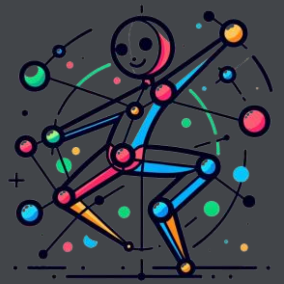
Pose Estimation

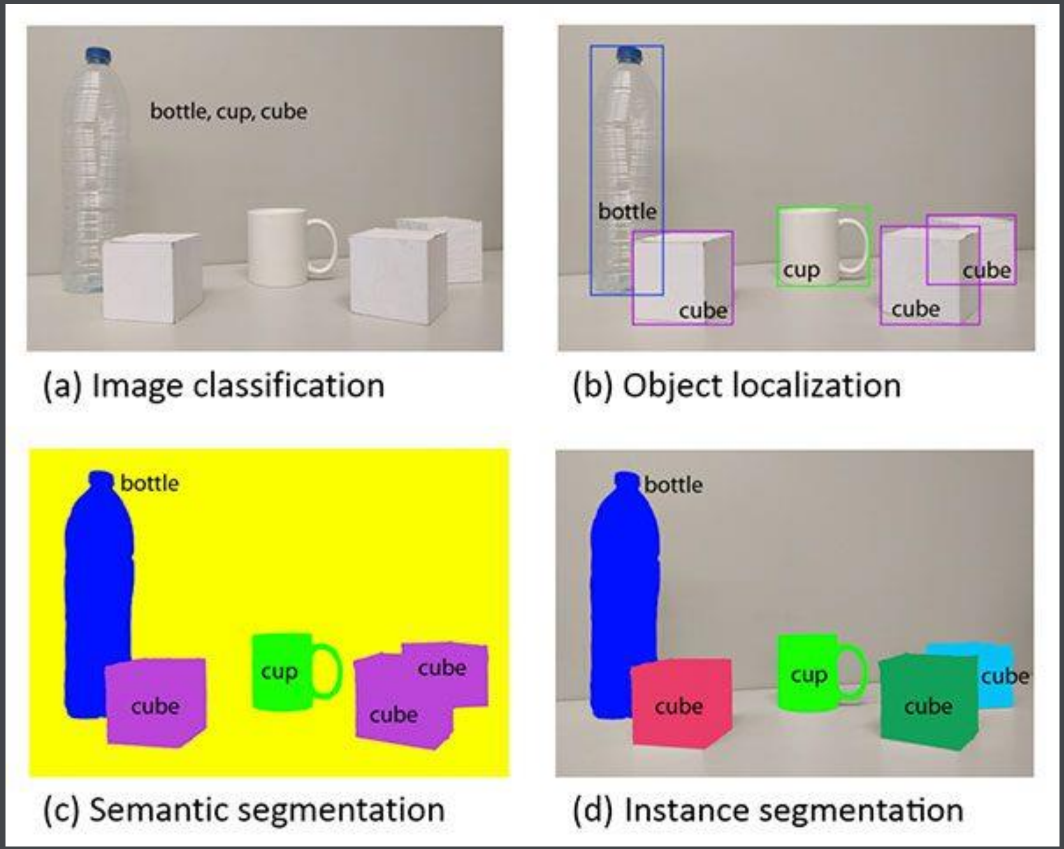
Mert Özmeral & Jerome Habanz



📃 Agenda

- 🚀 Einleitung
- 🏔️ Datensatz
- 🏗️ Architektur
- 🏋️ Training
- ✅ Testing
- ♾️ Fazit

🚀 Einleitung



(a) Image classification (b) Object localization

(c) Semantic segmentation (d) Instance segmentation

📃 Agenda

- 🚀 Einleitung
- 🏔️ Datensatz
- 🏗️ Architektur
- 🏋️ Training
- ✅ Testing
- ♾️ Fazit

