kubwa®

BIGDATA & AI ANALYTICS
EXPERT COMPANY

Segmentation
Mask RCNN

kubwa®

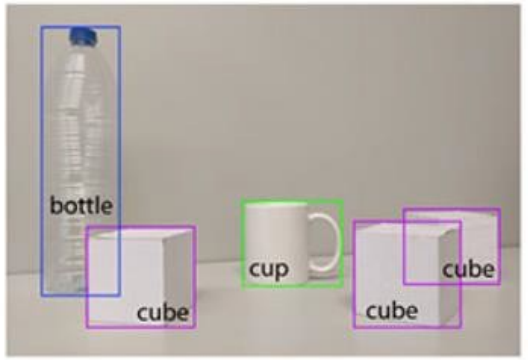
BIGDATA & AI ANALYTICS
EXPERT COMPANY

Segmentation 개요

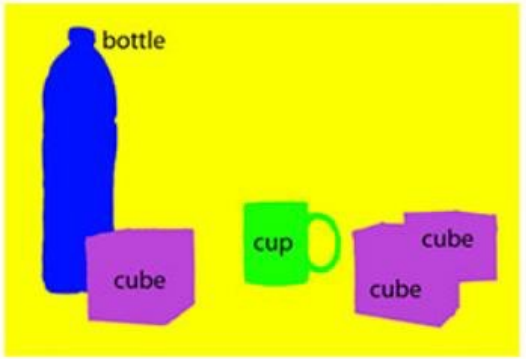
Object Detection과 Segmentation

Image Classification

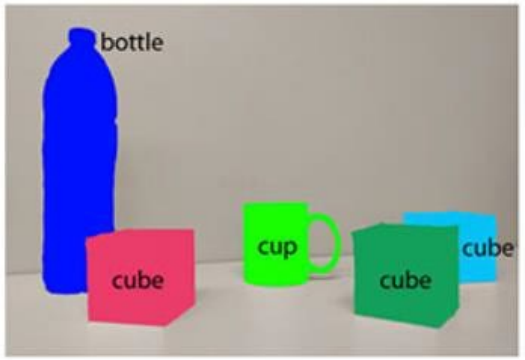
Object Detection



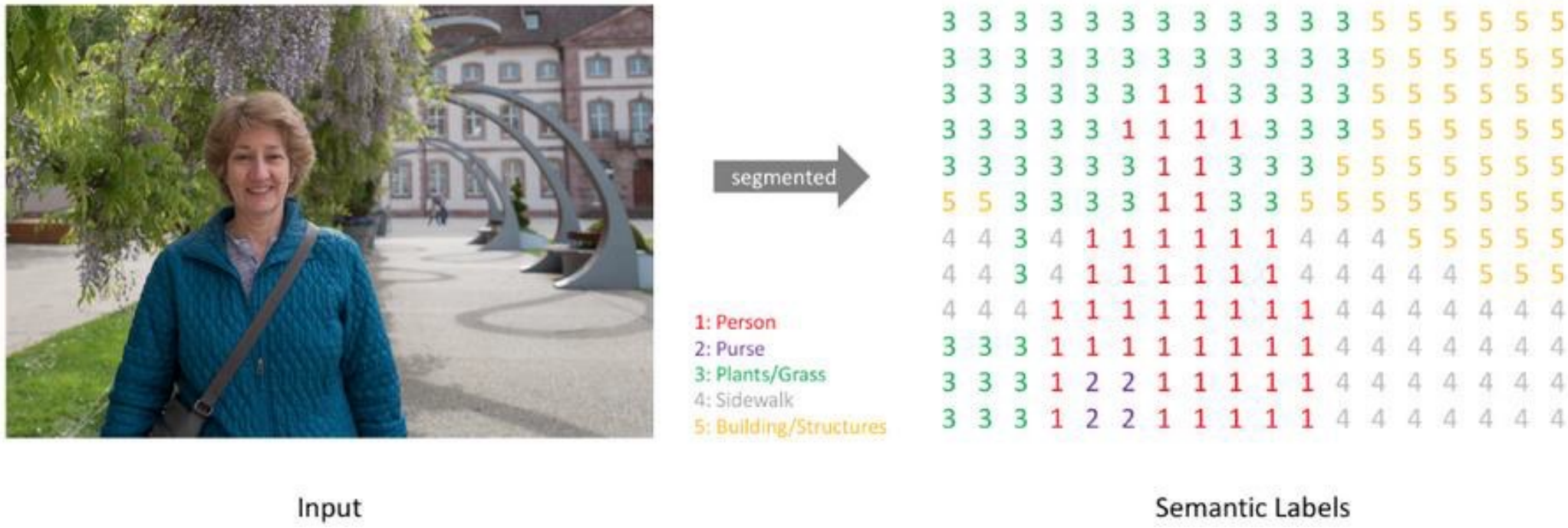
Semantic Segmentation



Instance Segmentation



이미지의 개별 Pixel별 Classification

