partial Network (CSPNet)



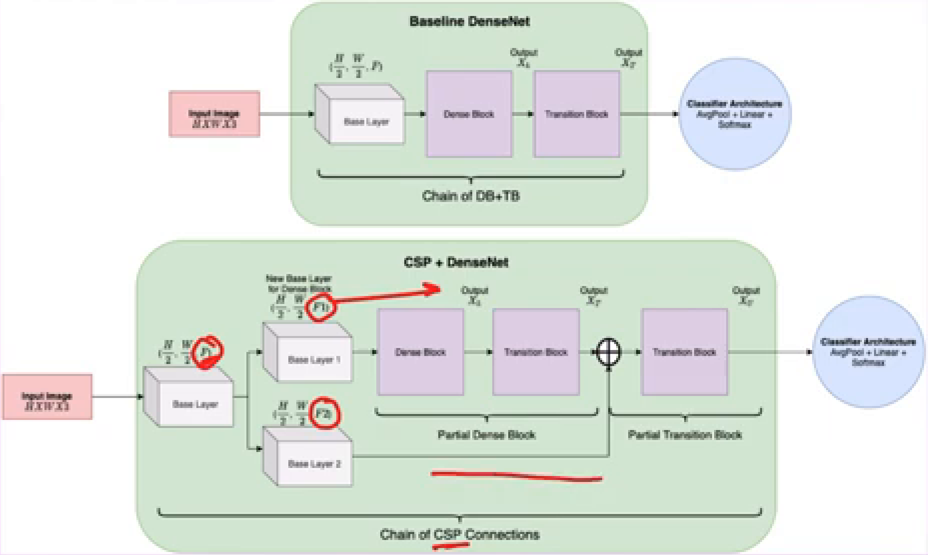
Cross Stage Partial Network (CSPNet)



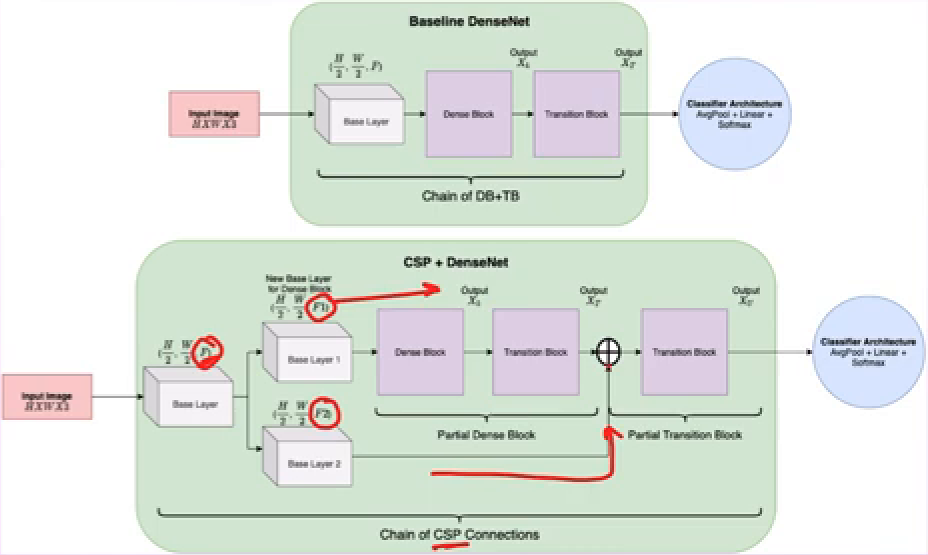
Cross Stage Partial Network (CSPNet)



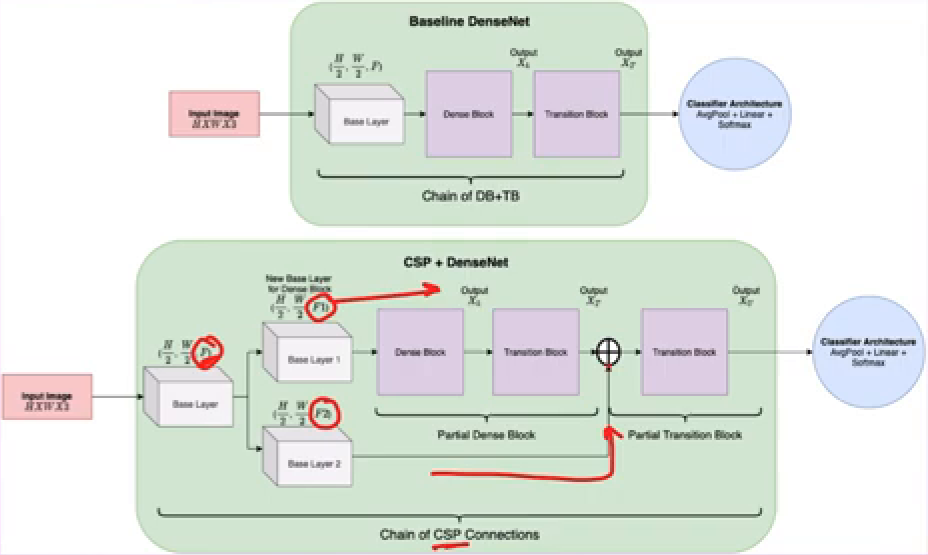
Cross Stage Partial Network (CSPNet)



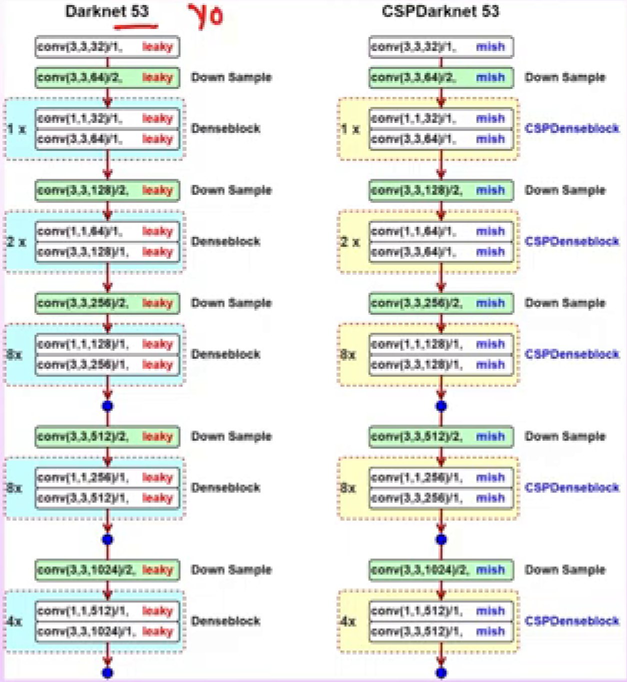
Cross Stage Partial Network (CSPNet)



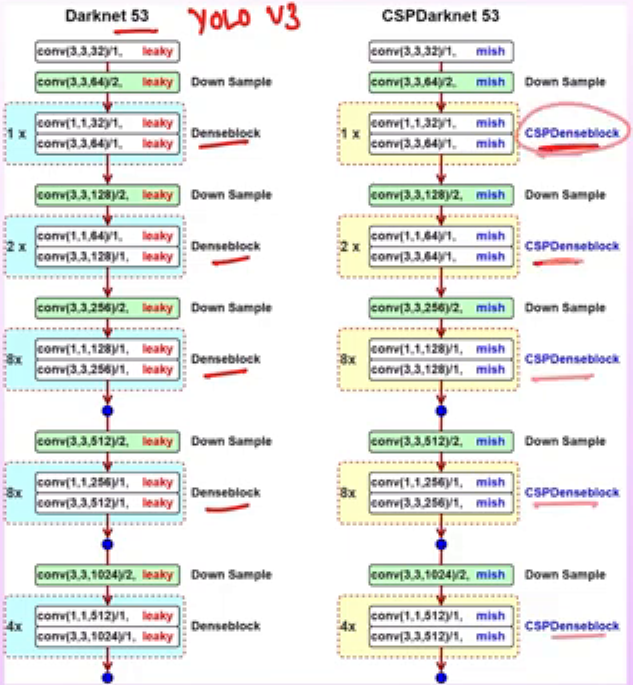
Cross Stage Partial Network (CSPNet)



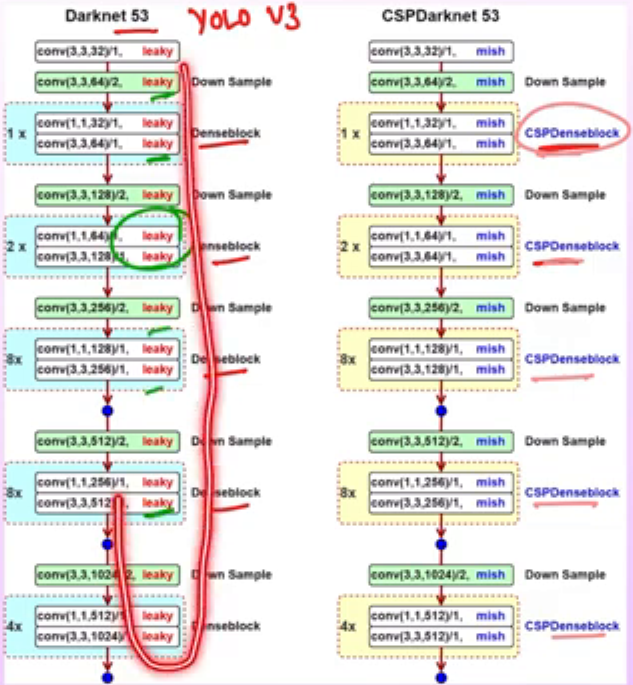
CSPDarkNet-53



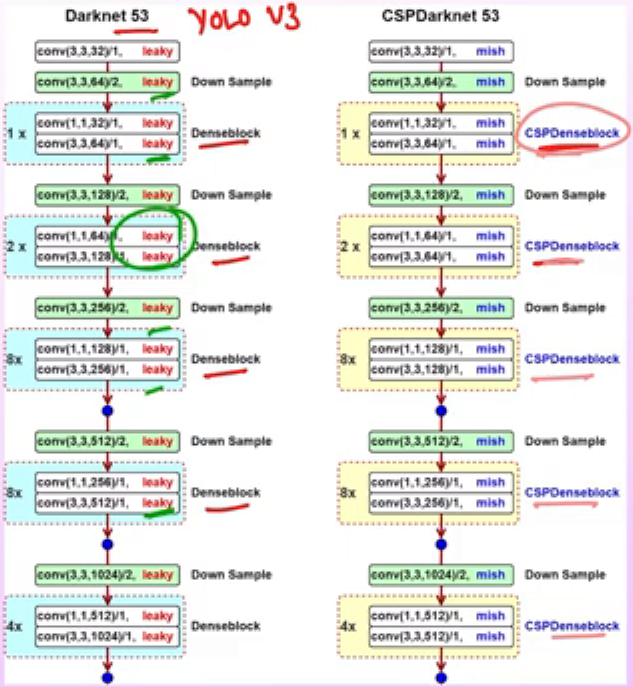
CSPDarkNet-53



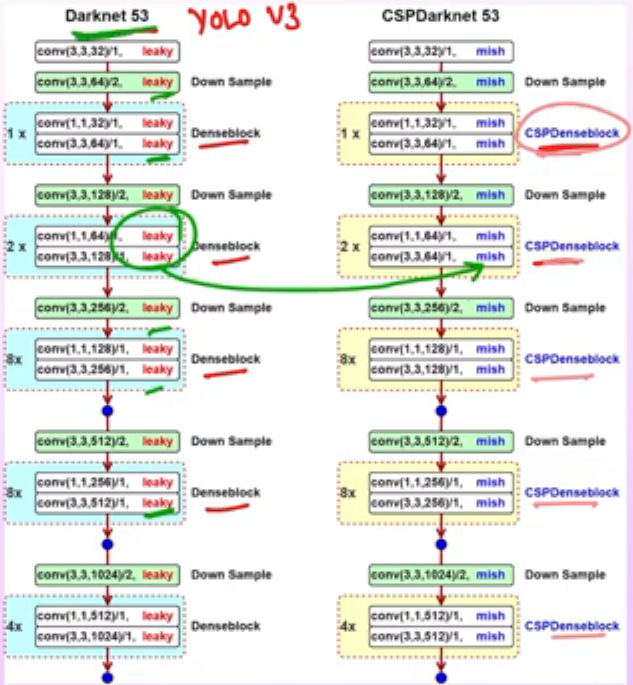
CSPDarkNet-53



CSPDarkNet-53



CSPDarkNet-53



In this video..

- Backbone
  - DenseNet
  - CSPNet
  - CSPDarknet-53
- Neck
  - FPN
  - SPP
  - PAN
- Spatial Attention Module

In this video..

- Backbone
  - DenseNet
  - CSPNet
  - CSPDarknet-53
- Neck → feature Agg
  - FPN
  - SPP
  - PAN
- Spatial Attention Module

In this video..

- Backbone
  - DenseNet
  - CSPNet
  - CSPDarknet-53
- Neck → feature Agg
  - FPN
  - SPP
  - PAN
- Spatial Attention Module

In this video..

- Backbone
  - DenseNet
  - CSPNet
  - CSPDarknet-53
- Neck → feature Agg
  - FPN
  - SPP
  - PAN
- Spatial Attention Module

In this video..

- Backbone
  - DenseNet
  - CSPNet
  - CSPDarknet-53
- Neck → feature Agg
  - FPN
  - SPP
  - PAN
- Spatial Attention Module

In this video..

- Backbone
  - DenseNet
  - CSPNet
  - CSPDarknet-53
- Neck → feature Agg
  - FPN
  - SPP
  - PAN
- Spatial Attention Module

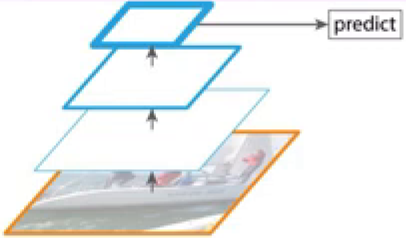
In this video..

- Backbone
  - DenseNet
  - CSPNet
  - CSPDarknet-53
- Neck → feature Agg
  - FPN
  - SPP
  - PAN
- Spatial Attention Module

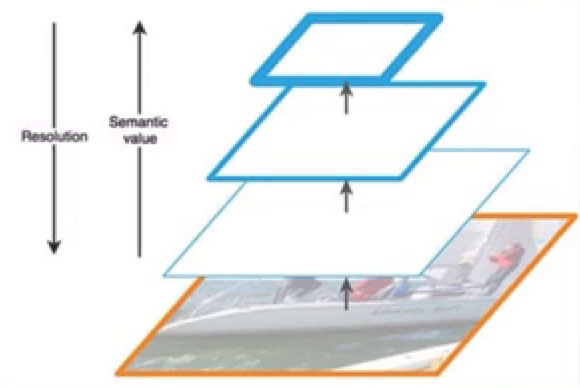
In this video..

- Backbone
    - DenseNet
    - CSPNet
    - CSPDarknet-53
- Neck → feature Agg
    - FPN
    - SPP
    - PAN
- Spatial Attention Module

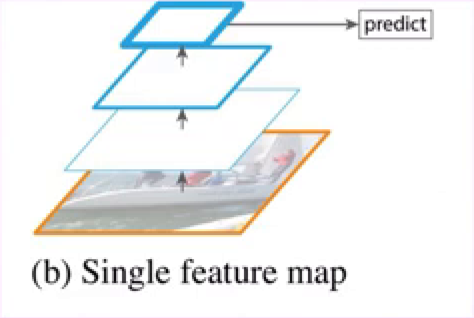
Small Objects Detection

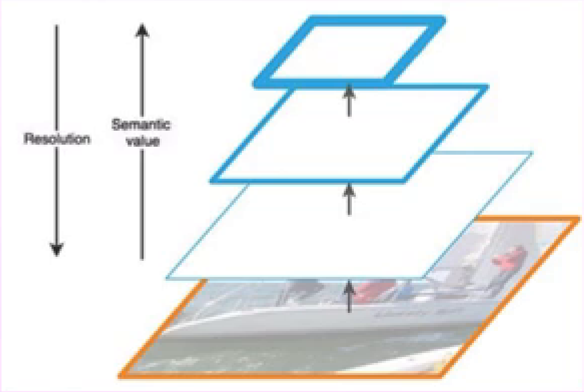


(b) Single feature map

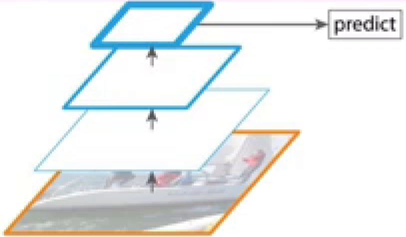


Small Objects Detection

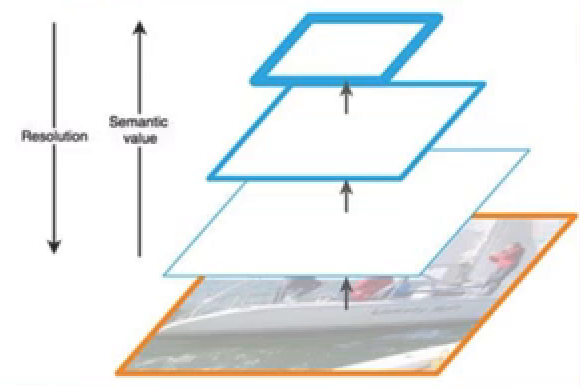




Small Objects Detection

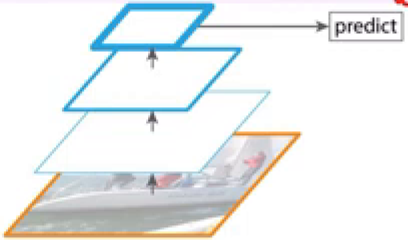


(b) Single feature map

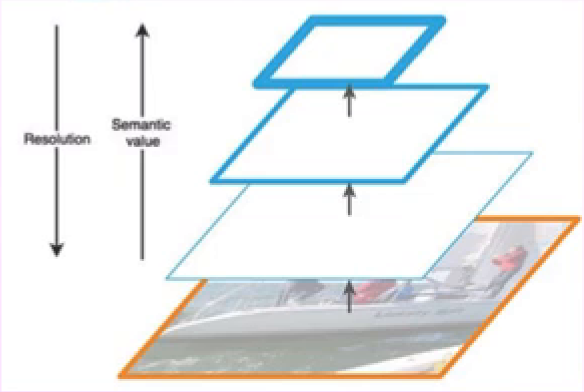


Small Objects Detection

416 Img 416 → 100 CNN → 7x7

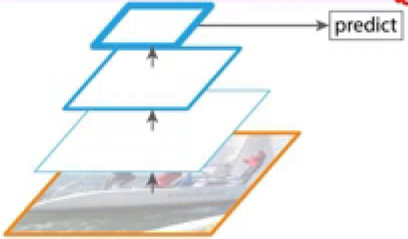


(b) Single feature map

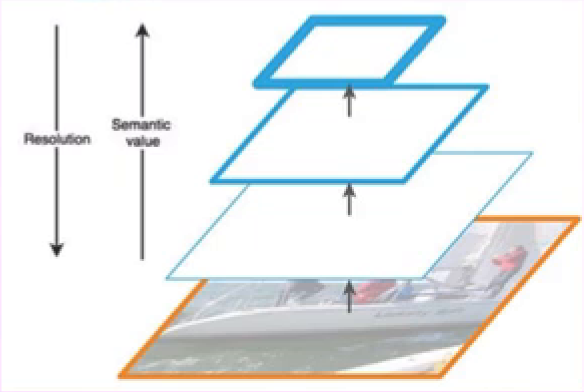


Small Objects Detection

416 Img 416 → 100 CNN → 7x7 →



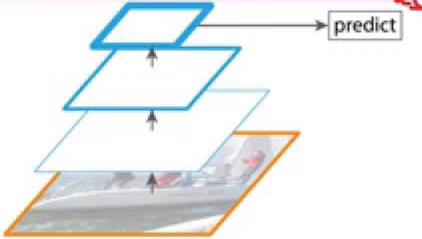
(b) Single feature map



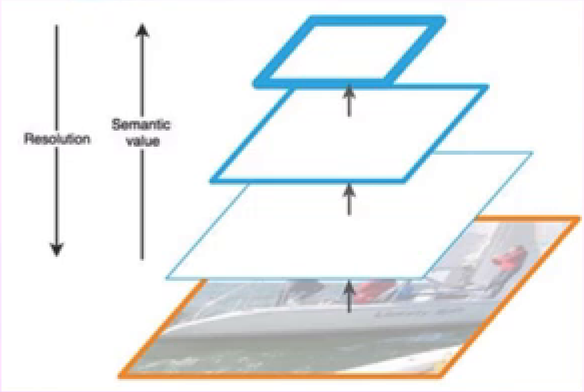
Small Objects Detection

416 Img 416 → CNN 100 → 7x7 → semantic

class object



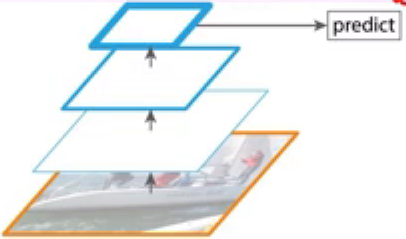
(b) Single feature map



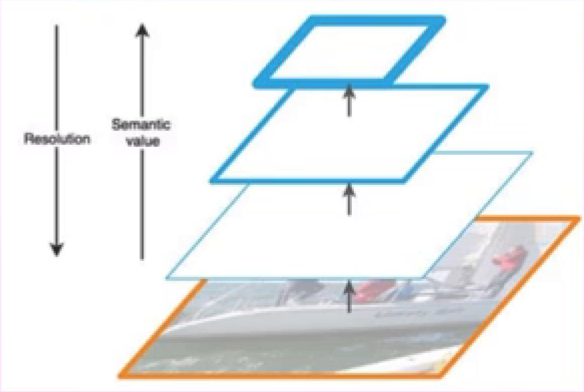
Small Objects Detection

416 Img 416 → CNN 100 → 7x7 → semantic

class object

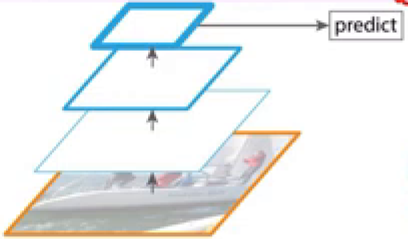


(b) Single feature map

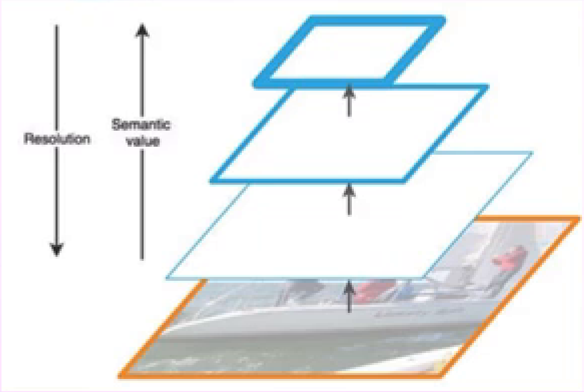


Small Objects Detection

416 Img 416 → 100 CNN → 7x7 → semantic class object

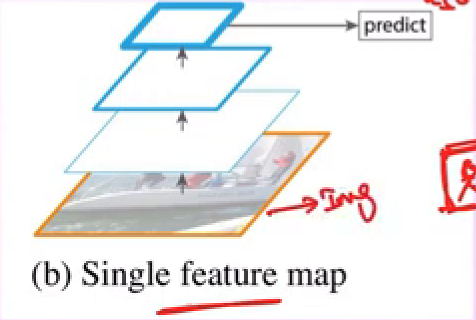


(b) Single feature map

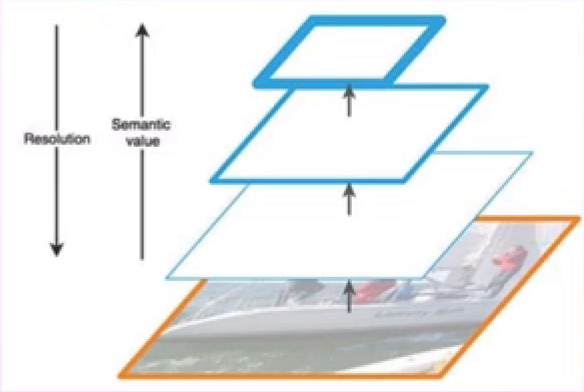


Small Objects Detection

416 Img 416 → CNN 100 → 7x7 → semantic
class object

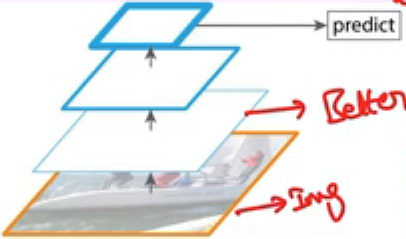


(b) Single feature map



Small Objects Detection

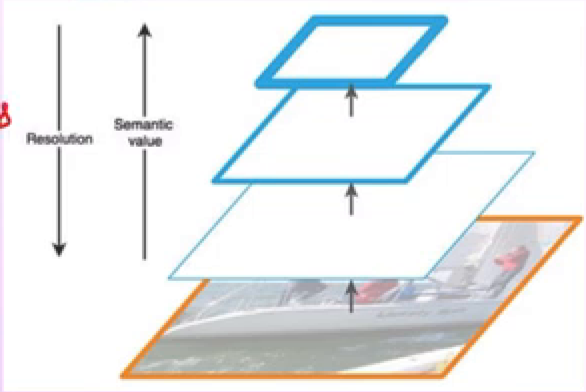
416 Img 416 → CNN (100) → 7x7 → semantic class object



Better spatial features

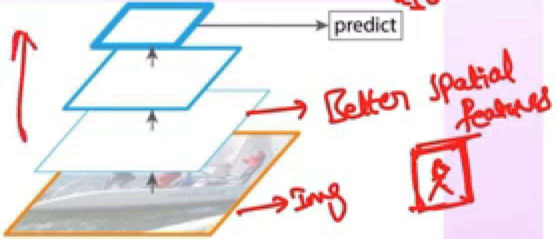
Img

(b) Single feature map

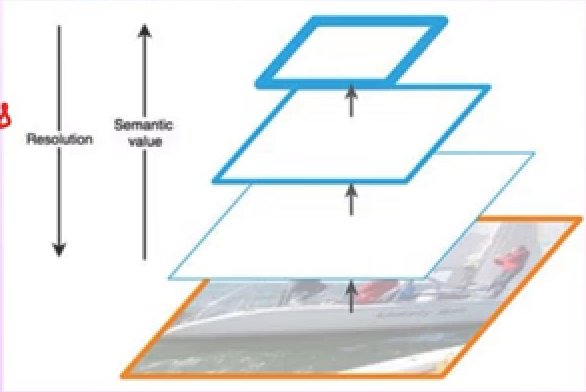


Small Objects Detection

416 Img 416 → 100 CNN → 7x7 → semantic
class object



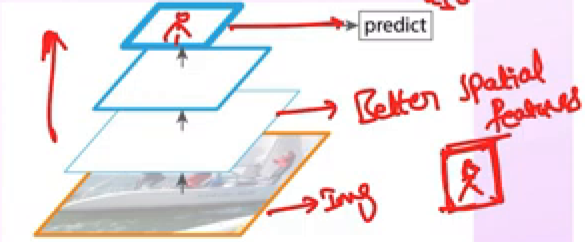
(b) Single feature map



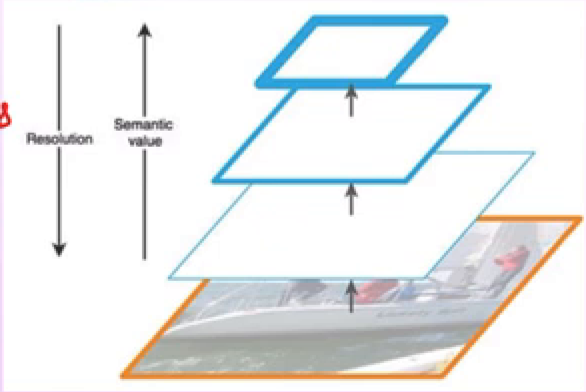
Small Objects Detection

416 Img 416 → CNN 100 → 7x7 → semantic

class object



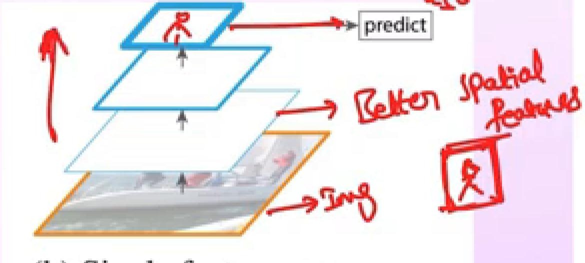
(b) Single feature map



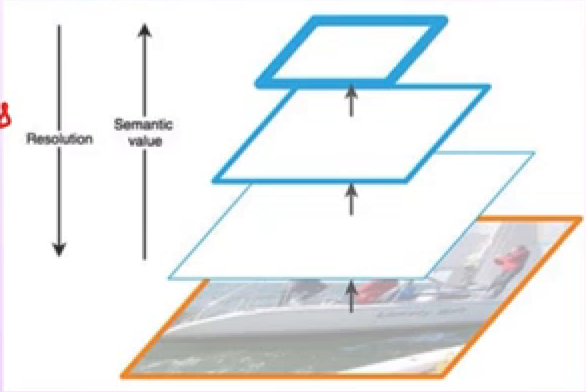
Small Objects Detection

416 Img → CNN (100) → 7x7 → semantic class object

416

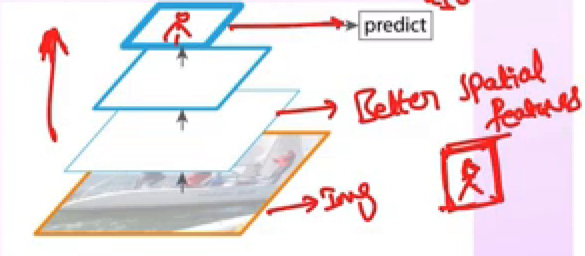


(b) Single feature map

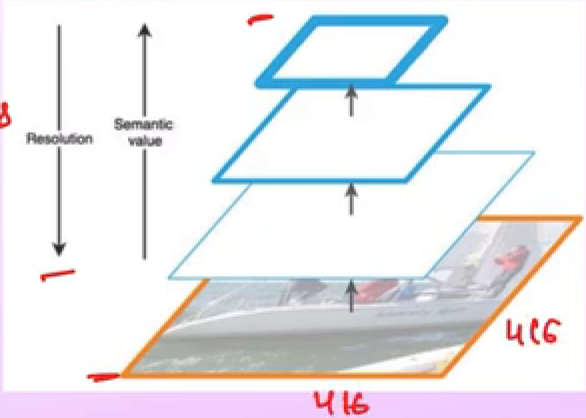


Small Objects Detection

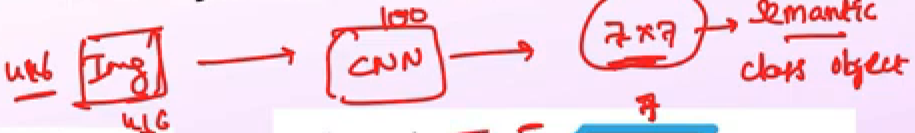
416 Img → 100 CNN → 7x7 → semantic
class object

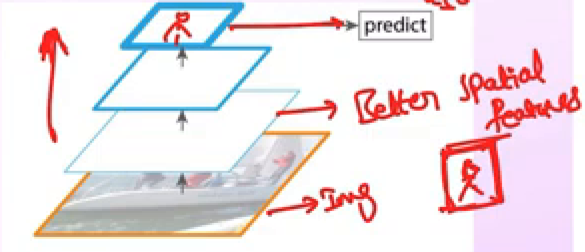


(b) Single feature map

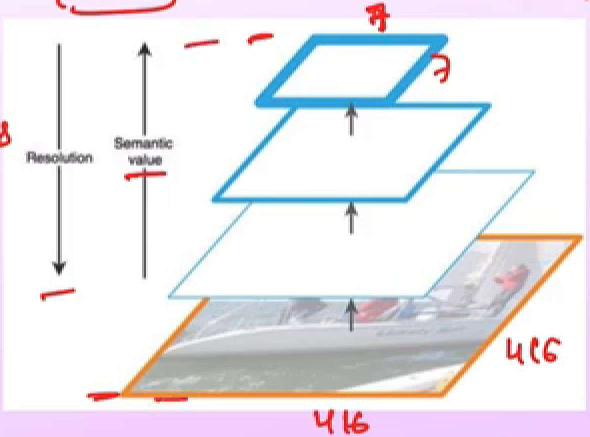


Small Objects Detection





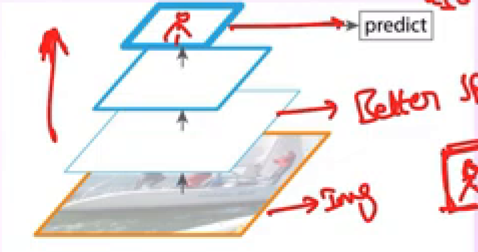
(b) Single feature map



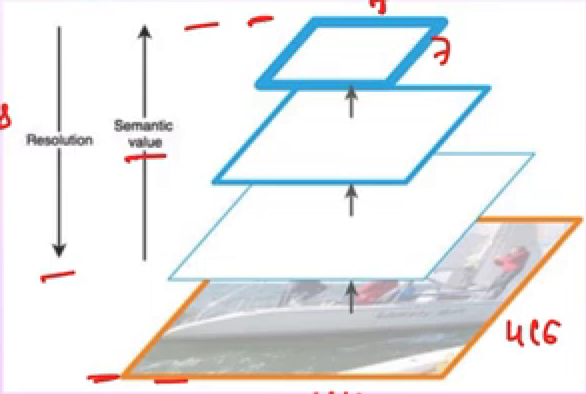
Small Objects Detection

416 Img → CNN → 7x7 → semantic class object

100



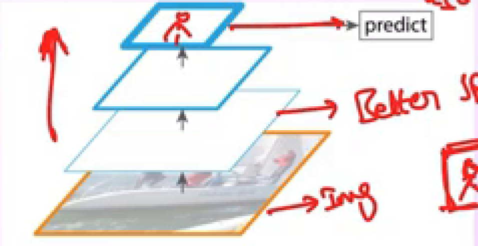
(b) Single feature map



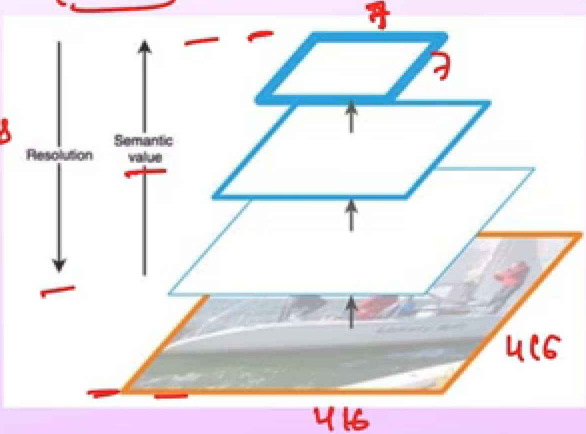
Small Objects Detection

416 Img → CNN (100) → 7x7 → semantic class object

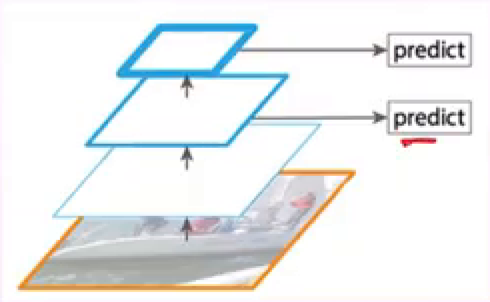
416



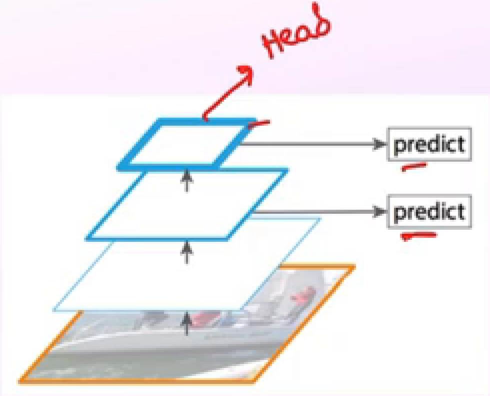
(b) Single feature map



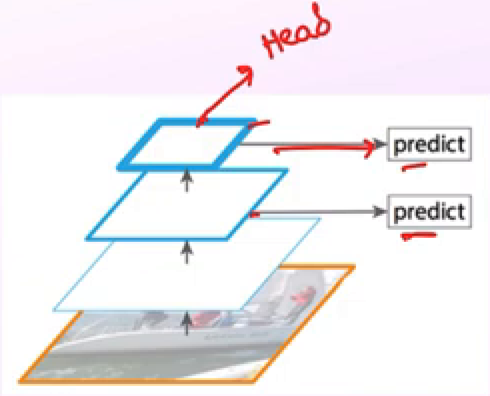
SSD



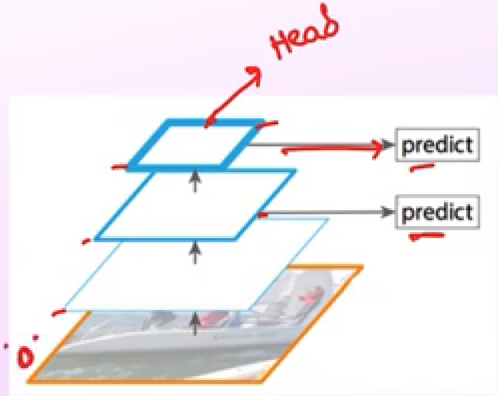
SSD



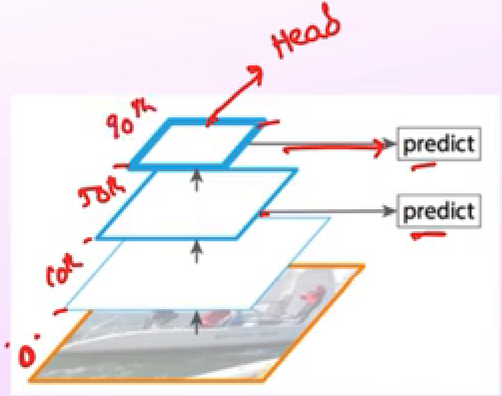
SSD



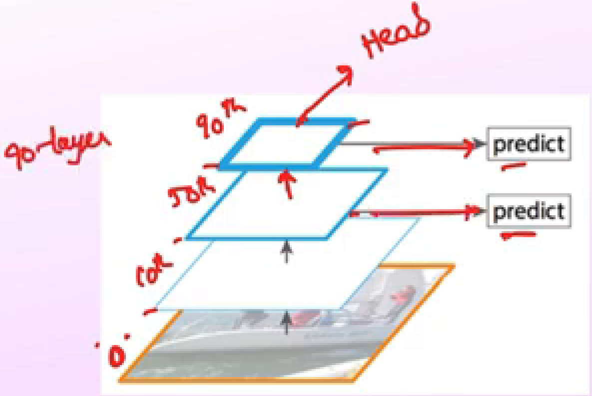
SSD



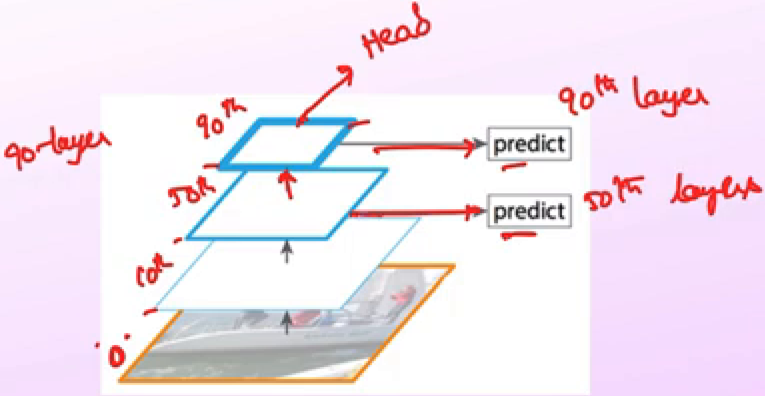
SSD



SSD



SSD

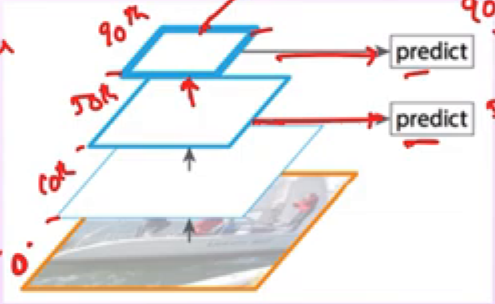


Head
90th layer
90-layer
90th
50th
50th layer
predict
predict
'0'