Datensatz

COCO 2017

~ 118k Biler

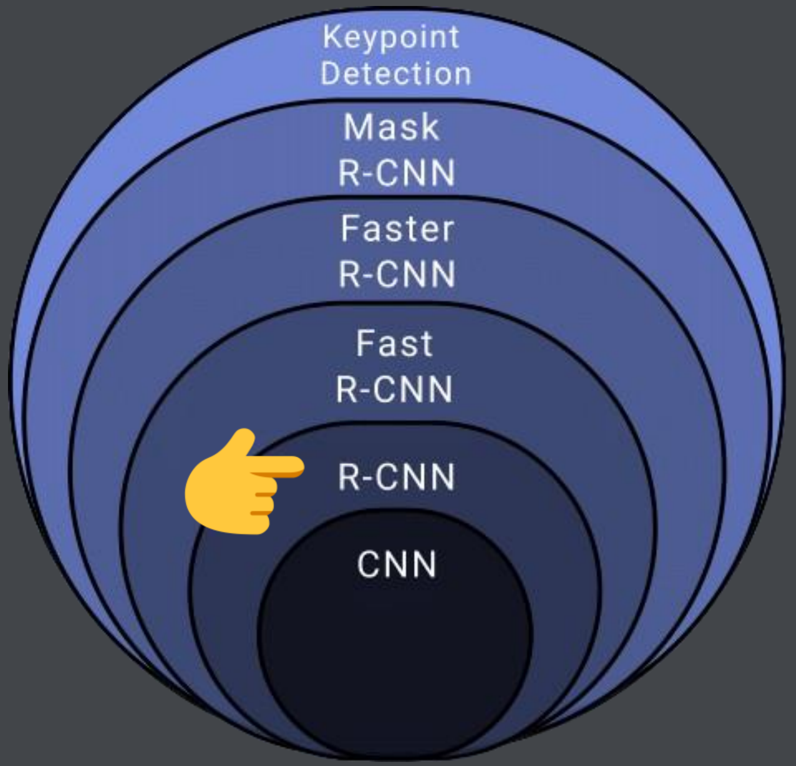
~ 250k Personen

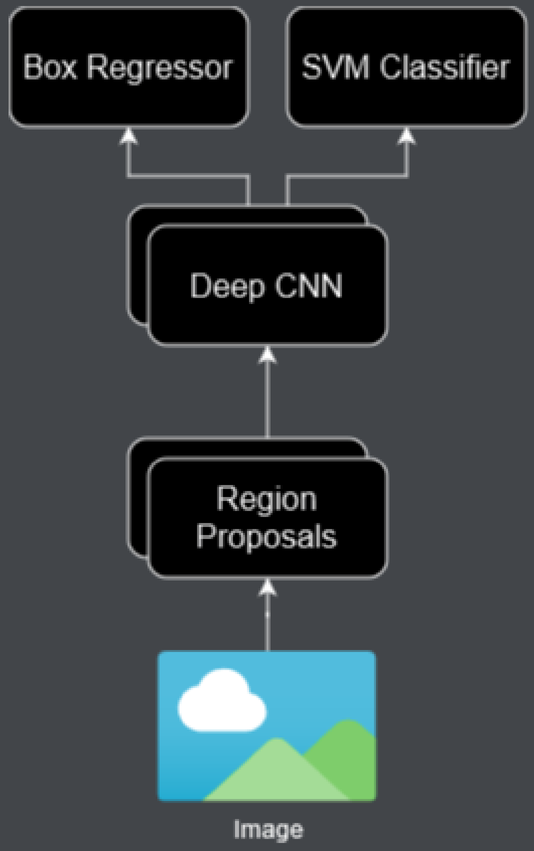
Training	➡	18 GB
Validation	➡	1 GB
Annotations	➡	241 MB

📃 Agenda

- 🚀 Einleitung
- 🏔️ Datensatz
- 🏗️ Architektur
- 🏋️ Training
- ✅ Testing
- ♾️ Fazit