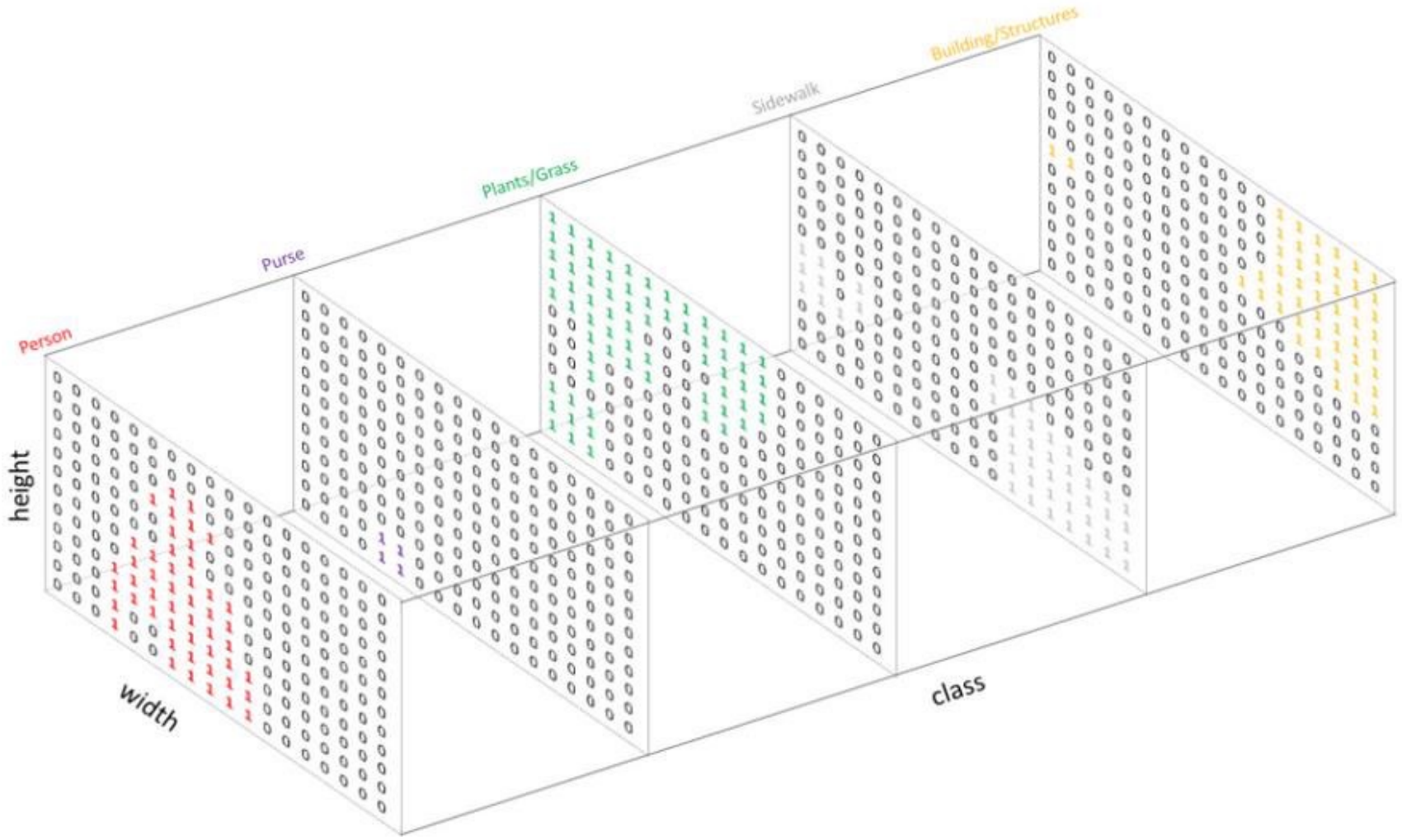


Input

Semantic Labels

이미지의 개별 Pixel별 Classification

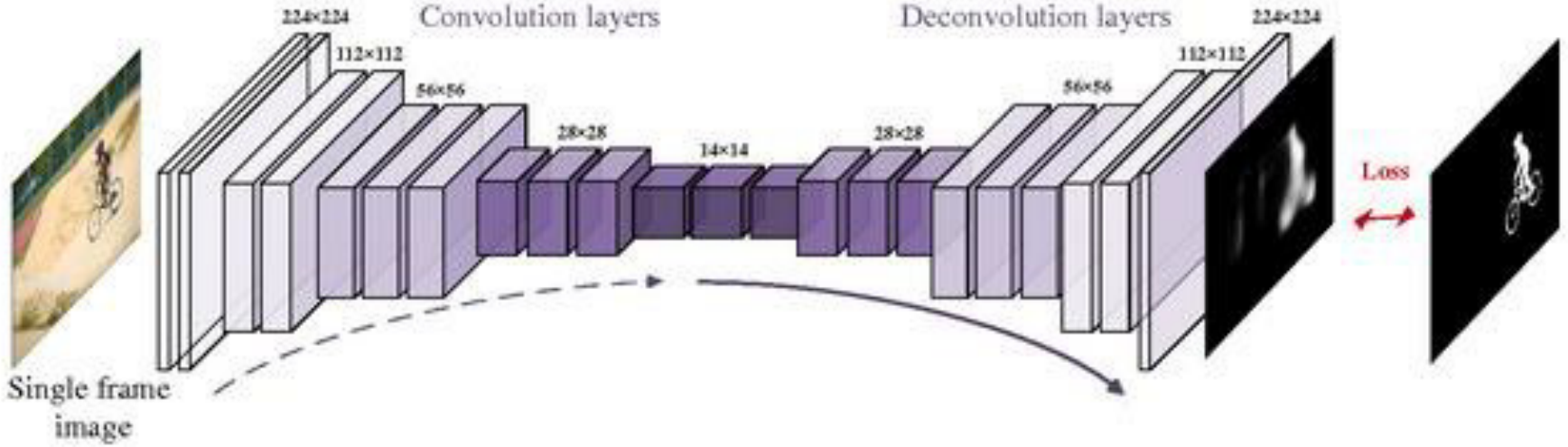
kubwa



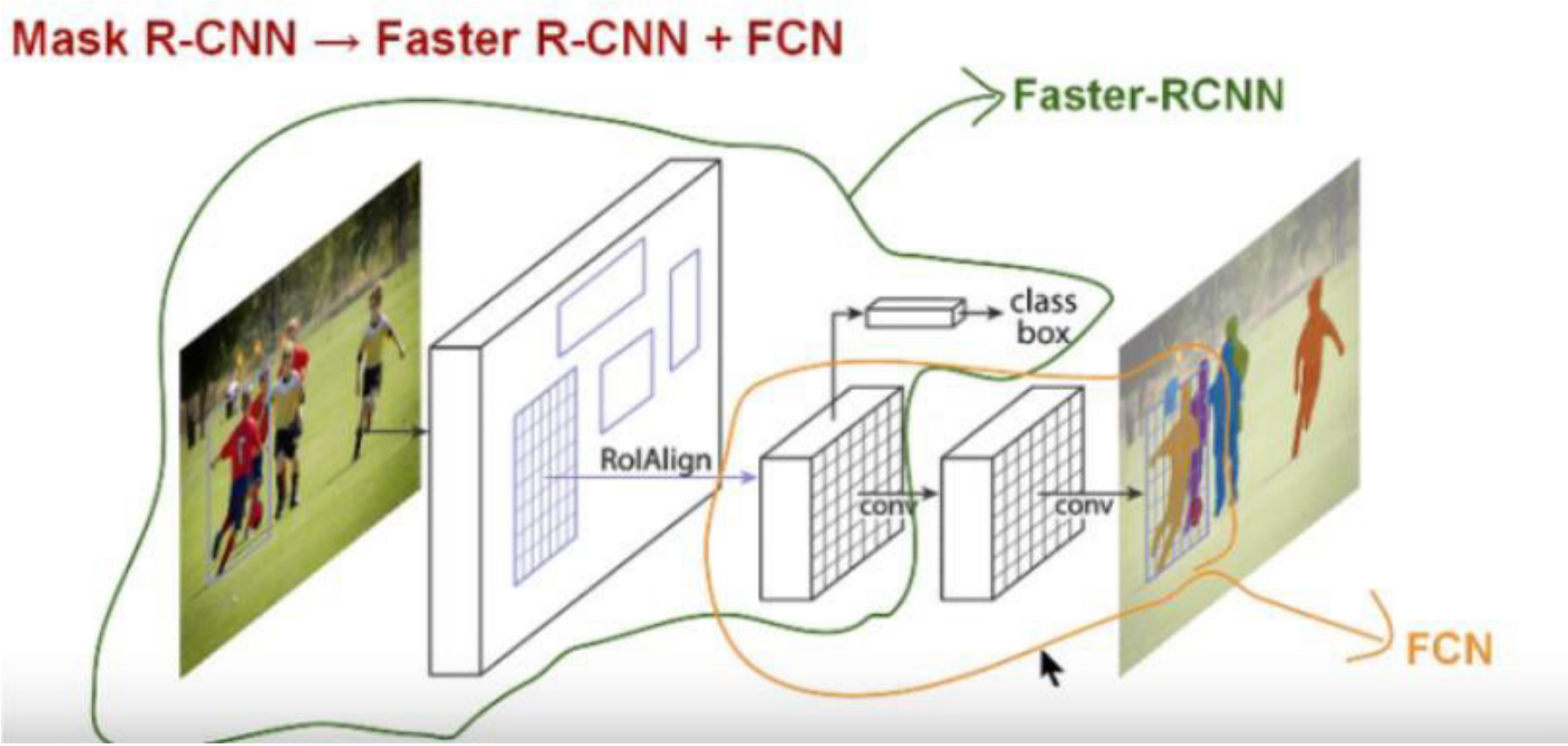
Semantic Segmentation Encoder-Decoder Model

kubwa®

FCN, Segnet, U-NET, Dilated FCN



Mask R-CNN



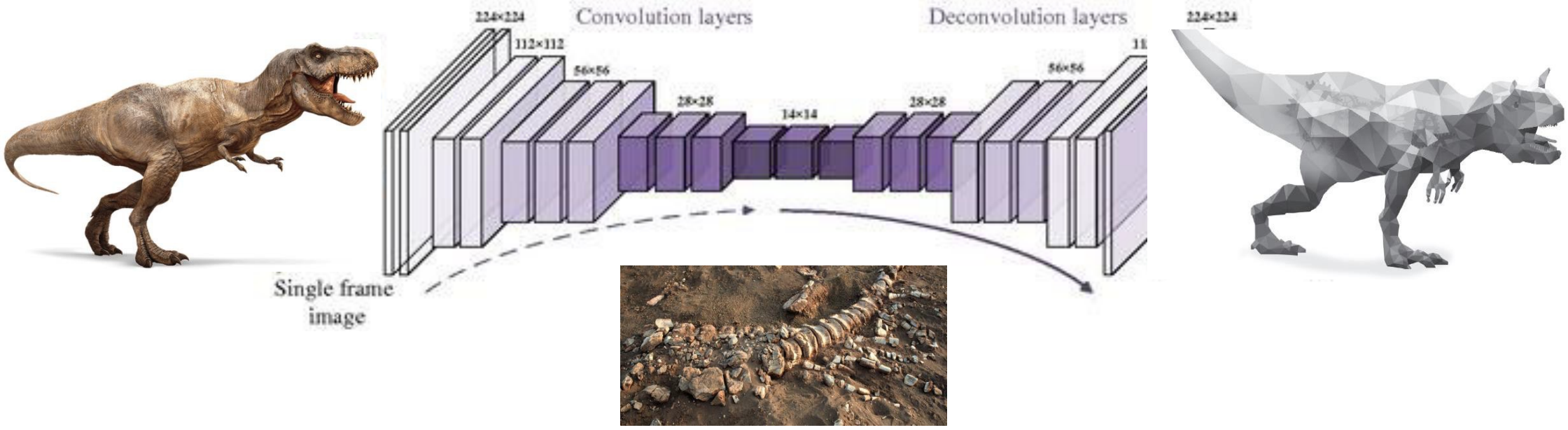
BIGDATA & AI ANALYTICS
EXPERT COMPANY

FCN(Fully Covolution Network)

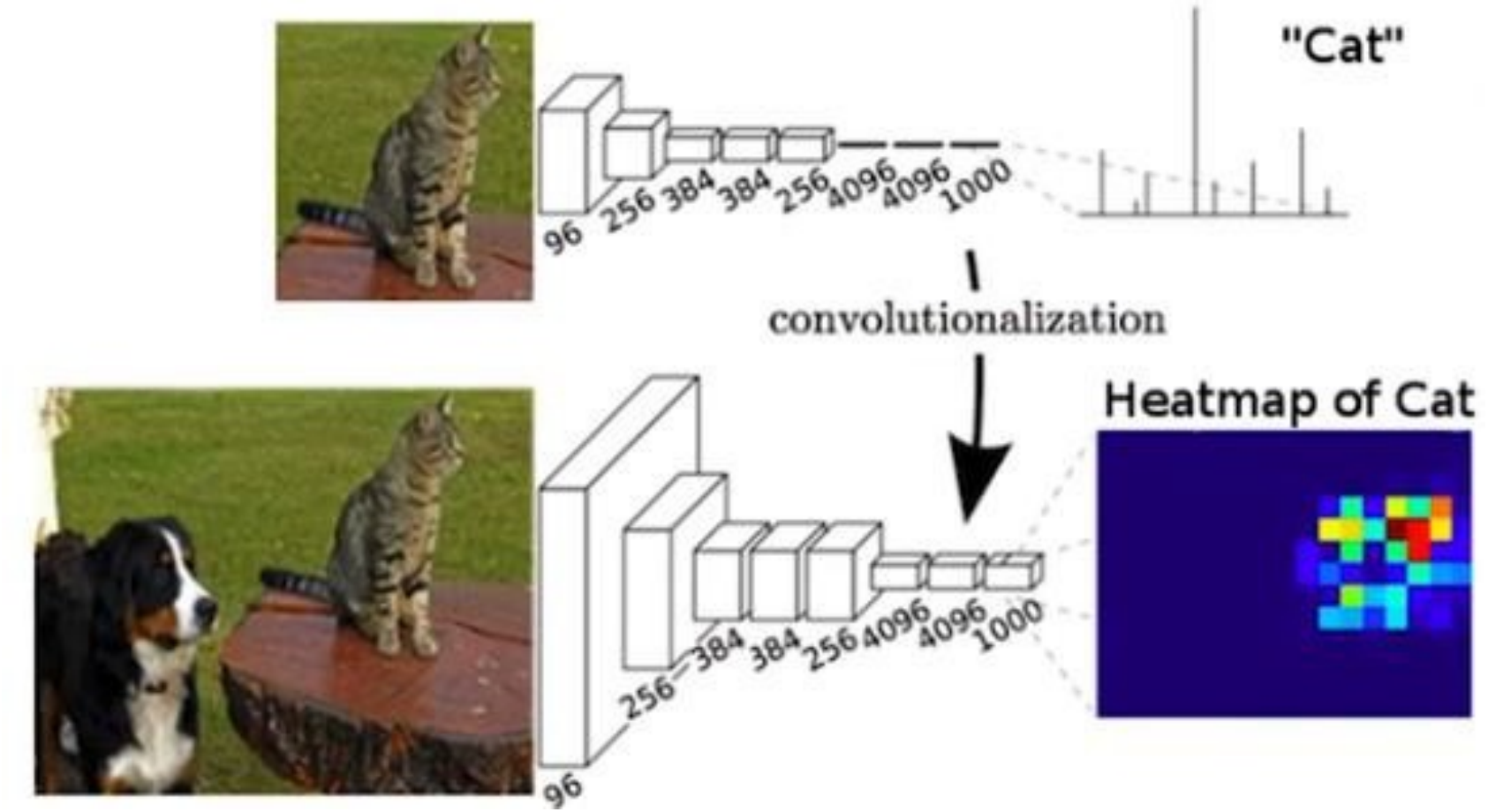
Semantic Segmentation Encoder-Decoder Model