SSD

Head

90-layer



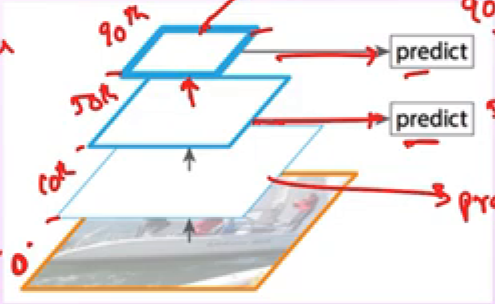
90th layer → semantic

50th layer → spati

SSD

Head

90-layer



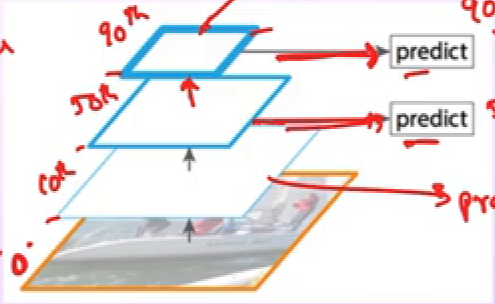
90th layer → semantic

50th layer → spatial

& sema

predict → spatial info

SSD



Head

90-layer

90th

50th

'0'

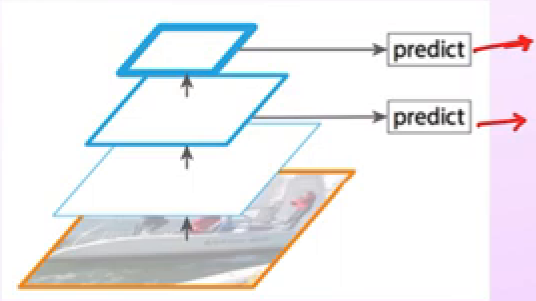
90th layer → semantic

50th layer → spatial & sema

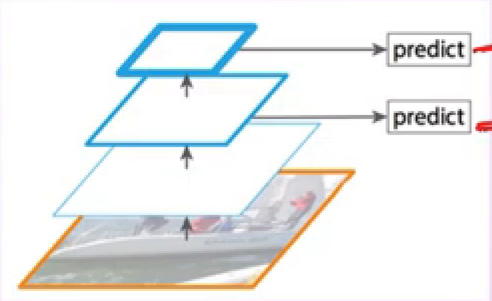
predict → spatial info

Bounding

SSD

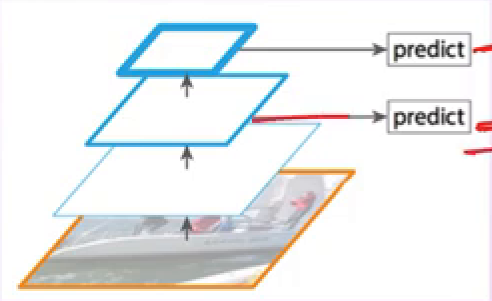


SSD



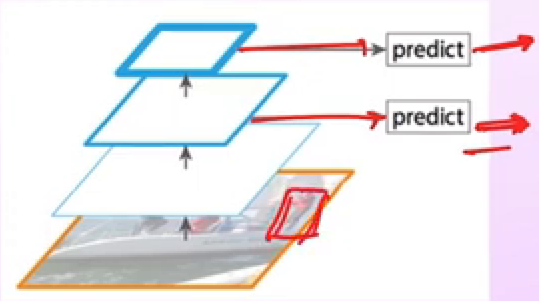
independent
P

SSD



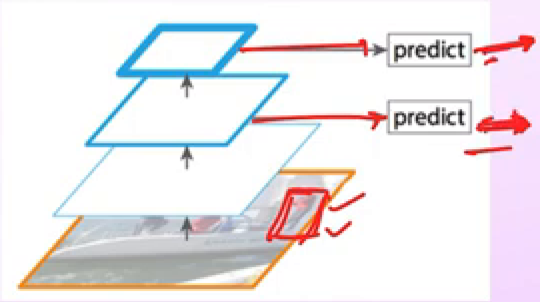
independent
predictions

SSD



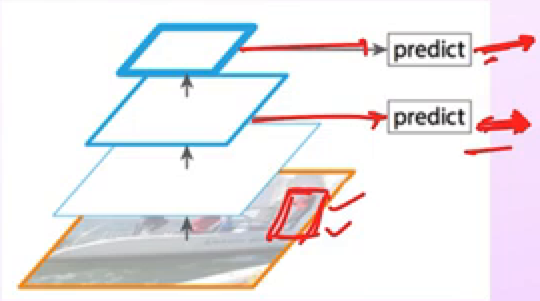
independent
predictions

SSD



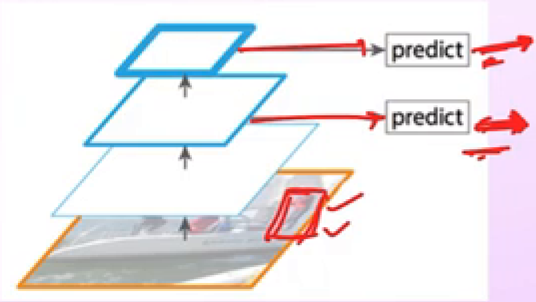
independent
predictions

SSD



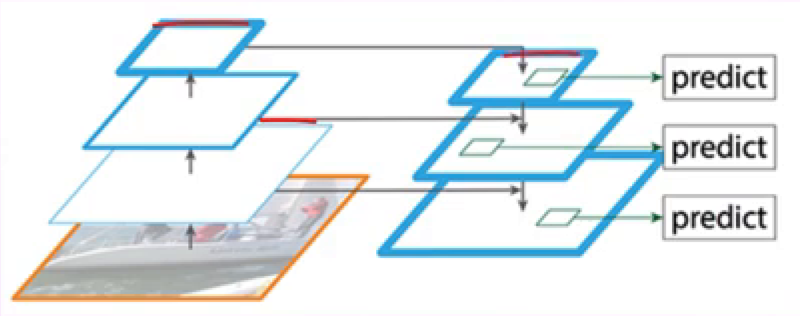
independent
predictions

SSD

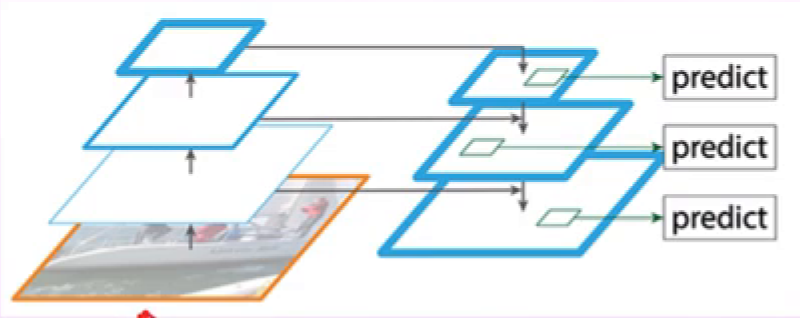


independent
predictions

Feature Pyramid Network (FPN)

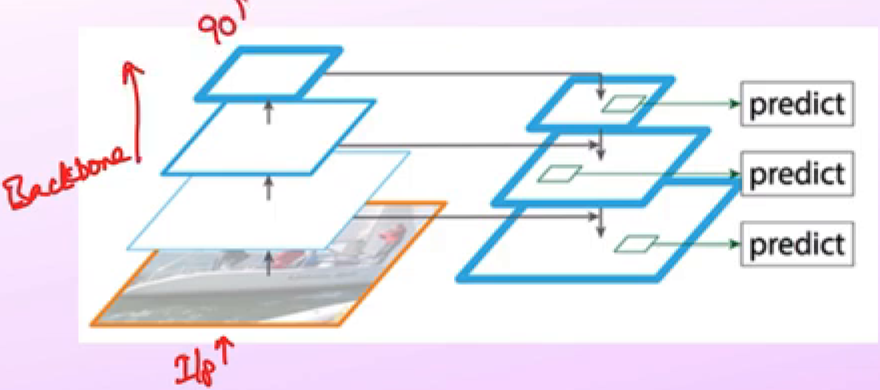


Feature Pyramid Network (FPN)

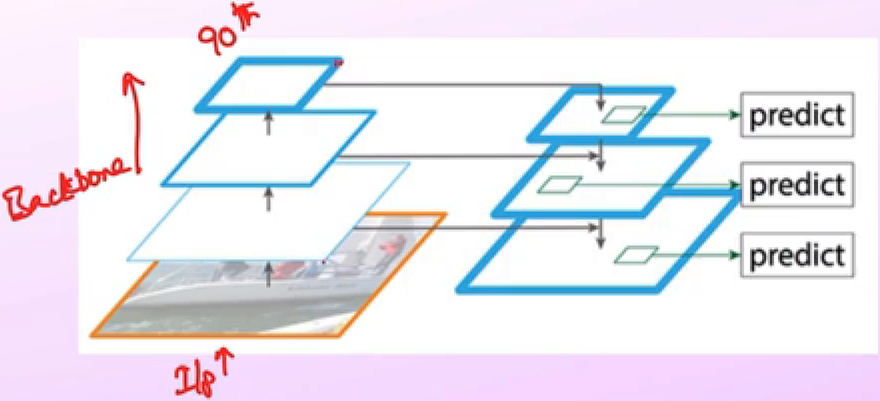


218

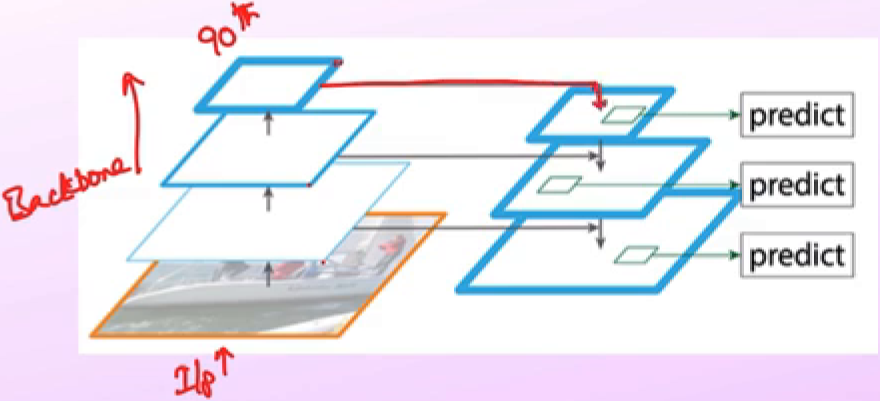
Feature Pyramid Network (FPN)



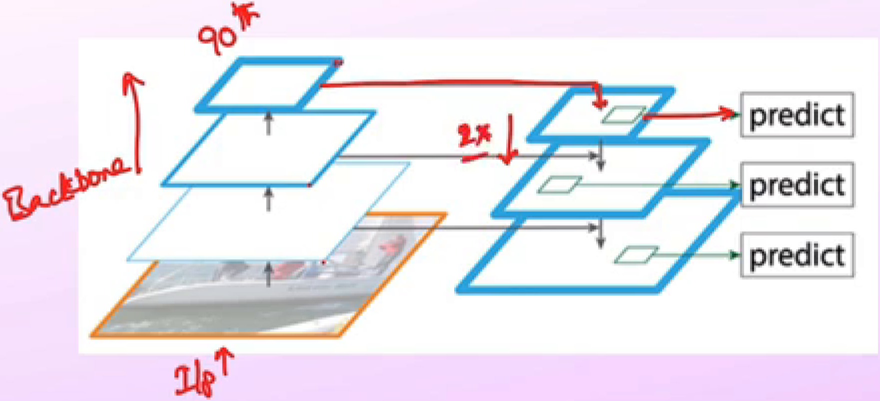
Feature Pyramid Network (FPN)



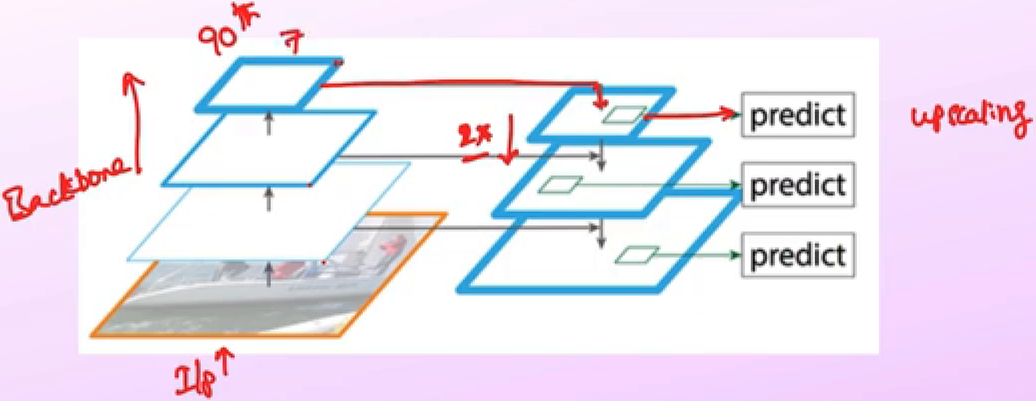
Feature Pyramid Network (FPN)



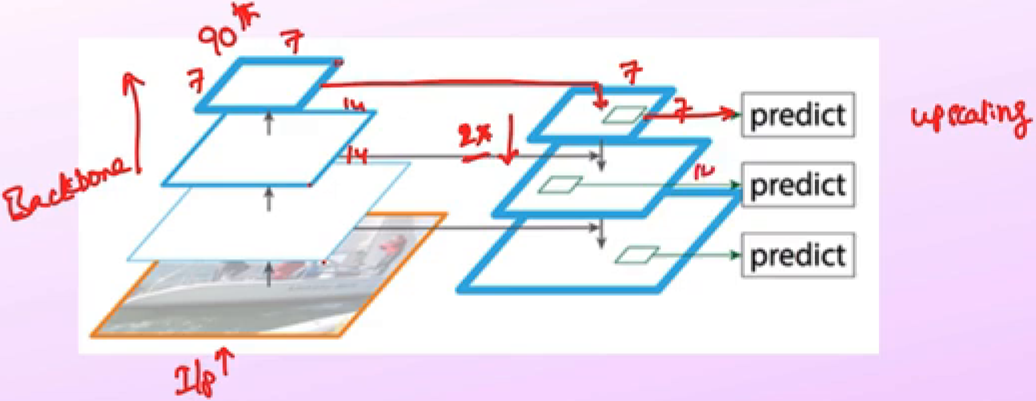
Feature Pyramid Network (FPN)



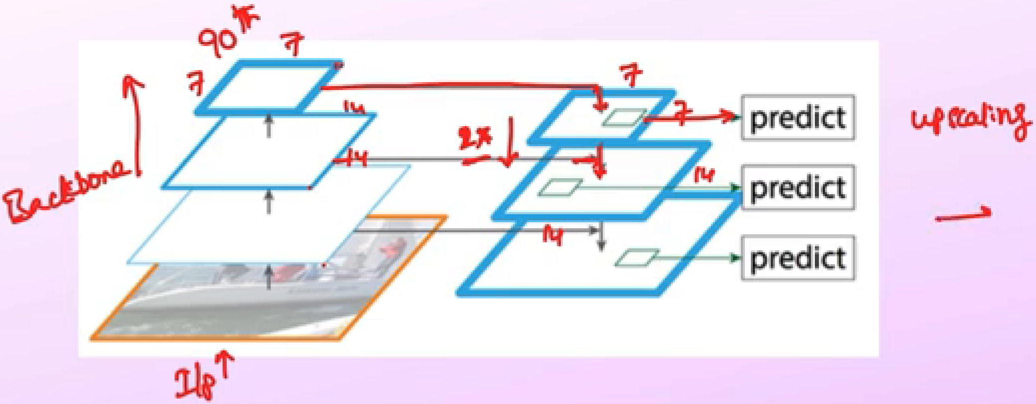
Feature Pyramid Network (FPN)



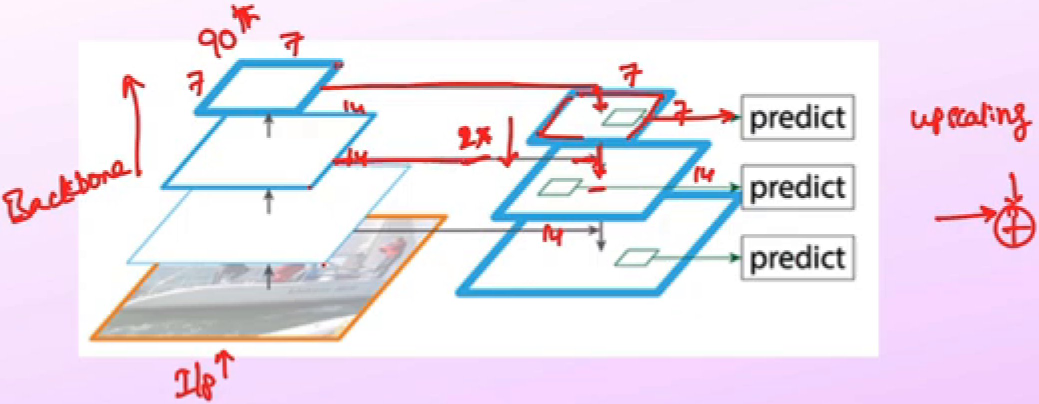
Feature Pyramid Network (FPN)



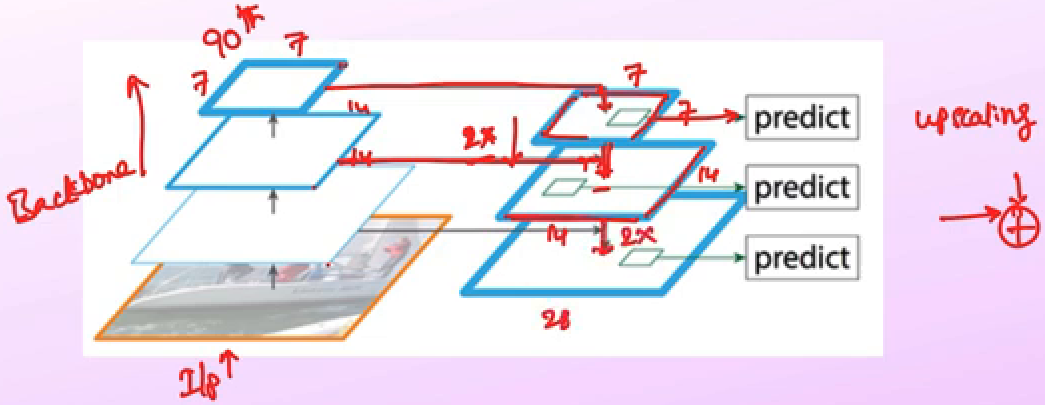
Feature Pyramid Network (FPN)



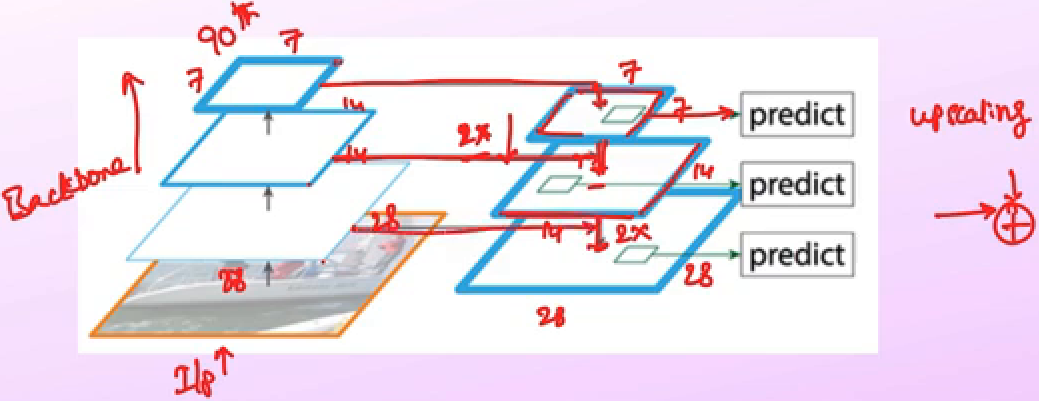
Feature Pyramid Network (FPN)



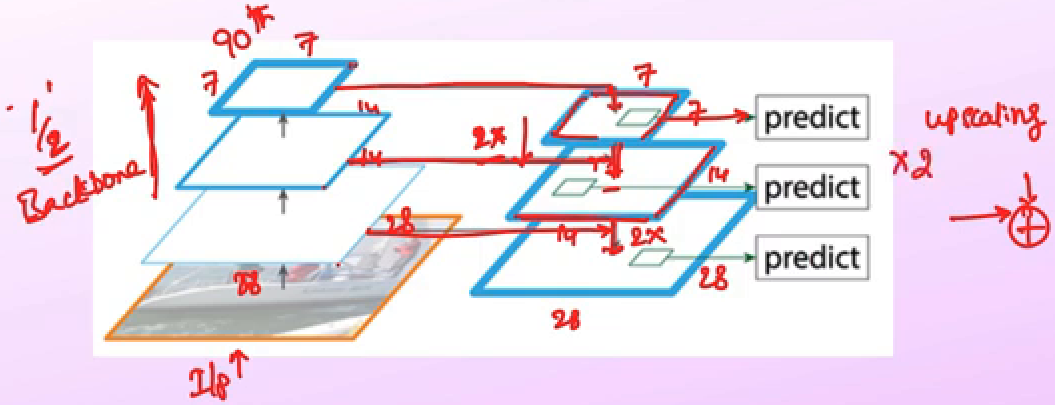
Feature Pyramid Network (FPN)



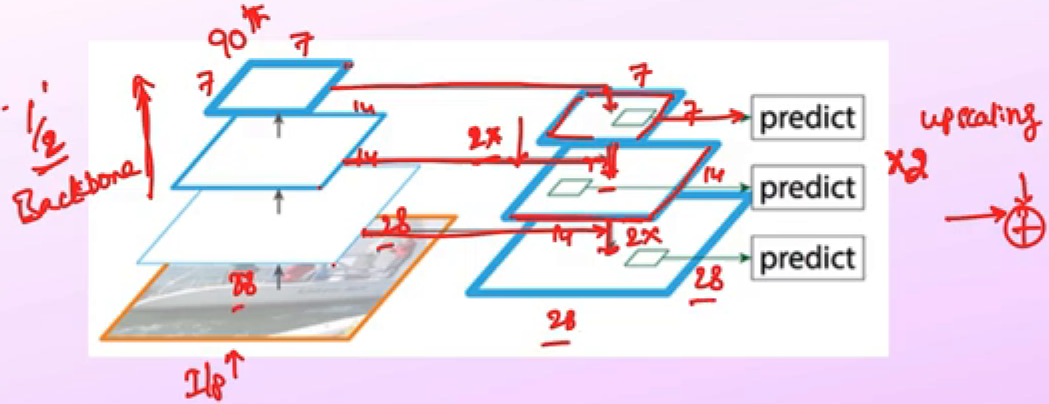
Feature Pyramid Network (FPN)



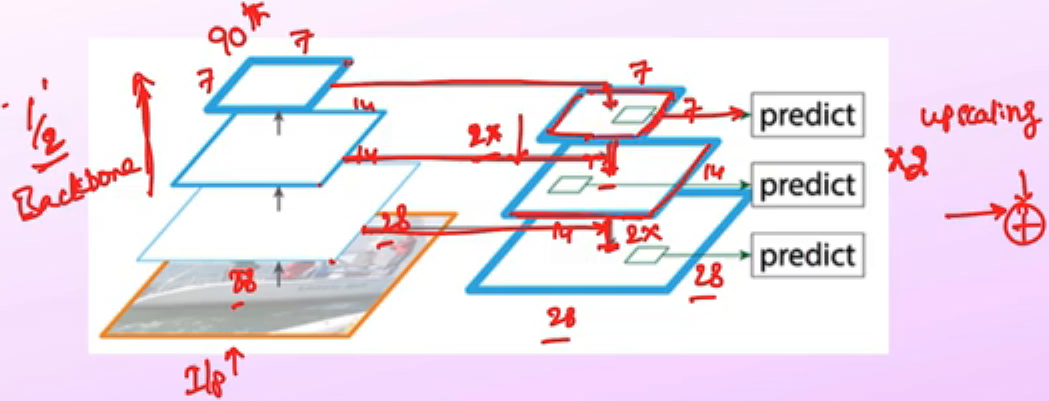
Feature Pyramid Network (FPN)



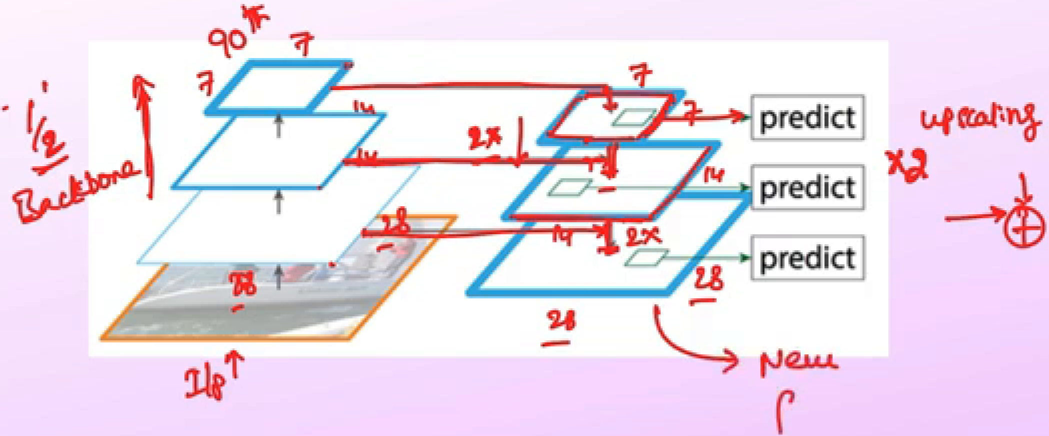
Feature Pyramid Network (FPN)



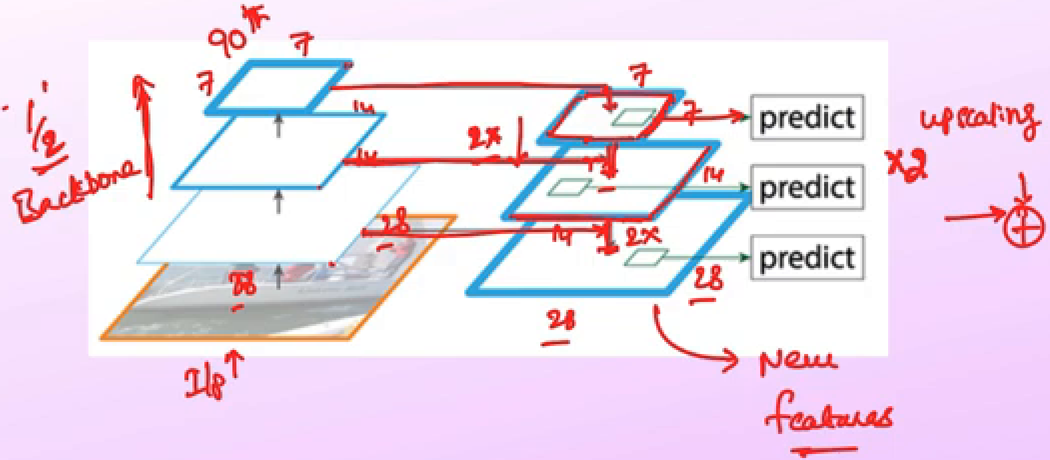
Feature Pyramid Network (FPN)



Feature Pyramid Network (FPN)



Feature Pyramid Network (FPN)



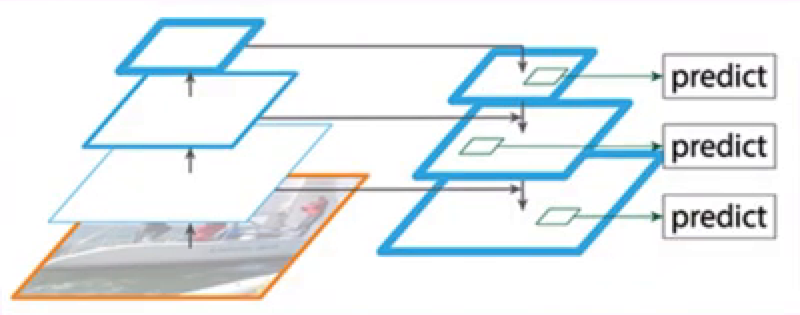
Feature Pyramid Network (FPN)

upscaling

x2

New features

Feature Pyramid Network (FPN)



This is not used in YoloV4

Path Aggregation Network

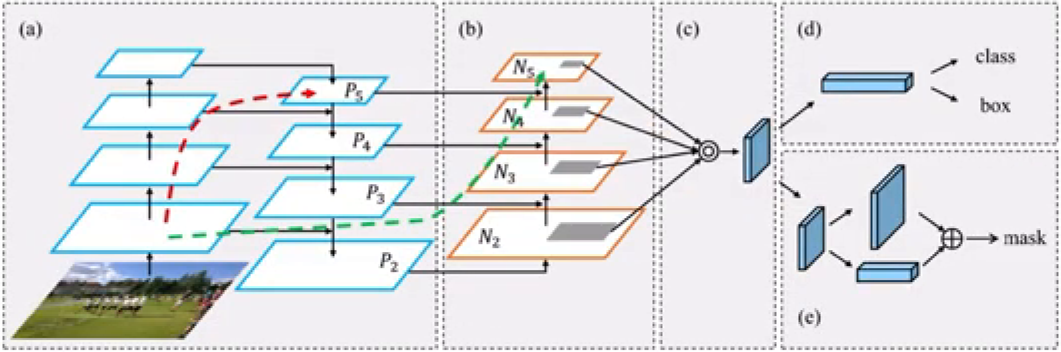


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path augmentation. (c) Adaptive feature pooling. (d) Box branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature maps in (a) and (b) for brevity.

Path Aggregation Network

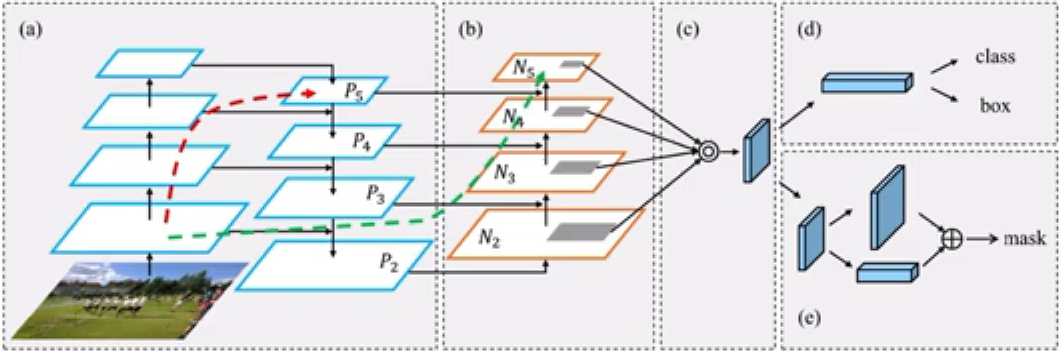


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path augmentation. (c) Adaptive feature pooling. (d) Box branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature maps in (a) and (b) for brevity.

Path Aggregation Network

FPN

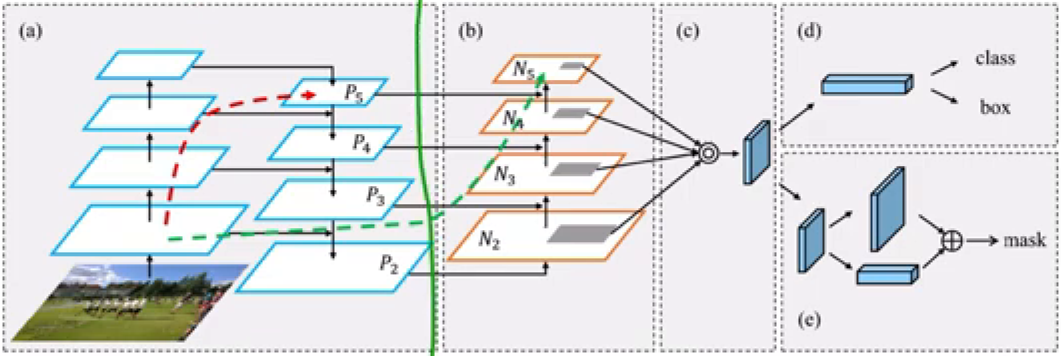


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path augmentation. (c) Adaptive feature pooling. (d) Box branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature maps in (a) and (b) for brevity.

Path Aggregation Network

FPN

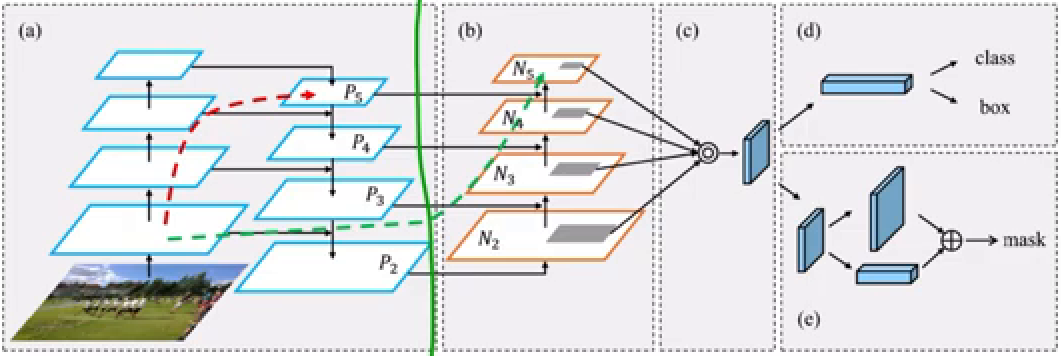


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path augmentation. (c) Adaptive feature pooling. (d) Box branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature maps in (a) and (b) for brevity.

Path Aggregation Network

FPN

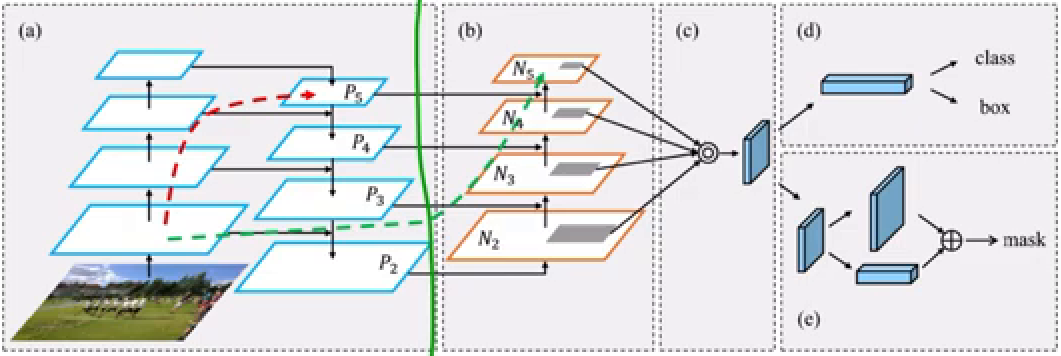


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path augmentation. (c) Adaptive feature pooling. (d) Box branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature maps in (a) and (b) for brevity.

Path Aggregation Network

FPN

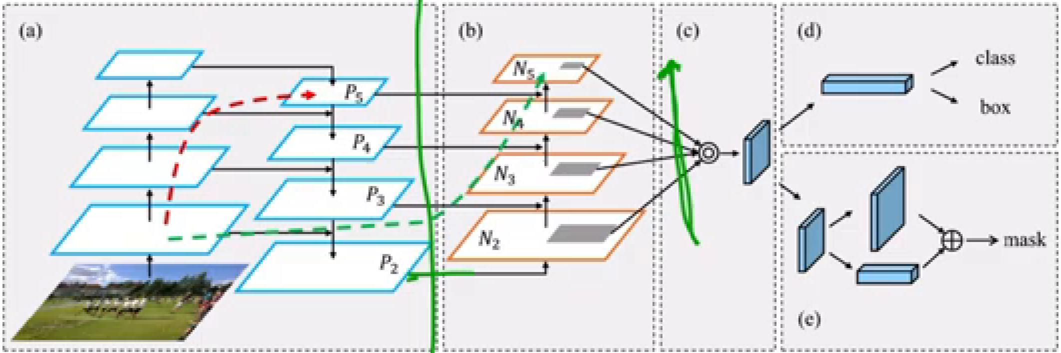


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path augmentation. (c) Adaptive feature pooling. (d) Box branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature maps in (a) and (b) for brevity.

Path Aggregation Network

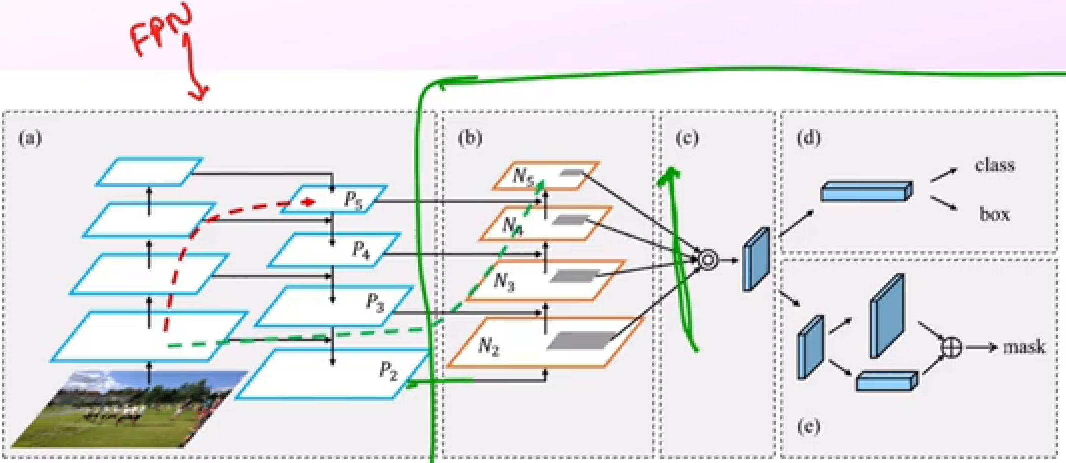


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path augmentation. (c) Adaptive feature pooling. (d) Box branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature maps in (a) and (b) for brevity.