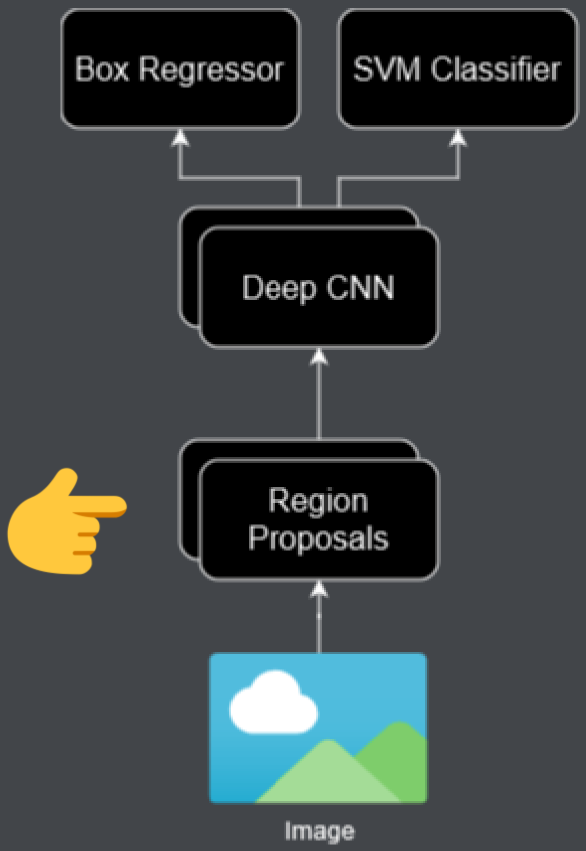
🏗 Architektur





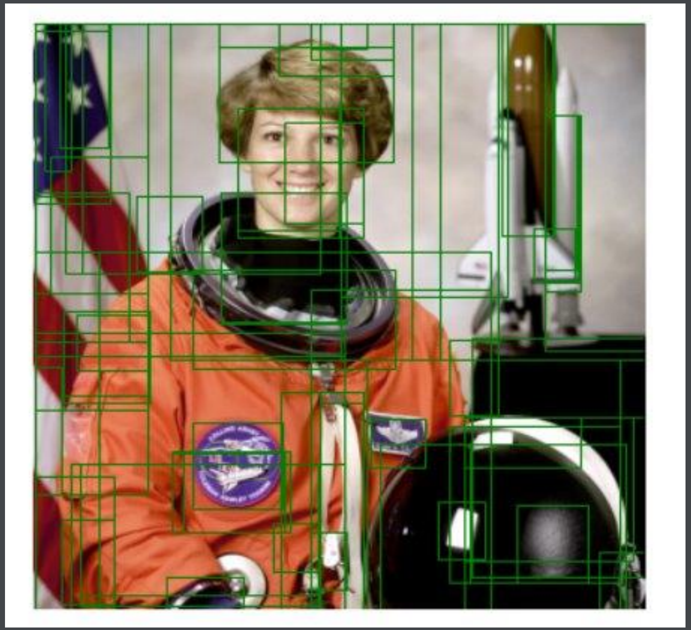
R-CNN

📦 Region Proposals

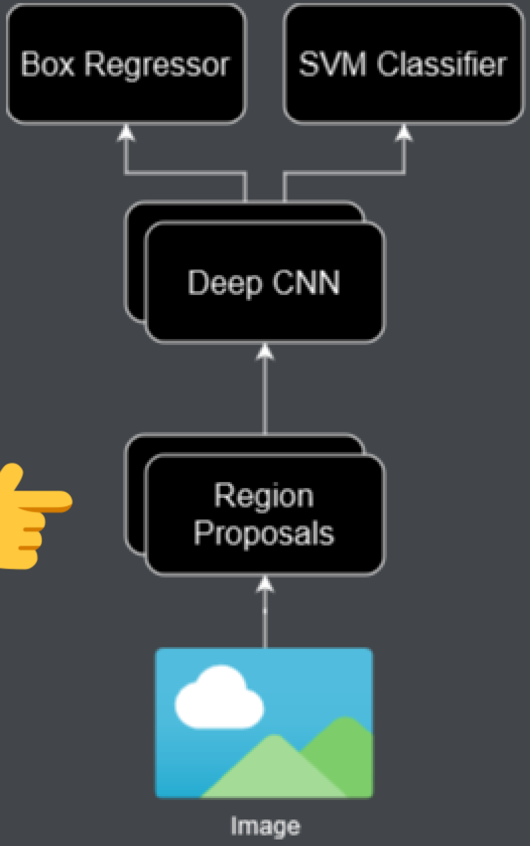


R-CNN

Selective Search

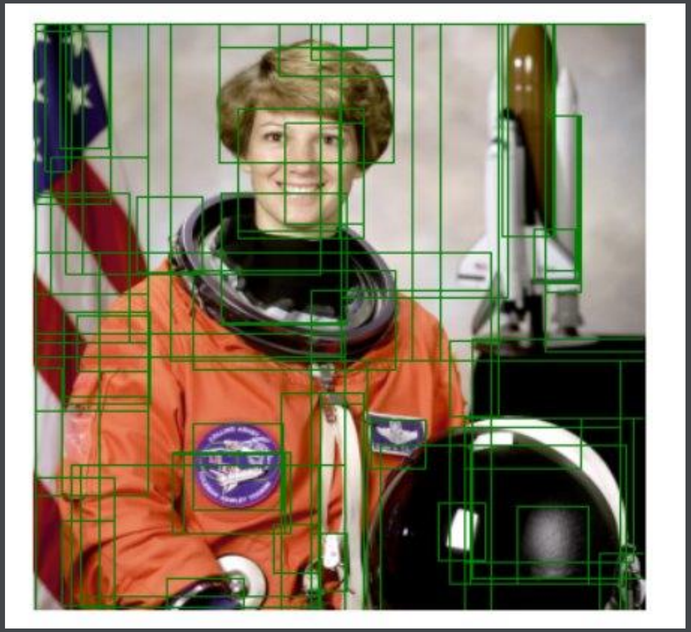