kubwa

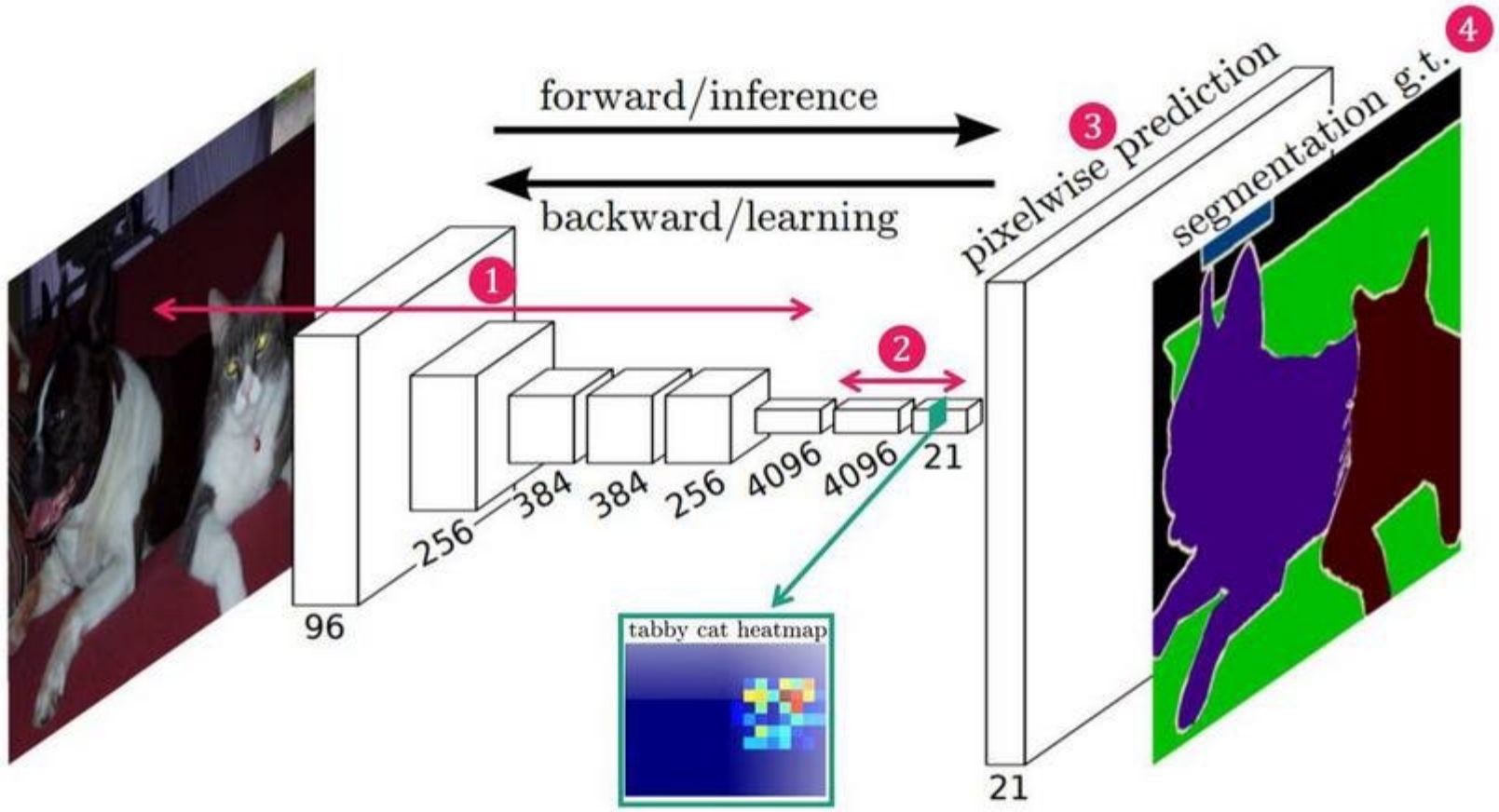
- 원본 이미지를 Convolution으로 차원 축소(Dimension Reduction)하여 응축된 정보를 가지고, 이를 다시 복원하면서 필요한 정보를 학습
- 이렇게 학습된 정보를 기반으로 Segmentation 수행



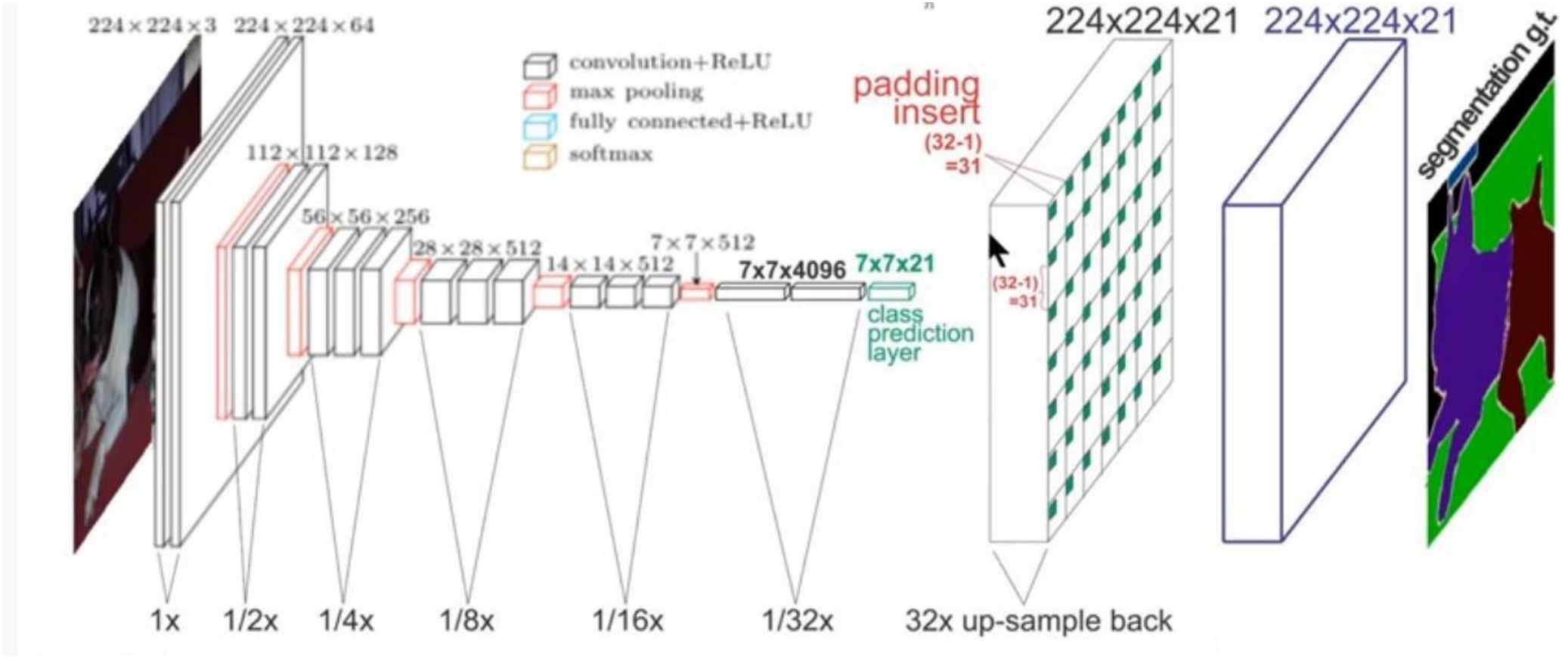
Fully Connected Layer vs Fully Convolutional layer



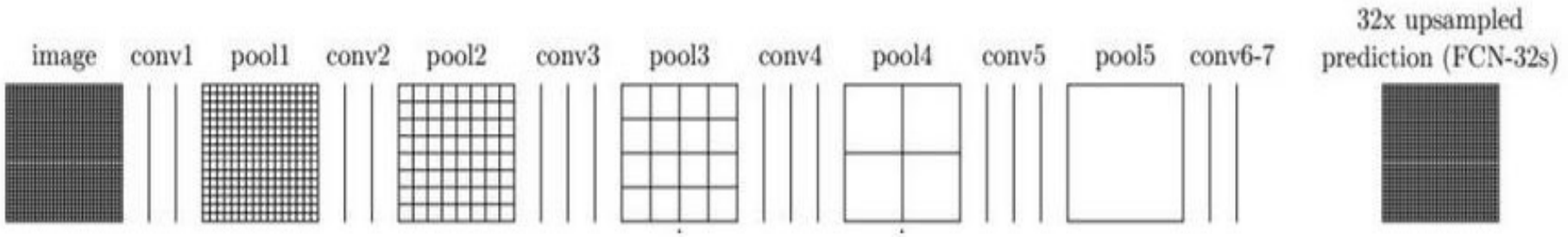
FCN(Fully Convolutional Network) for Semantic Segmentation