Path Aggregation Network

FPN

"To shorten the info path"

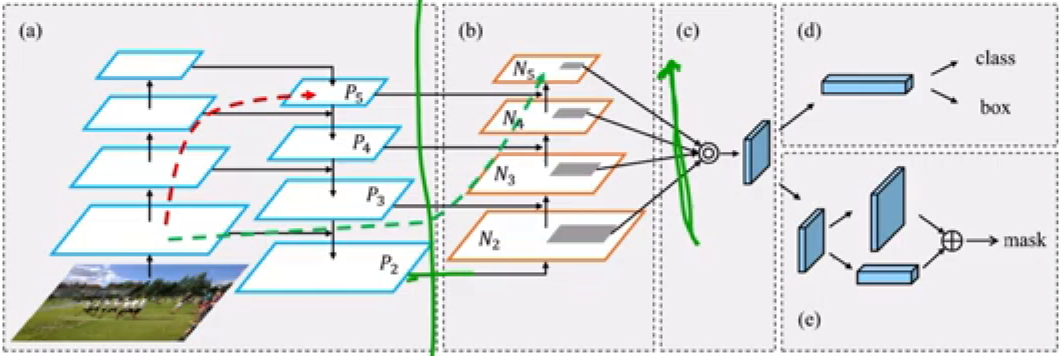
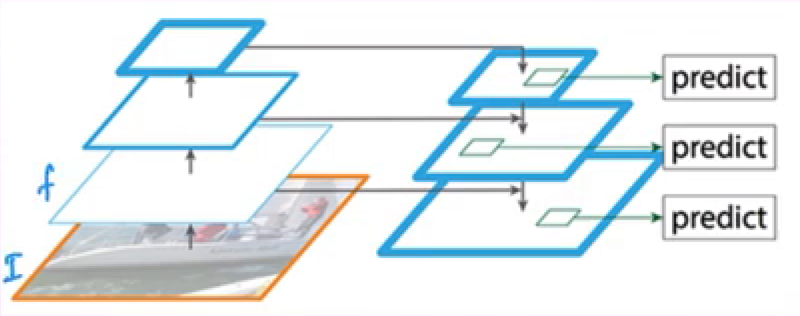
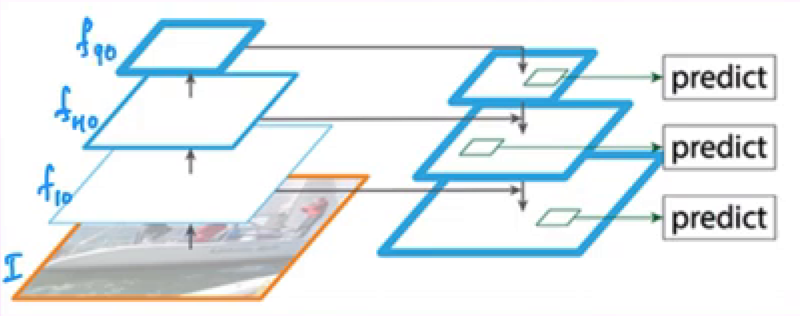


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path augmentation. (c) Adaptive feature pooling. (d) Box branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature maps in (a) and (b) for brevity.

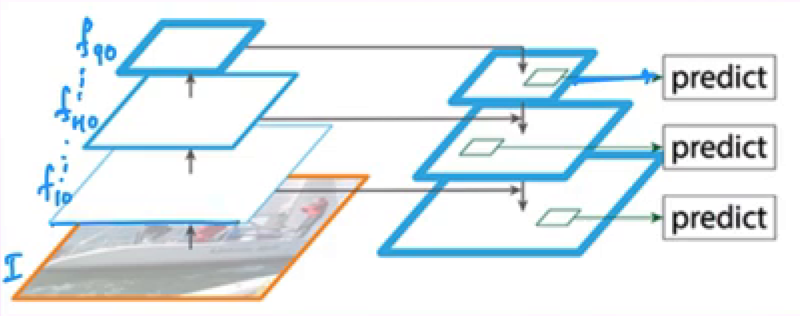
Feature Pyramid Network (FPN)



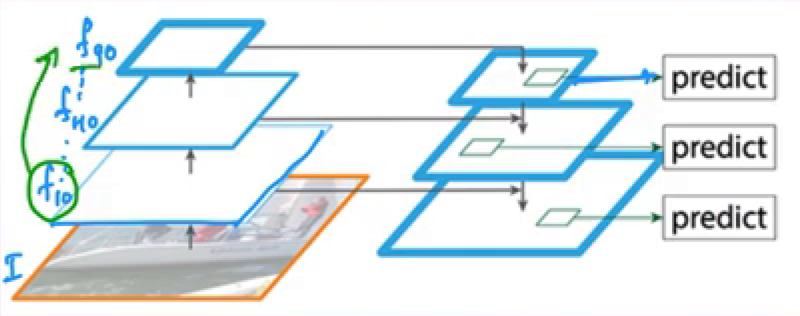
Feature Pyramid Network (FPN)



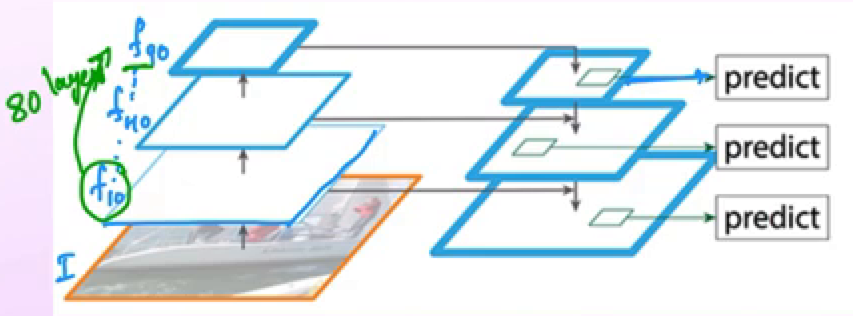
Feature Pyramid Network (FPN)



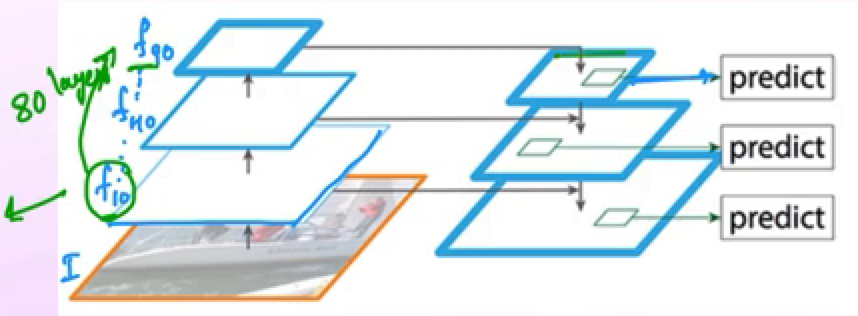
Feature Pyramid Network (FPN)



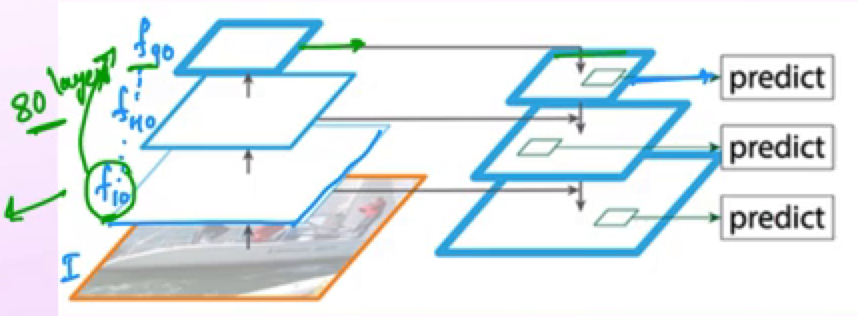
Feature Pyramid Network (FPN)



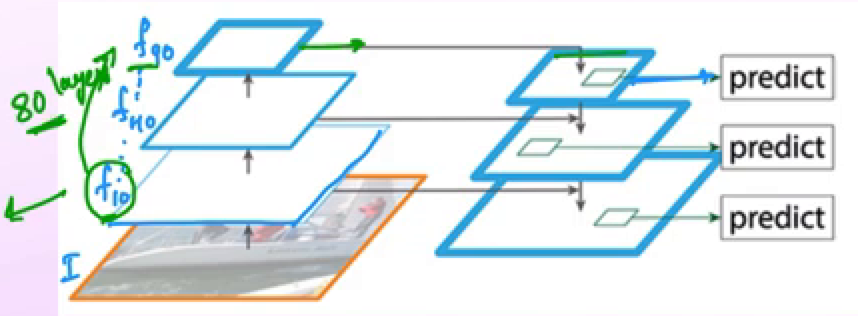
Feature Pyramid Network (FPN)



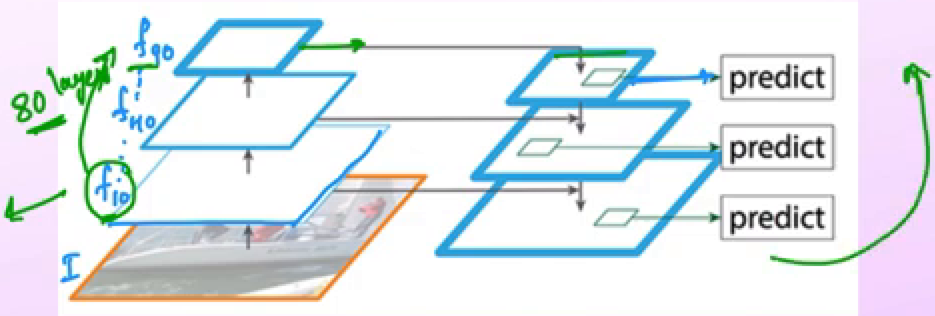
Feature Pyramid Network (FPN)



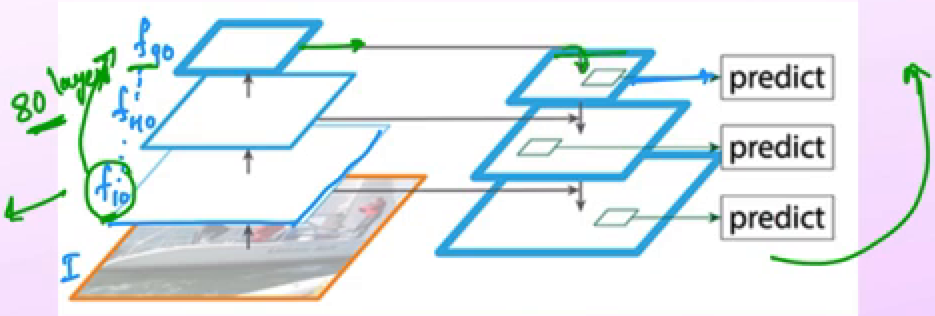
Feature Pyramid Network (FPN)



Feature Pyramid Network (FPN)



Feature Pyramid Network (FPN)



Path Aggregation Network

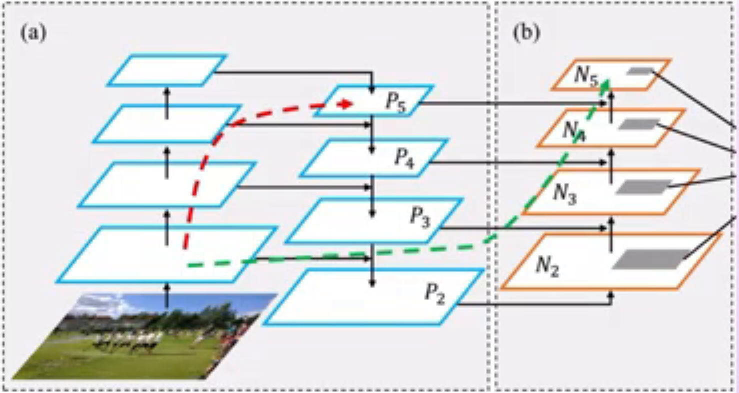


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path
branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature

Path Aggregation Network

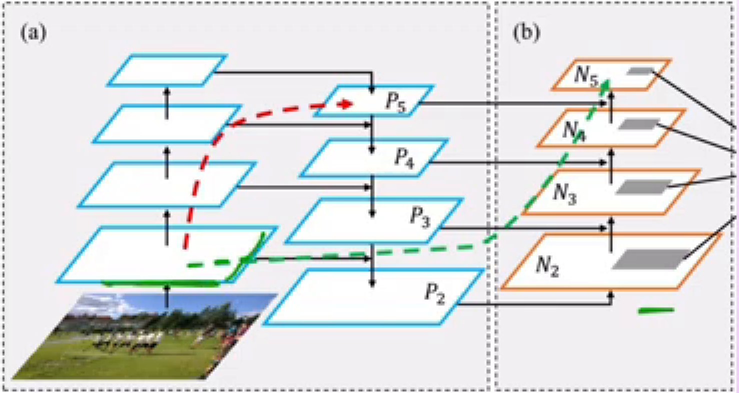


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path
branch. (e) Fully-connected fusion. Note that we omit channel dimension of featur

Path Aggregation Network

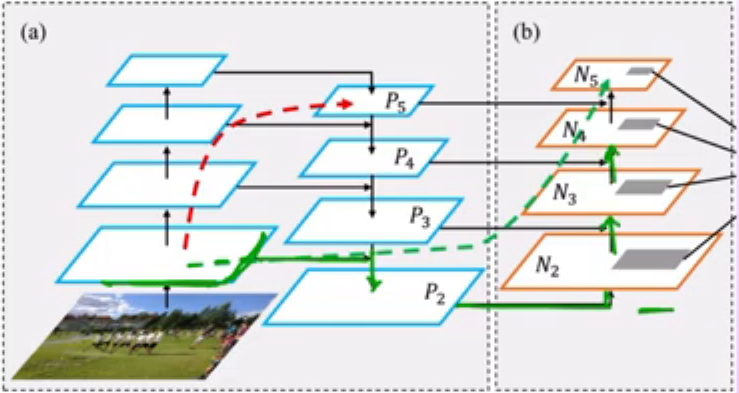


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path
branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature

Path Aggregation Network

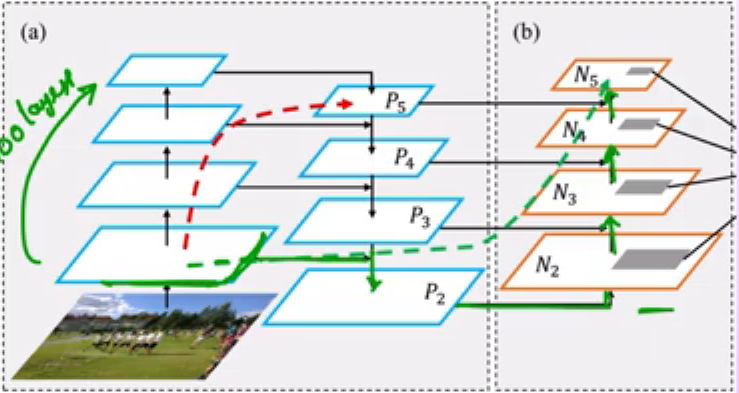


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path
branch. (e) Fully-connected fusion. Note that we omit channel dimension of featur

Path Aggregation Network

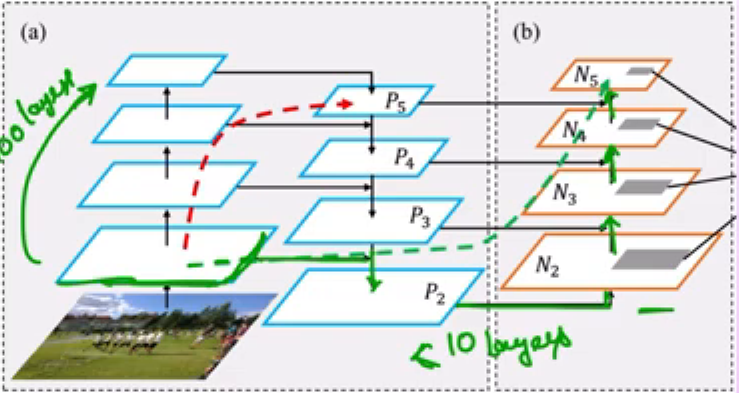


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path
branch. (e) Fully-connected fusion. Note that we omit channel dimension of featur

Path Aggregation Network

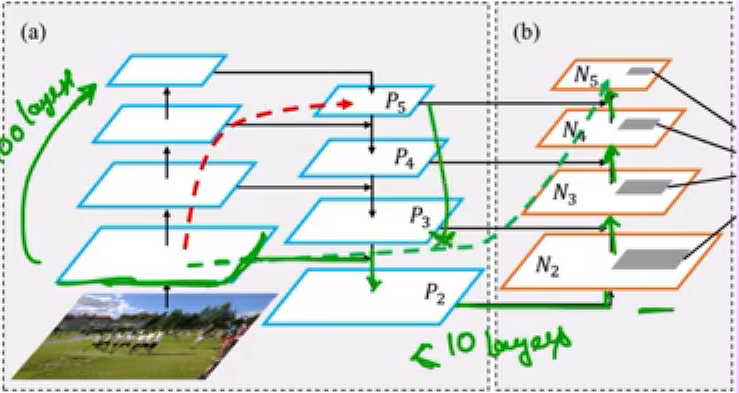


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path
branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature

Path Aggregation Network

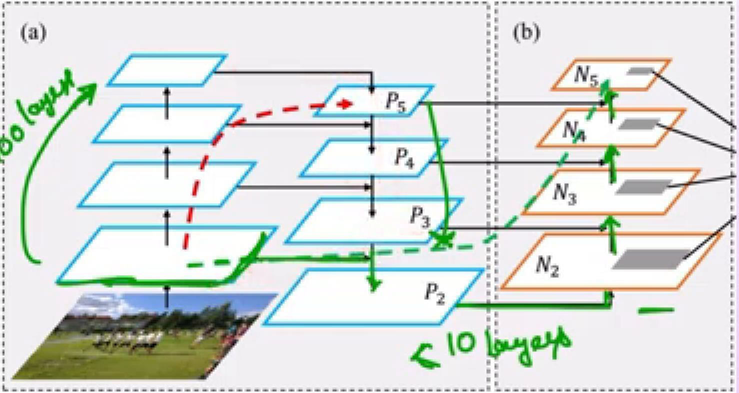


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path
branch. (e) Fully-connected fusion. Note that we omit channel dimension of featur

Path Aggregation Network

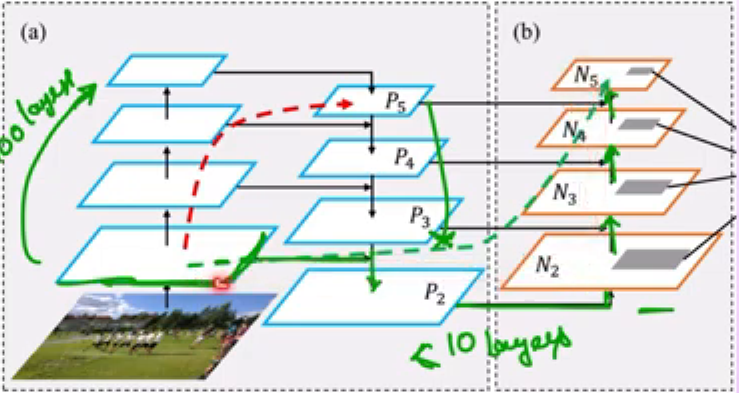


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path
branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature

Path Aggregation Network

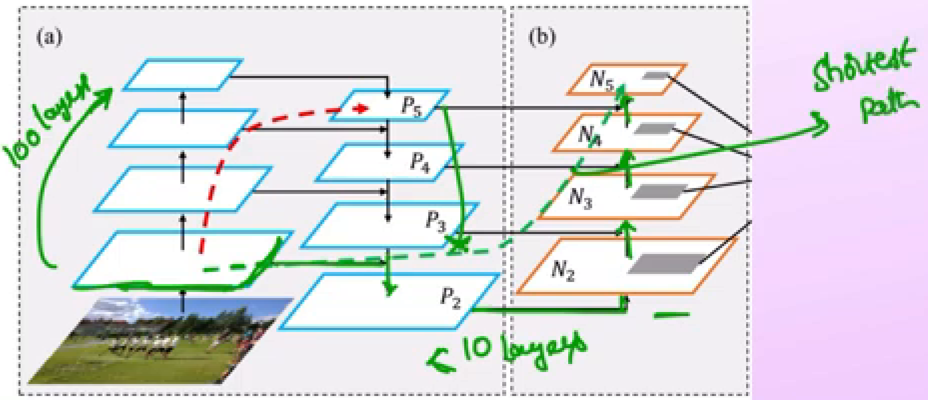


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path
branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature

Path Aggregation Network

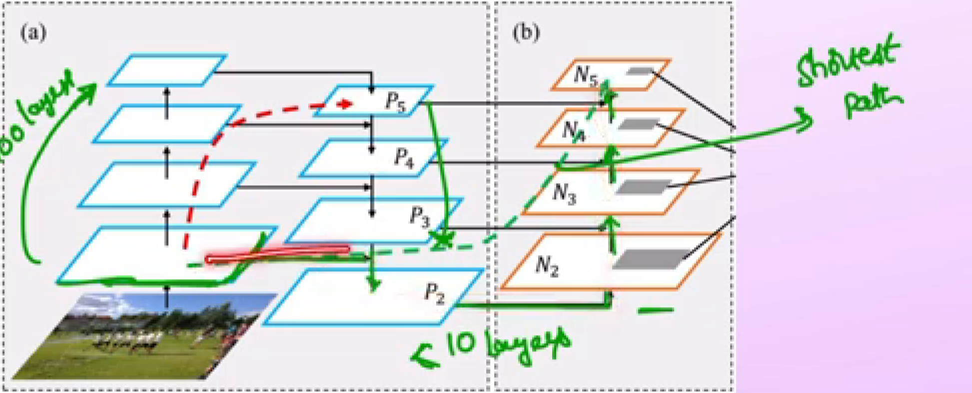


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path
branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature

Path Aggregation Network

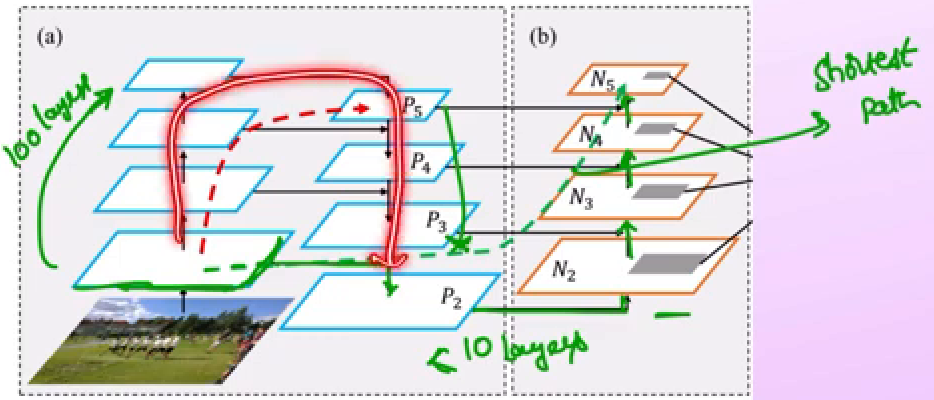


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path
branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature

Path Aggregation Network

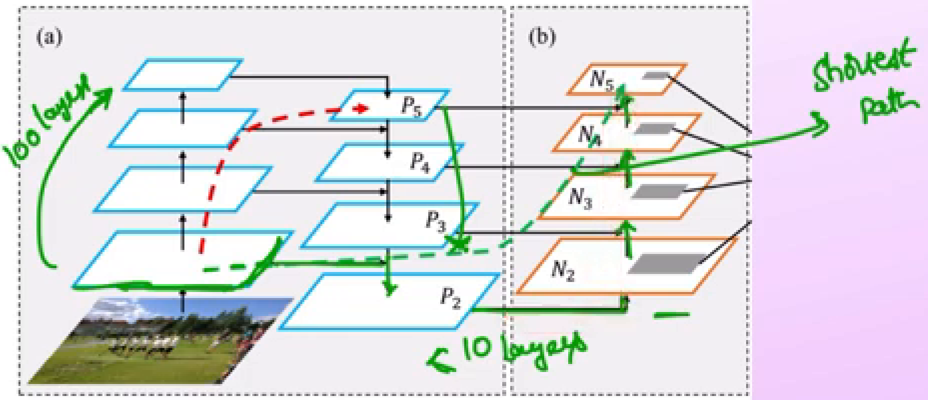


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path
branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature

Path Aggregation Network

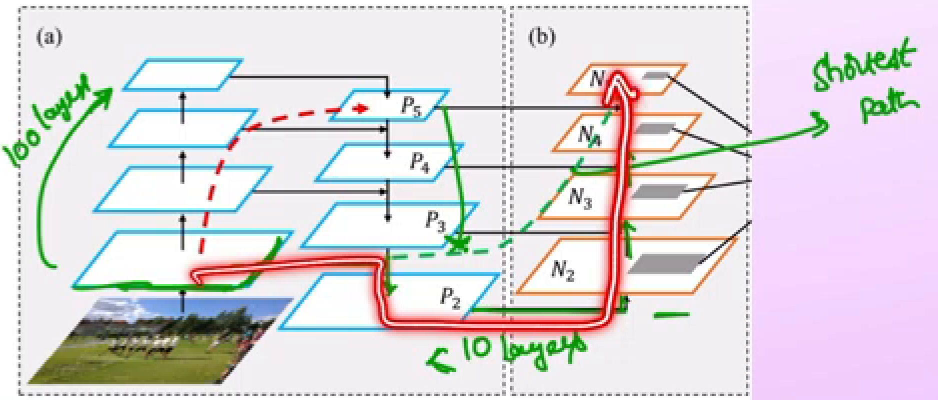


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path
branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature

Path Aggregation Network

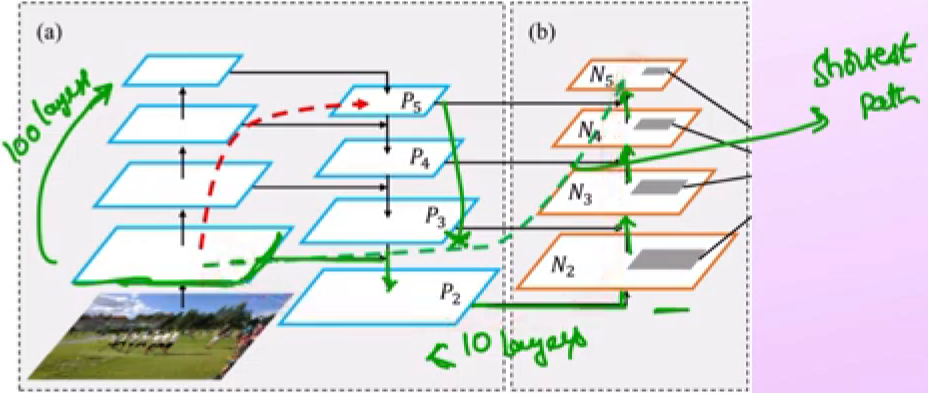


Figure 1. Illustration of our framework. (a) FPN backbone. (b) Bottom-up path
branch. (e) Fully-connected fusion. Note that we omit channel dimension of feature

Path Aggregation Network

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Backbone
Champion 2016 [33]	37.6	59.9	40.4	17.1	41.0	56.0	6×ResNet-101
Mask R-CNN [21]+FPN [35]	35.7	58.0	37.8	15.5	38.1	52.4	ResNet-101
Mask R-CNN [21]+FPN [35]	37.1	60.0	39.4	16.9	39.9	53.5	ResNeXt-101
PANet / PANet [ms-train]	36.6 / 38.2	58.0 / 60.2	39.3 / 41.4	16.3 / 19.1	38.1 / 41.1	53.1 / 52.6	ResNet-50
PANet / PANet [ms-train]	40.0 / **42.0**	62.8 / **65.1**	43.1 / **45.7**	18.8 / **22.4**	42.3 / **44.7**	57.2 / **58.1**	ResNeXt-101

Path Aggregation Network

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Backbone
Champion 2016 [33]	37.6	59.9	40.4	17.1	41.0	56.0	6×ResNet-101
Mask R-CNN [21]+FPN [35]	35.7	58.0	37.8	15.5	38.1	52.4	ResNet-101
Mask R-CNN [21]+FPN [35]	37.1	60.0	39.4	16.9	39.9	53.5	ResNeXt-101
PANet / PANet [ms-train]	36.6 / 38.2	58.0 / 60.2	39.3 / 41.4	16.3 / 19.1	38.1 / 41.1	53.1 / 52.6	ResNet-50
PANet / PANet [ms-train]	40.0 / **42.0**	62.8 / **65.1**	43.1 / **45.7**	18.8 / **22.4**	42.3 / **44.7**	57.2 / **58.1**	ResNeXt-101

Path Aggregation Network

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Backbone
Champion 2016 [33]	37.6	59.9	40.4	17.1	41.0	56.0	6×ResNet-101
Mask R-CNN [21]+FPN [35]	35.7	58.0	37.8	15.5	38.1	52.4	ResNet-101
Mask R-CNN [21]+FPN [35]	37.1	60.0	39.4	16.9	39.9	53.5	ResNeXt-101
PANet / PANet [ms-train]	36.6 / 38.2	58.0 / 60.2	39.3 / 41.4	16.3 / 19.1	38.1 / 41.1	53.1 / 52.6	ResNet-50
PANet / PANet [ms-train]	40.0 / **42.0**	62.8 / **65.1**	43.1 / **45.7**	18.8 / **22.4**	42.3 / **44.7**	57.2 / **58.1**	ResNeXt-101

Path Aggregation Network

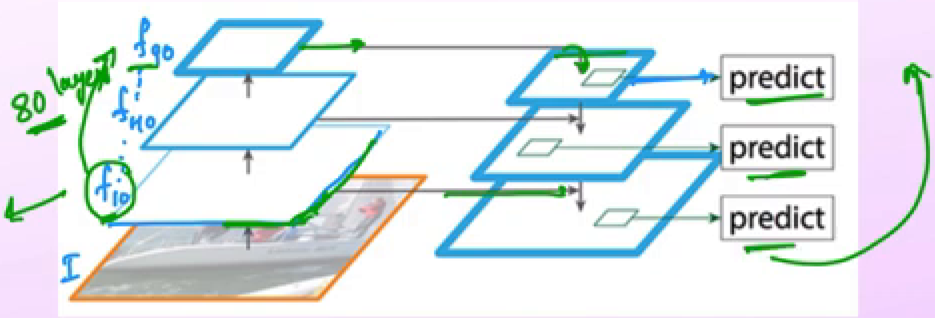
Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Backbone
Champion 2016 [33]	37.6	59.9	40.4	17.1	41.0	56.0	6×ResNet-101
Mask R-CNN [21]+FPN [35]	35.7	58.0	37.8	15.5	38.1	52.4	ResNet-101
Mask R-CNN [21]+FPN [35]	37.1	60.0	39.4	16.9	39.9	53.5	ResNeXt-101
PANet / PANet [ms-train]	36.6 / 38.2	58.0 / 60.2	39.3 / 41.4	16.3 / 19.1	38.1 / 41.1	53.1 / 52.6	ResNet-50
PANet / PANet [ms-train]	40.0 / **42.0**	62.8 / **65.1**	43.1 / **45.7**	18.8 / **22.4**	42.3 / **44.7**	57.2 / **58.1**	ResNeXt-101

Path Aggregation Network

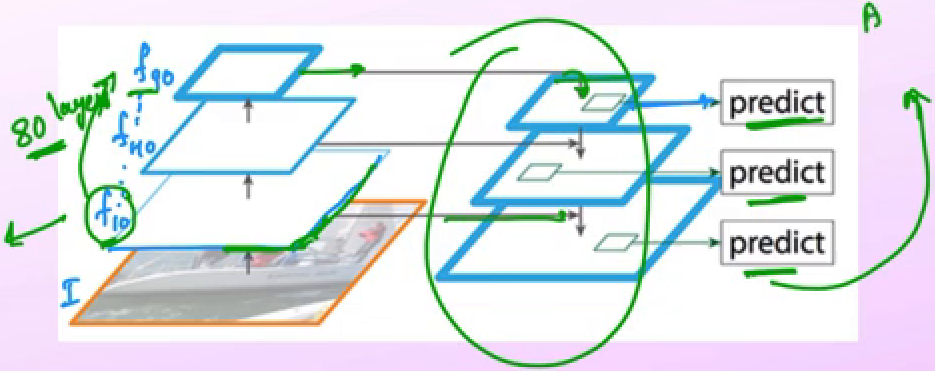
Method	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP_L	Backbone
Champion 2016 [33]	37.6	59.9	40.4	17.1	41.0	56.0	6×ResNet-101
Mask R-CNN [21]+FPN [35]	35.7	58.0	37.8	15.5	38.1	52.4	ResNet-101
Mask R-CNN [21]+FPN [35]	37.1	60.0	39.4	16.9	39.9	53.5	ResNeXt-101
PANet / PANet [ms-train]	36.6 / 38.2	58.0 / 60.2	39.3 / 41.4	16.3 / 19.1	38.1 / 41.1	53.1 / 52.6	ResNet-50
PANet / PANet [ms-train]	40.0 / **42.0**	62.8 / **65.1**	43.1 / **45.7**	18.8 / **22.4**	42.3 / **44.7**	57.2 / **58.1**	ResNeXt-101

2-3 % ↑

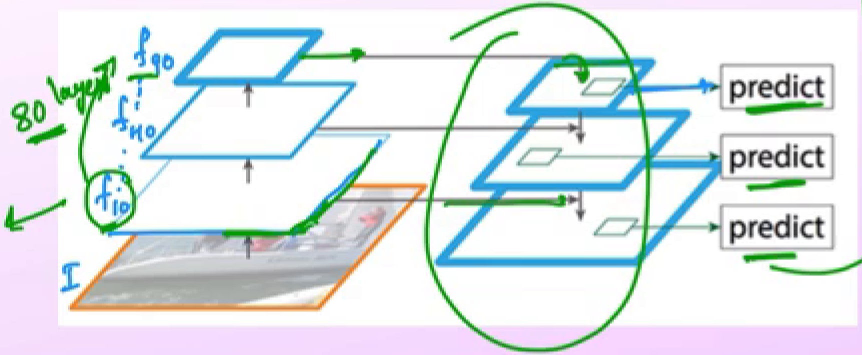
Feature Pyramid Network (FPN)



Feature Pyramid Network (FPN)

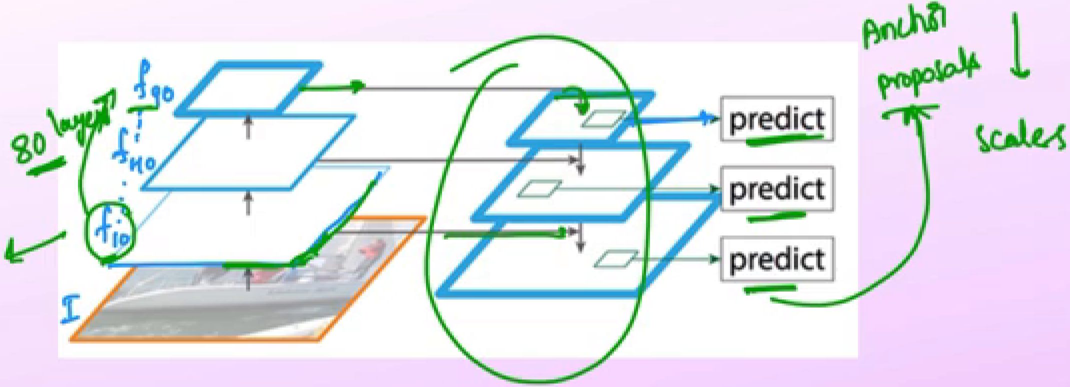


Feature Pyramid Network (FPN)

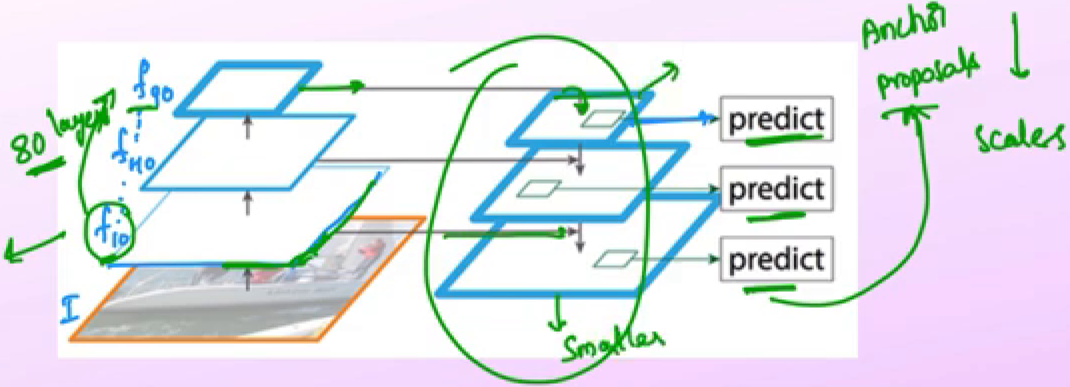


Anchor proposals

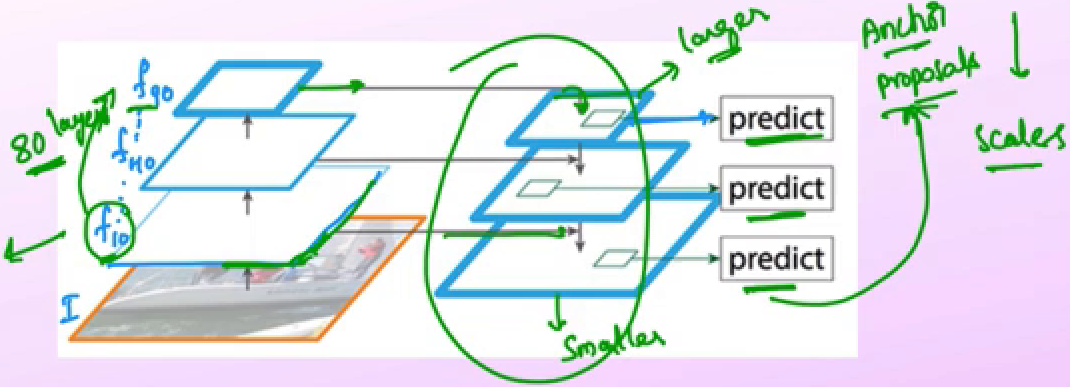
Feature Pyramid Network (FPN)



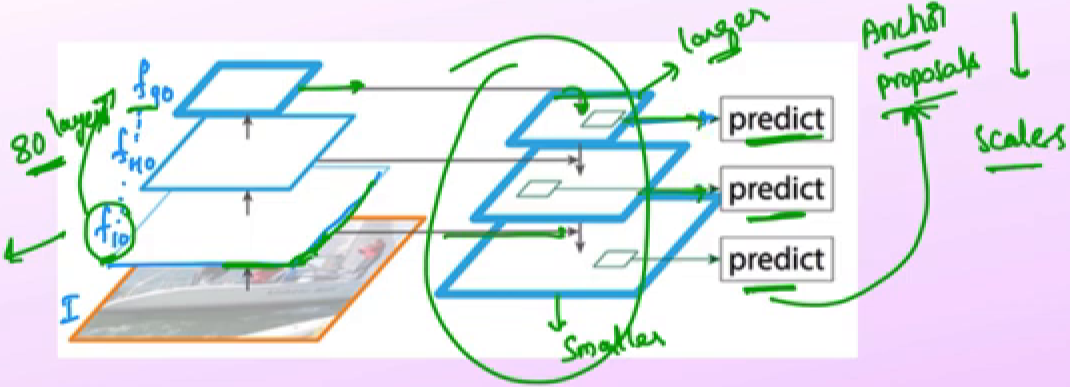
Feature Pyramid Network (FPN)



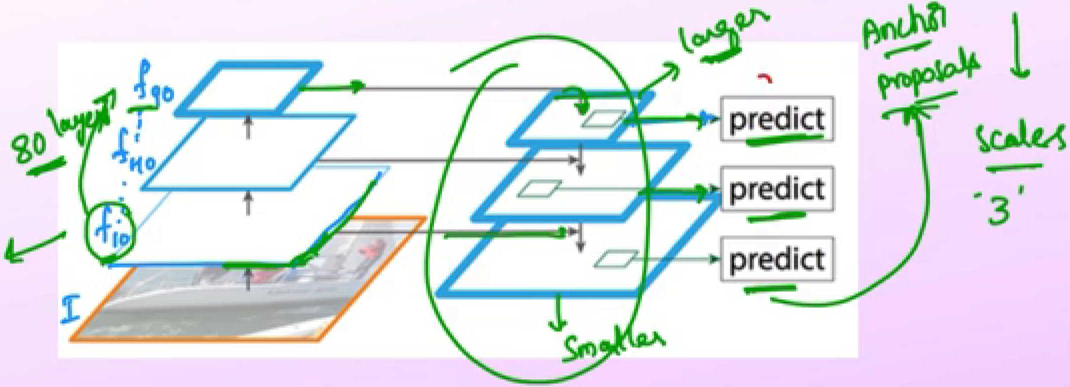
Feature Pyramid Network (FPN)