Region Proposals

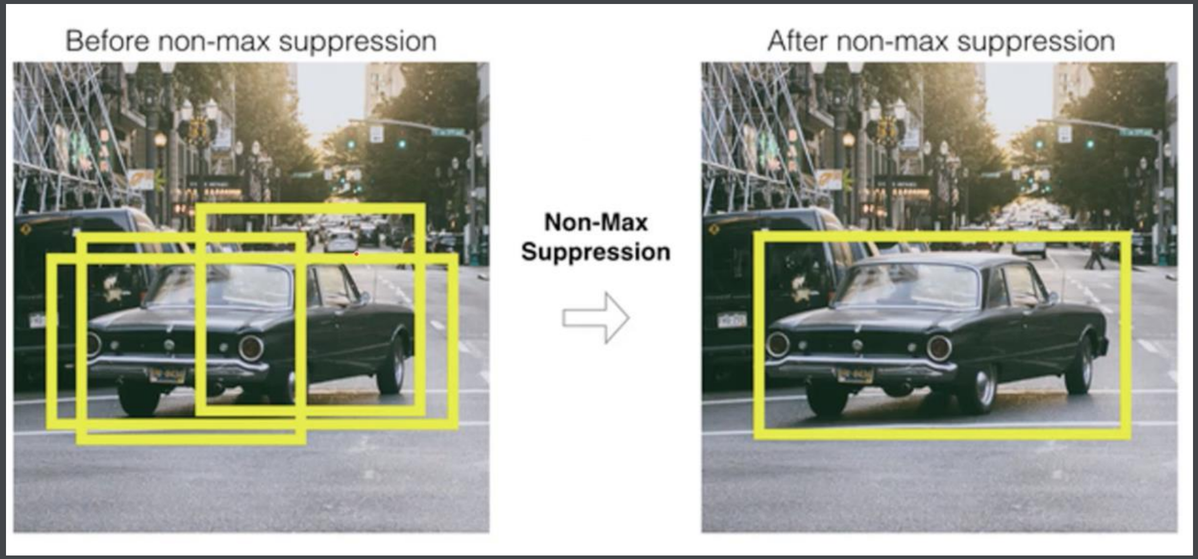


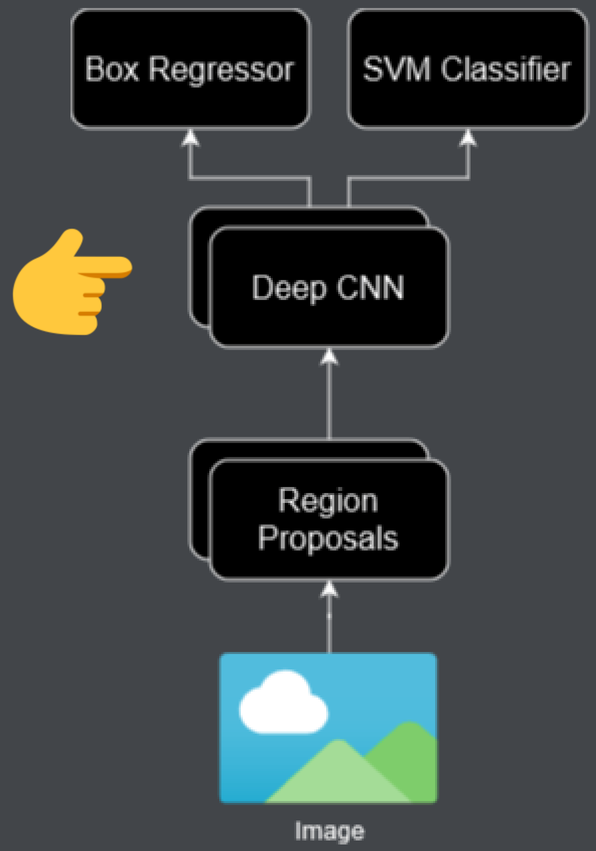
R-CNN

Selective Search

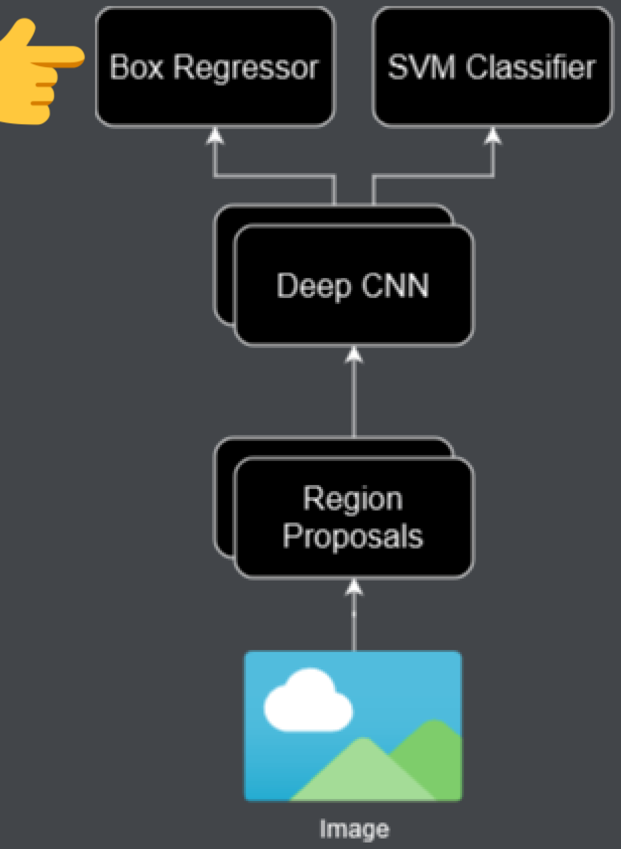


Non Maximum Suppression




Box Regressor
SVM Classifier
Deep CNN
Region Proposals