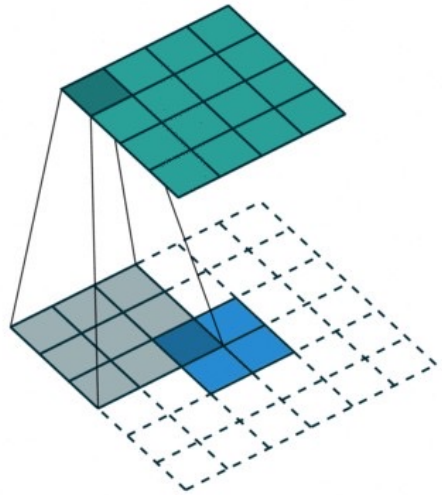
FCN – Down sampling과 Up sampling



32x Up sampling을 통한 Pixel wise prediction



De-Convolution을 통한 Upsampling

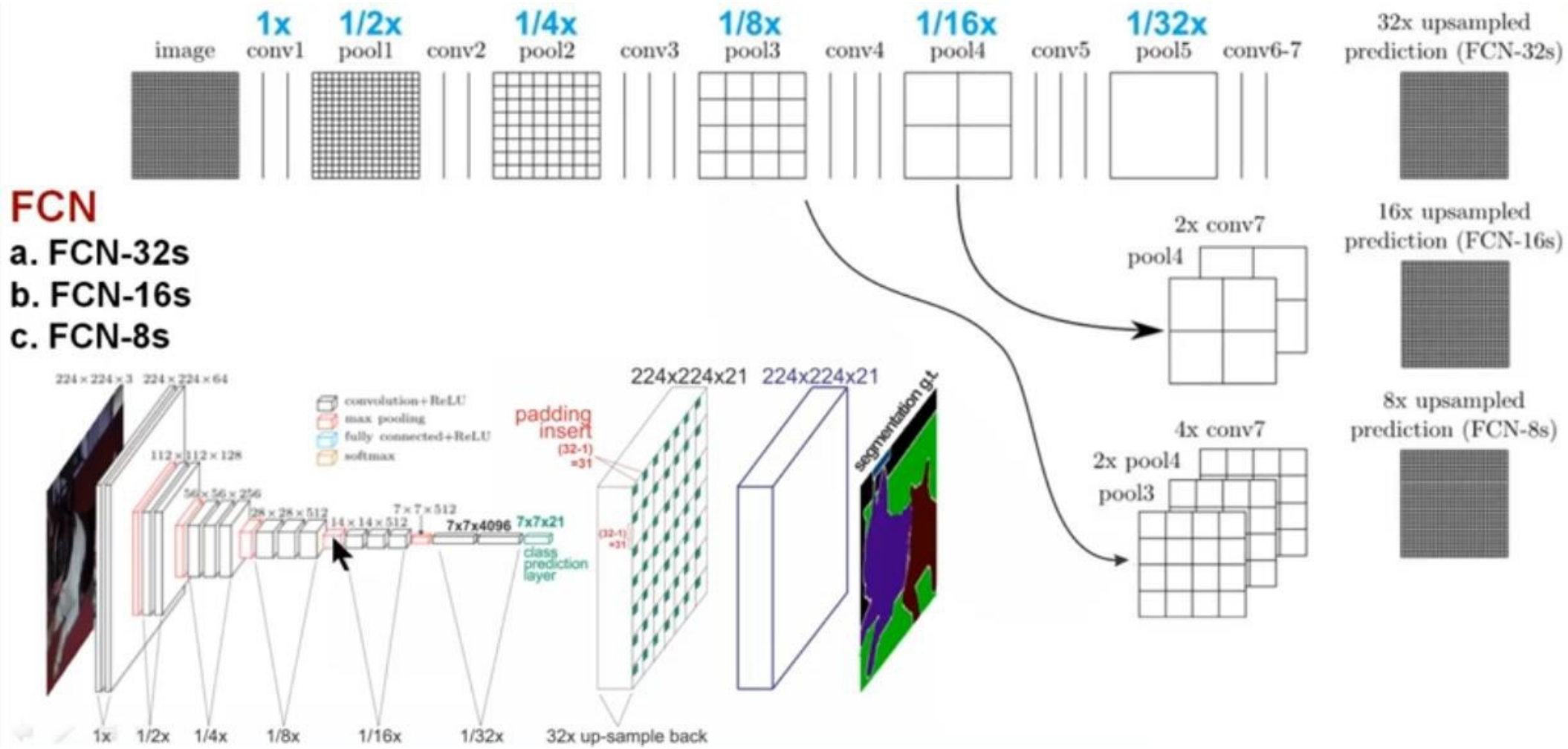


Ground truth

FCN-32s

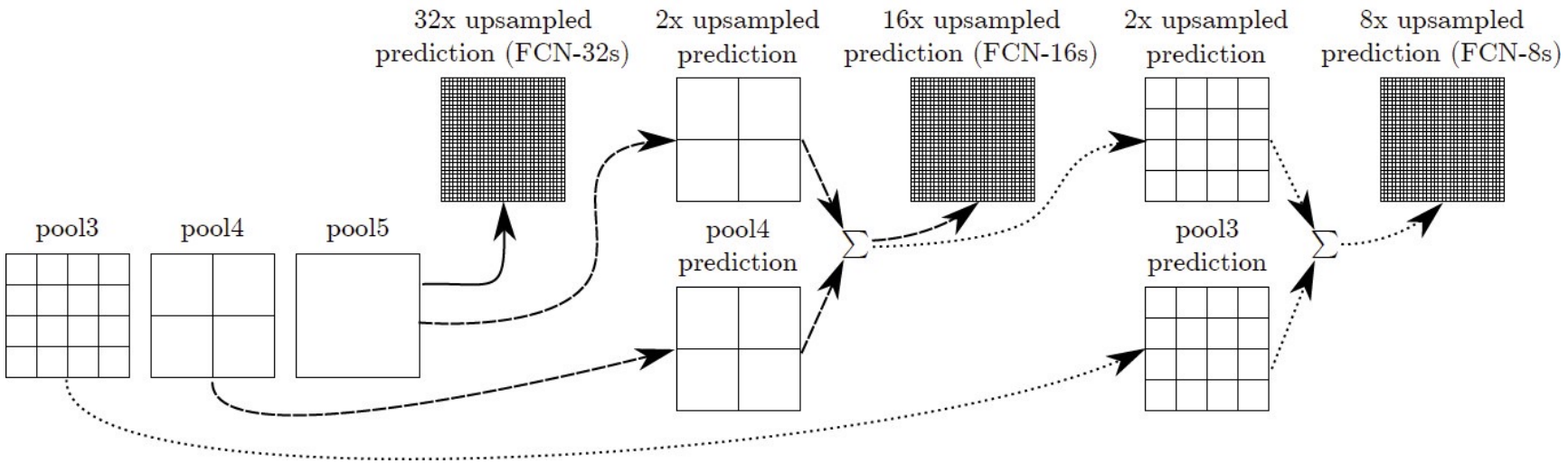
Feature Map을 혼합하여 Pixel Wise Prediction

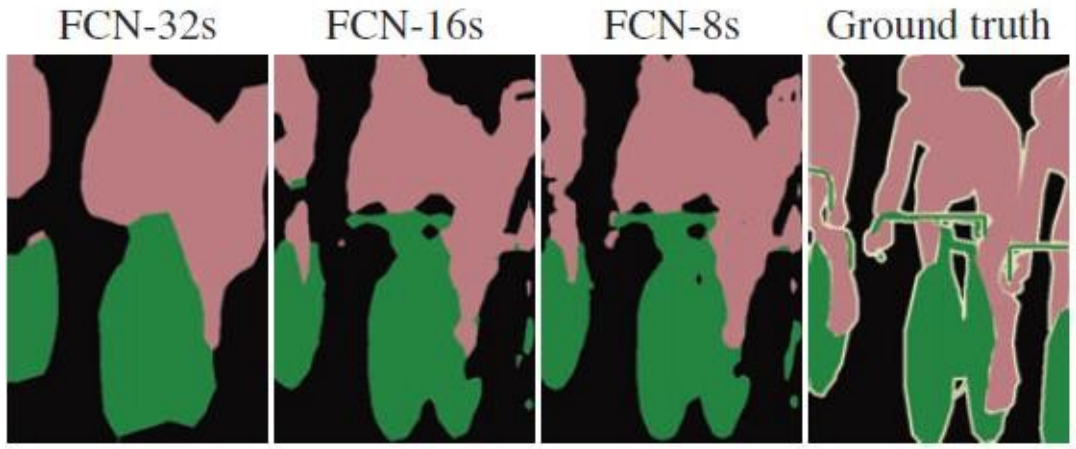
kubwa®



kubwa

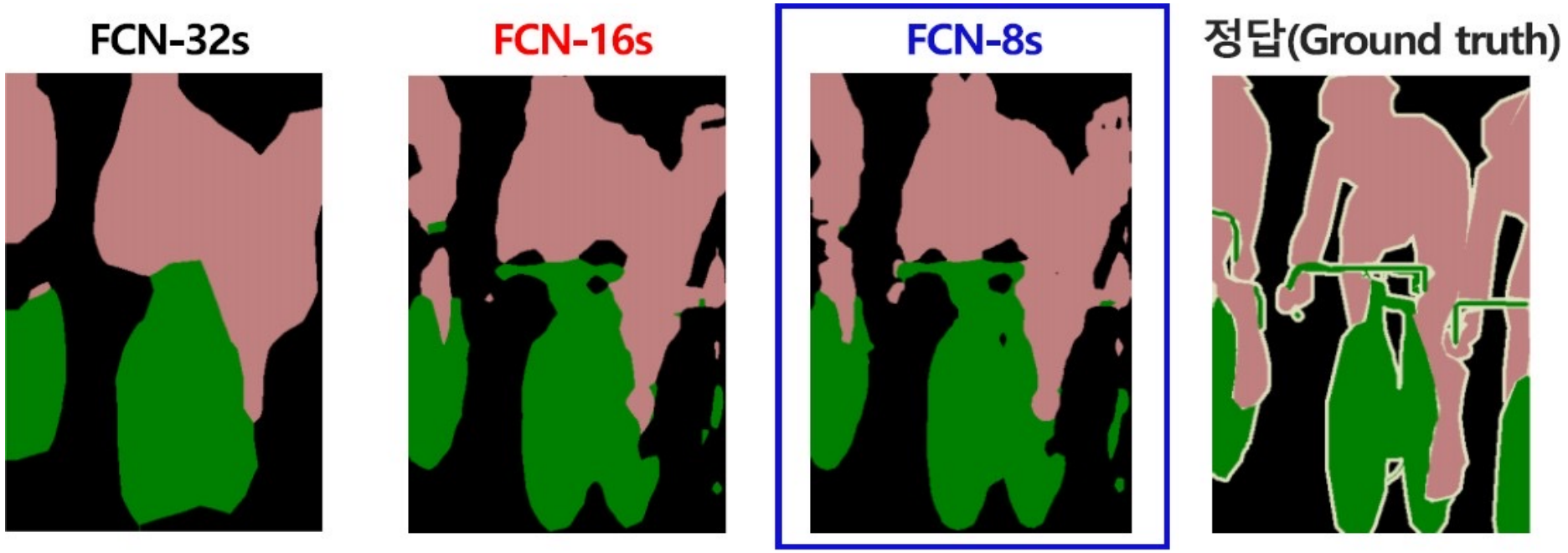
Feature Map을 혼합하여 Pixel Wise Prediction





kubwa

FCN-32s, 16s, 8s 별 성능



	FCN-32s	FCN-16s	FCN-8s
IoU (Intersection over Union)	59.4%	62.4%	62.7%