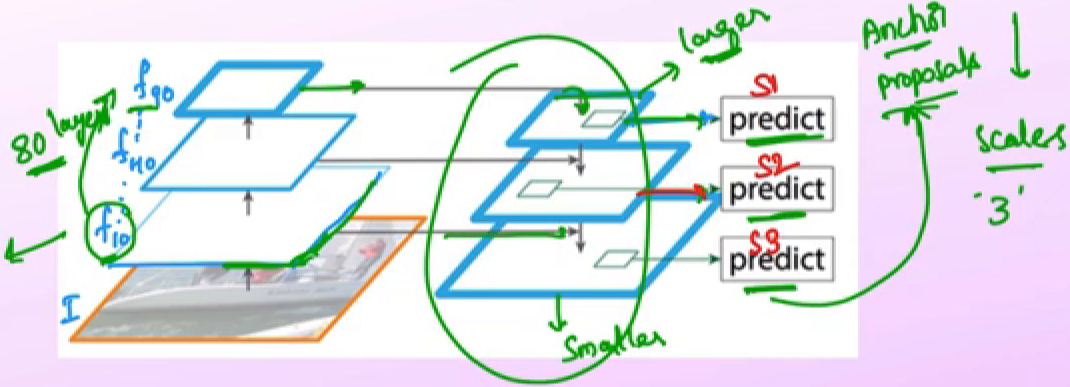
Feature Pyramid Network (FPN)



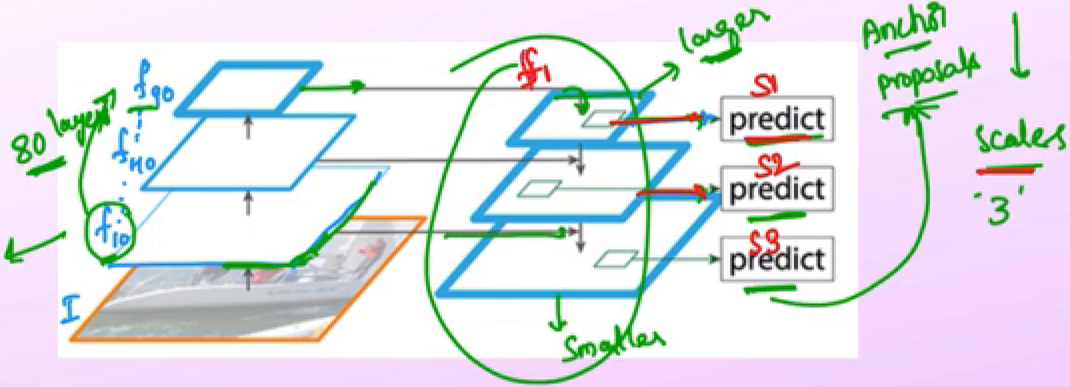
Feature Pyramid Network (FPN)



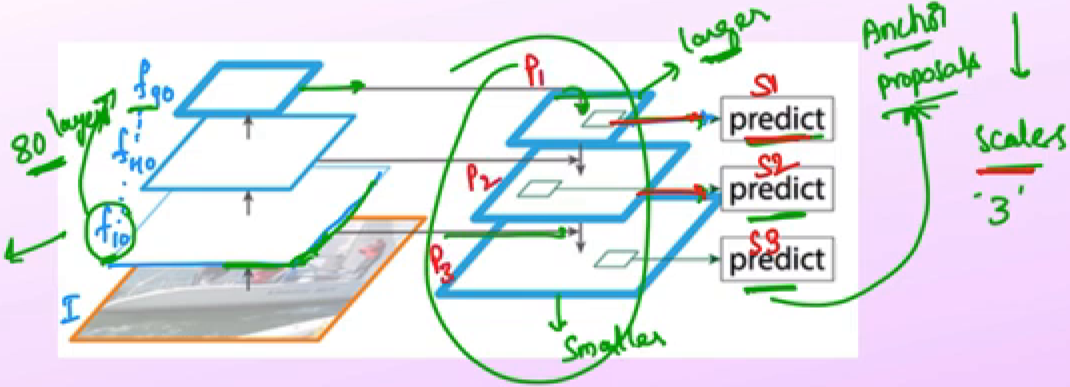
Feature Pyramid Network (FPN)



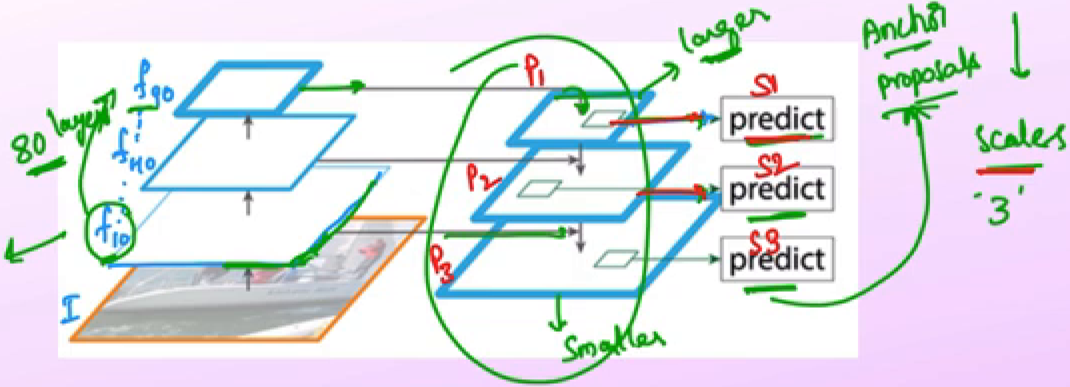
Feature Pyramid Network (FPN)



Feature Pyramid Network (FPN)



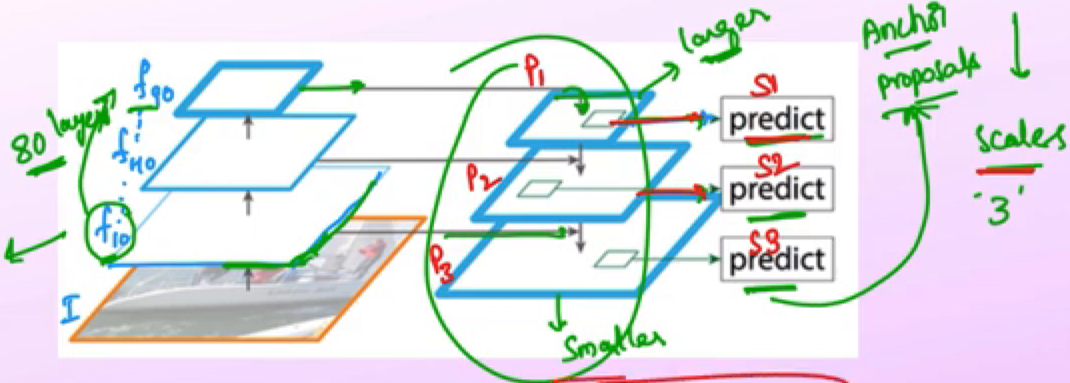
Feature Pyramid Network (FPN)



$P_1 \longrightarrow S_1$

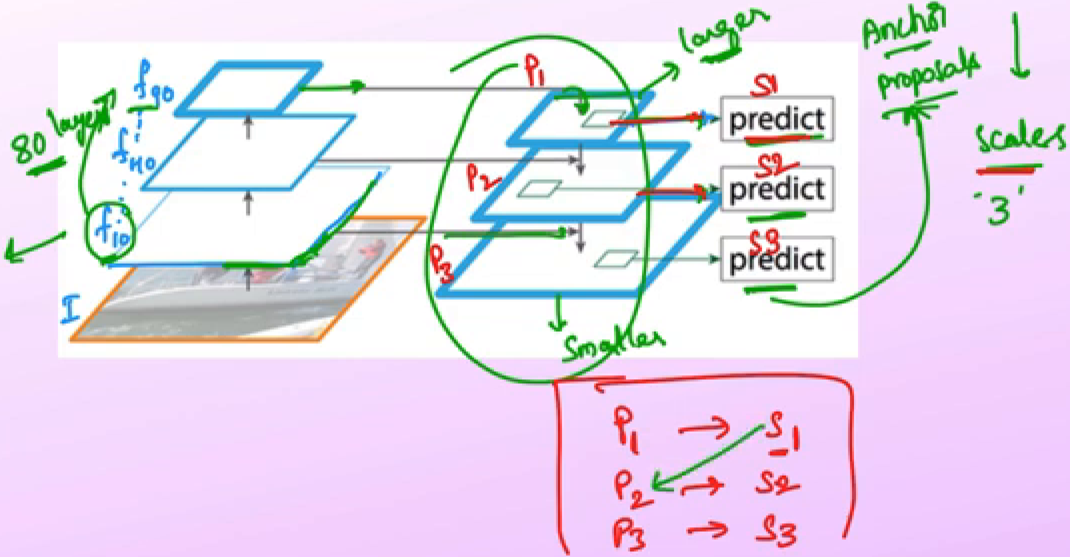
P_2

Feature Pyramid Network (FPN)

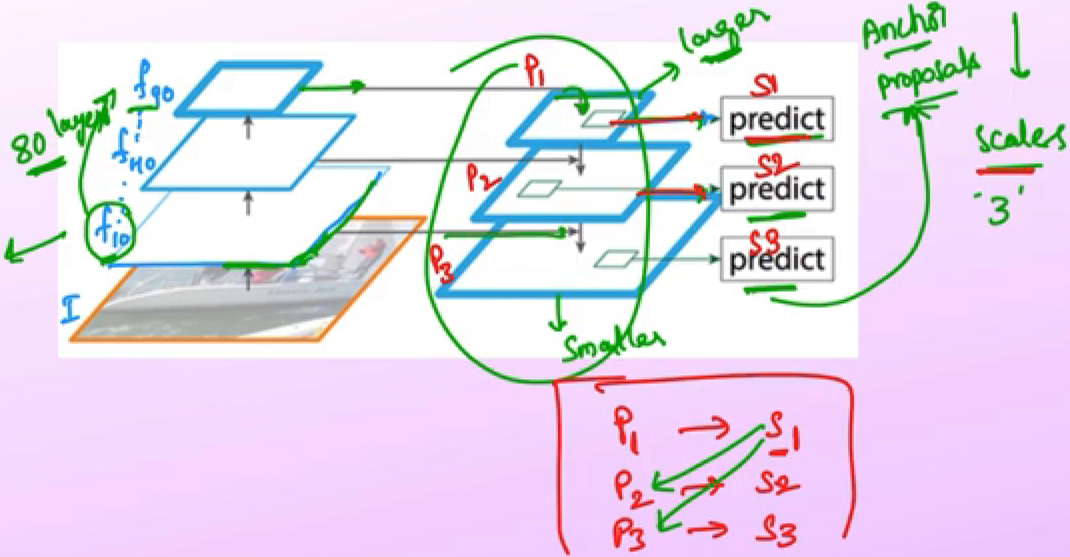


$P_1 \rightarrow S_1$
$P_2 \rightarrow S_2$
$P_3 \rightarrow S_3$

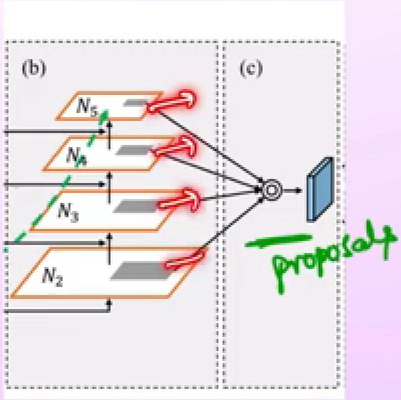
Feature Pyramid Network (FPN)



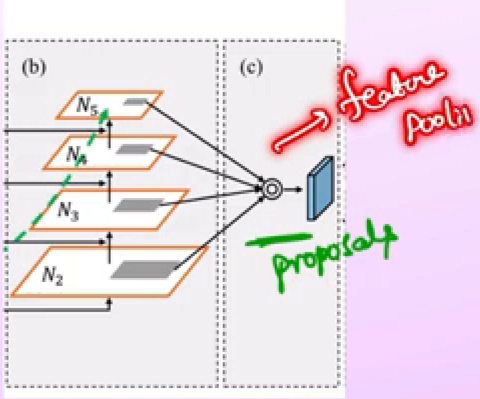
Feature Pyramid Network (FPN)



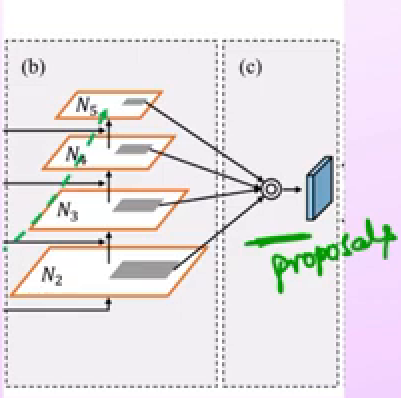
Path Aggregation Network



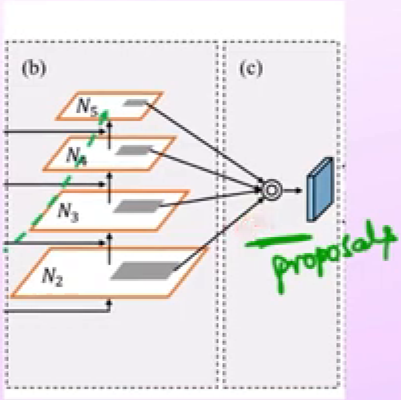
Path Aggregation Network



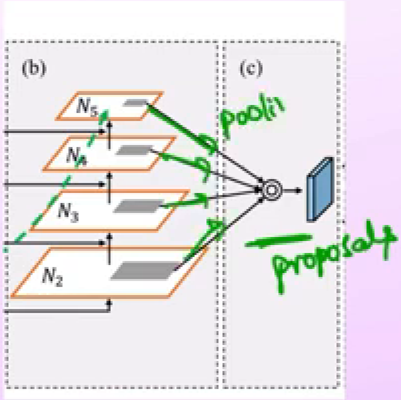
Path Aggregation Network



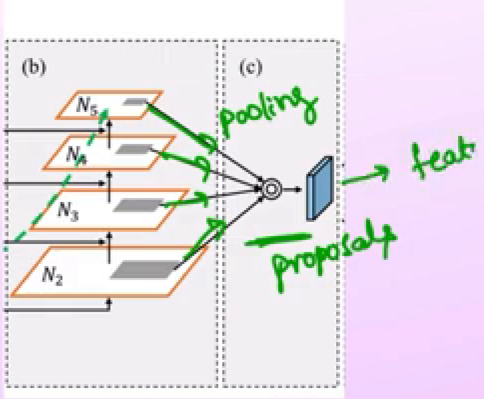
Path Aggregation Network



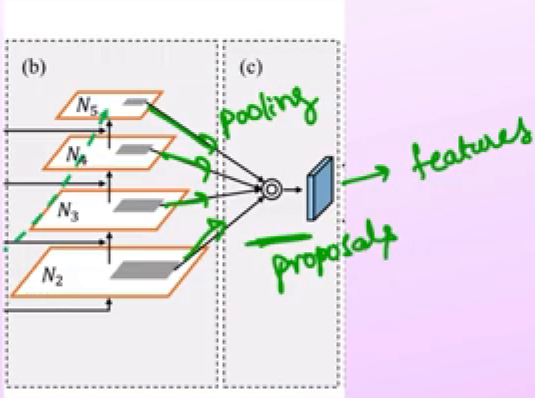
Path Aggregation Network



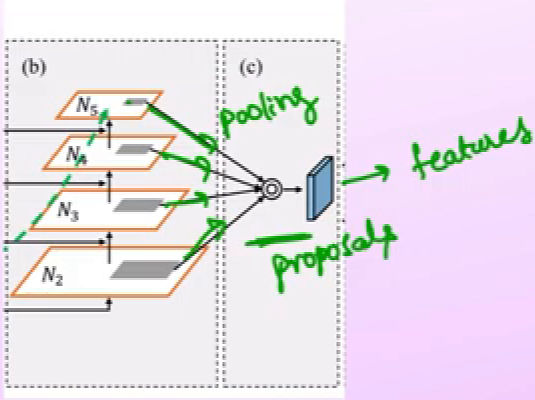
Path Aggregation Network



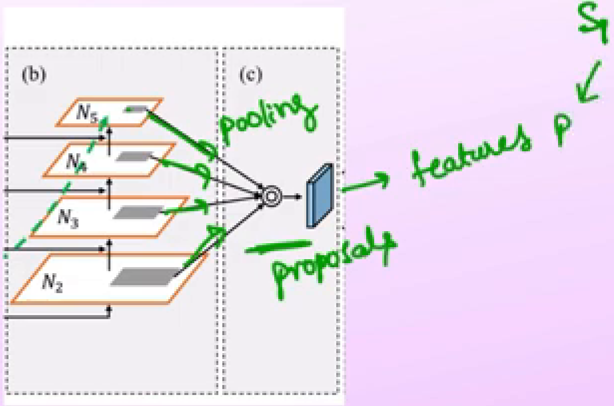
Path Aggregation Network



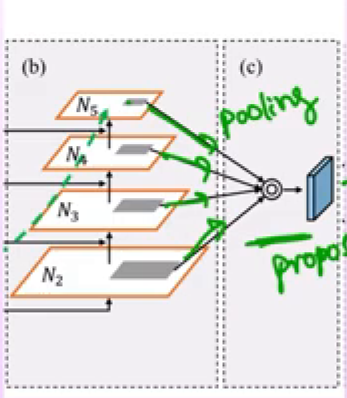
Path Aggregation Network



Path Aggregation Network



Path Aggregation Network

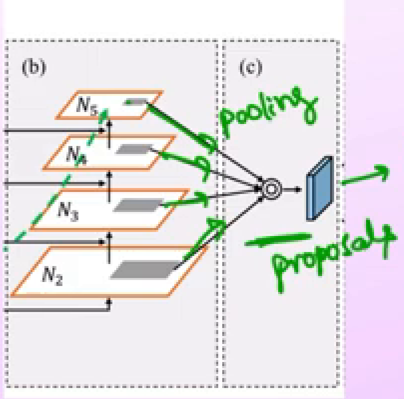


→ features

S

P_1 P_2 P_3 P_4

Path Aggregation Network

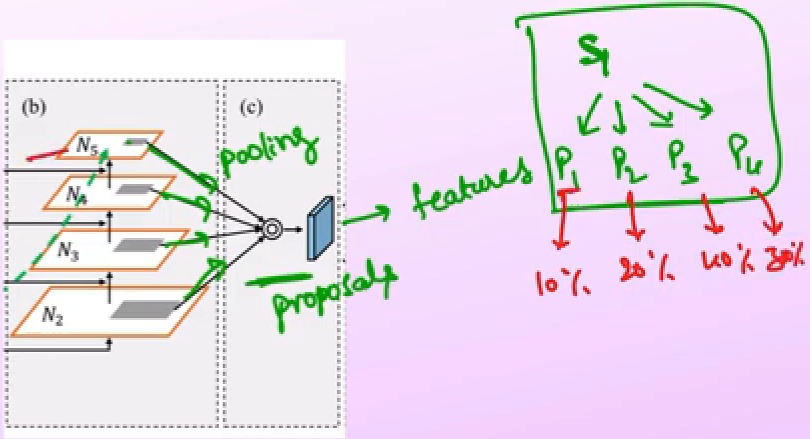


S_1

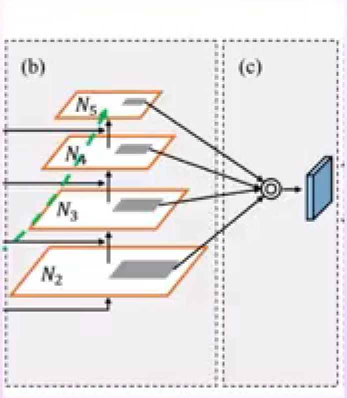
P_1 P_2 P_3 P_4

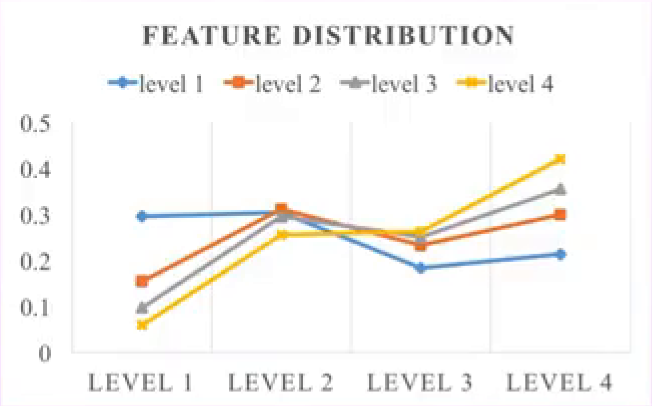
10%

Path Aggregation Network

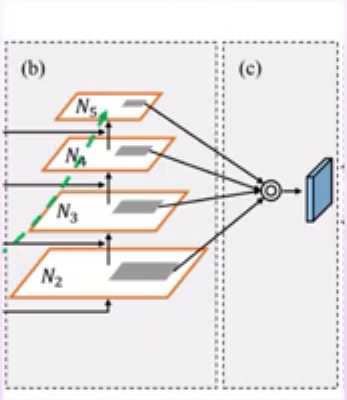


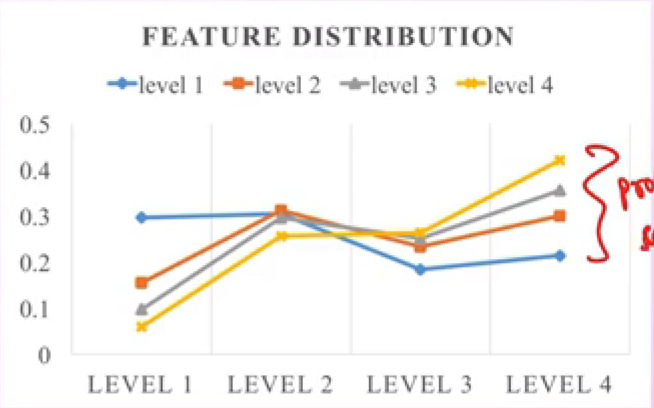
Path Aggregation Network





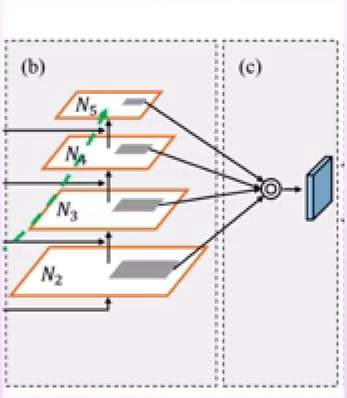
Path Aggregation Network

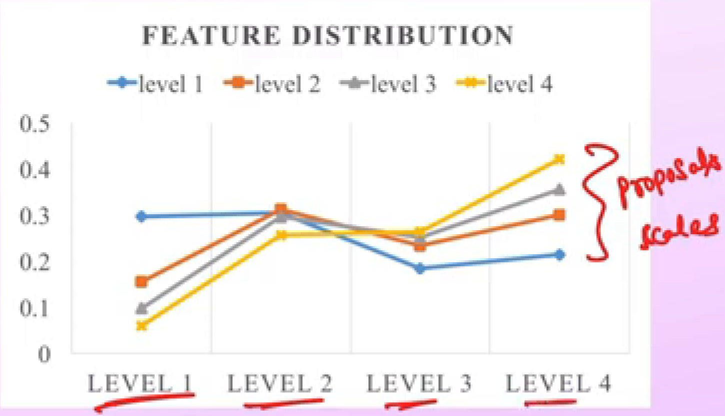




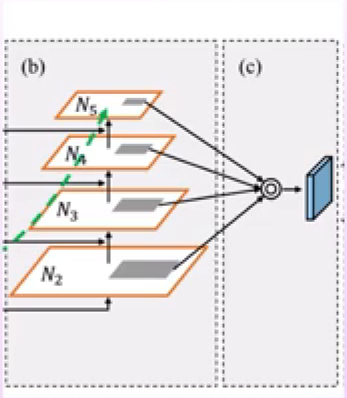
proposals scales

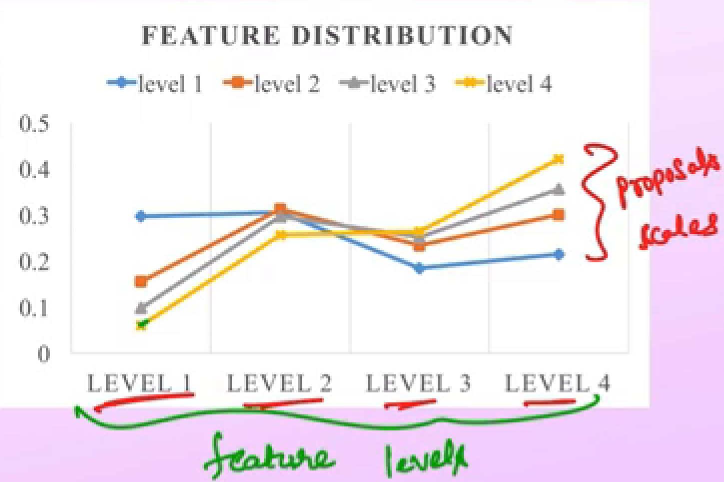
Path Aggregation Network



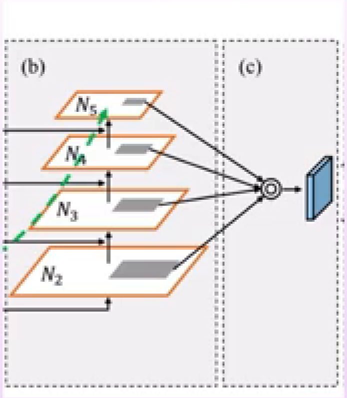


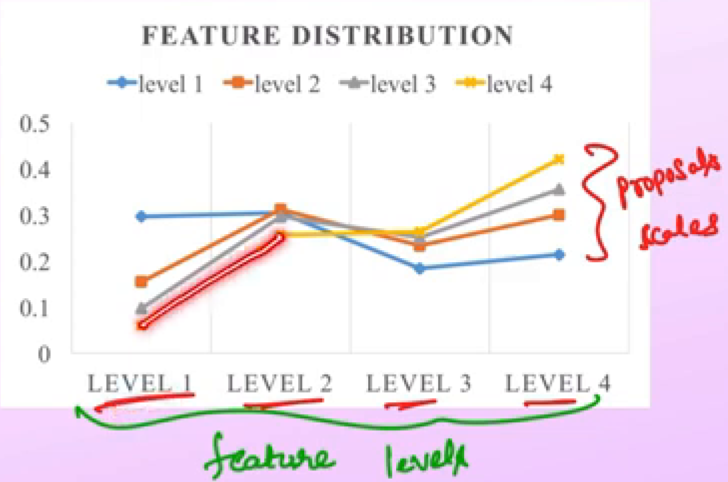
Path Aggregation Network



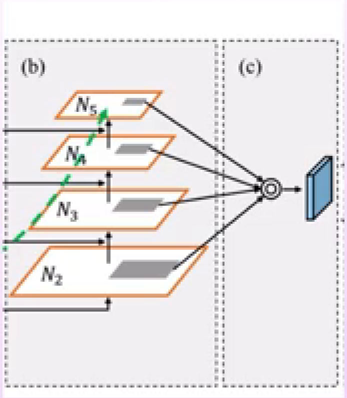


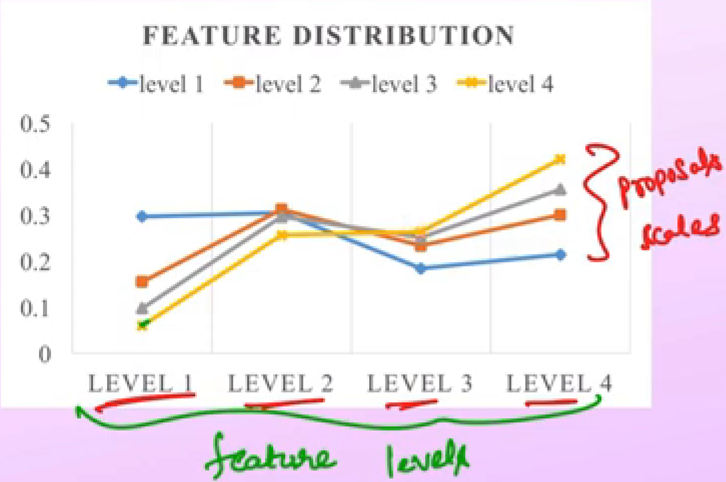
Path Aggregation Network



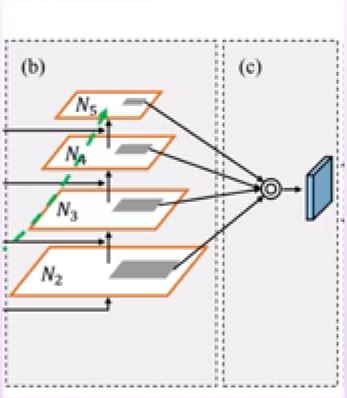


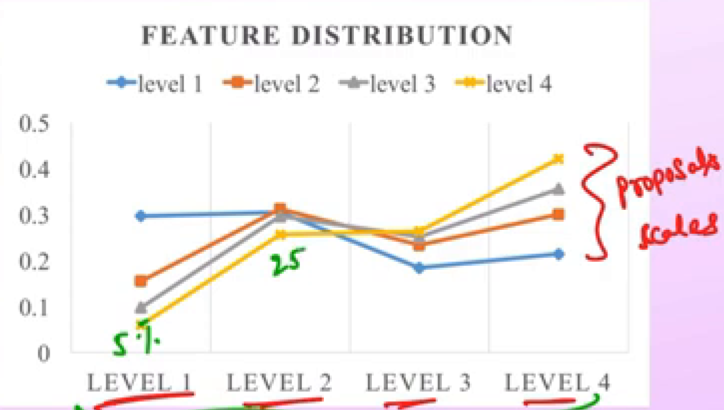
Path Aggregation Network





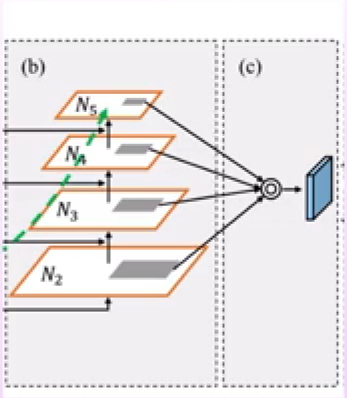
Path Aggregation Network

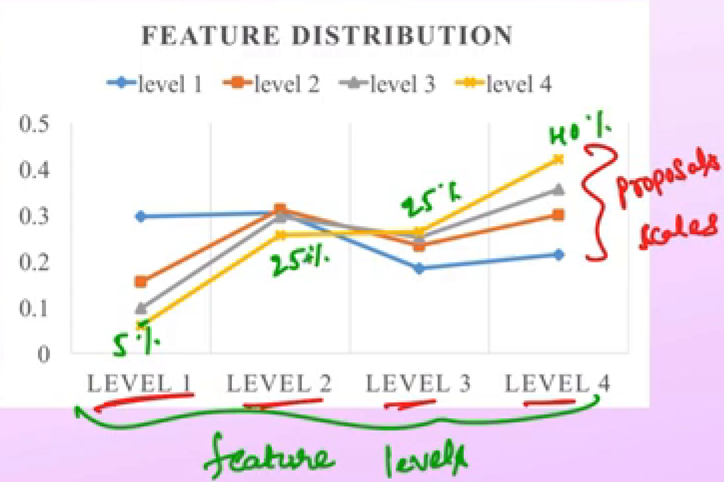




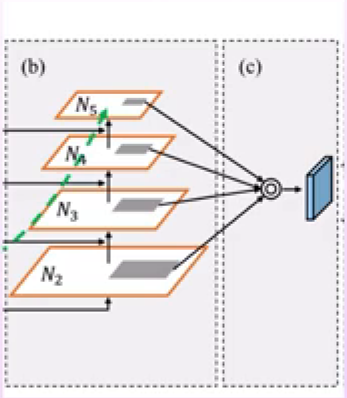
feature levels

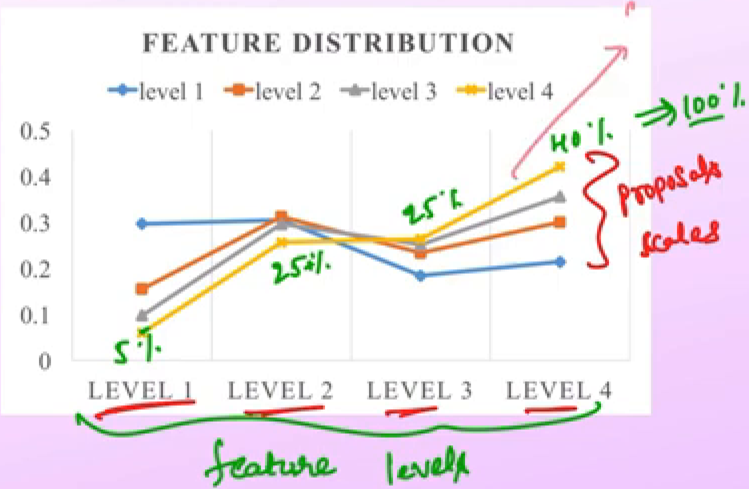
Path Aggregation Network



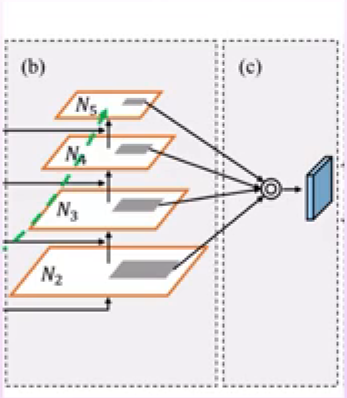


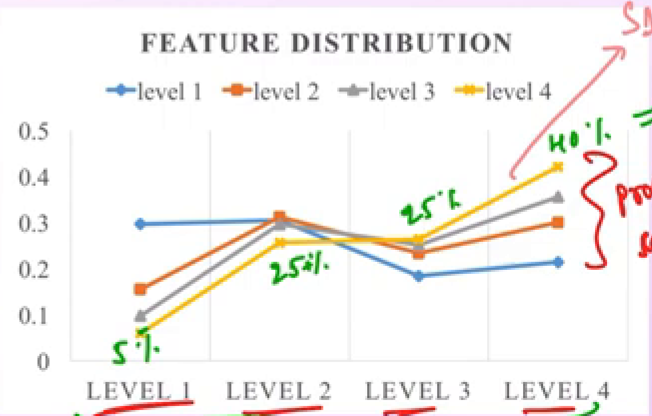
Path Aggregation Network





Path Aggregation Network





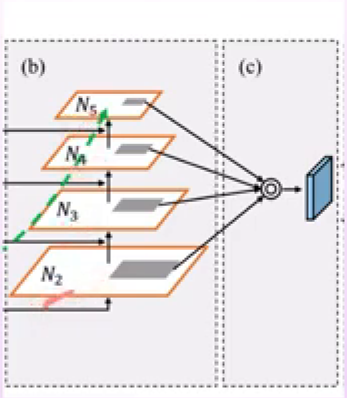
(larger S1 objects)