Image

R-CNN



R-CNN

R-CNN

Region proposal: (p_x, p_y, p_h, p_w)

Bbox

Transform: (t_x, t_y, t_h, t_w)

Output: (b_x, b_y, b_h, b_w)

Translation:

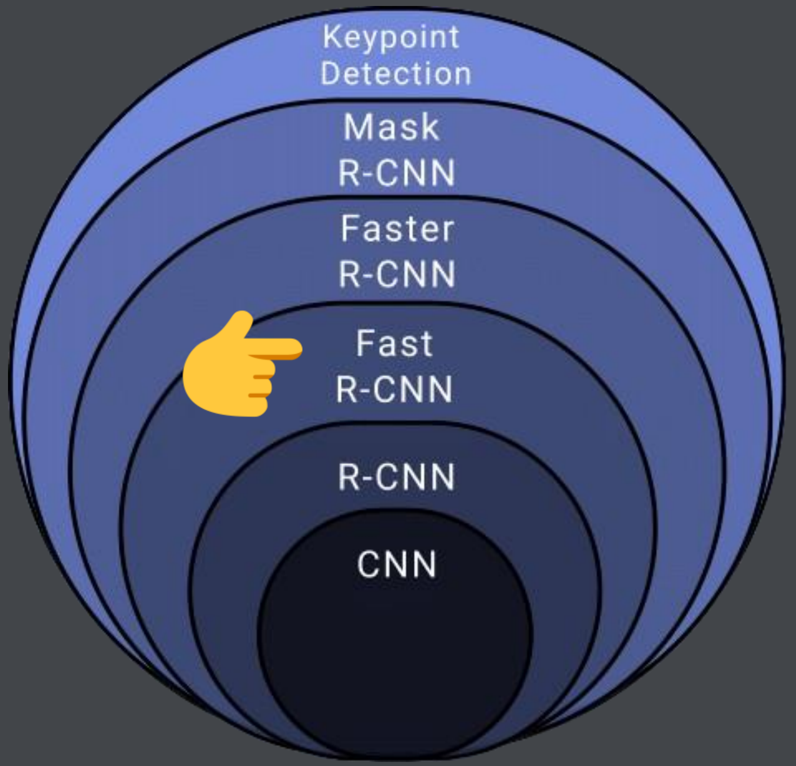
$b_x = p_x + p_w t_w$
(Horizontal translation)