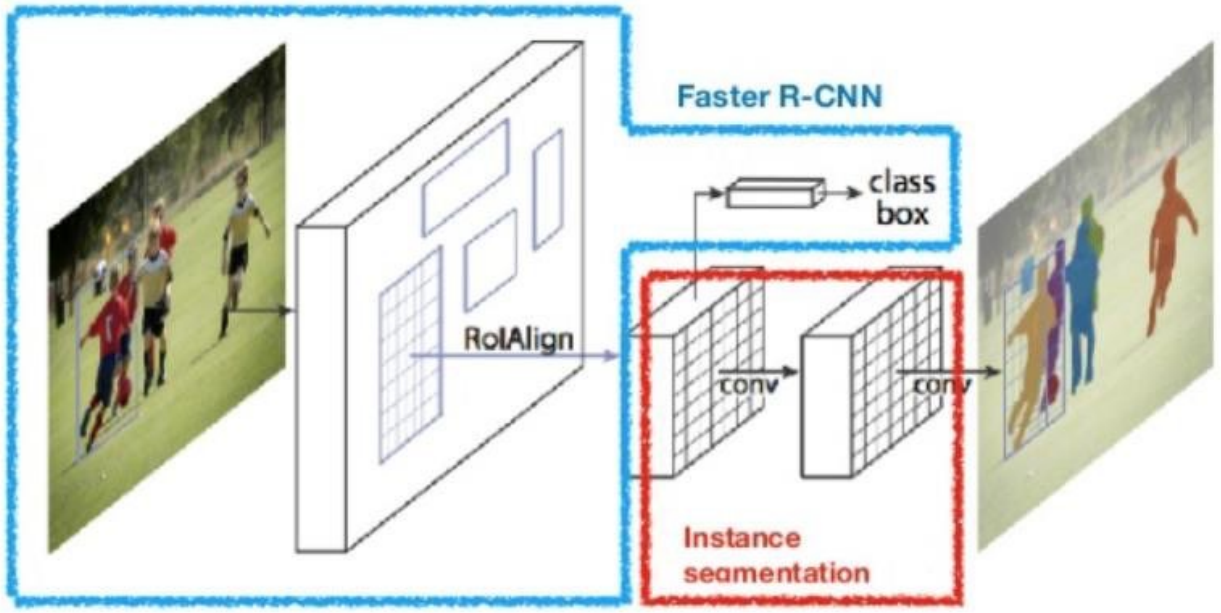
kubwa®

BIGDATA & AI ANALYTICS
EXPERT COMPANY

Mask RCNN

Mask RCNN

- Faster RCNN 과 FCN 기법 개선 및 결합
  - ROI-Align
  - 기존 Bounding box regression과 Classification에 Binary Mask Prediction 추가
- 비교적 빠른 Detection 시간과 높은 정확도
- 직관적이고 상대적으로 쉬운 구현

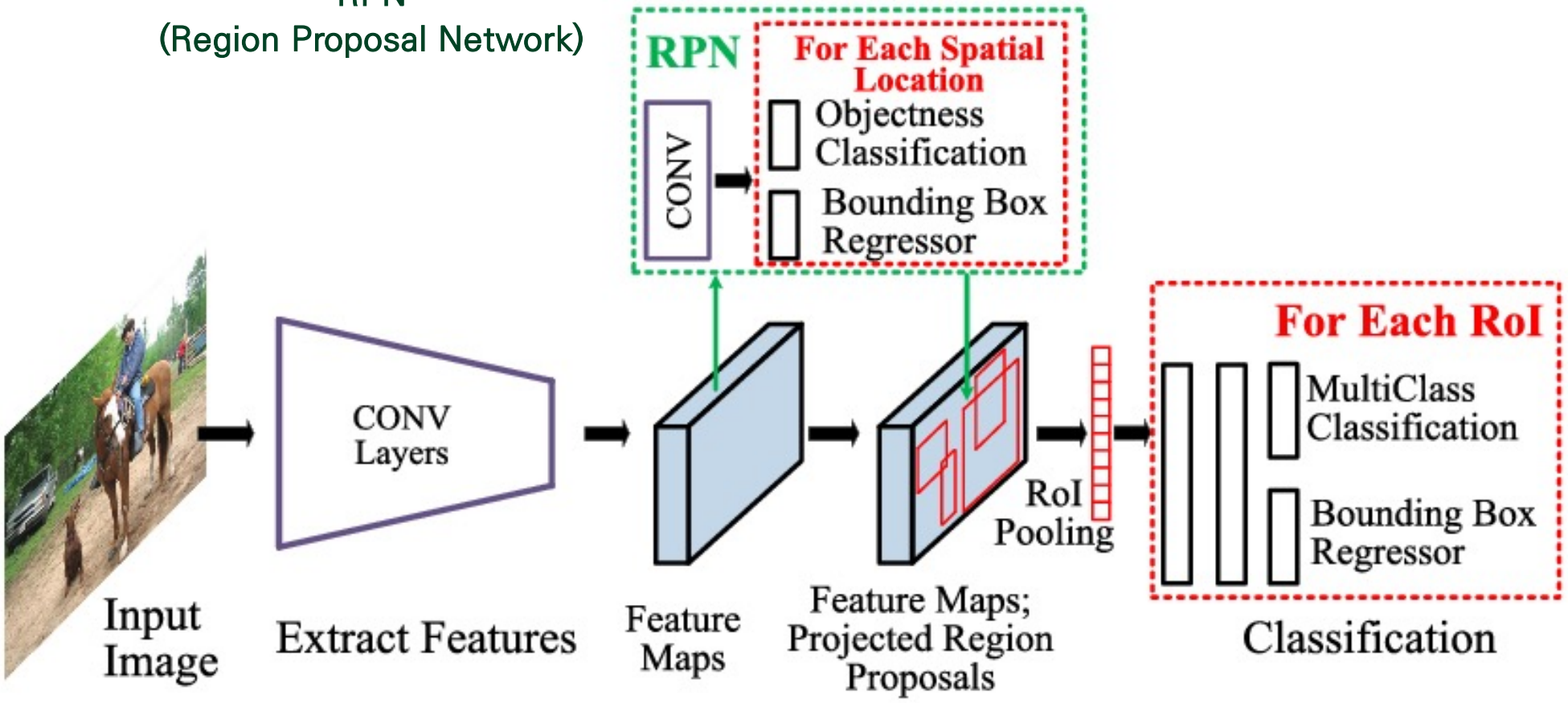


Faster R-CNN 구조

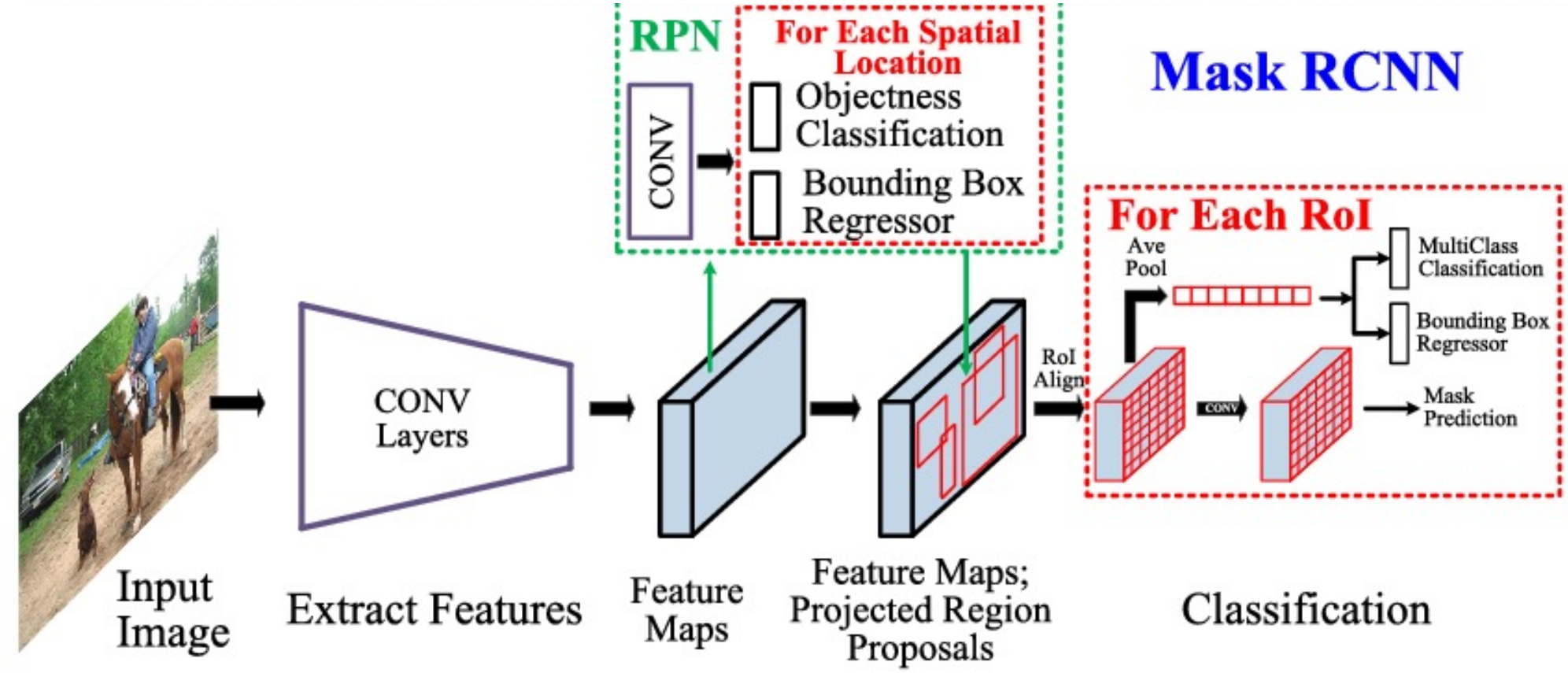
Selective Search를 Neural Network 구조로 변경

- GPU 사용으로 빠른 학습/Inference
- End to End Network 학습

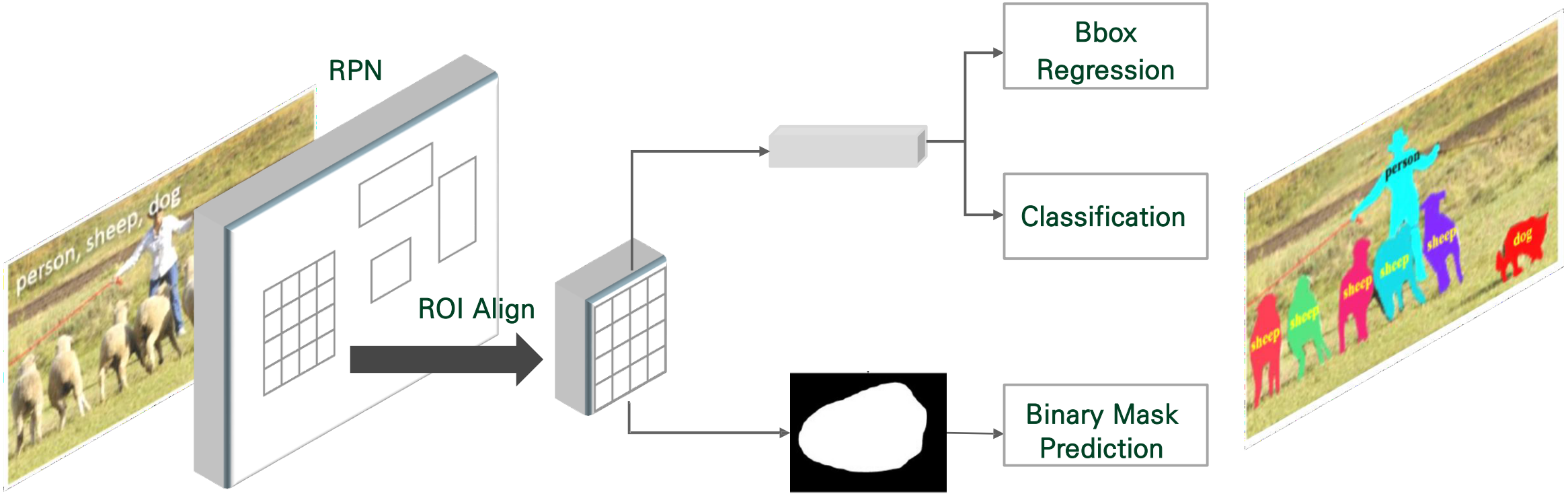
RPN
(Region Proposal Network)