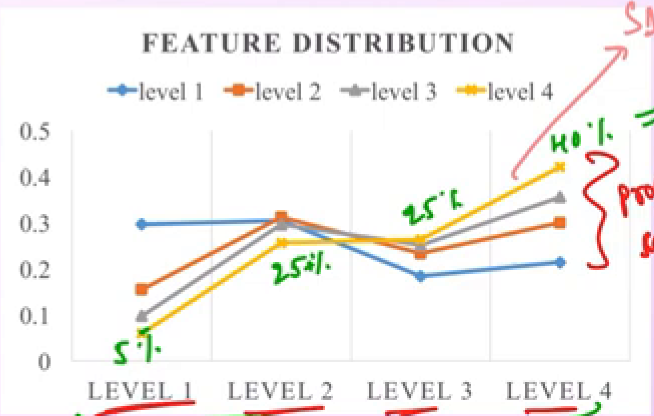
⇒ 100%

proposals scales

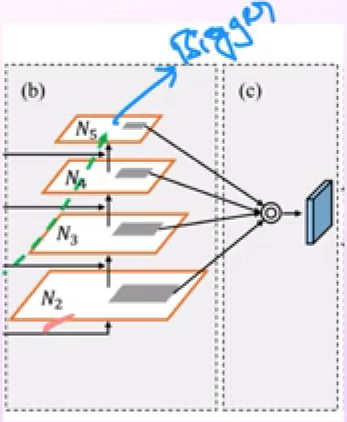
feature levels

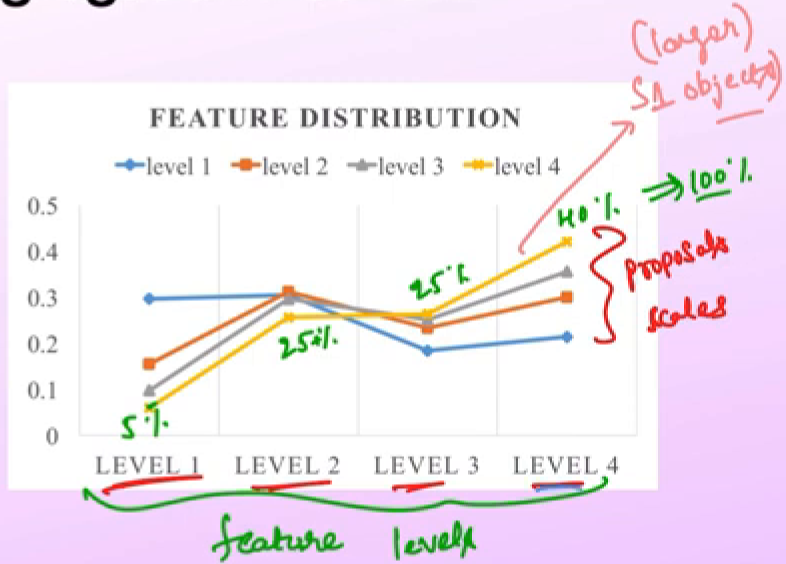
Path Aggregation Network



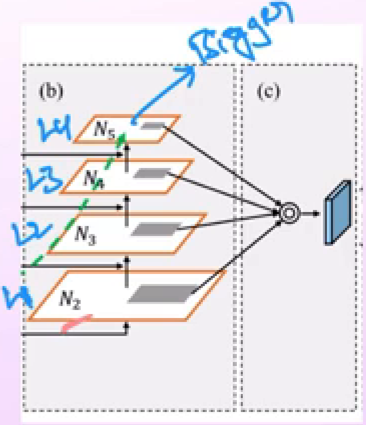


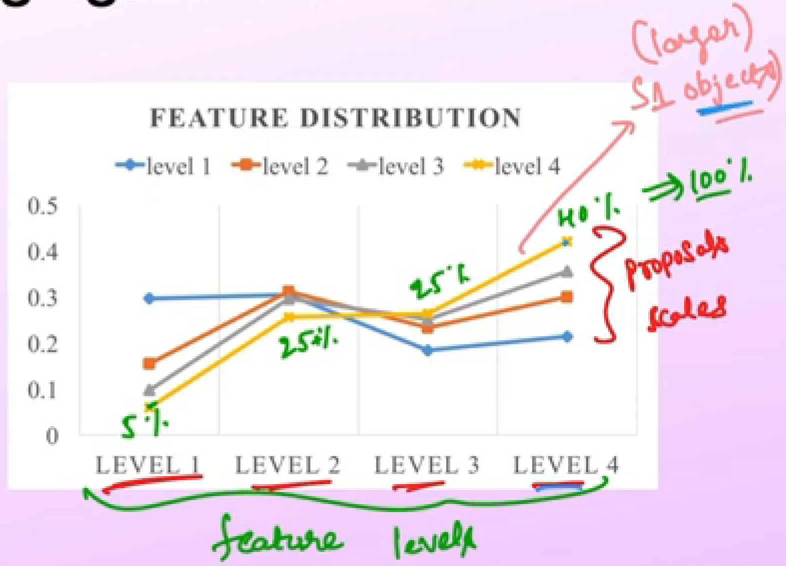
Path Aggregation Network



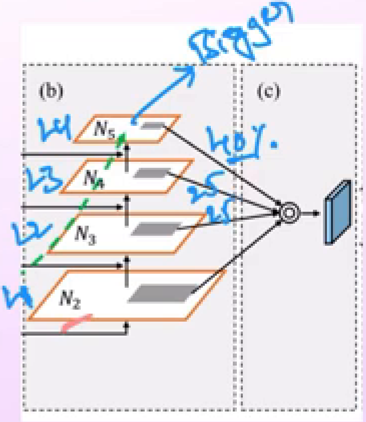


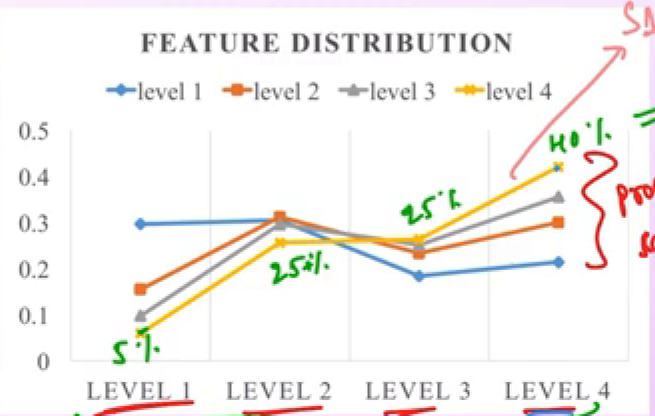
Path Aggregation Network



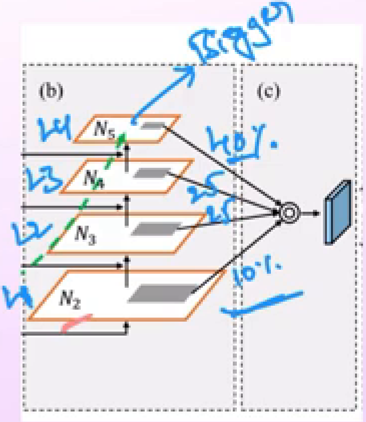


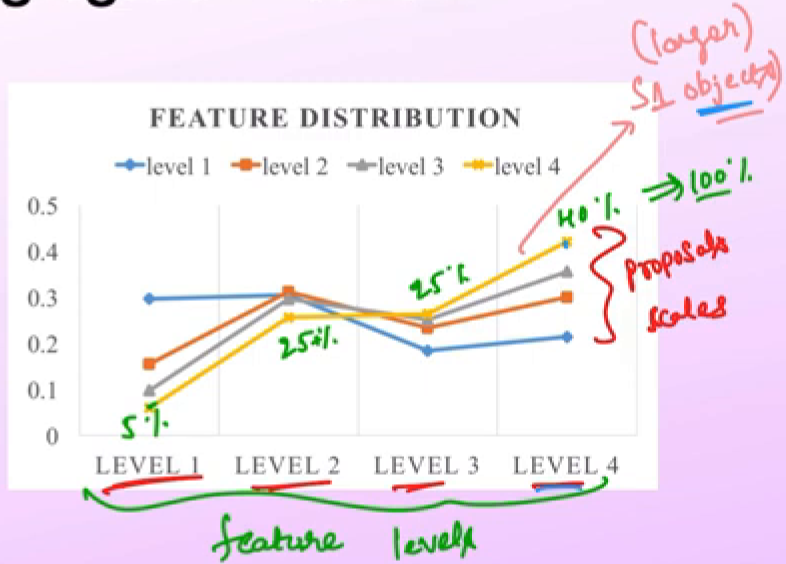
Path Aggregation Network



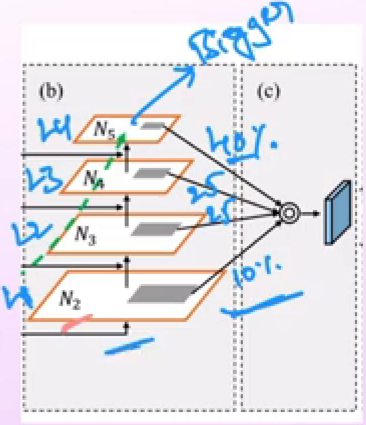


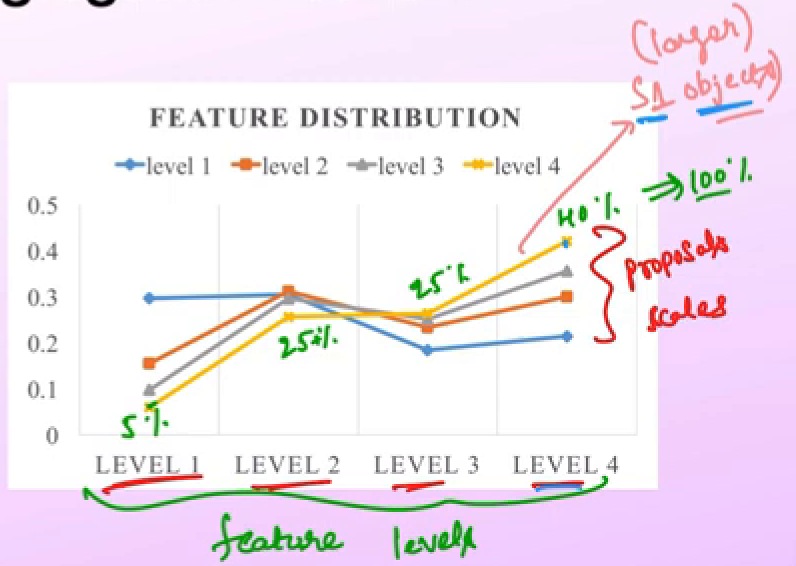
Path Aggregation Network



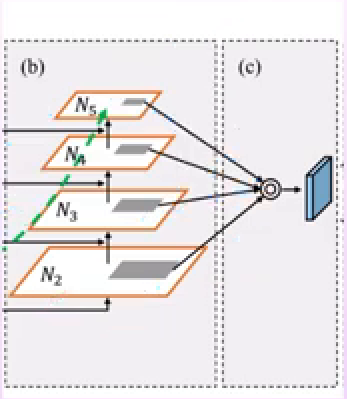


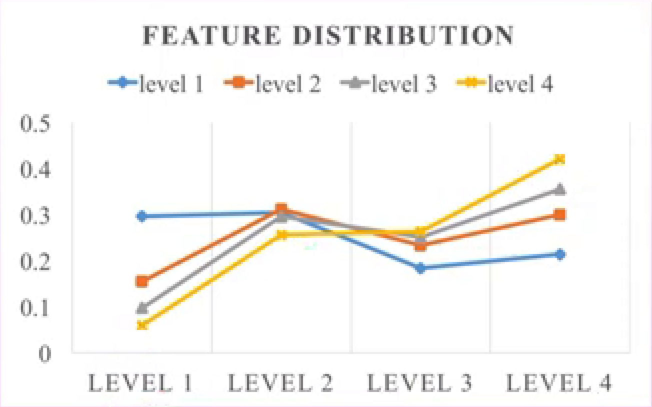
Path Aggregation Network



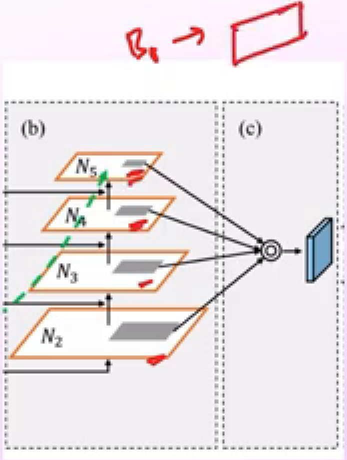


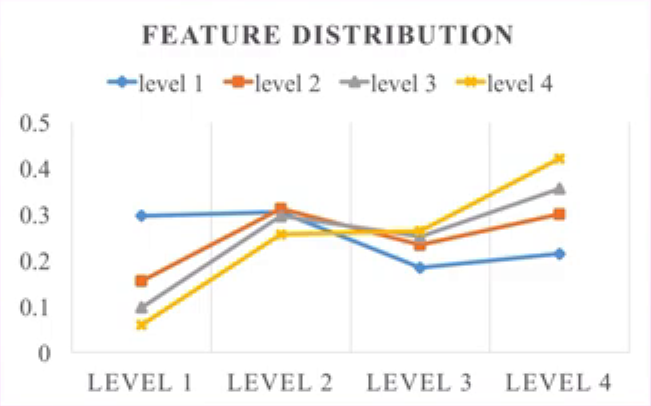
Path Aggregation Network



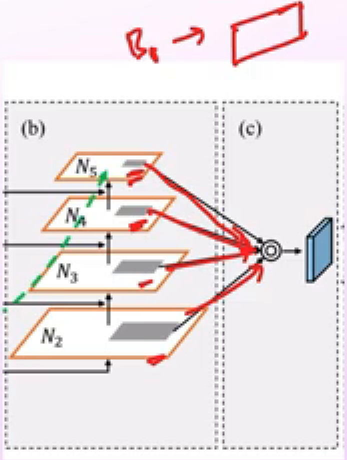


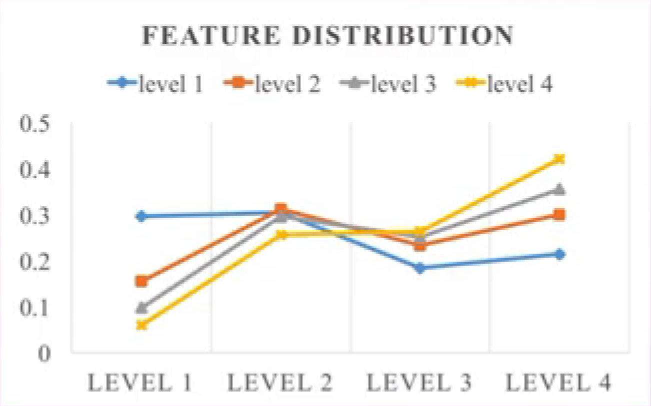
Path Aggregation Network



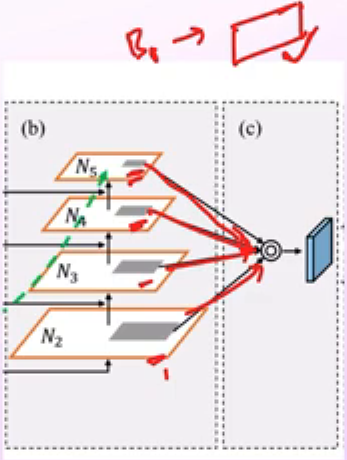


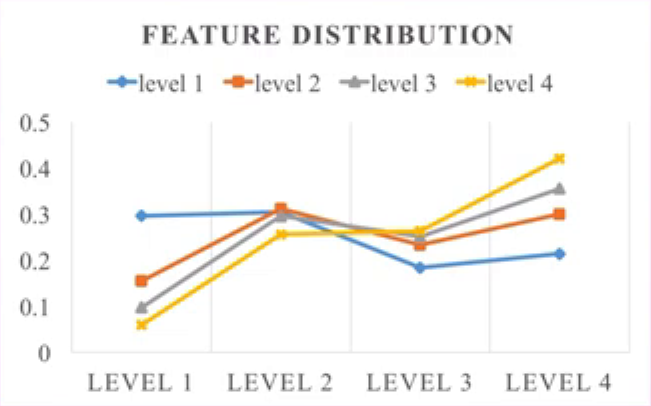
Path Aggregation Network



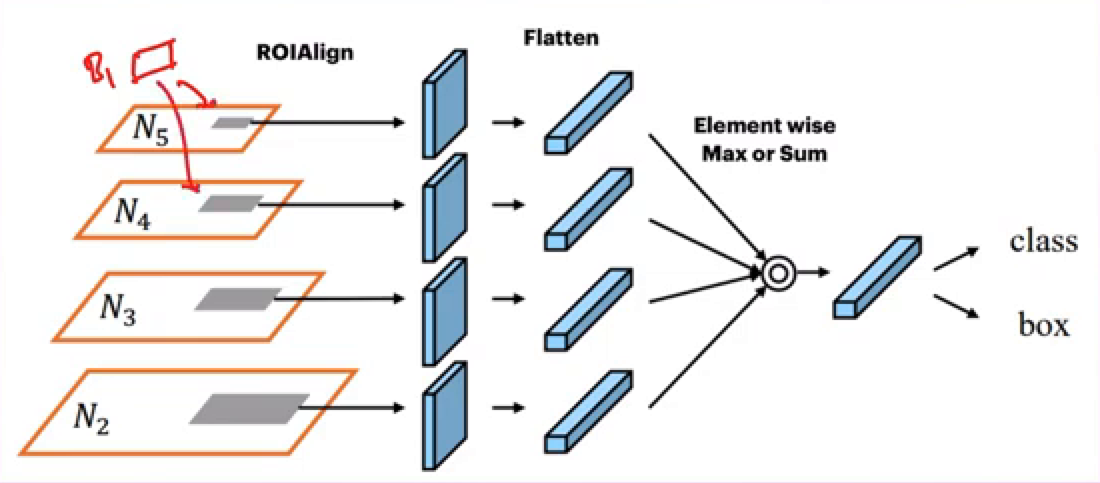


Path Aggregation Network

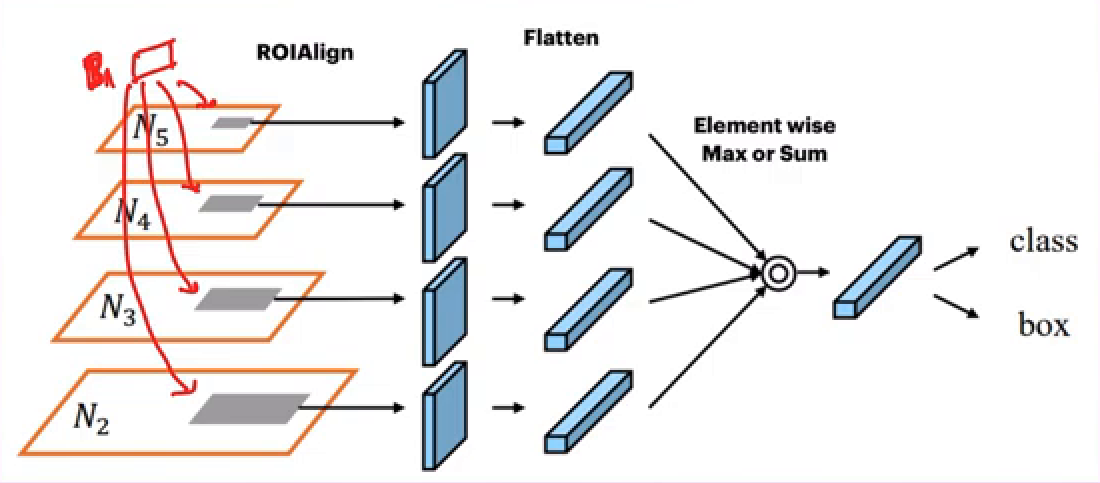




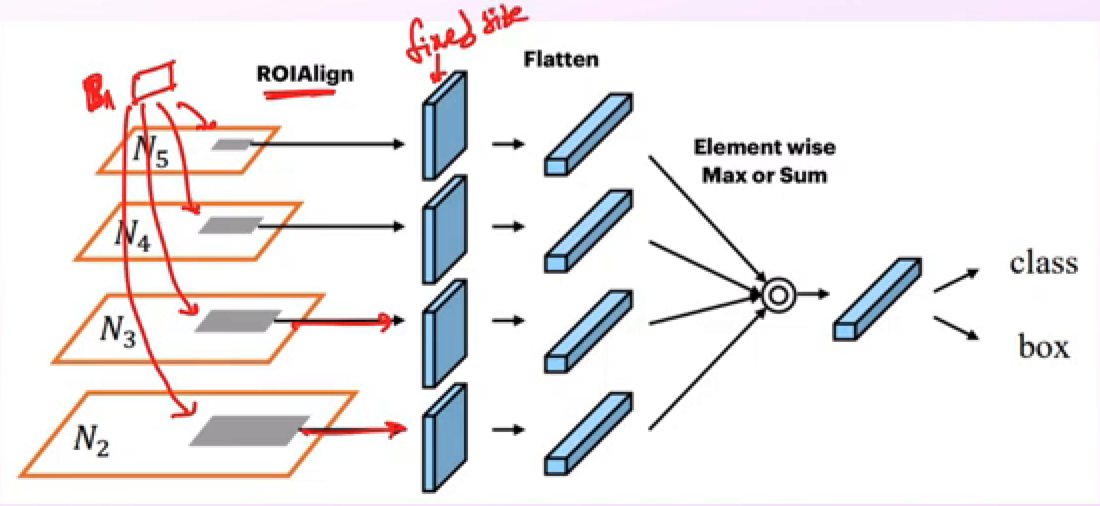
Path Aggregation Network



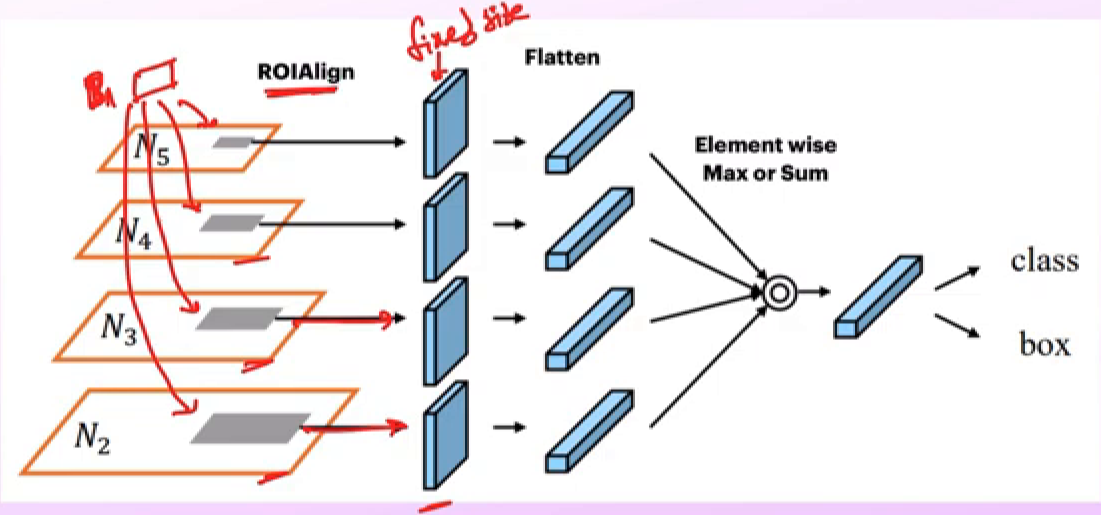
Path Aggregation Network



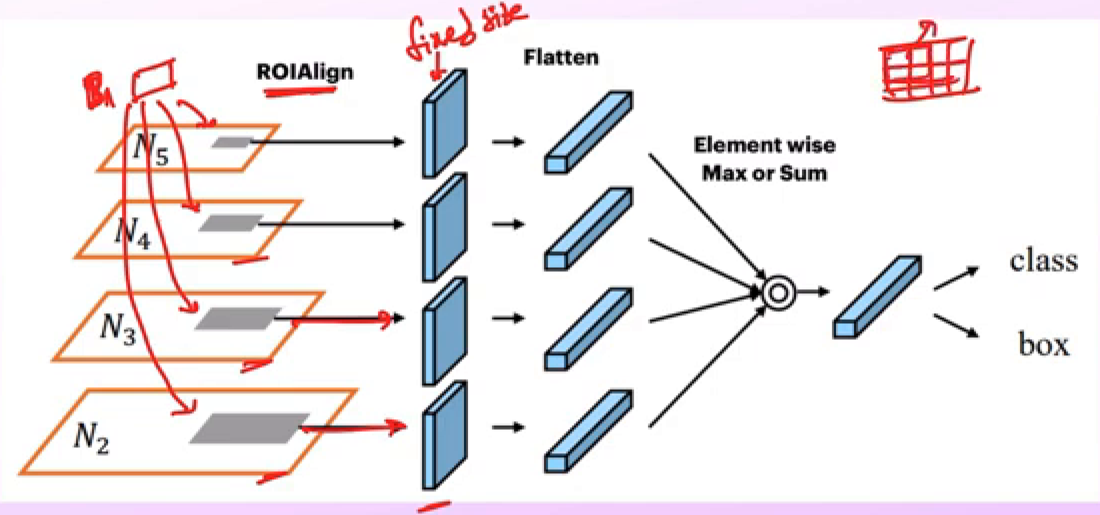
Path Aggregation Network



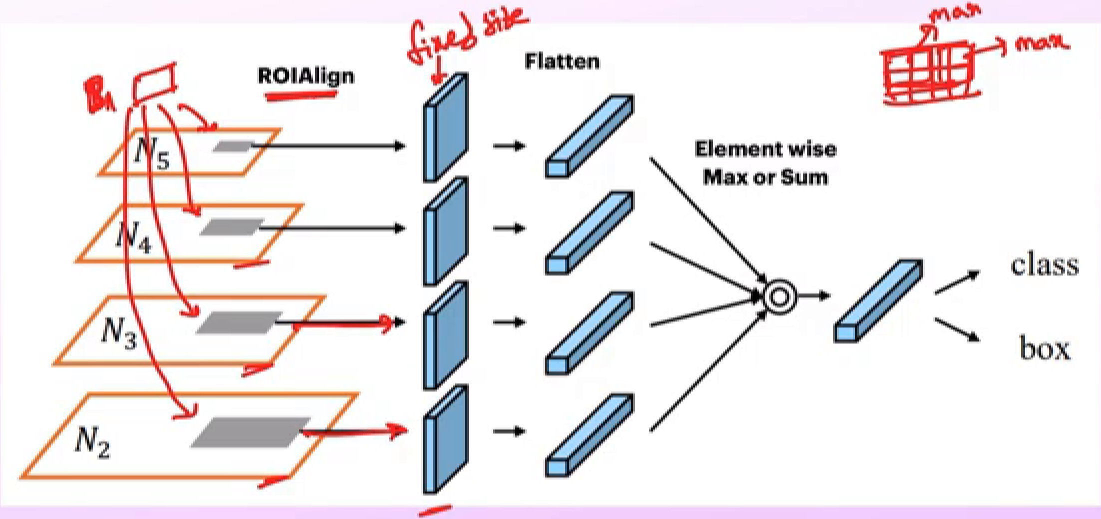
Path Aggregation Network



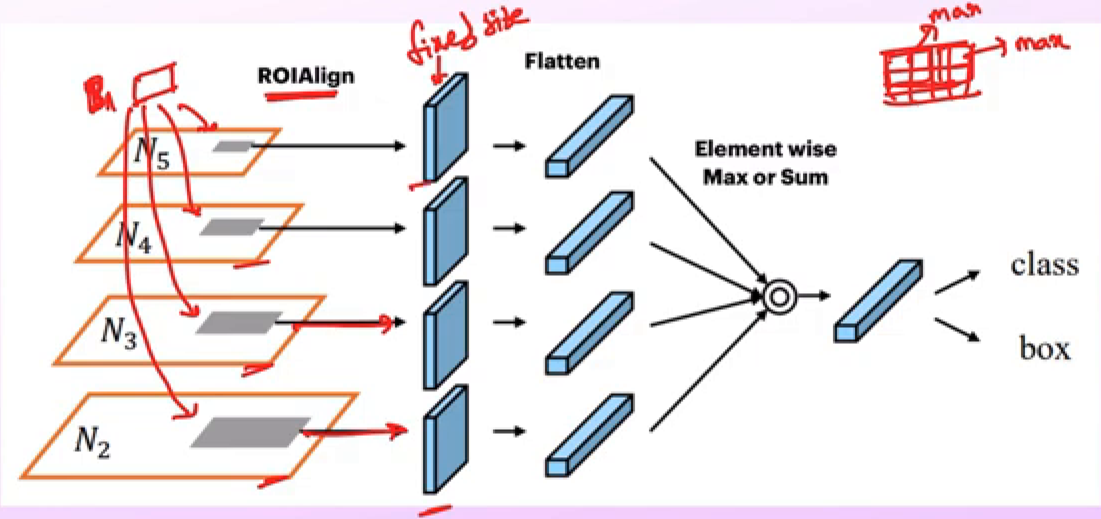
Path Aggregation Network



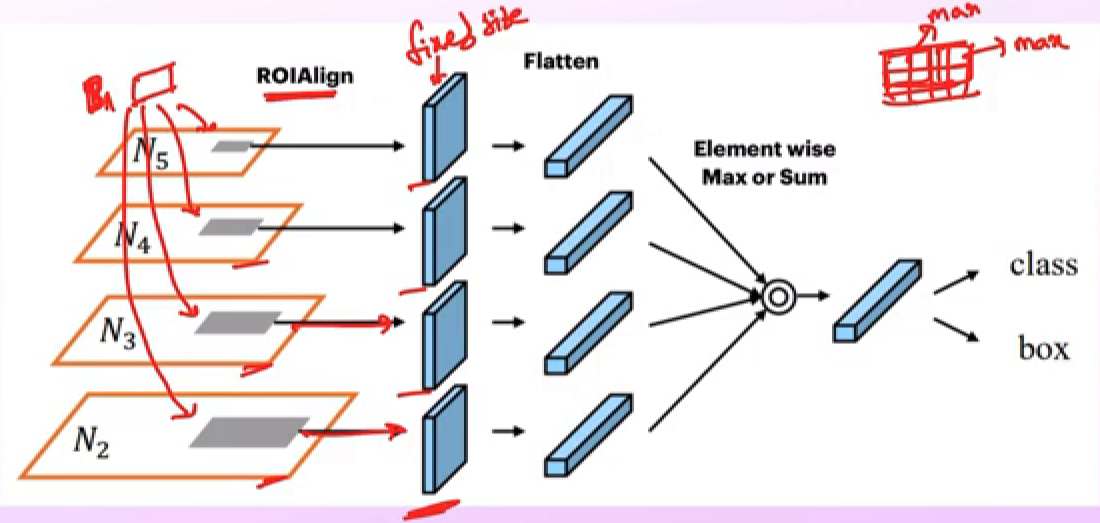
Path Aggregation Network



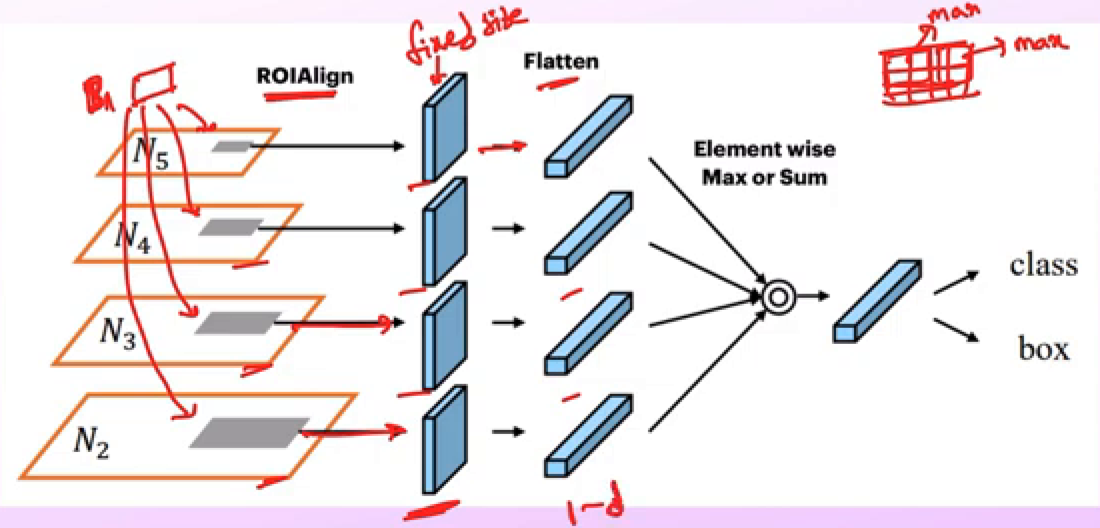
Path Aggregation Network



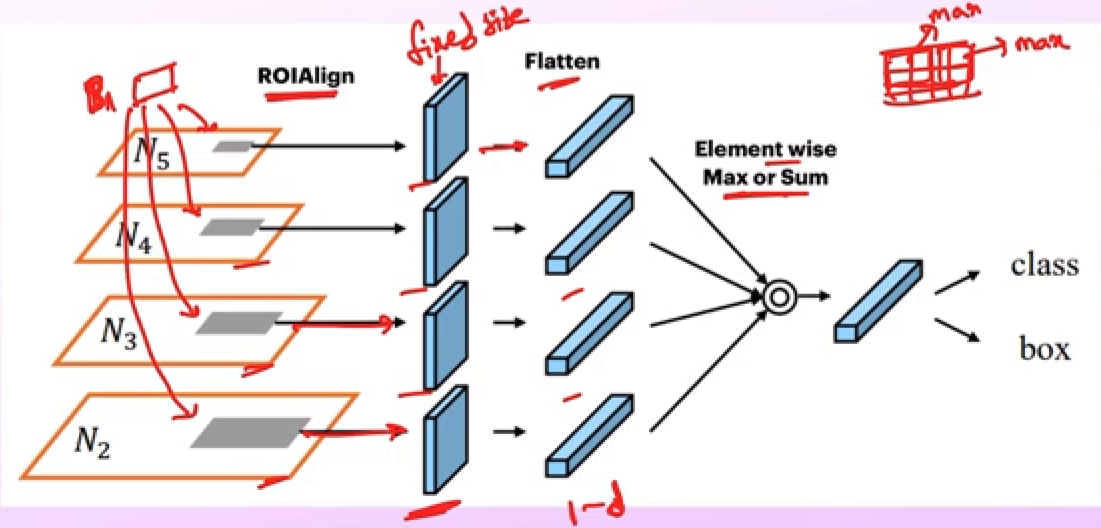
Path Aggregation Network



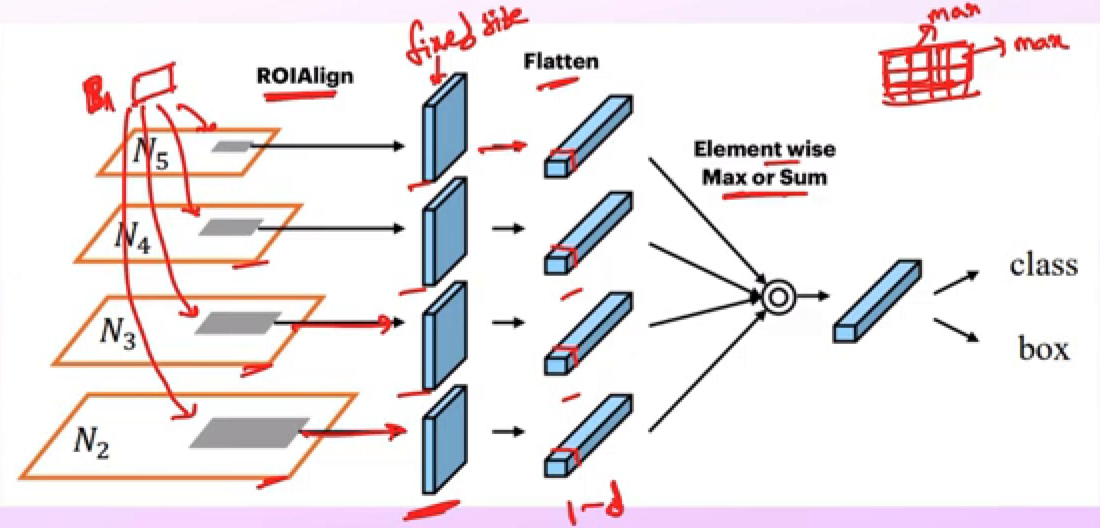
Path Aggregation Network



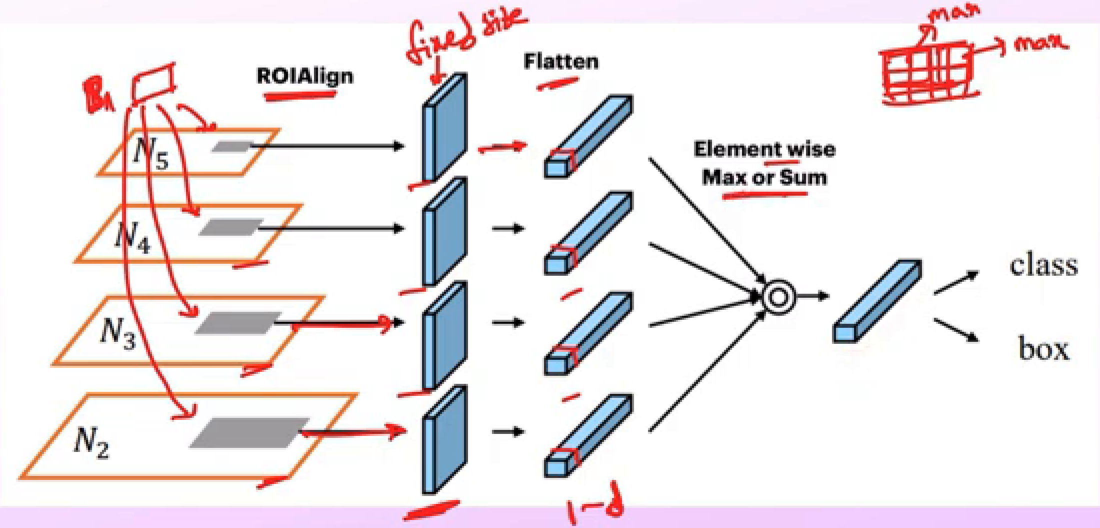
Path Aggregation Network



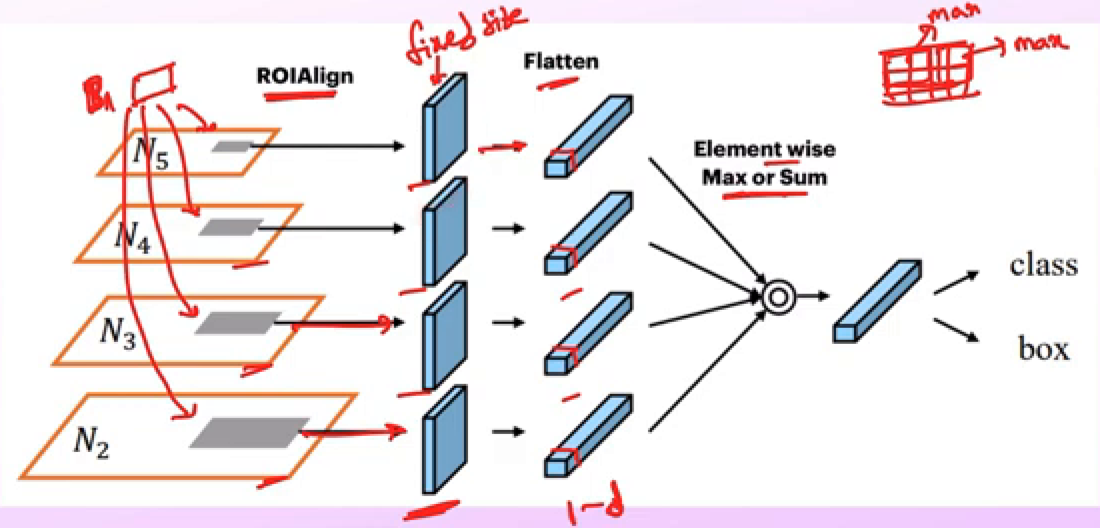
Path Aggregation Network



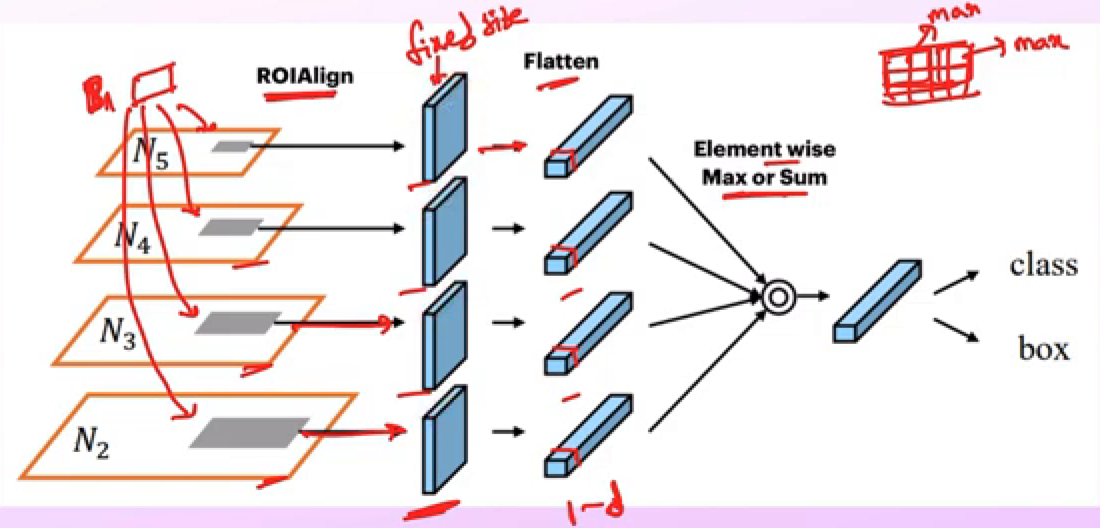
Path Aggregation Network



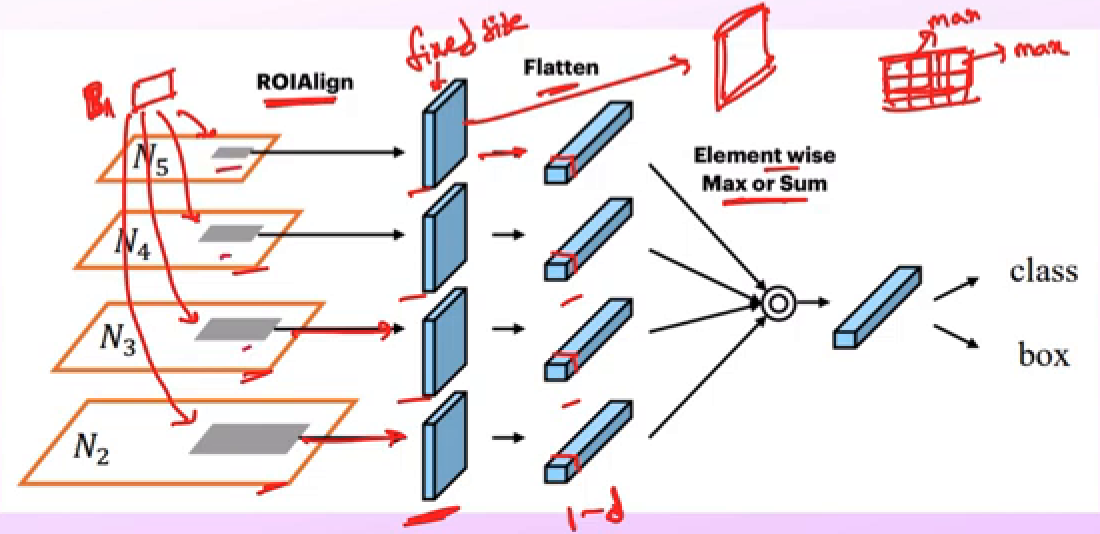
Path Aggregation Network