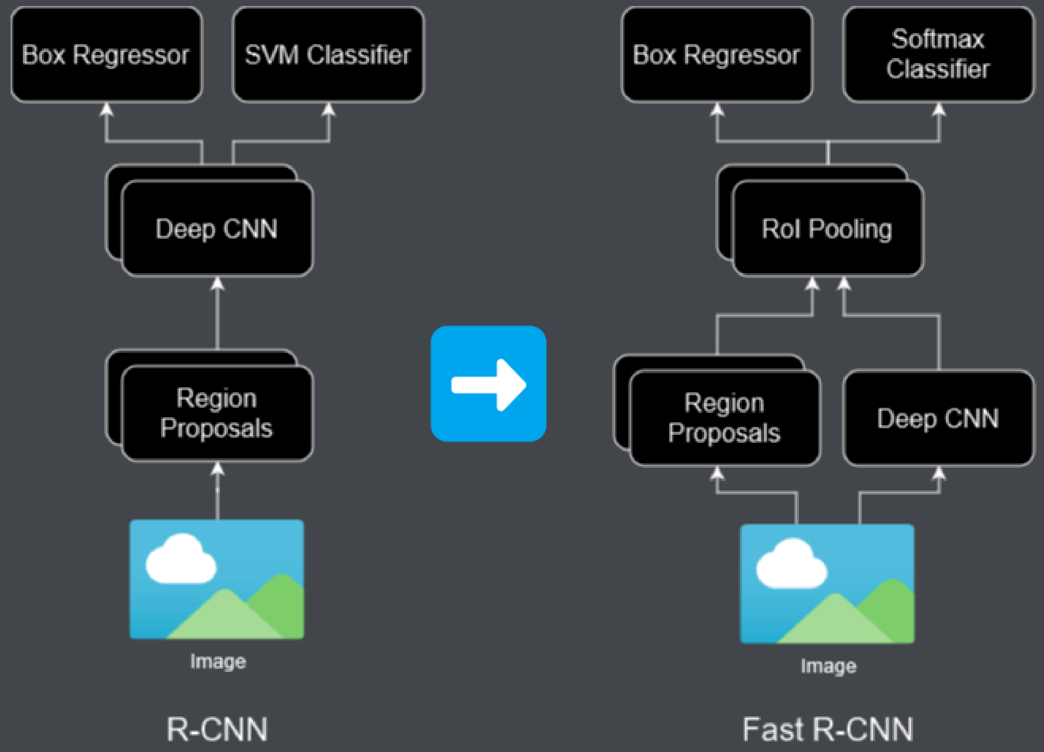
$b_y = p_y + p_h t_h$
(Vertical translation)

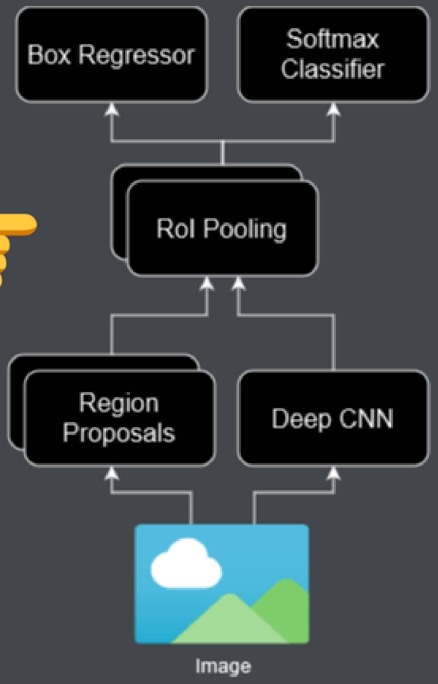
Log-space scale transform:

$b_w = p_w \exp(t_w)$
(Horizontal scale)