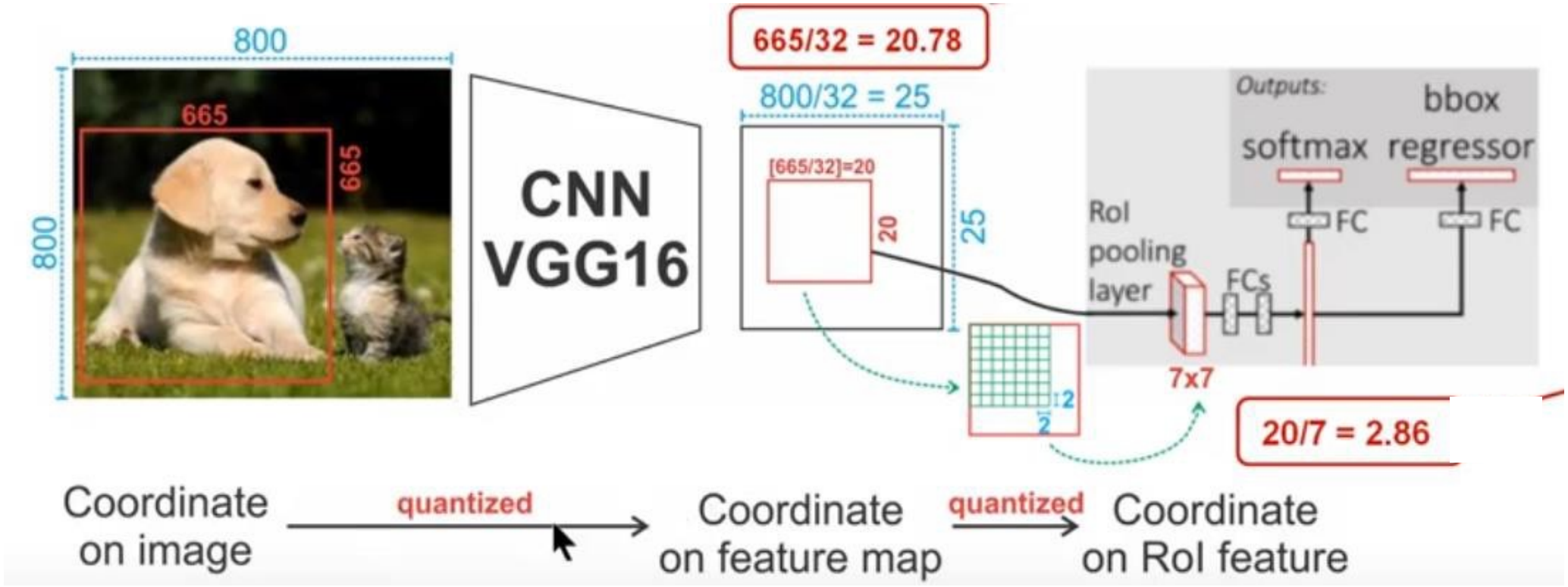
Mask RCNN 구조


RPN
For Each Spatial Location
CONV
Objectness Classification
Bounding Box Regressor
Mask RCNN
For Each RoI
Ave Pool
MultiClass Classification
Bounding Box Regressor
RoI Align
CONV
Mask Prediction
CONV Layers
Input Image
Extract Features
Feature Maps
Feature Maps; Projected Region Proposals
Classification

Mask RCNN 구조


person, sheep, dog
RPN
ROI Align
Bbox Regression
Classification
Binary Mask Prediction
person
sheep
sheep
sheep
sheep
sheep
dog

Segmentation에서 ROI Pooling 문제점



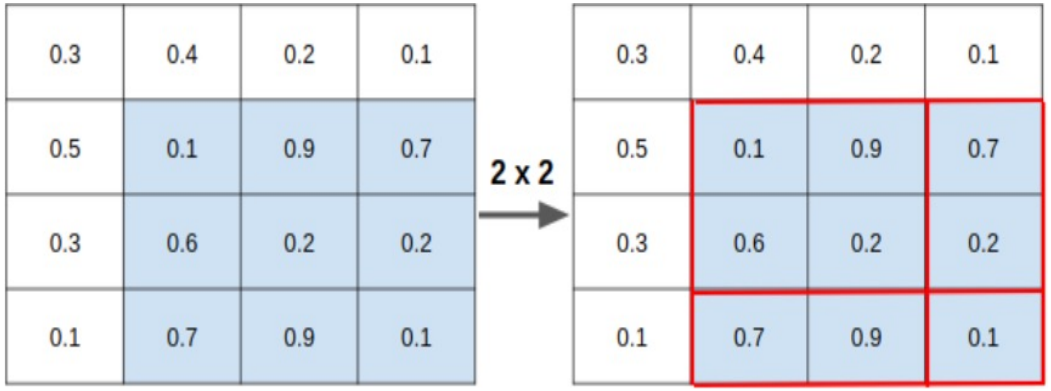
Ref. Mask RCNN Arc.(Part3) – How RoI Pooling, RoI Warping & RoI Align Work
https://www.youtube.com/watch?v=XGi-Mz3do2s&list=PLkRkKTC6HZMxZrxnHUDYSLiPZxiUUFD2C&index=4

Segmentation에서 ROI Pooling 문제점

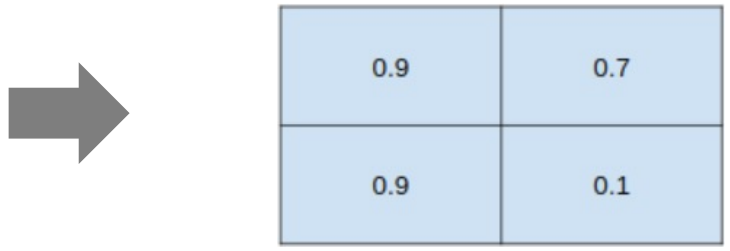
Feature Map

0.3	0.4	0.2	0.1
0.5	0.1	0.9	0.7
0.3	0.6	0.2	0.2
0.1	0.7	0.9	0.1

ROI 영역

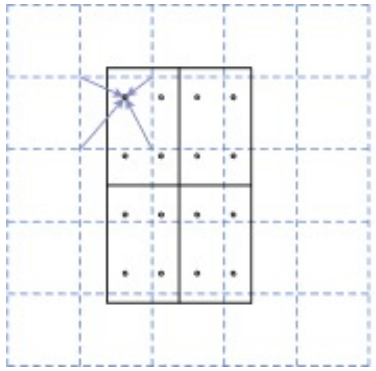


2X2 ROI (Max) Pooling



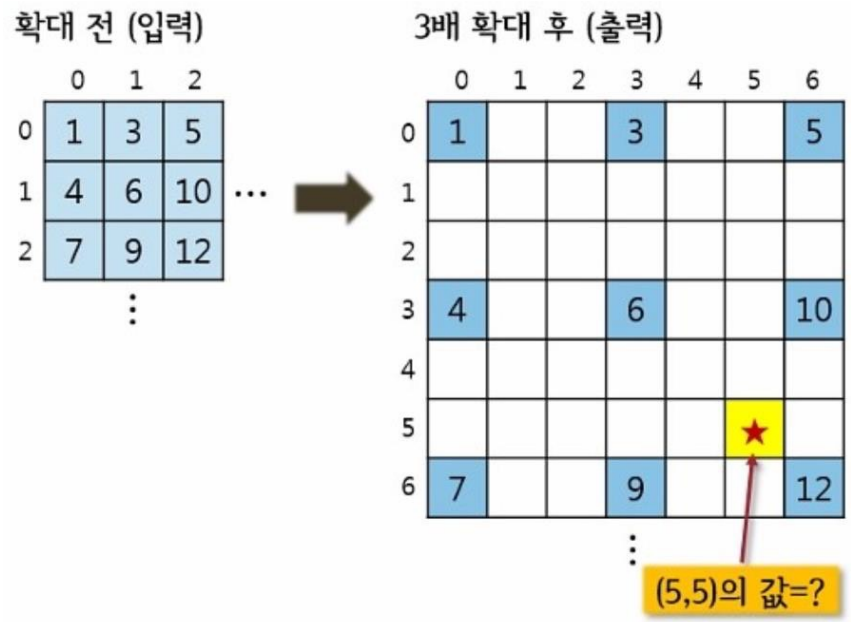
Bilinear Interpolation을 이용한 ROI-Align

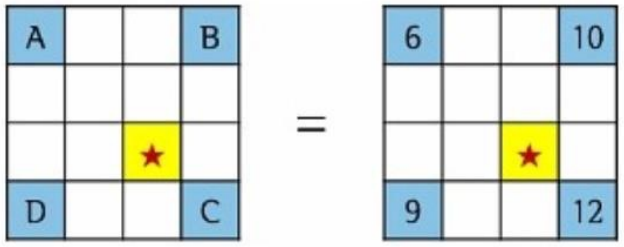
ROI-Align

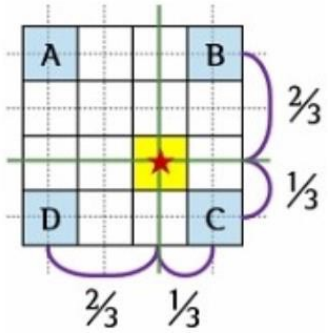


- ROI를 소수점 그대로 매핑하고 ROI의 개별 Grid 에 4개의 point를 균등하게 배열
- 개별 point에서 가장 가까운 feature map Grid를 고려하면서 포인트를 Weighted Sum 으로 계산.
- 계산된 포인트를 기반으로 Max Pooling 수행.