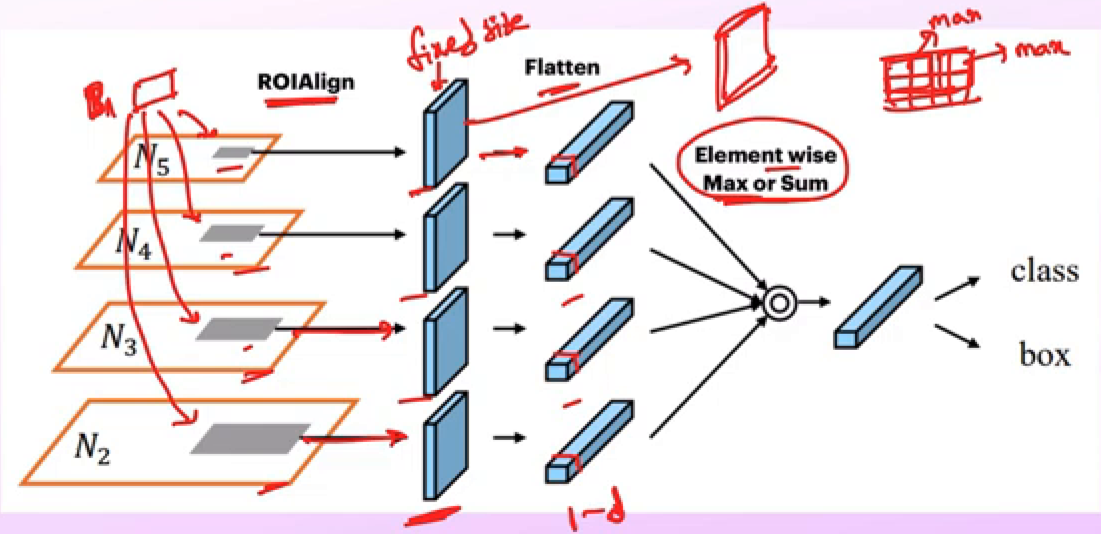
Path Aggregation Network



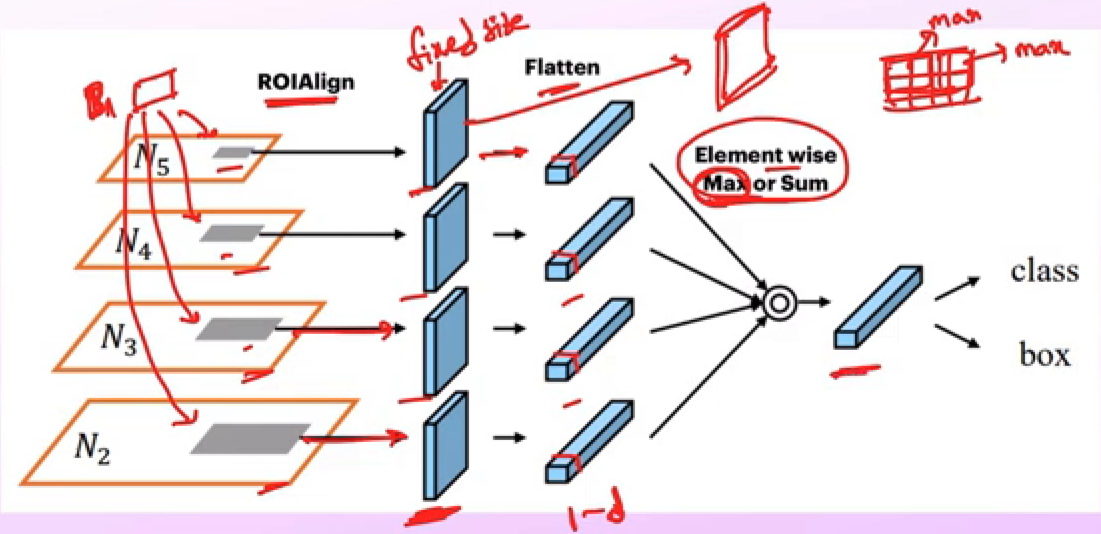
Path Aggregation Network



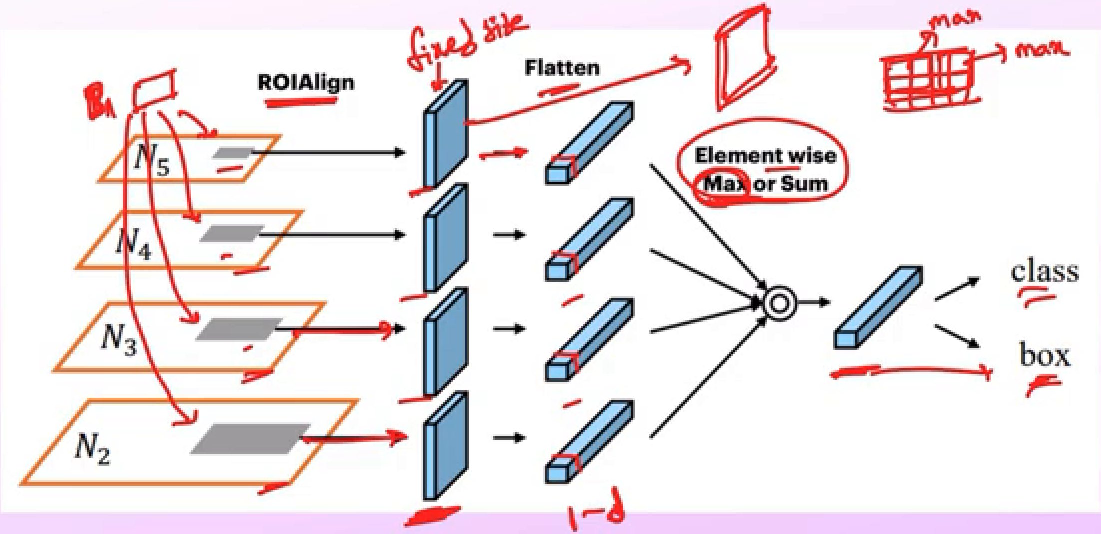
Path Aggregation Network



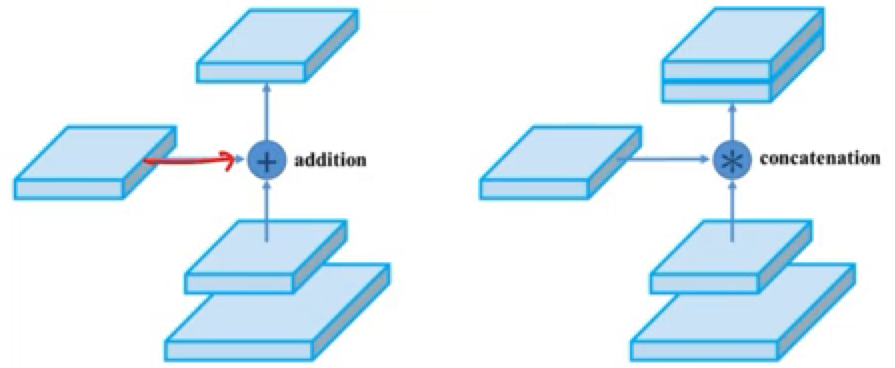
Path Aggregation Network



Path Aggregation Network



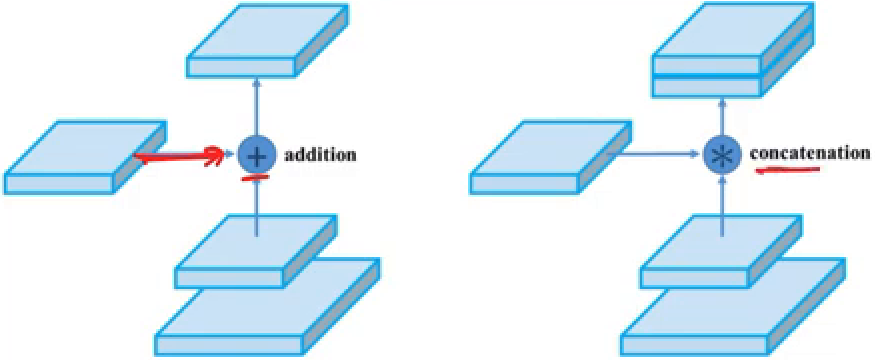
Modified PAN



(a) PAN [49]

(a) Our modified PAN

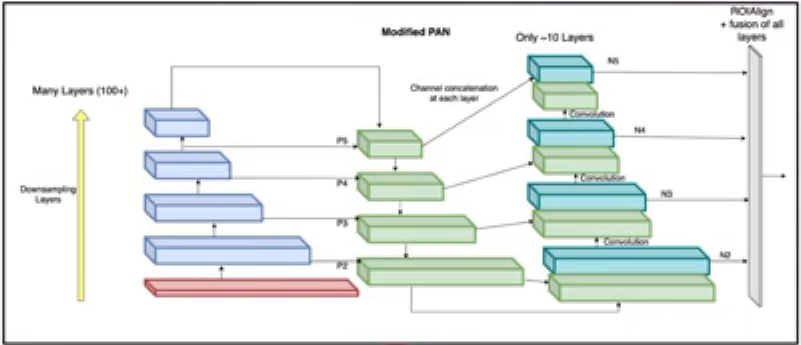
Modified PAN

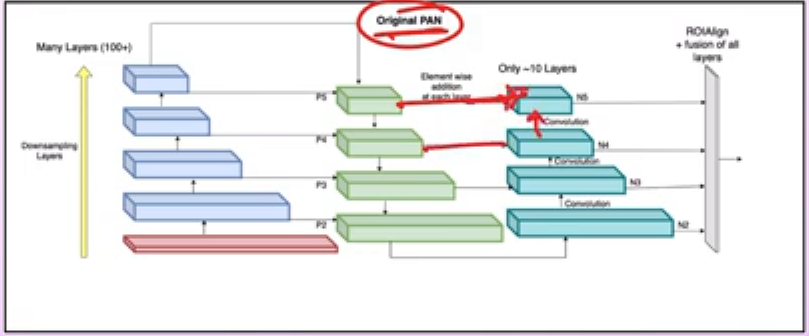


(a) PAN [49]

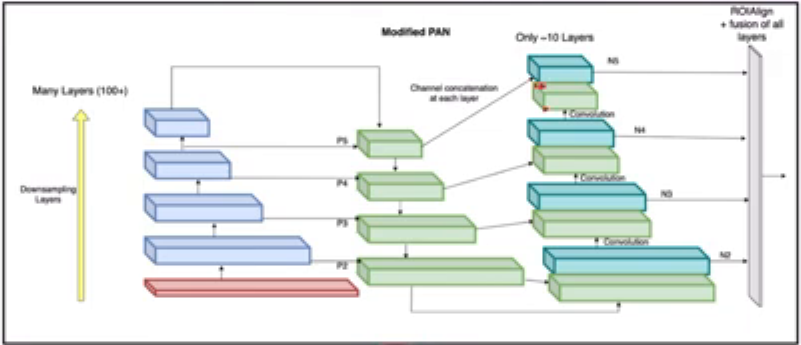
(a) Our modified PAN

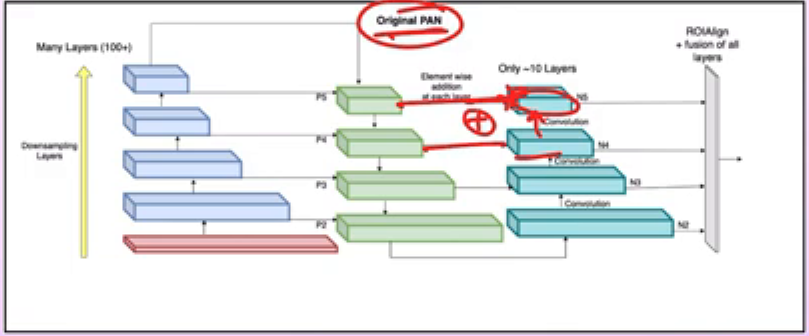
Modified PAN



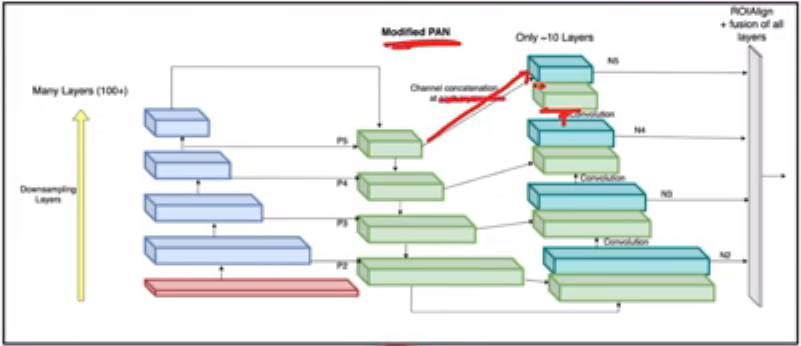


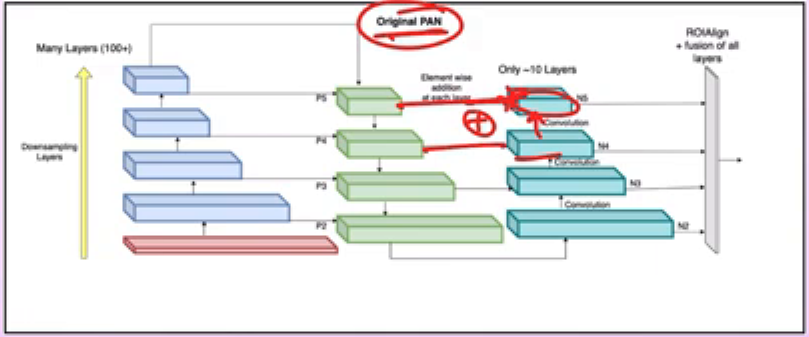
Modified PAN



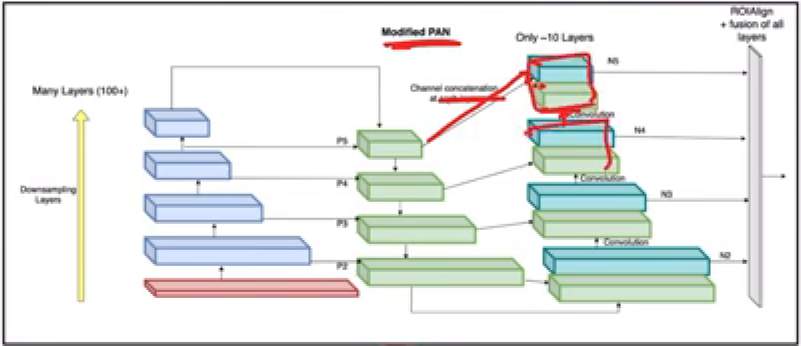


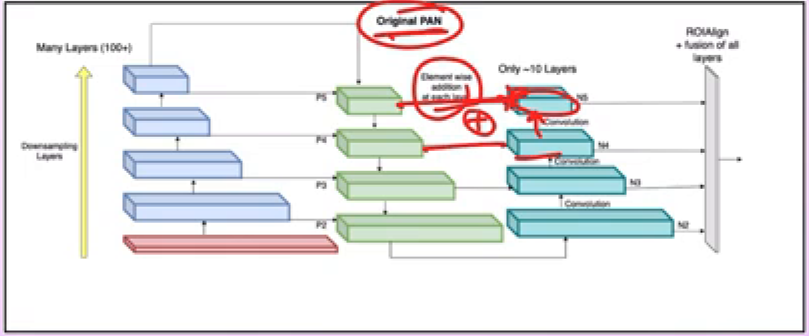
Modified PAN



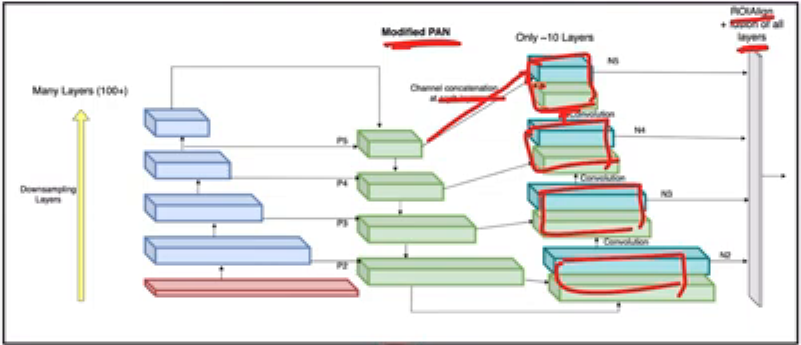


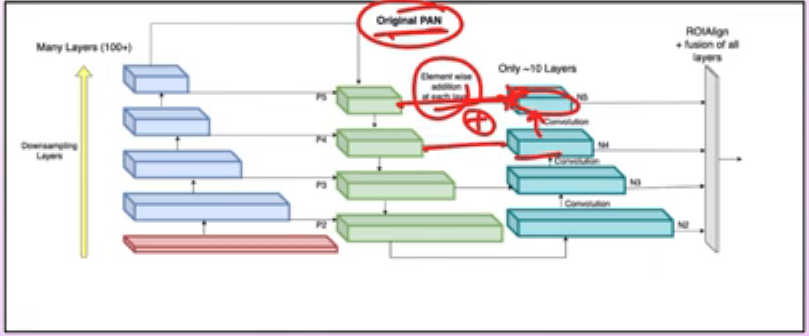
Modified PAN



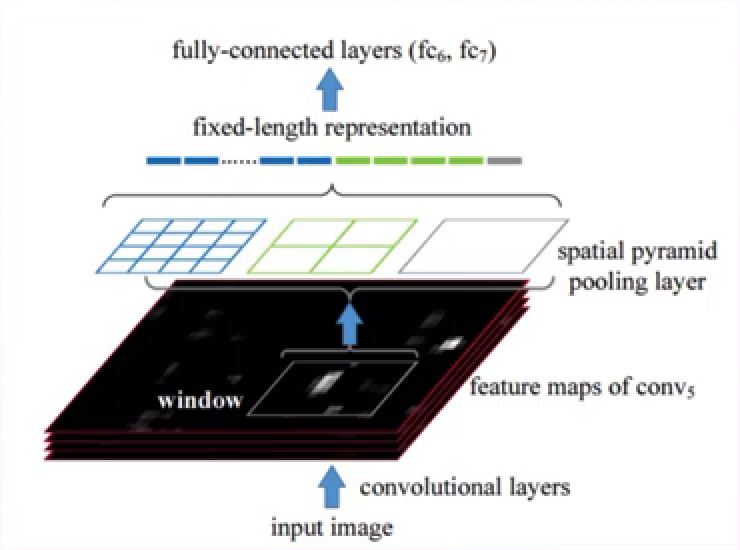


Modified PAN

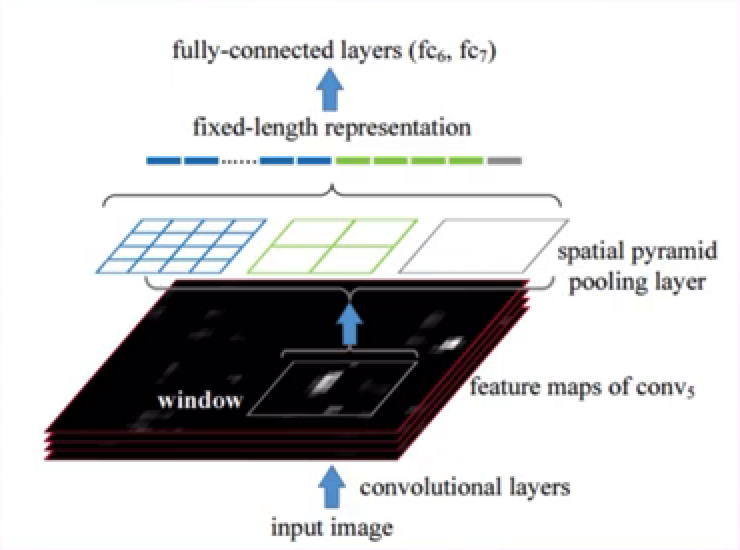




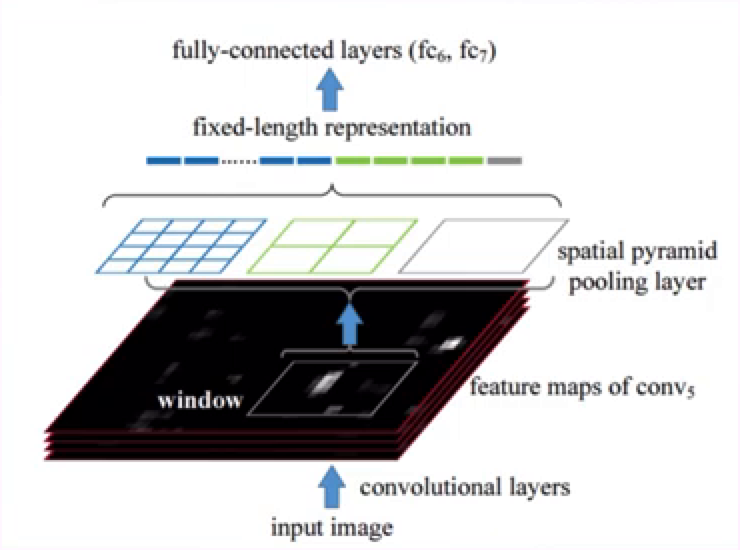
Spatial Pyramid Pooling (SPP)



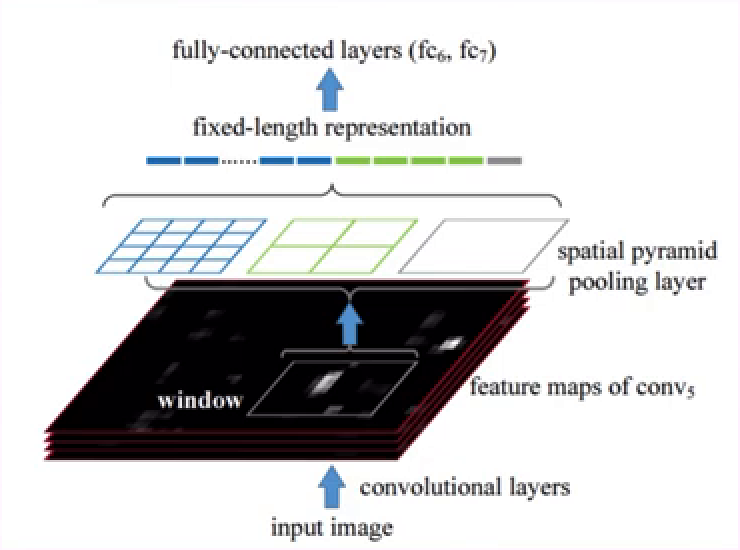
Spatial Pyramid Pooling (SPP)



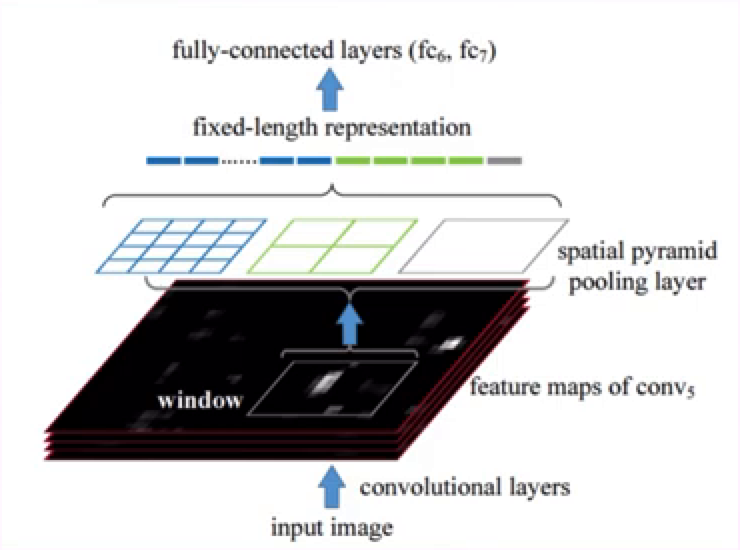
Spatial Pyramid Pooling (SPP)



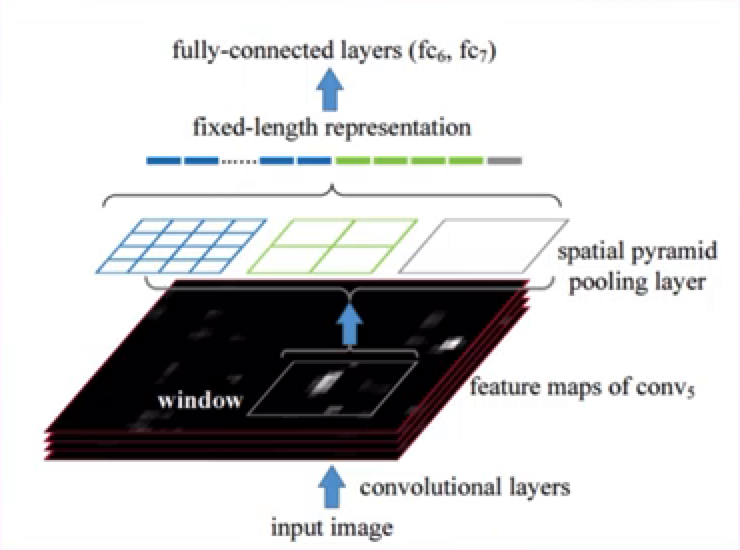
Spatial Pyramid Pooling (SPP)



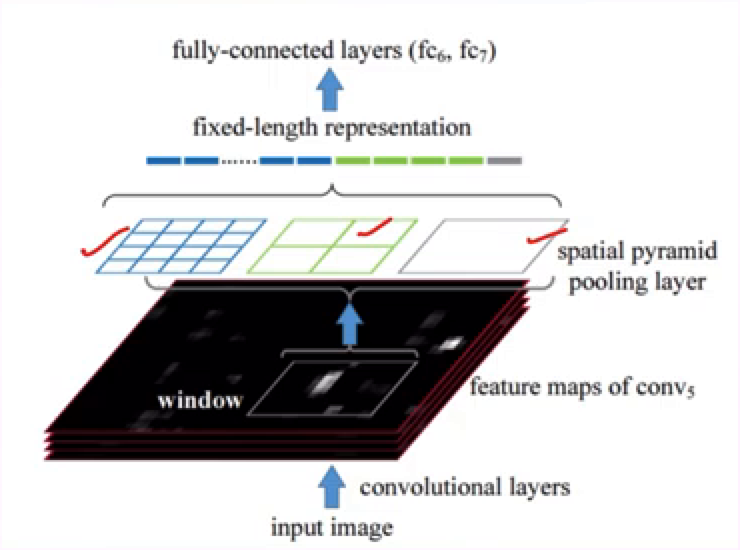
Spatial Pyramid Pooling (SPP)



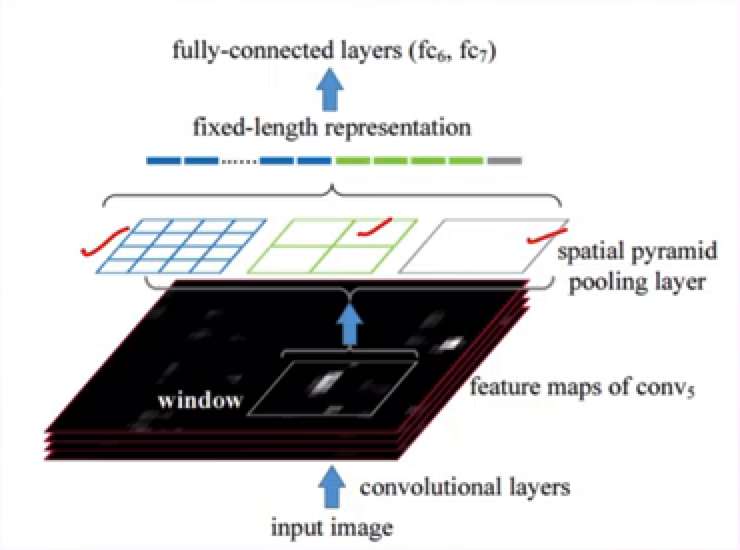
Spatial Pyramid Pooling (SPP)



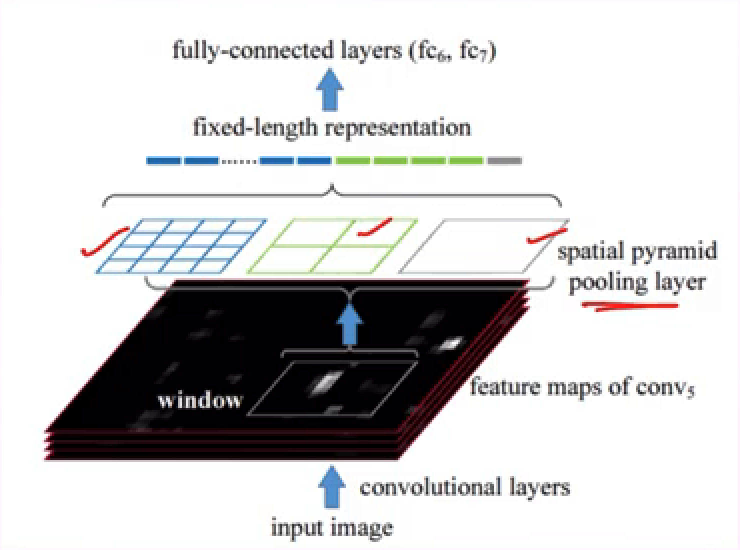
Spatial Pyramid Pooling (SPP)



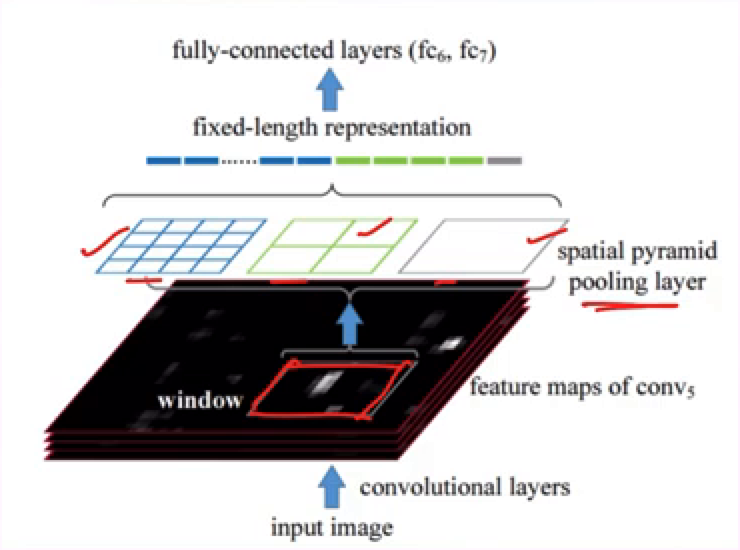
Spatial Pyramid Pooling (SPP)



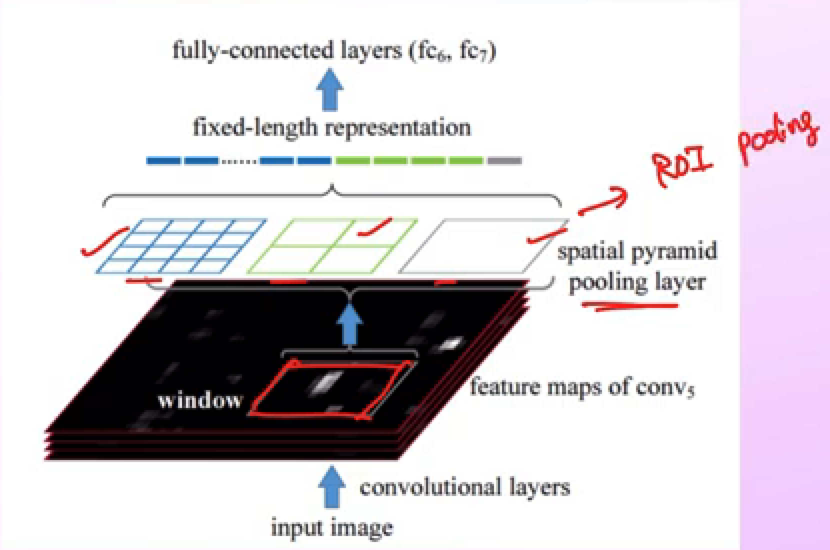
Spatial Pyramid Pooling (SPP)



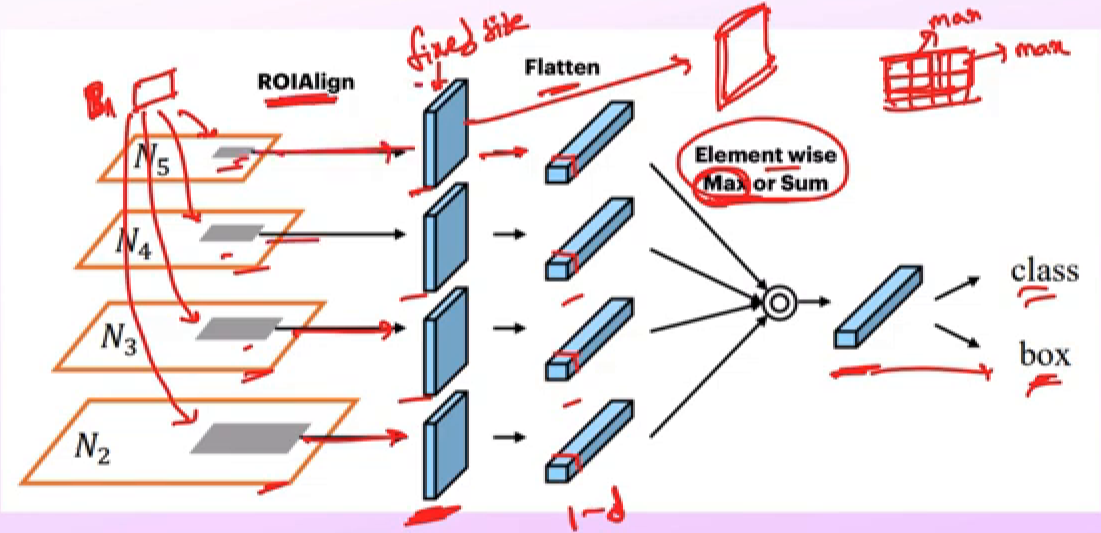
Spatial Pyramid Pooling (SPP)



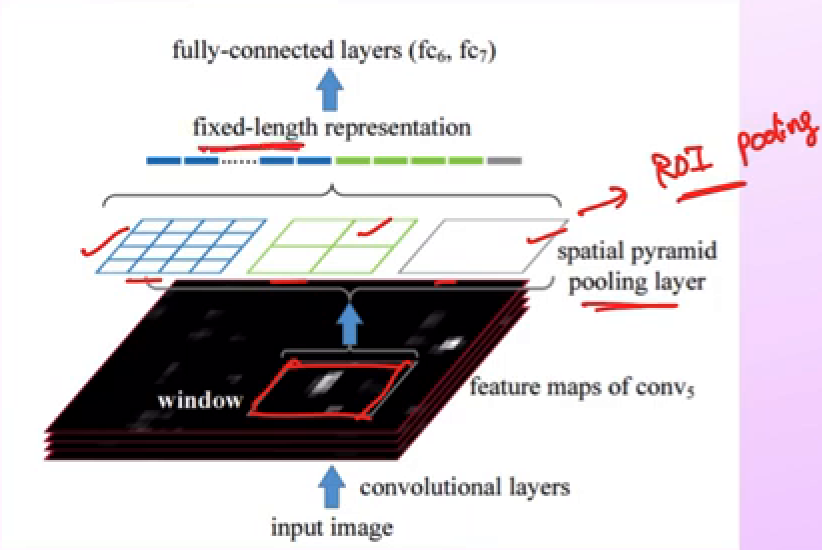
Spatial Pyramid Pooling (SPP)



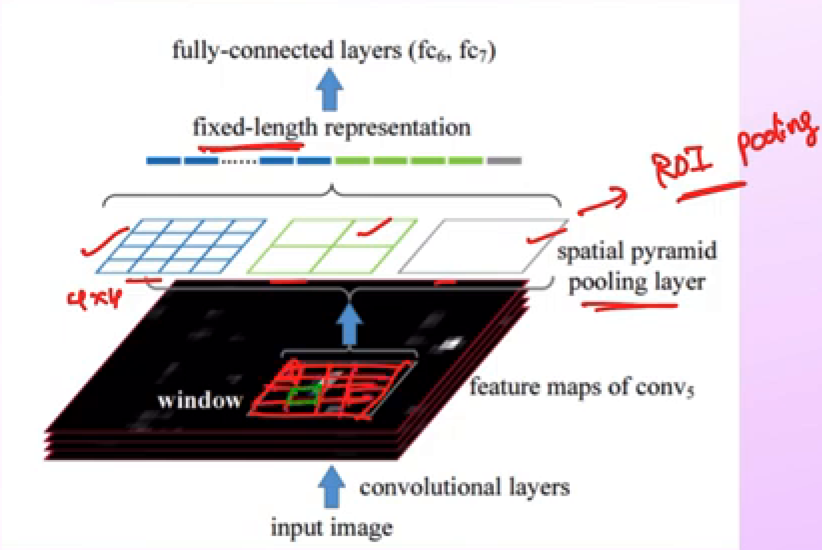
Path Aggregation Network



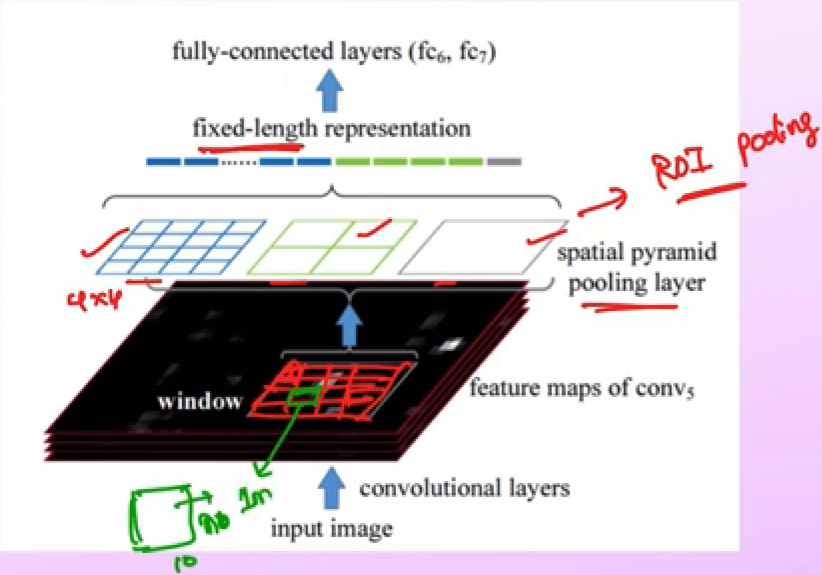
Spatial Pyramid Pooling (SPP)



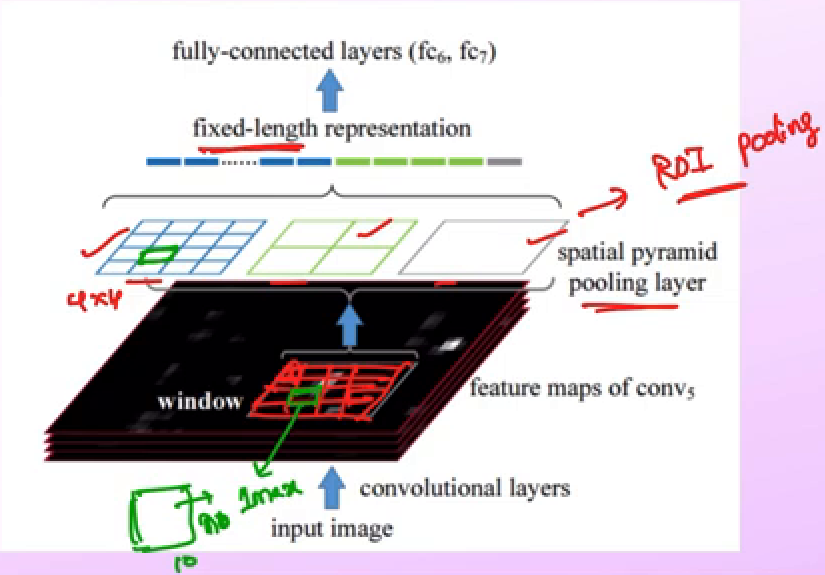
Spatial Pyramid Pooling (SPP)



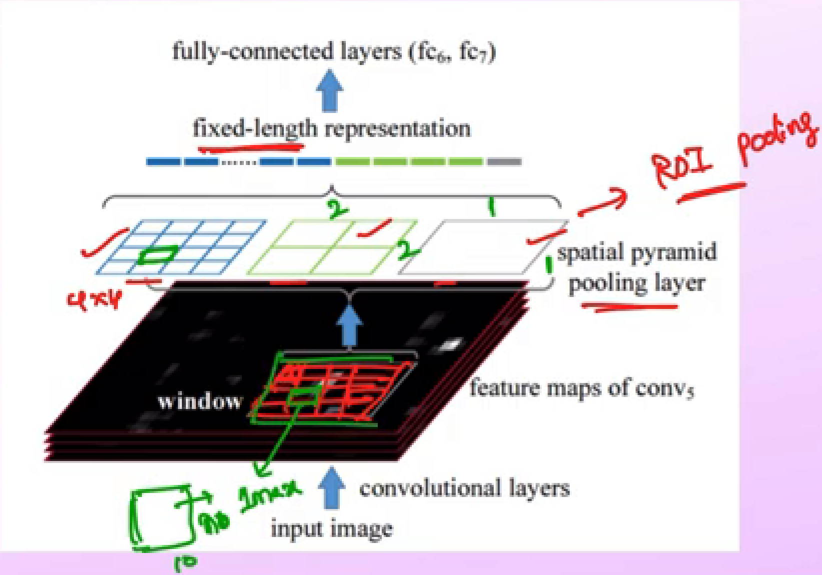
Spatial Pyramid Pooling (SPP)