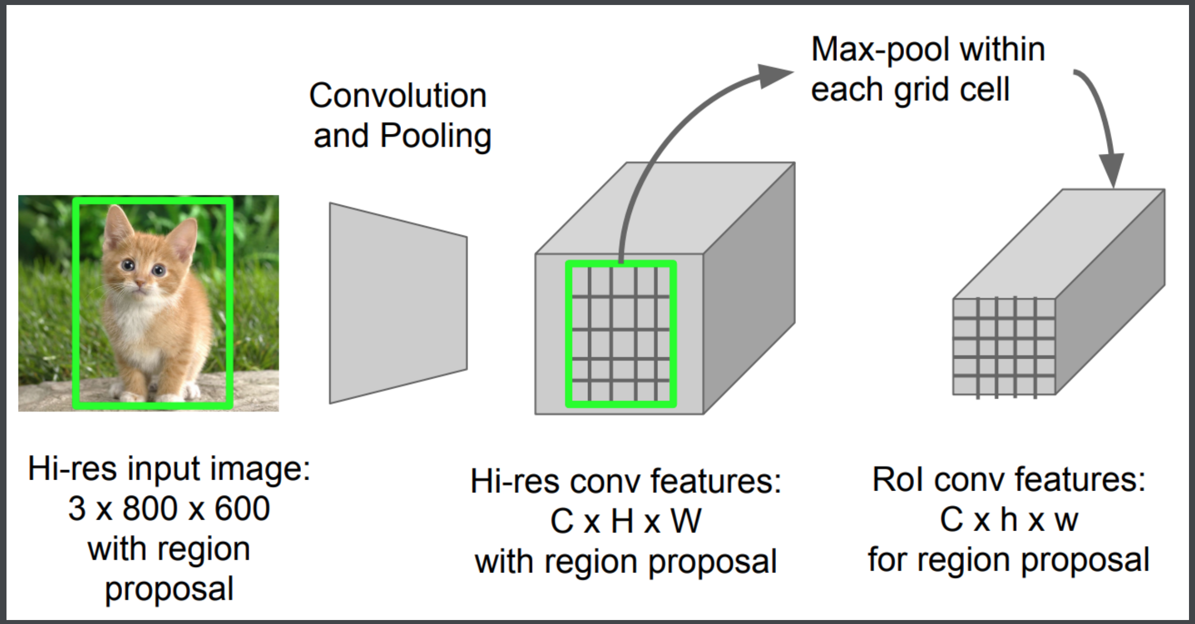
$b_h = p_h \exp(t_h)$
(Vertical scale)


Keypoint Detection
Mask R-CNN
Faster R-CNN
Fast R-CNN
R-CNN
CNN


Box Regressor
SVM Classifier
Deep CNN
Region Proposals
Image
R-CNN
Box Regressor
Softmax Classifier
RoI Pooling
Region Proposals
Deep CNN
Image
Fast R-CNN


Box Regressor
Softmax Classifier
RoI Pooling
Region Proposals
Deep CNN
Image

Fast R-CNN


Convolution and Pooling
Max-pool within each grid cell
Hi-res input image: 3 x 800 x 600 with region proposal
Hi-res conv features: C x H x W with region proposal
RoI conv features: C x h x w for region proposal

Singular Value Decomposition (SVD)

$$W_{m\times n} = \begin{bmatrix} \vec{w}_1 & \vec{w}_2 & \dots & \vec{w}_n \end{bmatrix} = U_{m\times m}\,\Sigma_{m\times n}\,V^T_{n\times n}$$

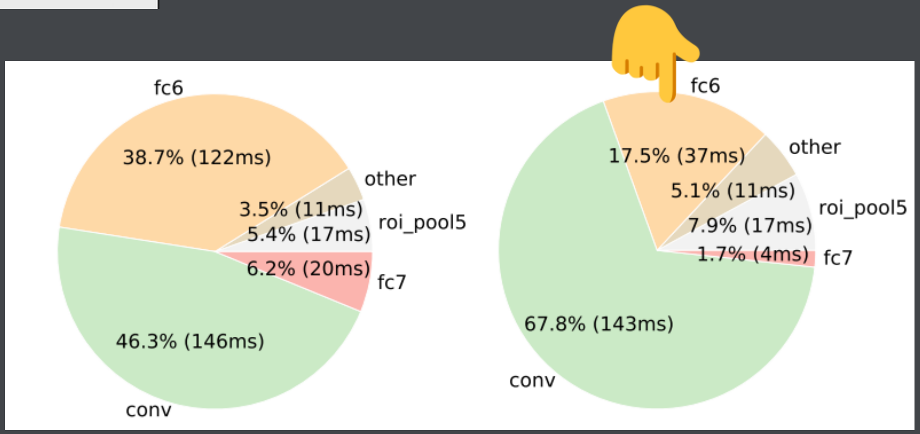
$$\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n$$