Bilinear Interpolation

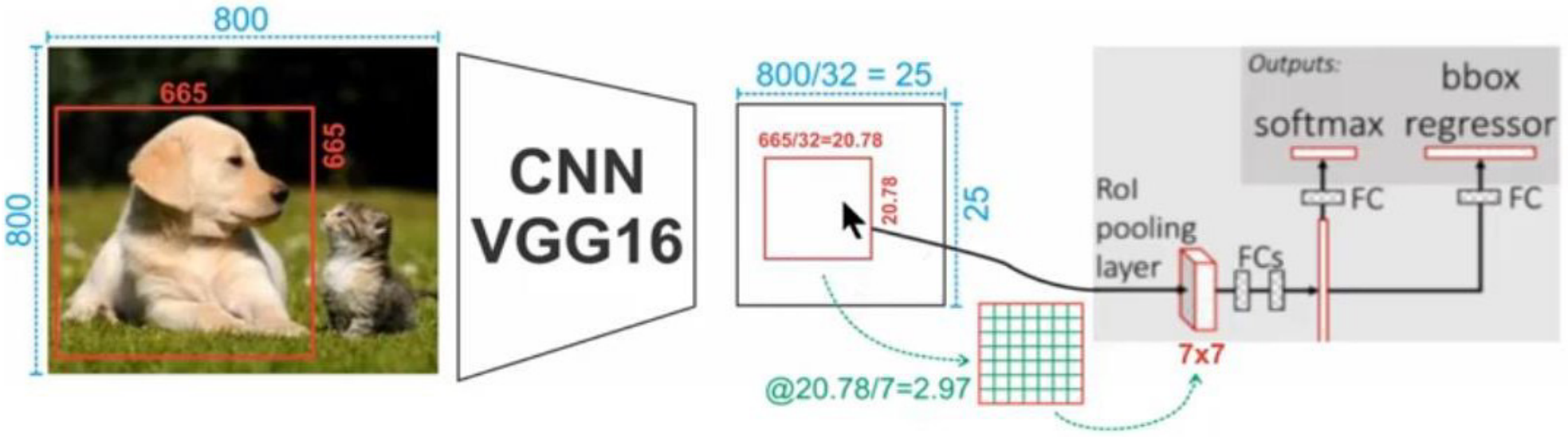






6	7.33	8.67	10
7	8.22	9.44	10.67
8	9.11	10.22	11.33
9	10	11	12

ROI-Align 적용


800
665
665
800
CNN
VGG16
800/32 = 25
665/32=20.78
20.78
25
@20.78/7=2.97
Outputs:
softmax
bbox
regressor
FC
FC
RoI
pooling
layer
FCs
7x7

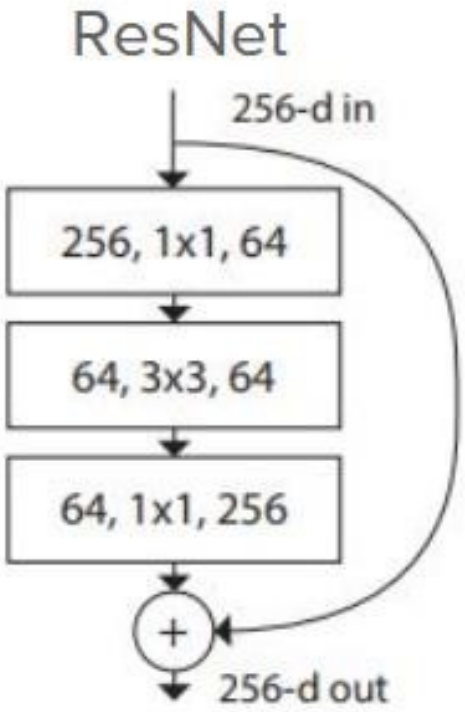
ROI-Align 적용 후 성능 향상

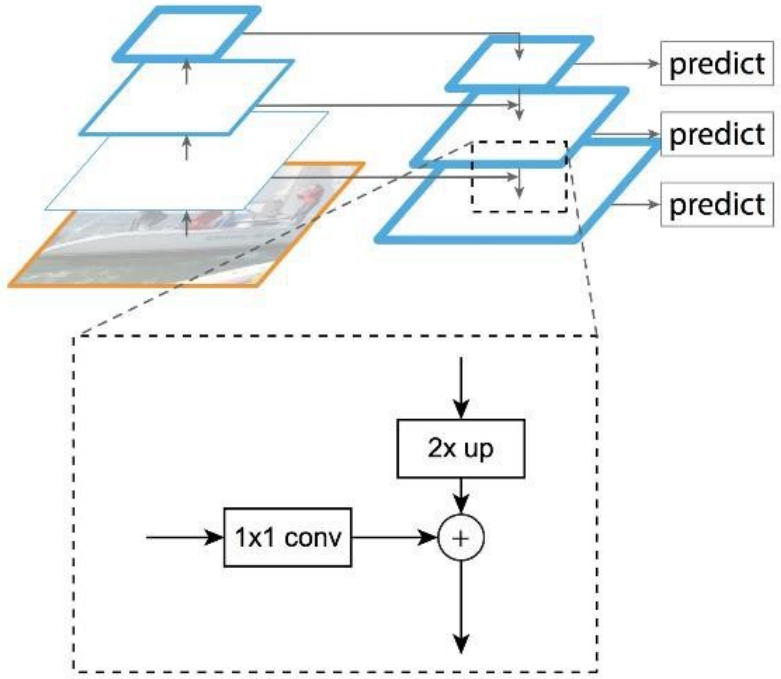
ROI Pooling 대비 약 3~4 % AP 성능 향상

	align?	bilinear?	agg.	AP	AP_{50}	AP_{75}
RoIPool [12]			max	26.9	48.8	26.4
RoIWarp [10]		✓	max	27.2	49.2	27.1
		✓	ave	27.1	48.9	27.1
RoIAlign	✓	✓	max	**30.2**	**51.0**	**31.8**
	✓	✓	ave	**30.3**	**51.2**	**31.5**

Feature Extractor

Resnet + Feature Pyramid Network





Mask RCNN Loss Function

Mask RCNN Loss

$$L = L_{cls} + L_{bbox} + L_{mask}$$

Classification Loss

Multiclass cross-entropy loss

Bounding box Loss

Smooth L1 loss

$$L_{\text{loc}}(t^u, v) = \sum_{i \in \{\text{x,y,w,h}\}} \text{smooth}_{L_1}(t_i^u - v_i),$$