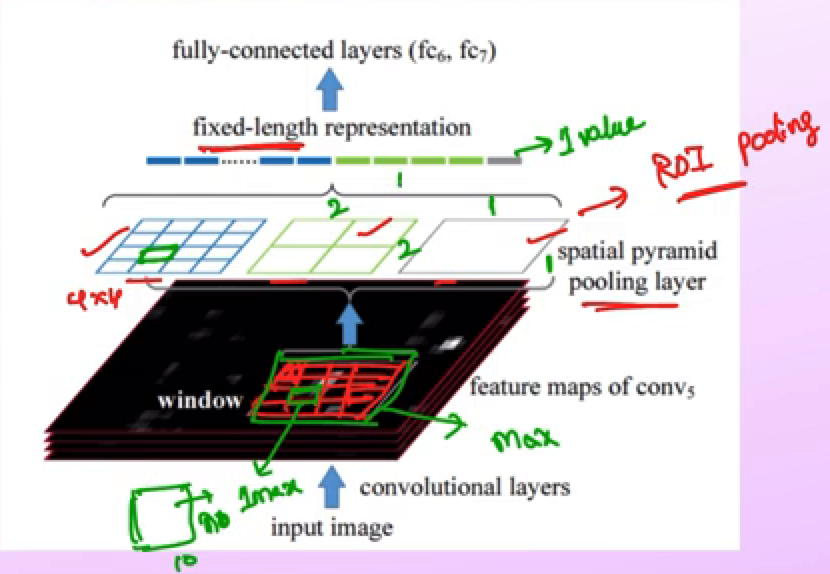
Spatial Pyramid Pooling (SPP)



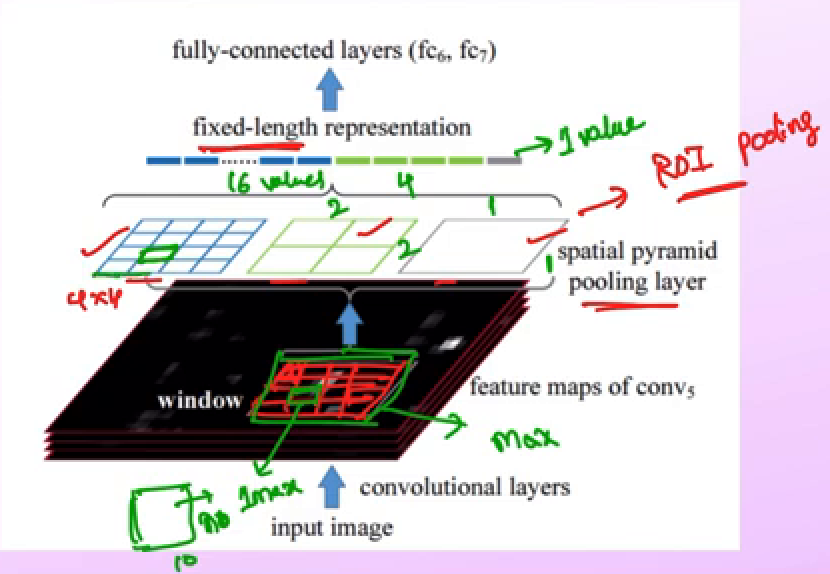
Spatial Pyramid Pooling (SPP)



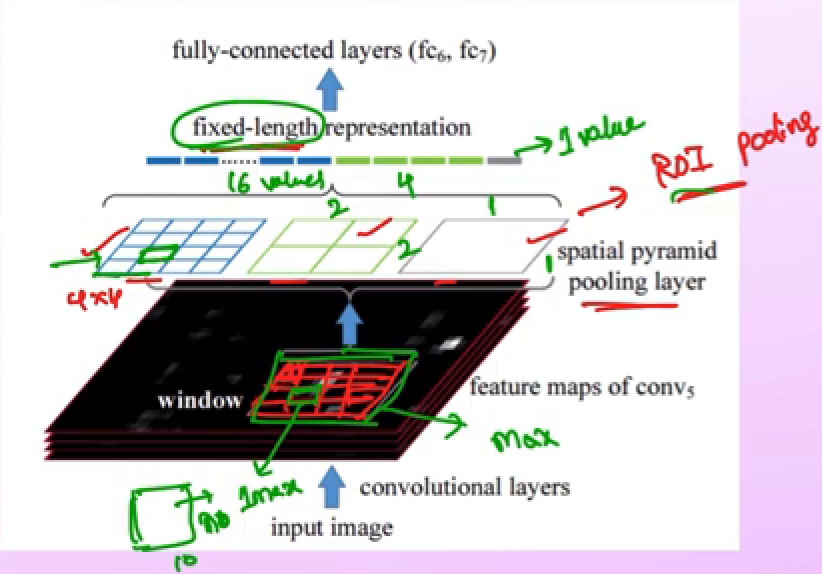
Spatial Pyramid Pooling (SPP)



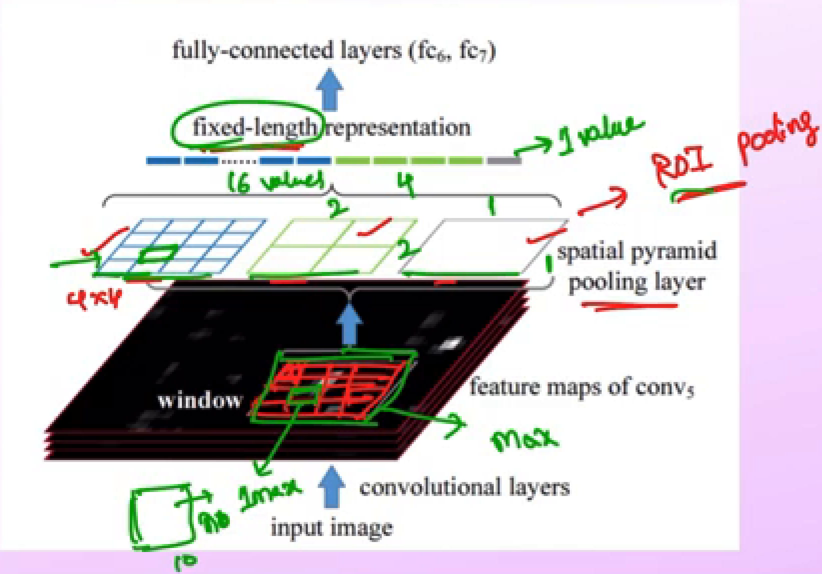
Spatial Pyramid Pooling (SPP)



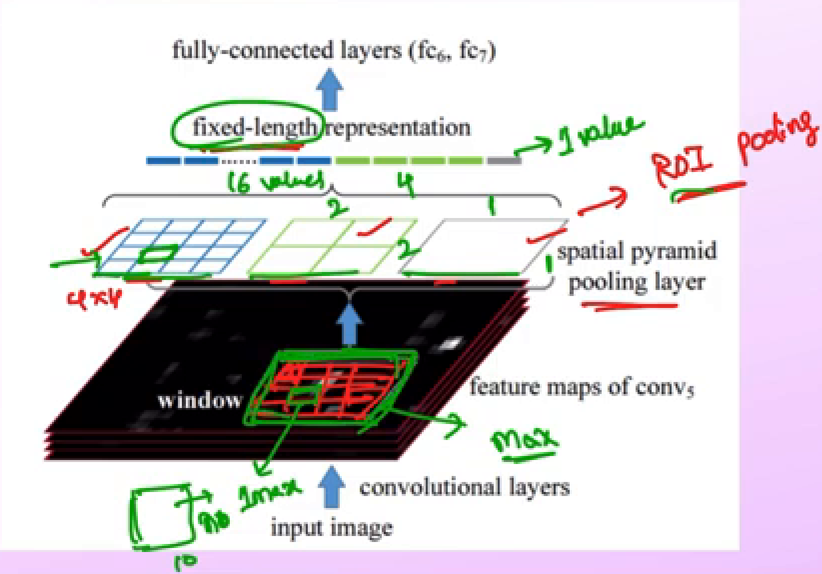
Spatial Pyramid Pooling (SPP)



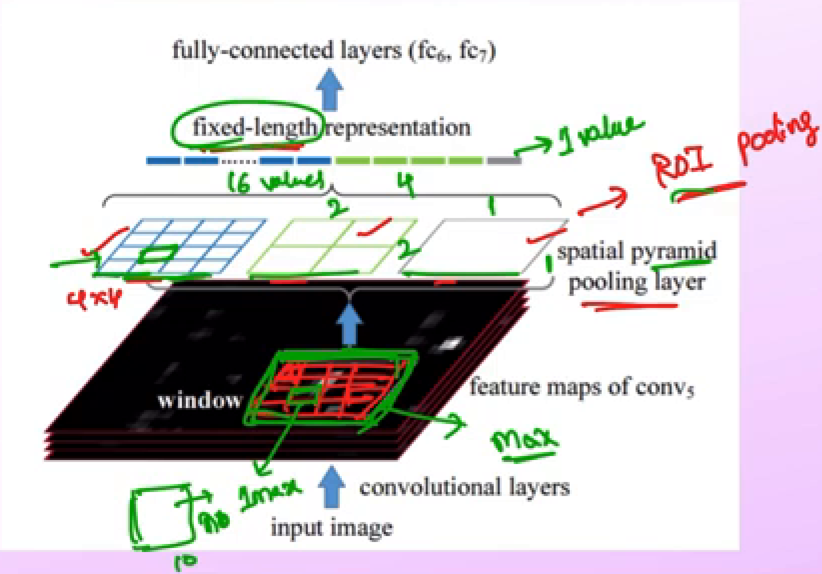
Spatial Pyramid Pooling (SPP)



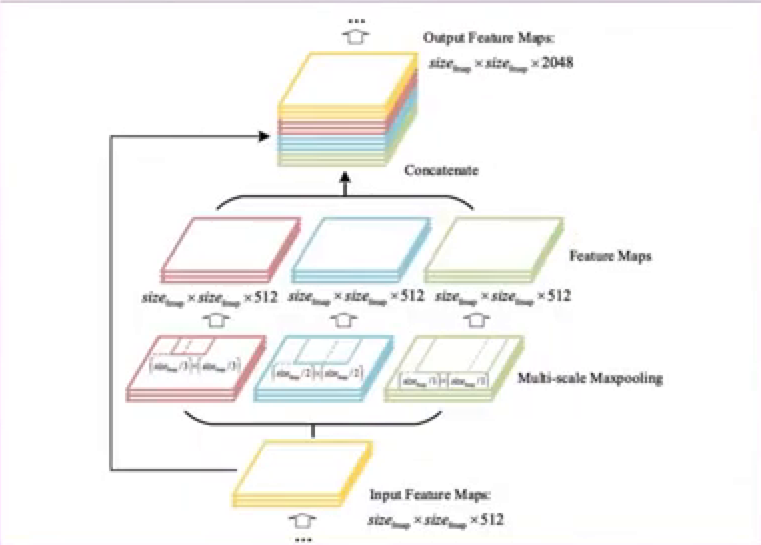
Spatial Pyramid Pooling (SPP)



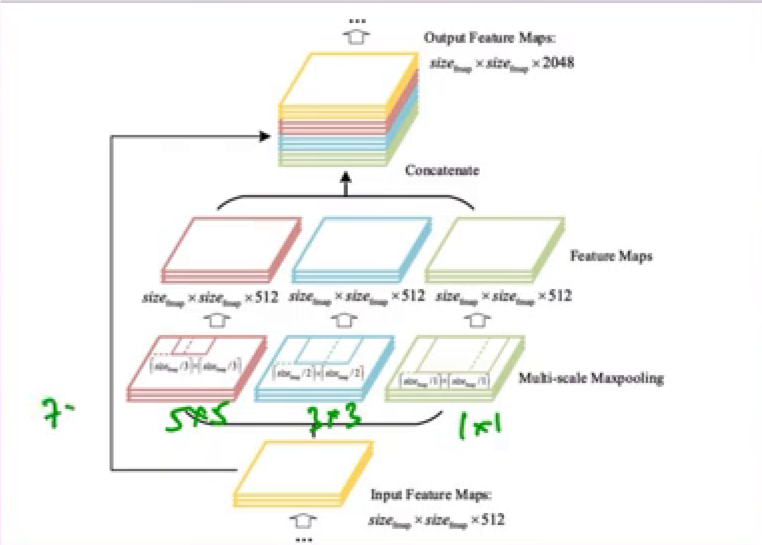
Spatial Pyramid Pooling (SPP)



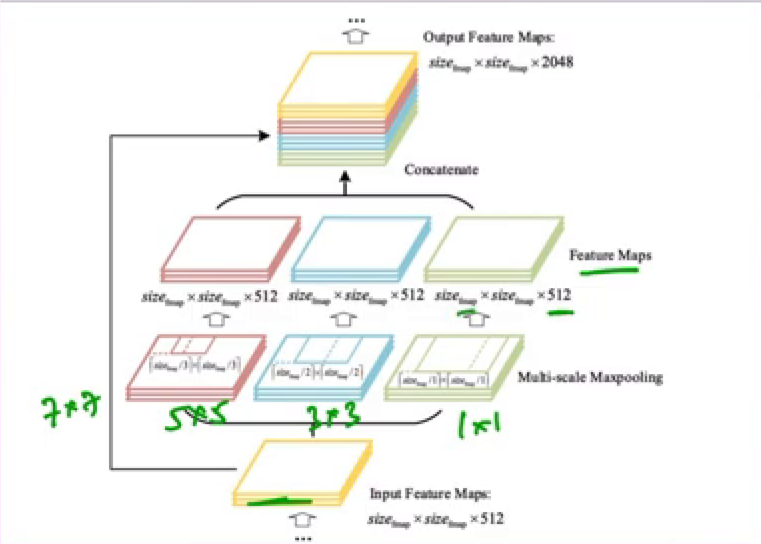
YOLO with SPP



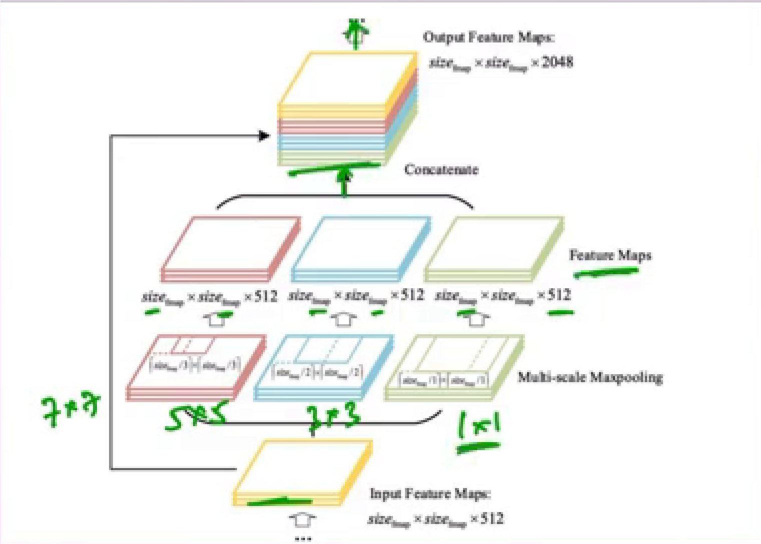
YOLO with SPP



YOLO with SPP

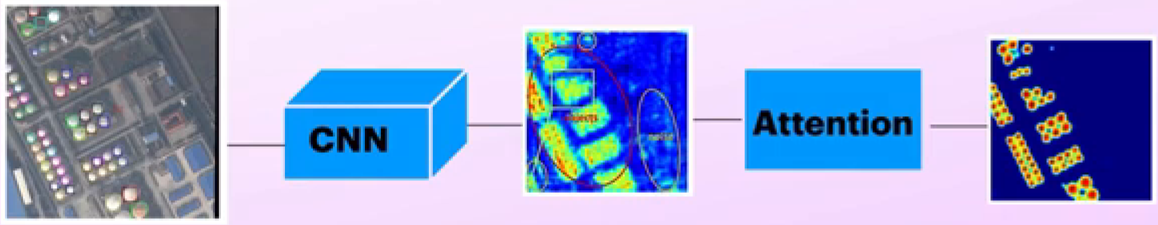


YOLO with SPP



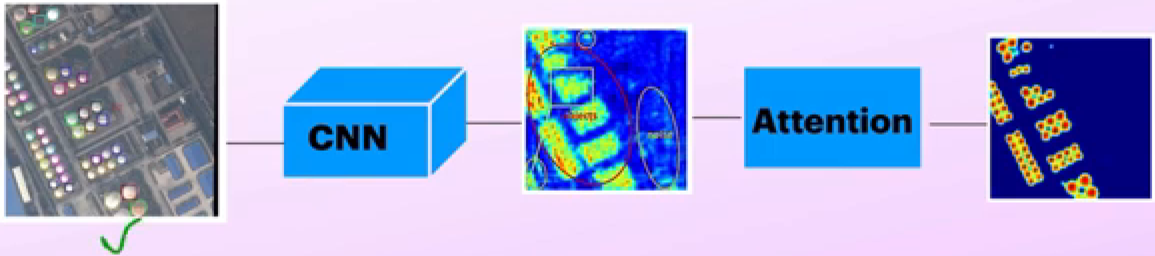
Attention Module

SAM



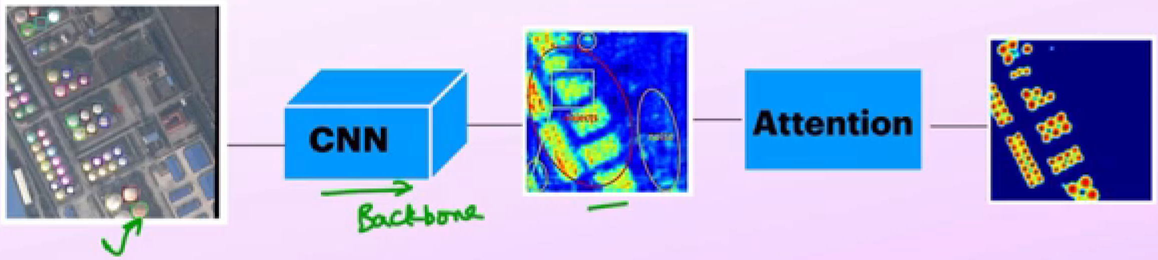
Attention Module

SAM



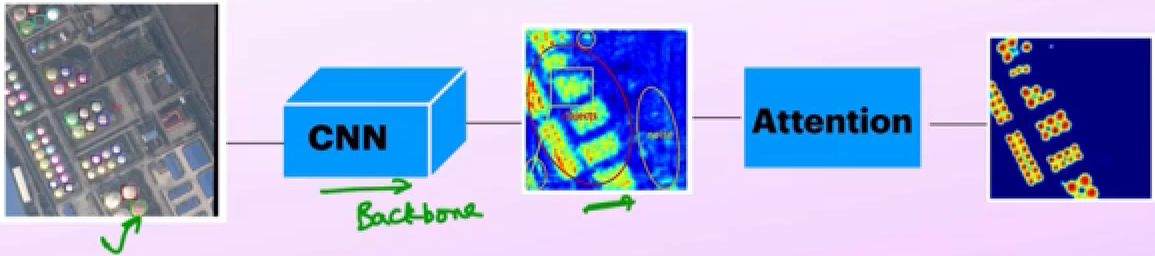
Attention Module

SAM



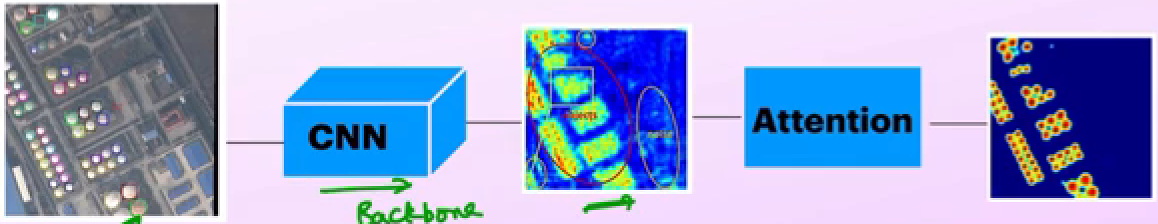
Attention Module

SAM



Attention Module

SAM

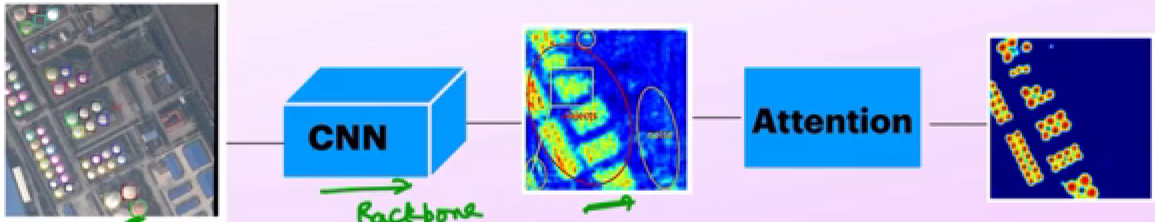


Backbone

DOTA

Attention Module

SAM

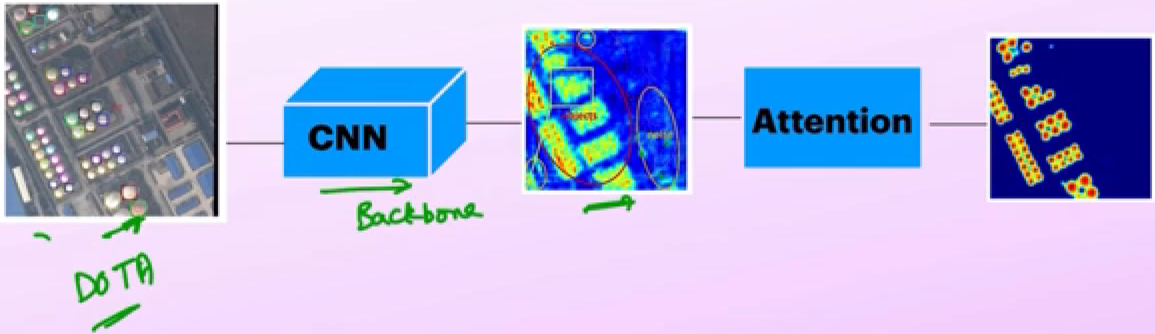


Backbone

DOTA

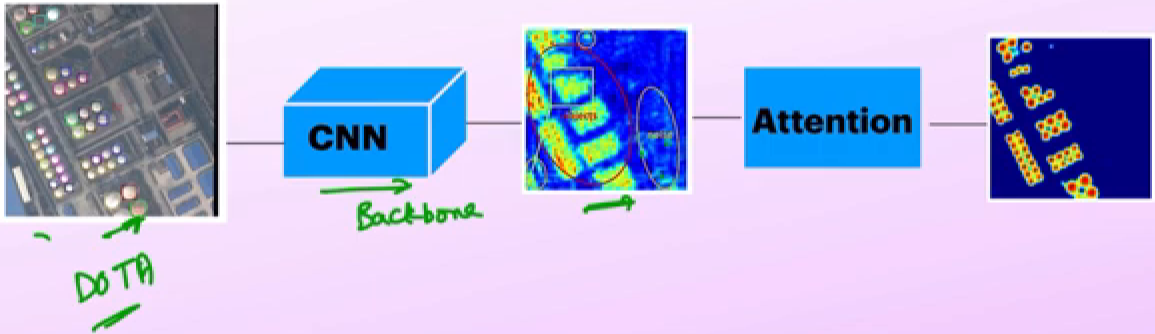
Attention Module

SAM



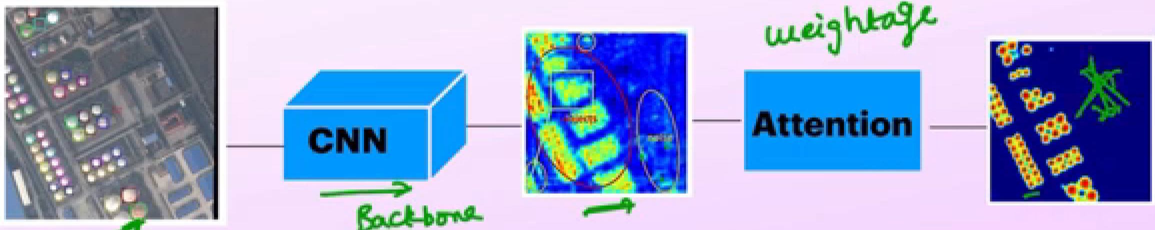
Attention Module

SAM



Attention Module

SAM

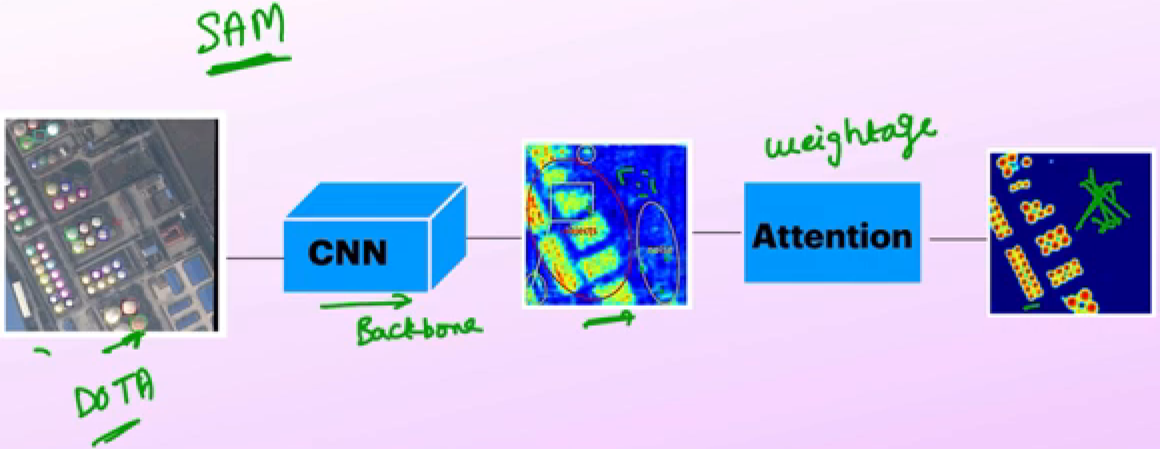


weightage

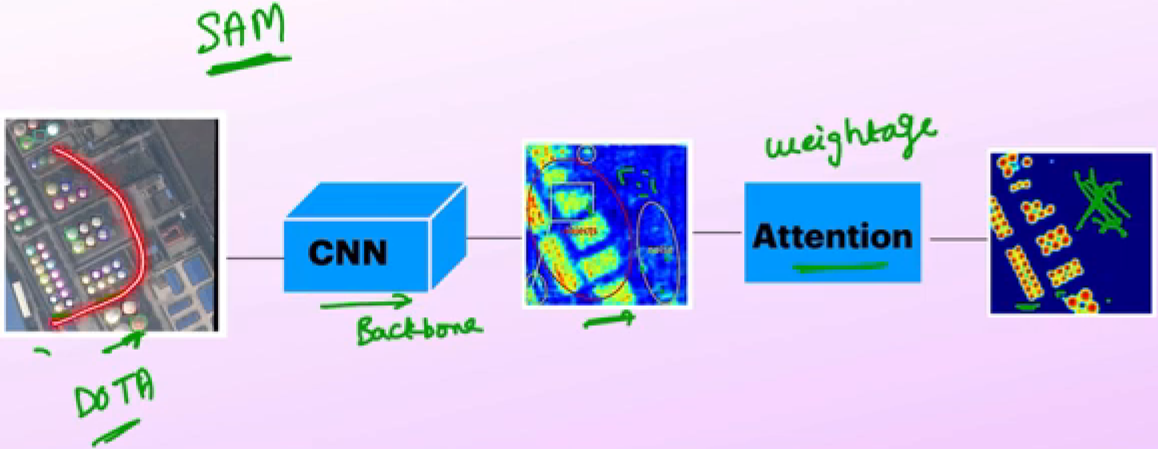
Backbone

DOTA

Attention Module

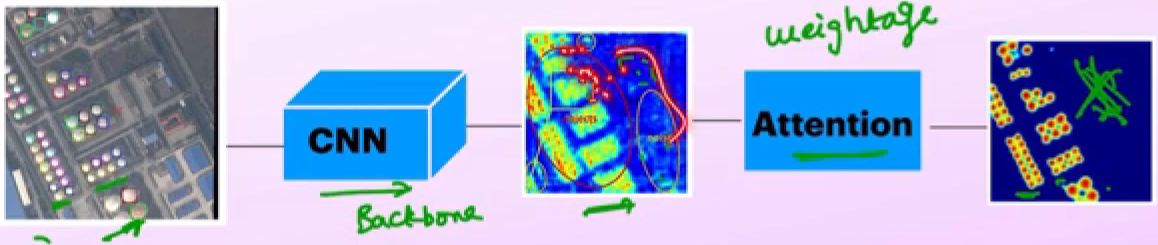


Attention Module



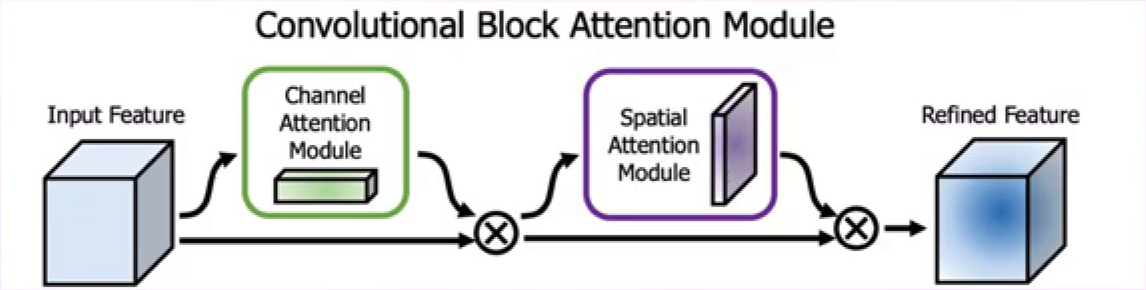
Attention Module

SAM

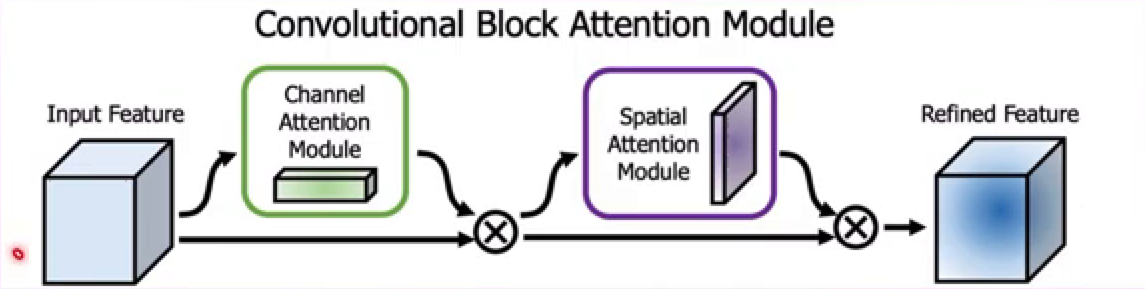


DOTA

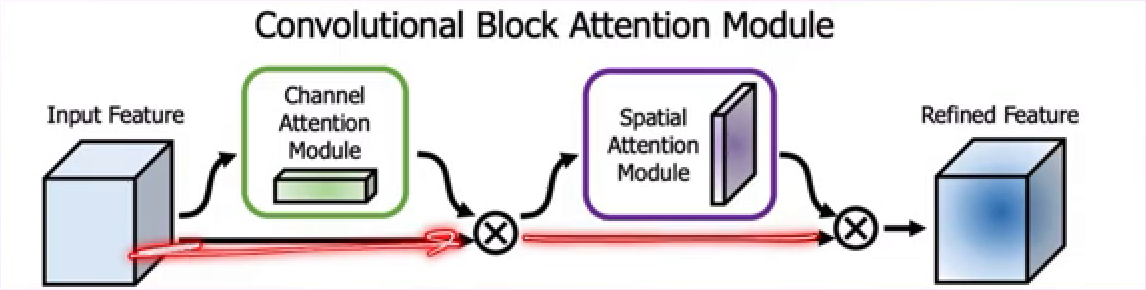
Attention Module



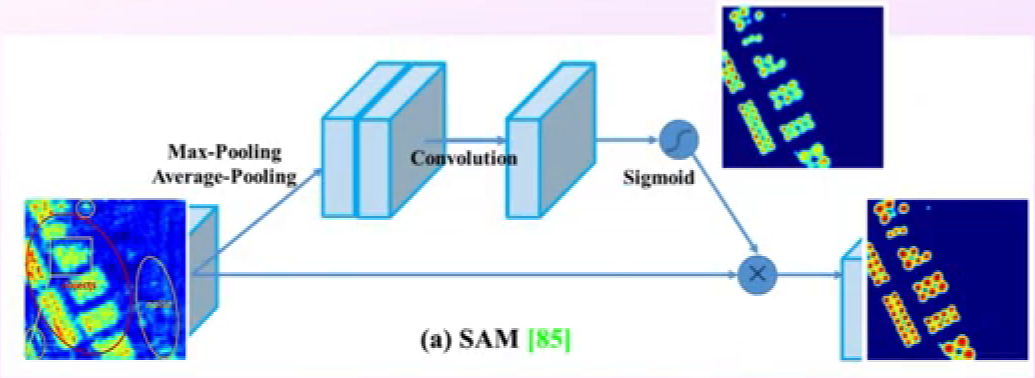
Attention Module



Attention Module

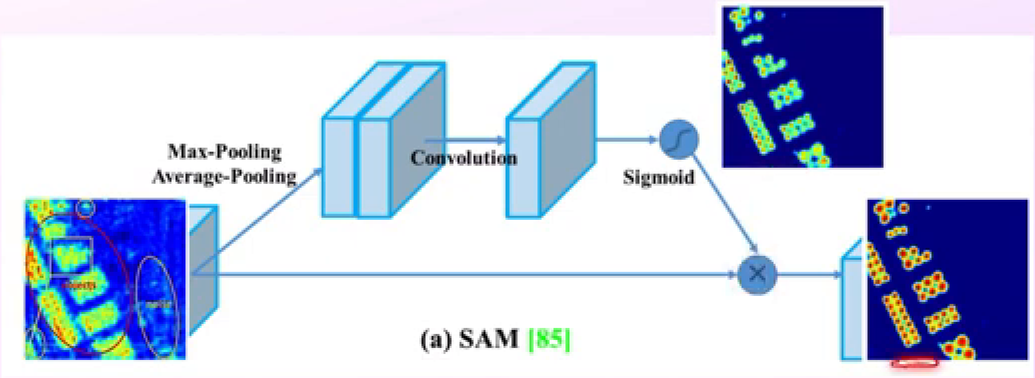


Spatial Attention Module (SAM)



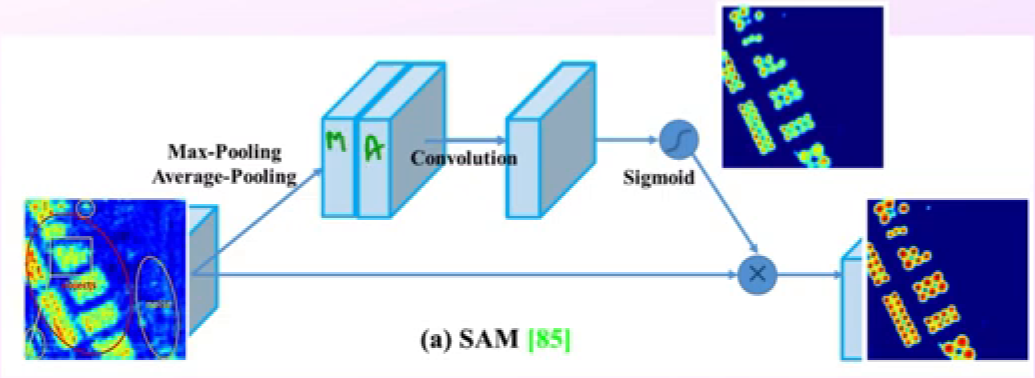
(a) SAM [85]

Spatial Attention Module (SAM)



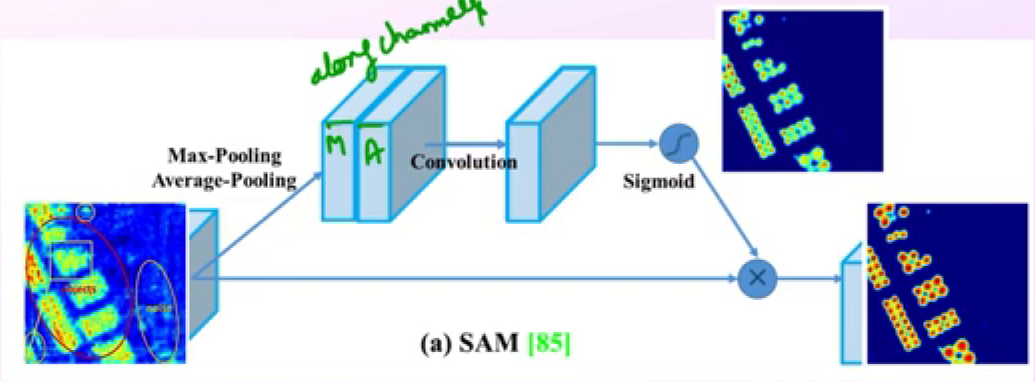
(a) SAM [85]

Spatial Attention Module (SAM)



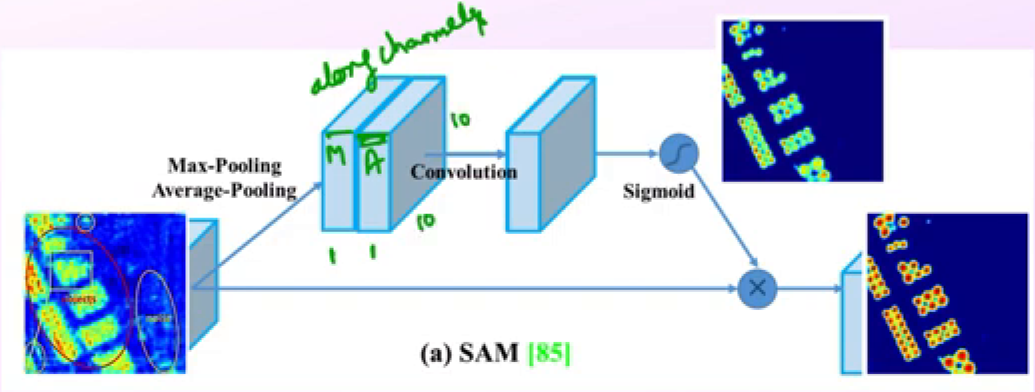
(a) SAM [85]

Spatial Attention Module (SAM)



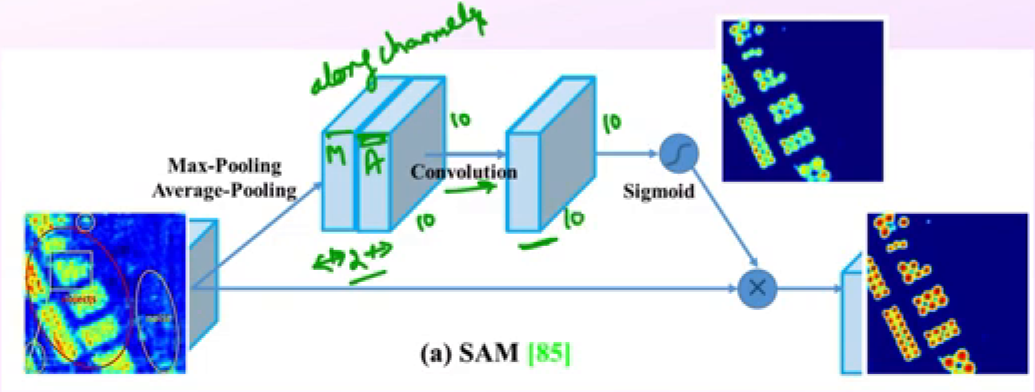
(a) SAM [85]