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise,} \end{cases}$$

Mask Loss

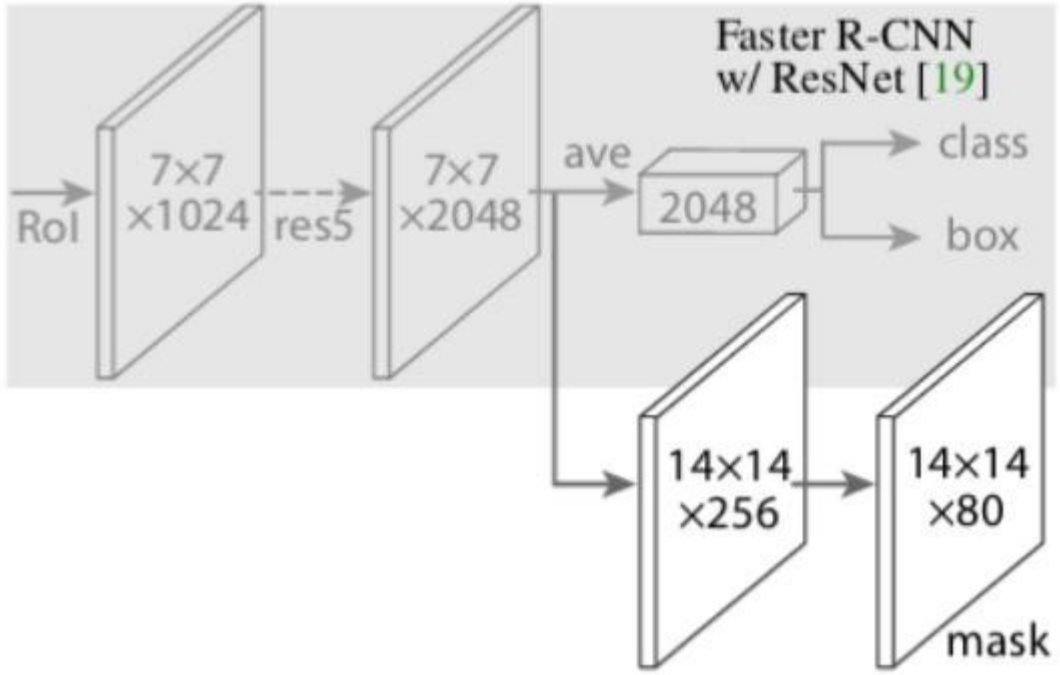
Binary cross-entropy loss

K개의 정해진 Class에 대해서 그 class에 pixel이 속하는지 그렇지 않는지 sigmoid로 결정

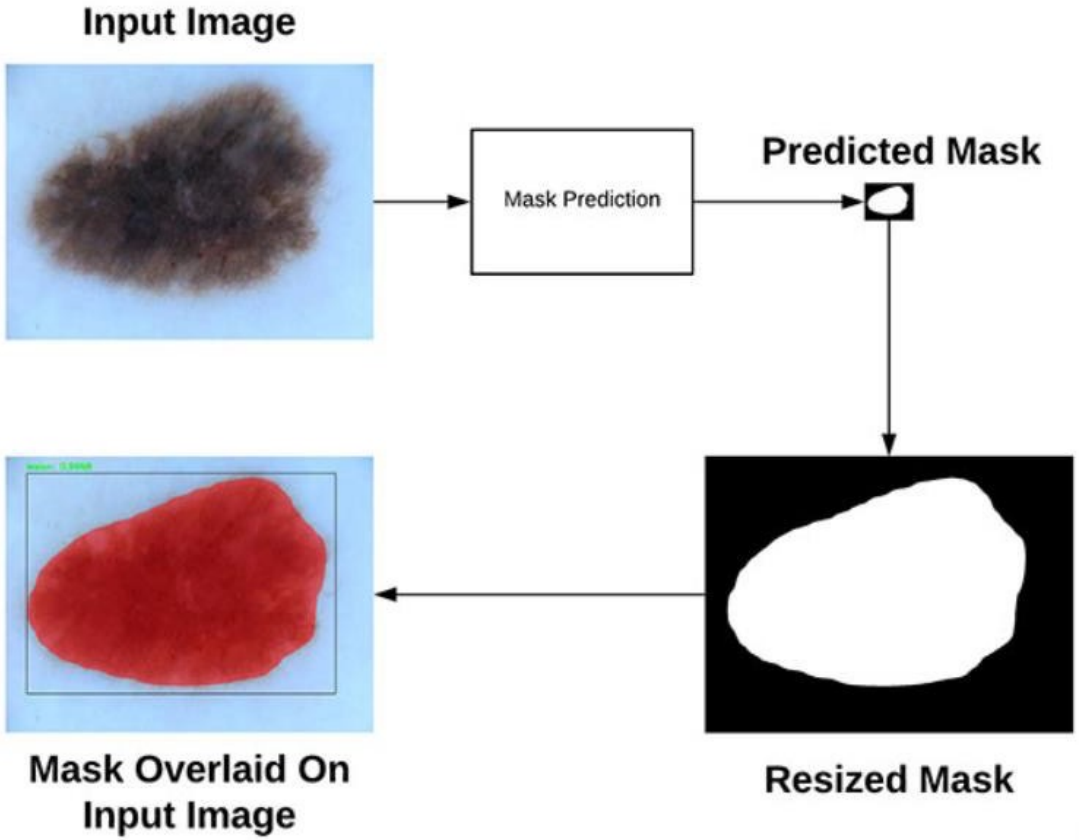
Mask Prediction

kubwa

Mask RCNN에서 Mask Prediction



Mask Prediction후 Resize하여 원본 이미지에 적용



Mask RCNN 성능 비교

MS-COCO Dataset 기준

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
MNC [10]	ResNet-101-C4	24.6	44.3	24.8	4.7	25.9	43.6
FCIS [26] +OHEM	ResNet-101-C5-dilated	29.2	49.5	-	7.1	31.3	50.0
FCIS+++ [26] +OHEM	ResNet-101-C5-dilated	33.6	54.5	-	-	-	-
Mask R-CNN	ResNet-101-C4	33.1	54.9	34.8	12.1	35.6	51.1
Mask R-CNN	ResNet-101-FPN	35.7	58.0	37.8	15.5	38.1	52.4
Mask R-CNN	ResNeXt-101-FPN	**37.1**	**60.0**	**39.4**	**16.9**	**39.9**	**53.5**

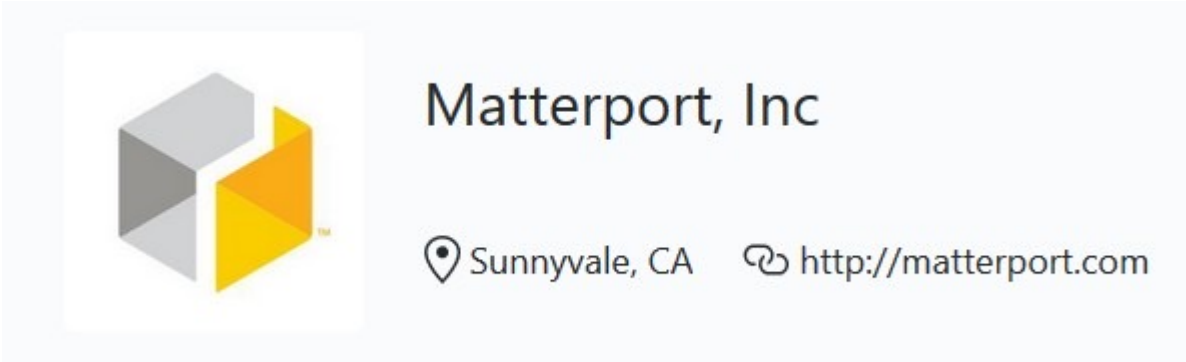
BIGDATA & AI ANALYTICS
EXPERT COMPANY

Matterport Mask RCNN

Matterport MASK RCNN 패키지

Mask RCNN을 위한 최적의 Open Source 패키지

https://github.com/matterport/Mask_RCNN



Matterport MASK RCNN 패키지 장단점

kubwa

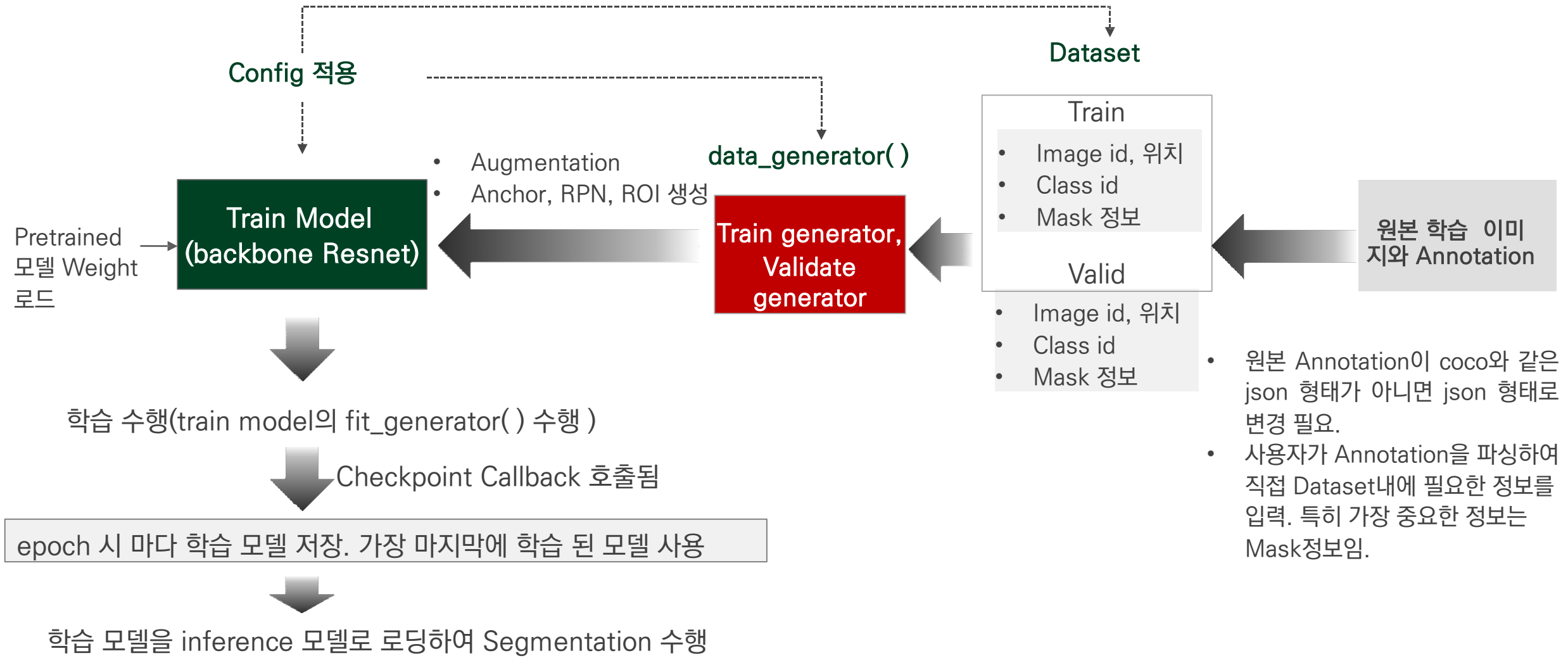
장점

- 최적화 된 Mask RCNN 알고리즘으로 보다 뛰어난 성능
- 전 세계에서 다양한 응용 분야에서 사용 중
- 다양한 편의 기능을 제공 및 이해를 위한 여러 샘플 예제 및 Document 제공
- 쉬운 Config 설정

단점

- Coco Dataset 포맷을 기반 데이터 Generator로 사용하므로 Custom Dataset을 Coco형태로 변환해야 함.
- 자체 시각화 기능이 matplotlib으로 되어 있어서 영상 Segmentation 시 customization이 필요.

Matterport Mask RCNN Train 프로세스



- 원본 Annotation이 coco와 같은 json 형태가 아니면 json 형태로 변경 필요.
- 사용자가 Annotation을 파싱하여 직접 Dataset내에 필요한 정보를 입력. 특히 가장 중요한 정보는 Mask정보임.

Train 코드

kubwa

utils.Dataset

```
self._image_ids = [ ]
self.image_info = [ ]
self.class_info = [{……}]
self.source_class_ids = {}
```

def prepare(self, class_map=None):

def load_mask(self, image_id):

```
…..
mask = np.empty([0, 0, 0])
class_ids = np.empty([0], np.int32)
return mask, class_ids
```

BalloonDataset(utils.Dataset)

Json 형태의 annotion을 parsing하여 image 정보, class정보등을 생성. (정보 parsing 후 self.add_image()로 Dataset에 등록)

def load_mask(self, image_id) 구현

- 개별 Dataset 객체는 Dataset 클래스를 상속 받아서 load_mask( )를 overloading 재 구현 해야함.
- 형식은 image_id로 입력 받은 단일 이미지의 mask 정보와 class id 정보를 return하는 형식으로 구현

Mask 정보를 통한 Bounding Box 좌표 계산

- Mask 정보를 이용하여 별도의 Bounding box 좌표 값이 제공되지 않아도
- Mask 정보를 이용하여 Bounding box 좌표값 계산이 가능.

Matterport Data Generation 프로세스

kubwa

1 Customized Dataset 객체 생성

mrcnn.utils.Dataset을 상속받아서 새로운 Dataset 객체와 메소드 작성

1.load_balloon(self, dataset_dir…): # 메소드 명 및 인자는 자유롭게 가능

- self.add_class( ) 를 호출하여 Class id와 클래스 명을 생성하고 매핑
- self.add_image()를 호출하여 source(데이터세트의 고유명), image id, image id 별 개별 path 정보를 생성.

2.load_mask(self, image_id) : Dataset.load_mask( ) 메소드를 override 하여 작성함.

- Image_id 인자로 입력된 image에 해당하는 여러 개의 mask 정보들과, image에 해당하는 class id array 형태로 반환

3.image_reference( ) 메소드는 작성하지 않아도 무방

Matterport Train & Inference 프로세스

2 학습 및 Inference

mrcnn.utils.Dataset을 상속받아서 새로운 Dataset 객체와 메소드 작성

1.Config 설정

- BATCH_SIZE, BACKBONE, STEPS_PER_EPOCH등 결정

2.MaskRCNN객체로 기본 모델 생성.

3.COCO Weight로 MaskRCNN 모델 Weight 초기화하되 클래스 개수가 coco와 다를 경우 classifier Layer는 제외.

4.Model 학습 수행

5.학습 모델 중 가장 적은 Loss를 가지는 Weight 파일을 Inference 모델로 로딩하여 Segmentation 수행