Spatial Attention Module (SAM)



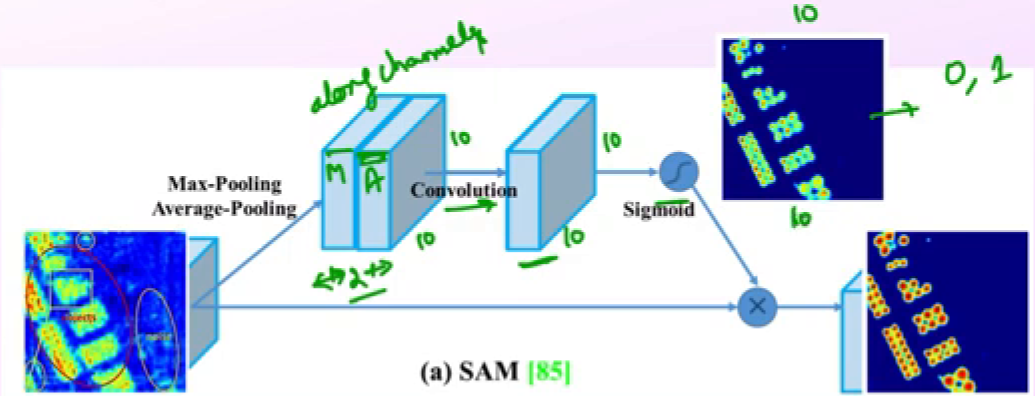
(a) SAM [85]

Spatial Attention Module (SAM)



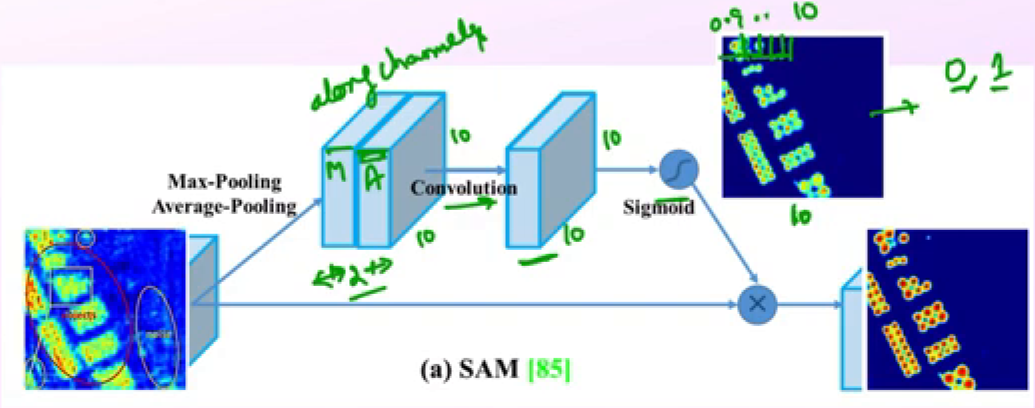
(a) SAM [85]

Spatial Attention Module (SAM)



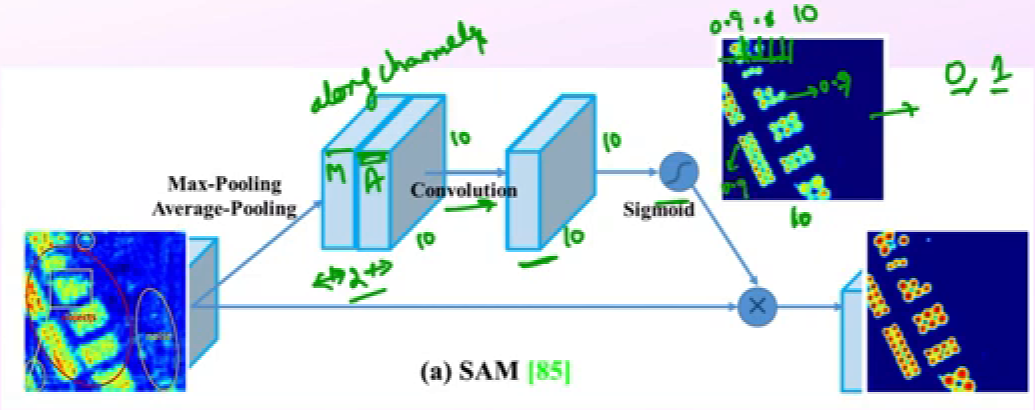
(a) SAM [85]

Spatial Attention Module (SAM)



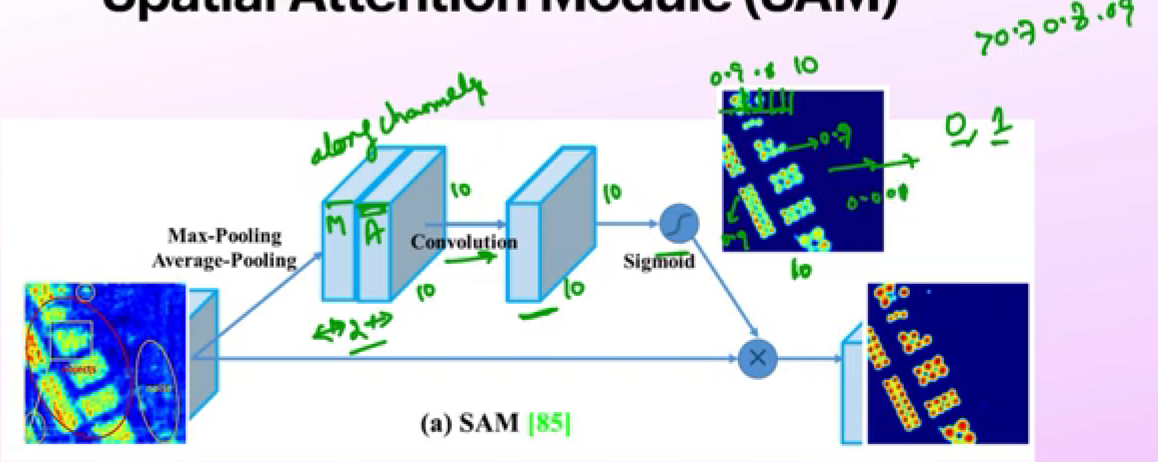
(a) SAM [85]

Spatial Attention Module (SAM)



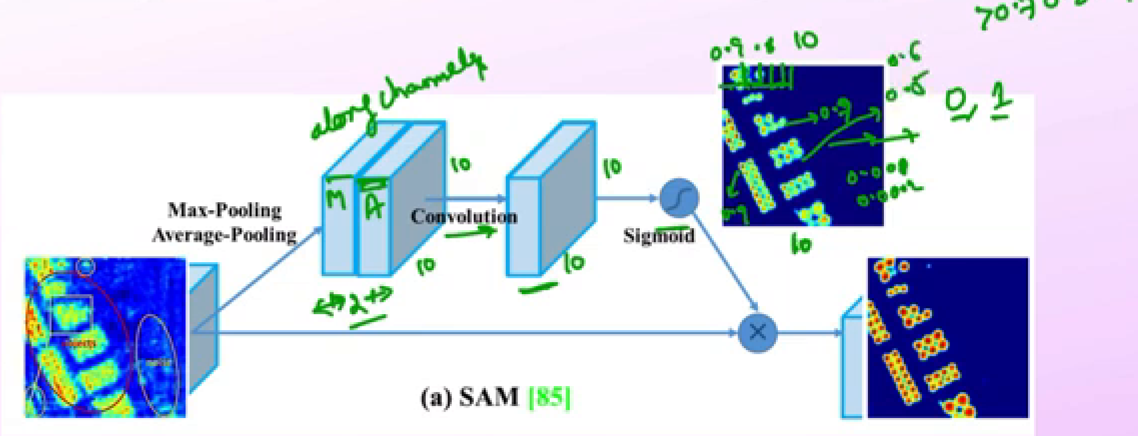
(a) SAM [85]

Spatial Attention Module (SAM)



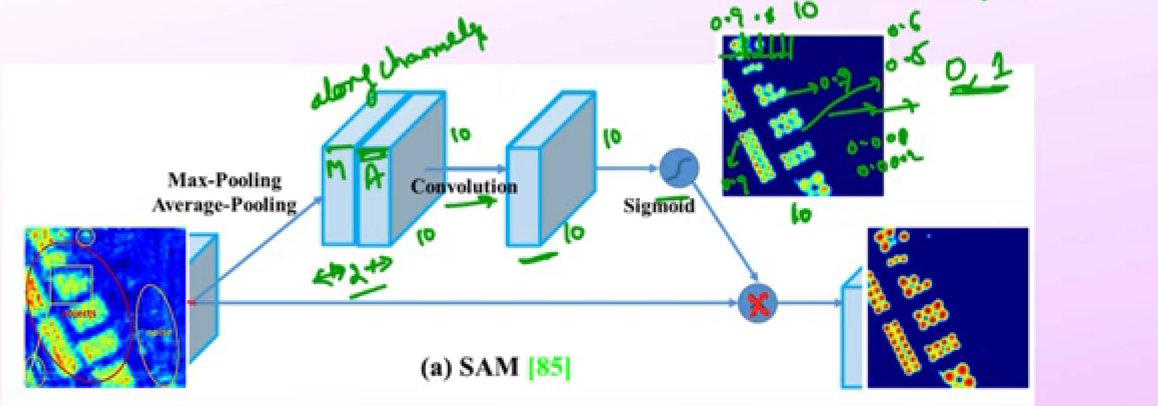
(a) SAM [85]

Spatial Attention Module (SAM)



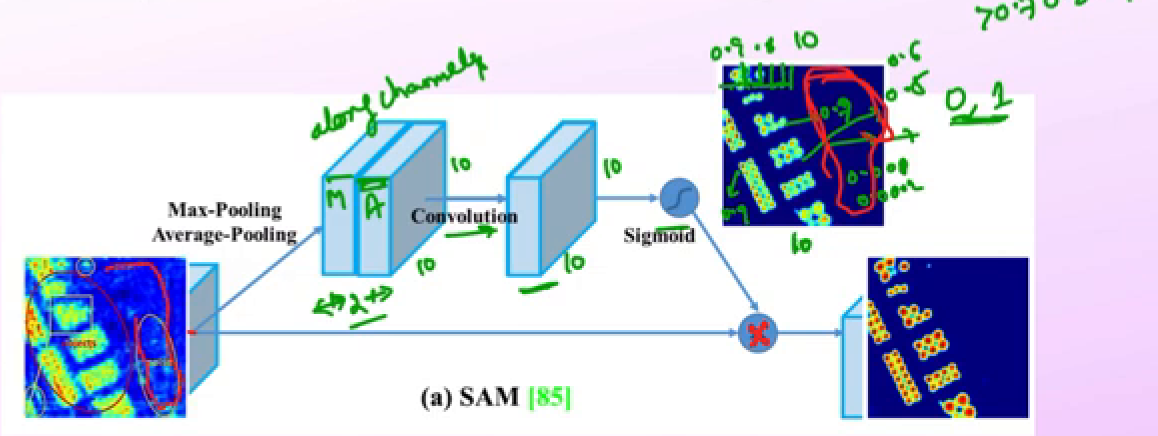
(a) SAM [85]

Spatial Attention Module (SAM)



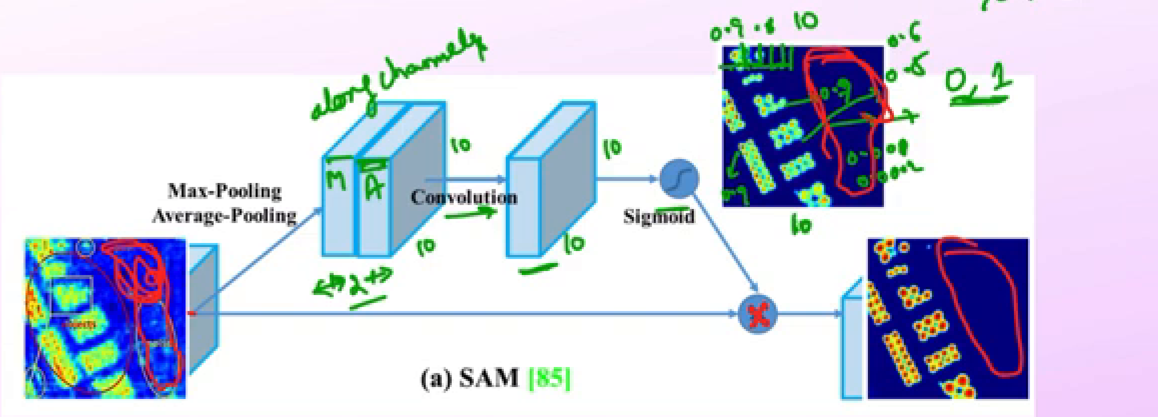
(a) SAM [85]

Spatial Attention Module (SAM)



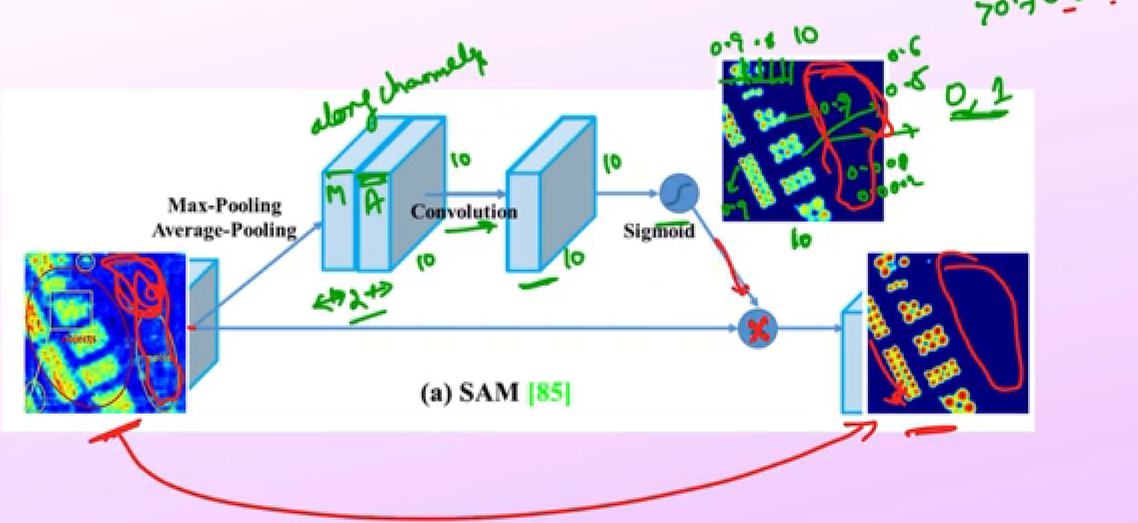
(a) SAM [85]

Spatial Attention Module (SAM)



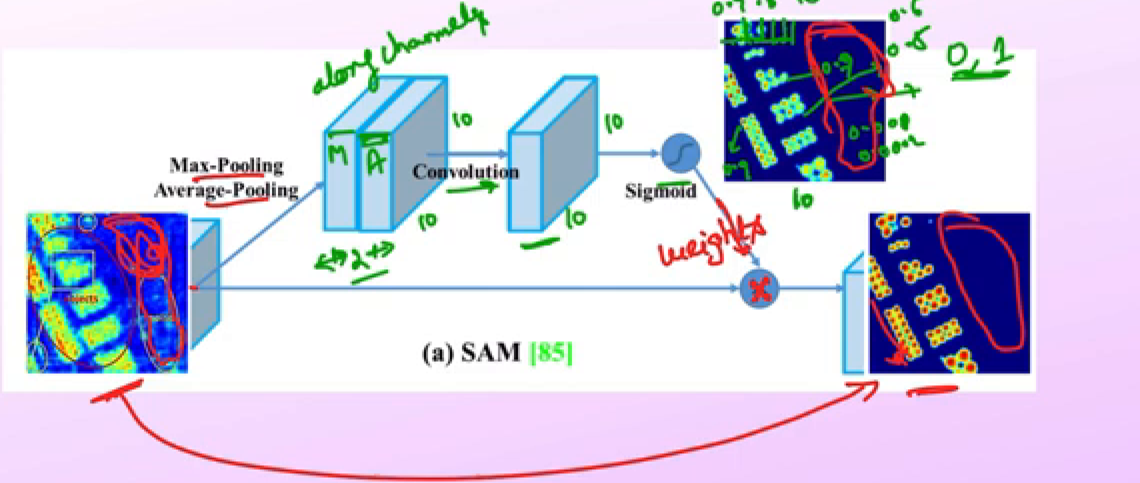
(a) SAM [85]

Spatial Attention Module (SAM)



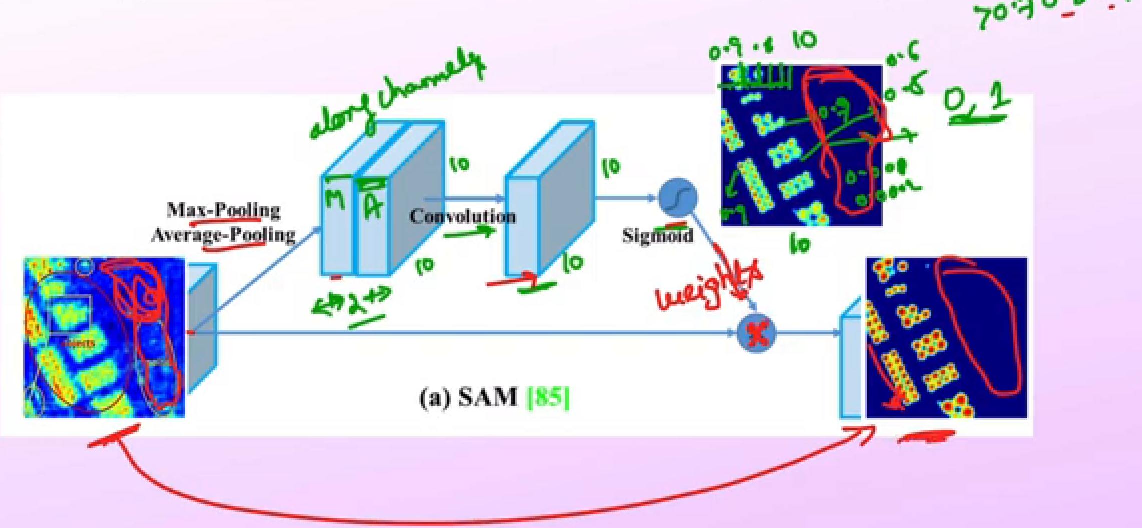
(a) SAM [85]

Spatial Attention Module (SAM)



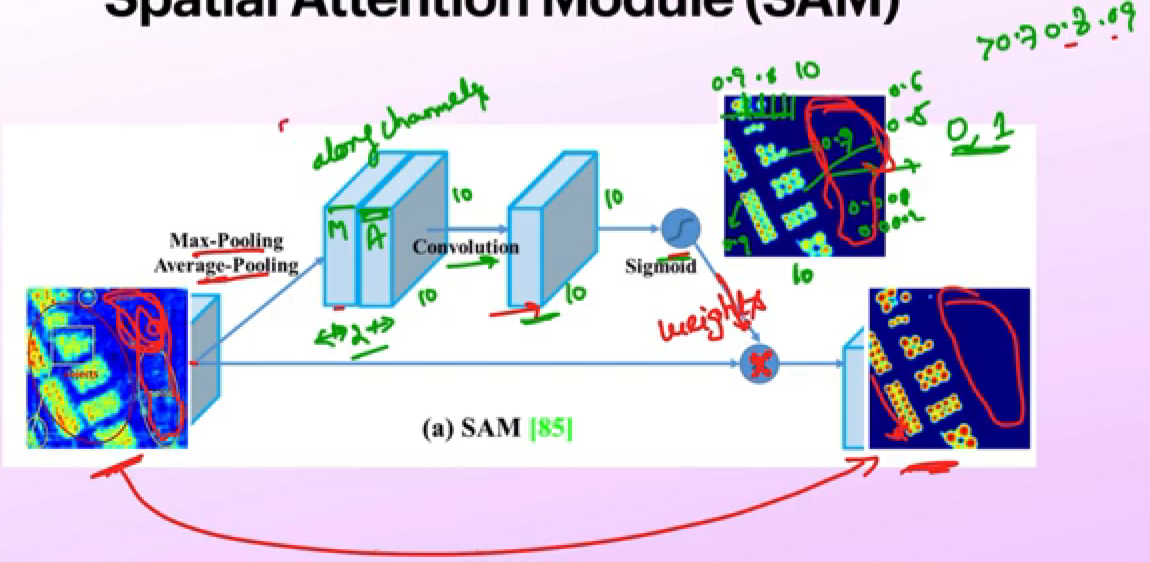
(a) SAM [85]

Spatial Attention Module (SAM)

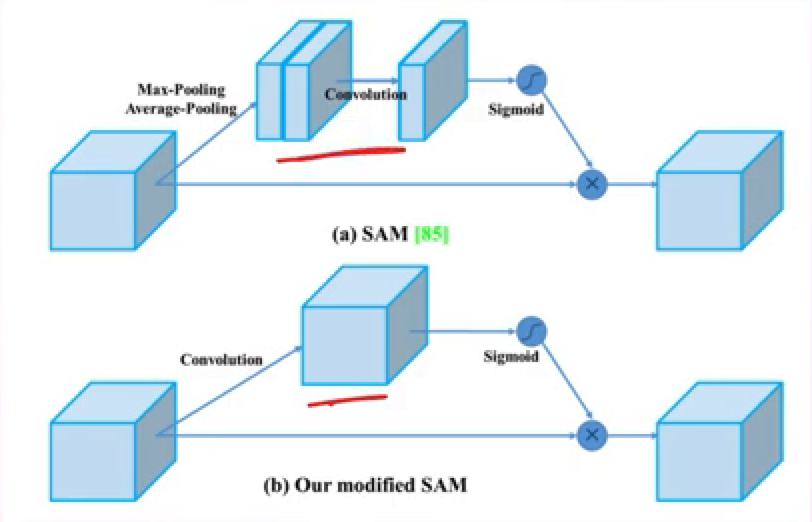


(a) SAM [85]

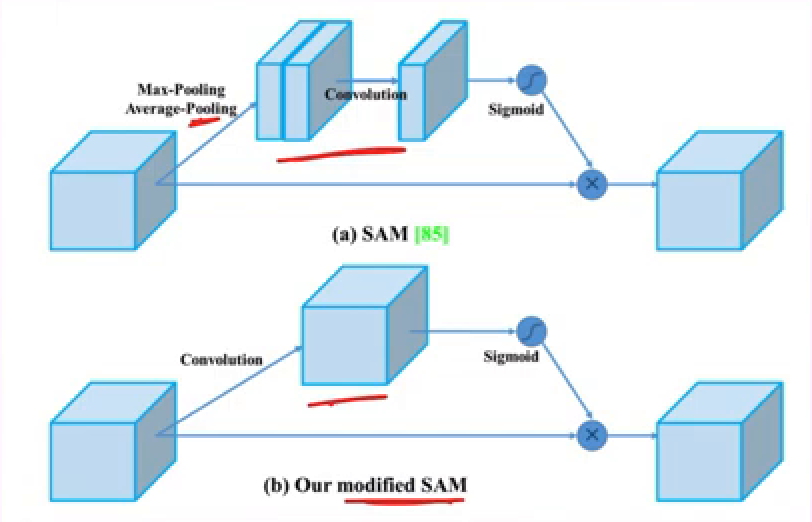
Spatial Attention Module (SAM)



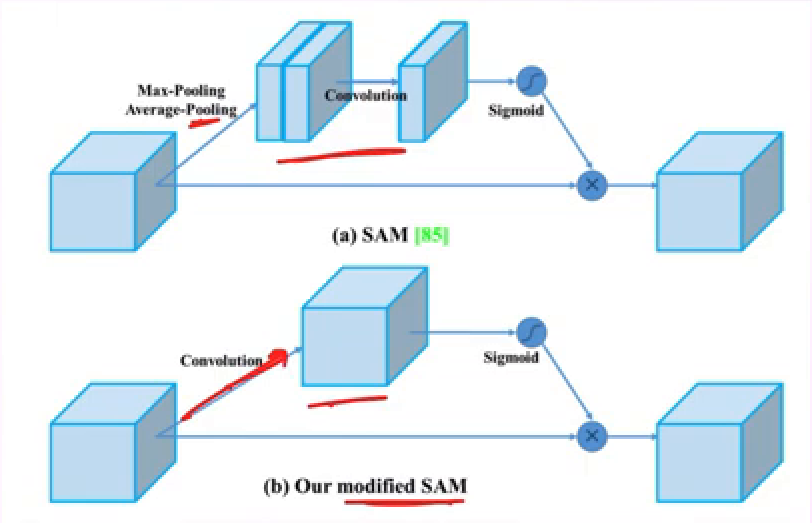
Modified SAM



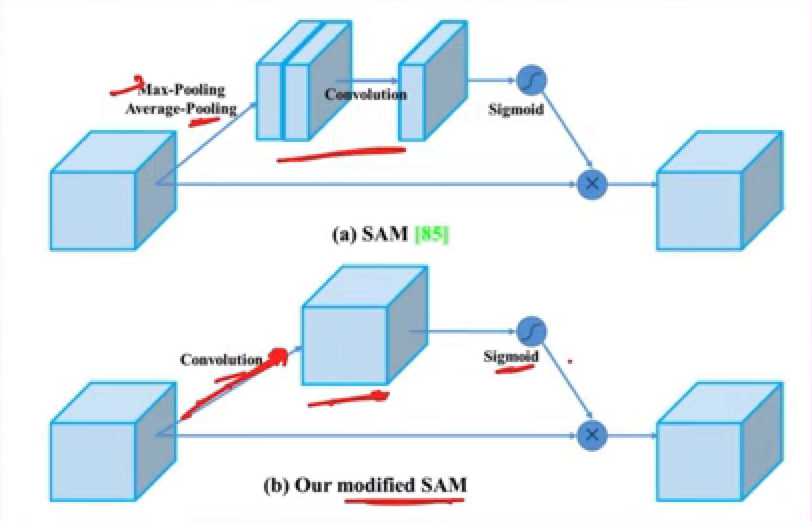
Modified SAM



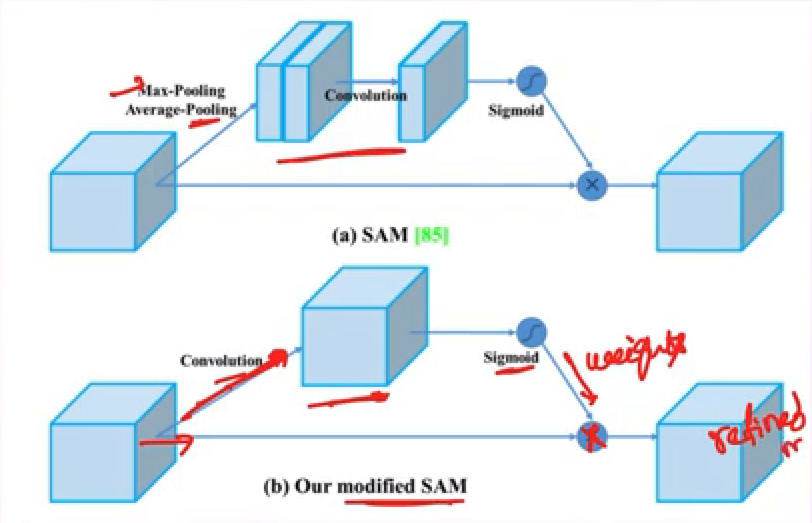
Modified SAM



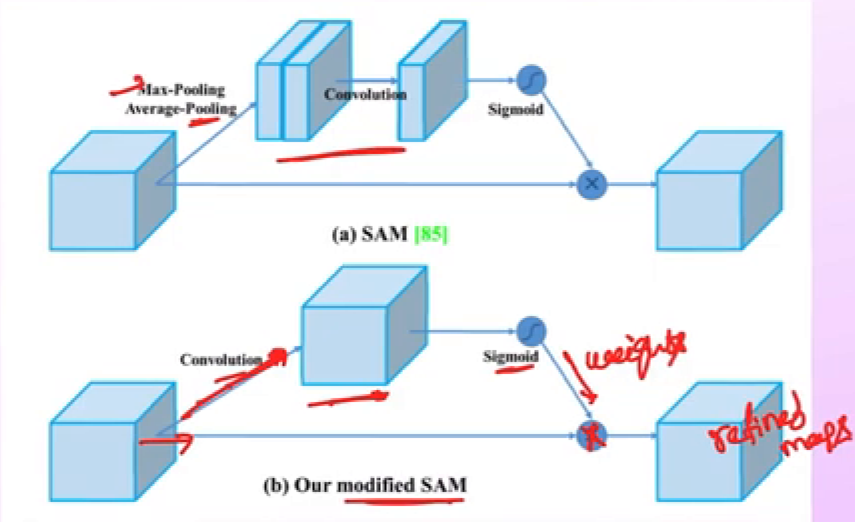
Modified SAM



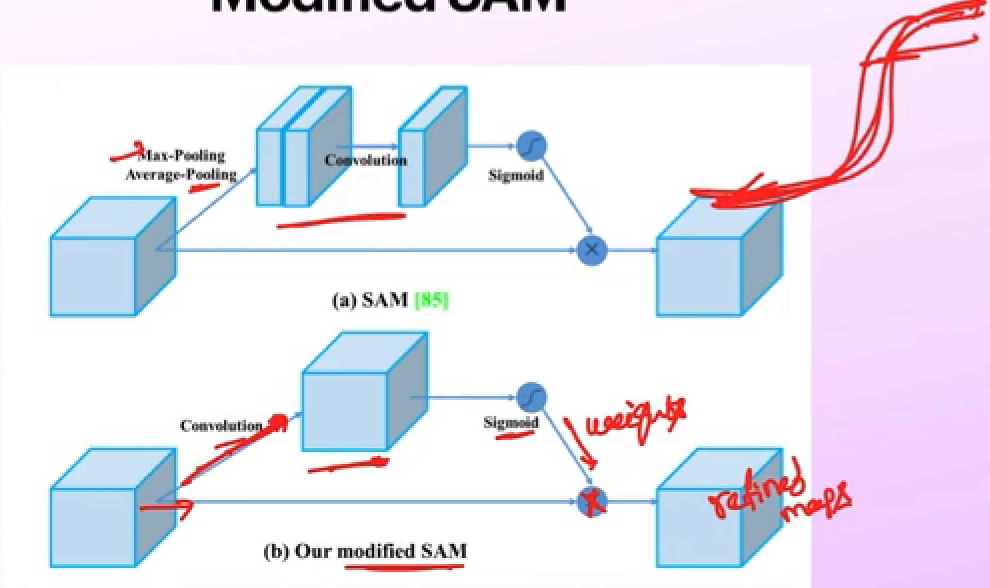
Modified SAM



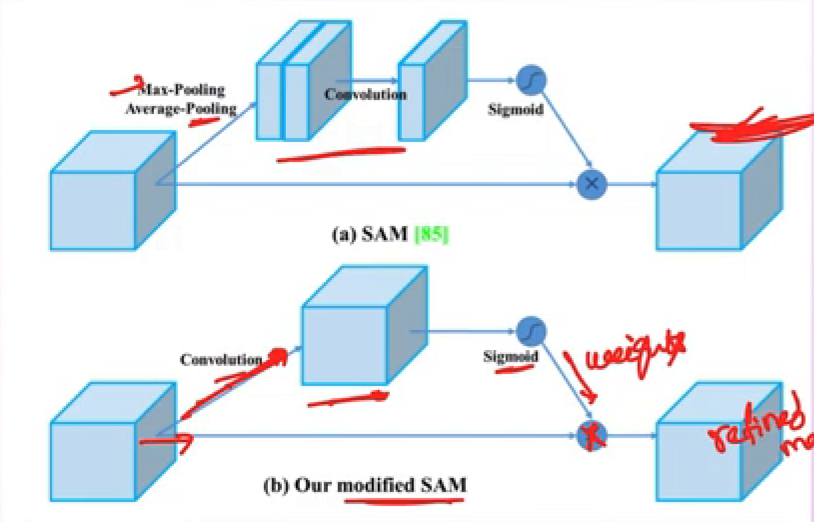
Modified SAM



Modified SAM



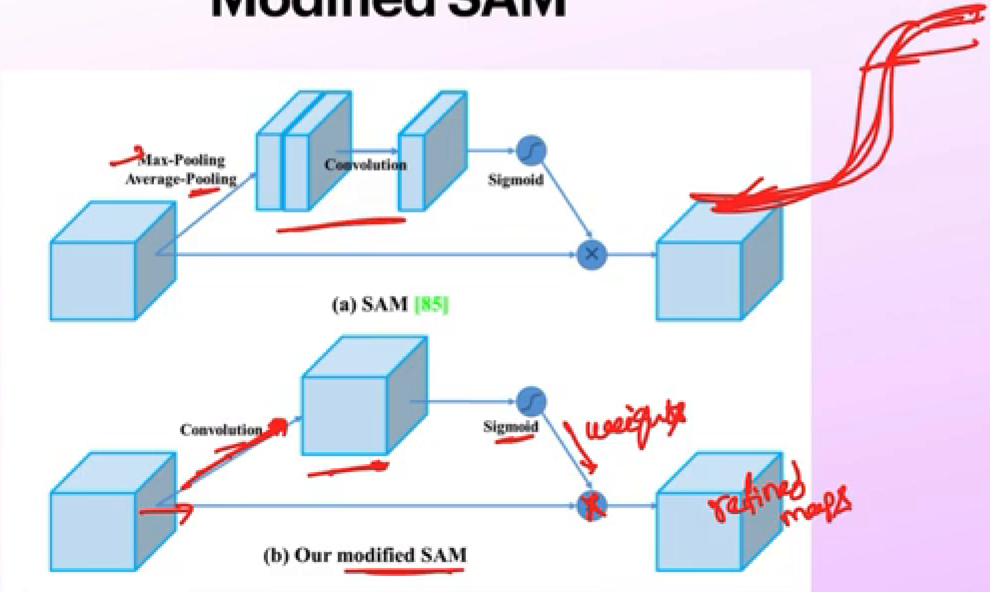
Modified SAM



(a) SAM [85]

(b) Our modified SAM

Modified SAM



BoF & BoS

	Backbone	Detector
Bag of Freebies (BoF)	• CutMix • Mosaic data augmentation • DropBlock • Class label smoothing	• CIoU-loss • Cross mini-Batch Normalization • DropBlock • Mosaic data augmentation • Self-Adversarial Training • Multiple anchors for a single ground truth • Cosine annealing scheduler • Optimal hyperparameters • Random training shapes
Bag of Specials (BoS)	• Mish activation • Cross-stage partial connections (CSP) • Multi-input weighted residual connections (MiWRC)	• Mish activation • SPP-block • SAM-block • PAN path-aggregation block • DIoU-NMS

BoF & BoS

	Backbone	Detector
Bag of Freebies (BoF)	• CutMix • Mosaic data augmentation • DropBlock • Class label smoothing	• CIoU-loss • Cross mini-Batch Normalization • DropBlock • Mosaic data augmentation • Self-Adversarial Training • Multiple anchors for a single ground truth • Cosine annealing scheduler • Optimal hyperparameters • Random training shapes
Bag of Specials (BoS)	• Mish activation • Cross-stage partial connections (CSP) • Multi-input weighted residual connections (MiWRC)	• Mish activation • SPP-block • SAM-block • PAN path-aggregation block • DIoU-NMS

BoF & BoS

	Backbone	Detector
Bag of Freebies (BoF)	• CutMix • Mosaic data augmentation • DropBlock • Class label smoothing	• CIoU-loss • Cross mini-Batch Normalization • DropBlock • Mosaic data augmentation • Self-Adversarial Training • Multiple anchors for a single ground truth • Cosine annealing scheduler • Optimal hyperparameters • Random training shapes
Bag of Specials (BoS)	• Mish activation • Cross-stage partial connections (CSP) • Multi-input weighted residual connections (MiWRC)	• Mish activation • SPP-block • SAM-block • PAN path-aggregation block • DIoU-NMS

BoF & BoS

	Backbone	Detector
Bag of Freebies (BoF)	• CutMix • Mosaic data augmentation	• CIoU-loss • Cross mini-Batch Normalization ugmentation al Training rs for a single ground • Cosine annealing scheduler • Optimal hyperparameters • Random training shapes
Bag of Specials (BoS)	• Mish activation • Cross-stage partial connections (CSP) • Multi-input weighted residual connections (MiWRC)	• Mish activation • SPP-block • SAM-block • PAN path-aggregation block • DIoU-NMS

Will cover in P3

BoF & BoS

Will cover in P3

	Backbone	Detector
Bag of Freebies (BoF)	• CutMix • Mosaic data augmentation	• CIoU-loss • Cross mini-Batch Normalization ugmentation al Training rs for a single ground • Cosine annealing scheduler • Optimal hyperparameters • Random training shapes
Bag of Specials (BoS)	• Mish activation • Cross-stage partial connections (CSP) • Multi-input weighted residual connections (MiWRC)	• Mish activation • SPP-block • SAM-block • PAN path-aggregation block • DIoU-NMS

BoF & BoS

	Backbone	Detector
Bag of Freebies (BoF)	• CutMix • Mosaic data augmentation	• CIoU-loss • Cross mini-Batch Normalization ugmentation al Training rs for a single ground • Cosine annealing scheduler • Optimal hyperparameters • Random training shapes
Bag of Specials (BoS)	• Mish activation • Cross-stage partial connections (CSP) • Multi-input weighted residual connections (MiWRC)	• Mish activation • SPP-block • SAM-block • PAN path-aggregation block • DIoU-NMS

Will cover in P3

w_1 w_2 w_3

BoF & BoS

	Backbone	Detector
Bag of Freebies (BoF)	• CutMix • Mosaic data augmentation • DropBlock • Class label smoothing	• CIoU-loss • Cross mini-Batch Normalization • DropBlock • Mosaic data augmentation • Self-Adversarial Training • Multiple anchors for a single ground truth • Cosine annealing scheduler • Optimal hyperparameters • Random training shapes
Bag of Specials (BoS)	• Mish activation • Cross-stage partial connections (CSP) • Multi-input weighted residual connections (MiWRC)	• Mish activation • SPP-block • SAM-block • PAN path-aggregation block • DIoU-NMS

w_1 w_2 w_3

BoF & BoS

	Backbone	Detector
Bag of Freebies (BoF)	• CutMix • Mosaic data augmentation • DropBlock • Class label smoothing	• CIoU-loss • Cross mini-Batch Normalization • DropBlock • Mosaic data augmentation • Self-Adversarial Training • Multiple anchors for a single ground truth • Cosine annealing scheduler • Optimal hyperparameters • Random training shapes
Bag of Specials (BoS)	• Mish activation • Cross-stage partial connections (CSP) • Multi-input weighted residual connections (MiWRC)	• Mish activation • SPP-block • SAM-block • PAN path-aggregation block • DIoU-NMS

w_1 w_2 w_3

BoF & BoS

	Backbone	Detector
Bag of Freebies (BoF)	• CutMix • Mosaic data augmentation • DropBlock • Class label smoothing	• CIoU-loss • Cross mini-Batch Normalization • DropBlock • Mosaic data augmentation • Self-Adversarial Training • Multiple anchors for a single ground truth • Cosine annealing scheduler • Optimal hyperparameters • Random training shapes
Bag of Specials (BoS)	• Mish activation • Cross-stage partial connections (CSP) • Multi-input weighted residual connections (MiWRC)	• Mish activation • SPP-block • SAM-block • PAN path-aggregation block • DIoU-NMS

w_1 w_2 w_3

BoF & BoS

	Backbone	Detector
Bag of Freebies (BoF)	• CutMix • Mosaic data augmentation • DropBlock • Class label smoothing	• CIoU-loss • Cross mini-Batch Normalization • DropBlock • Mosaic data augmentation • Self-Adversarial Training • Multiple anchors for a single ground truth • Cosine annealing scheduler • Optimal hyperparameters • Random training shapes
Bag of Specials (BoS)	• Mish activation • Cross-stage partial connections (CSP) • Multi-input weighted residual connections (MiWRC)	• Mish activation • SPP-block • SAM-block • PAN path-aggregation block • DIoU-NMS

w_1 w_2, w_3