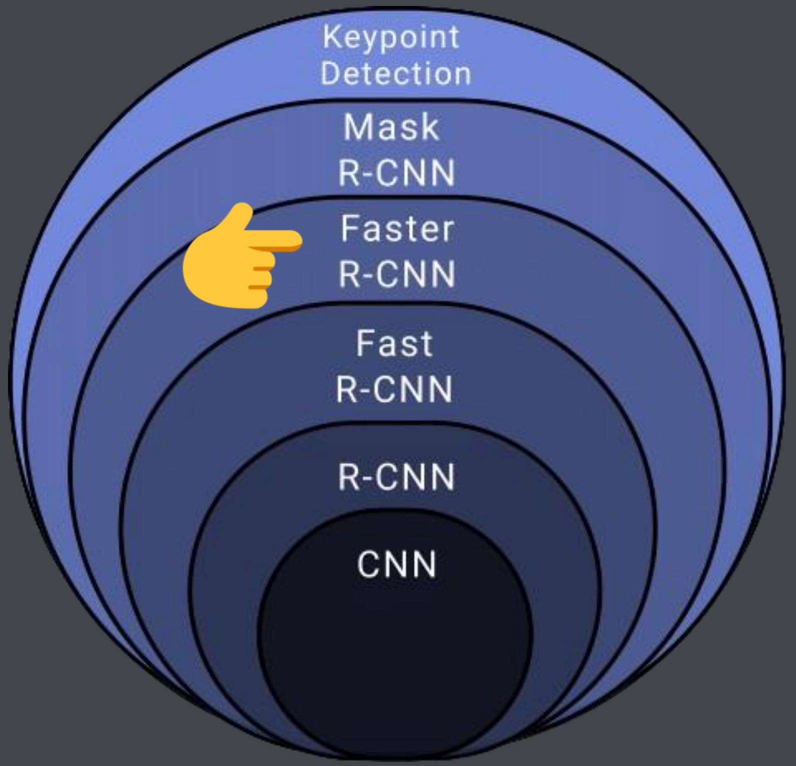
$$= \begin{bmatrix} \vec{u}_1 & \vec{u}_2 & \dots & \vec{u}_m \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 & 0 \\ \vdots & 0 & \ddots & 0 & 0 \\ 0 & 0 & 0 & \sigma_n & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \vec{v}_1 & \vec{v}_2 & \dots & \vec{v}_n \end{bmatrix}^T$$

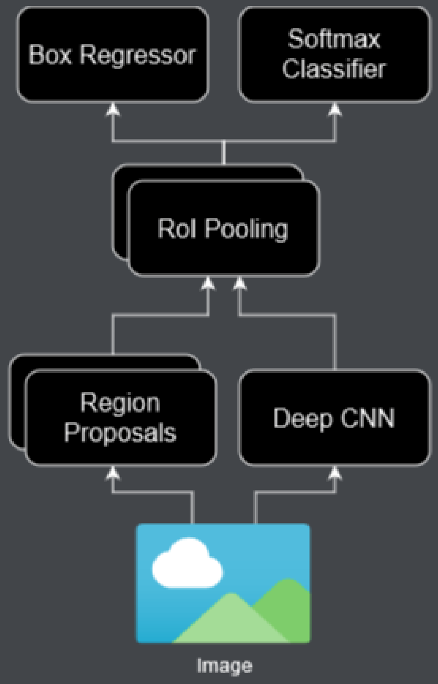
1. $m > n$ Coefficients of $\vec{u}_i$ where $i > n$ would be zero.
2. $m < n$ σ_i where $i > m$ would be zero.

Recall from Linear Algebra:

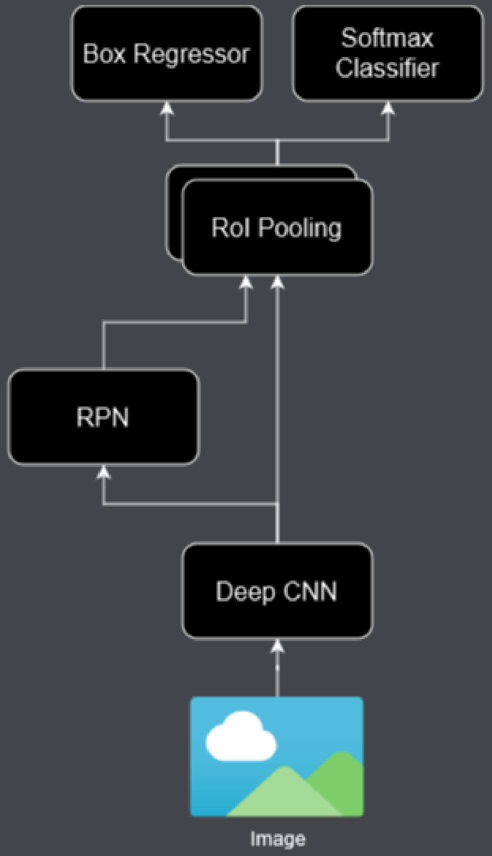
$$\begin{bmatrix} \vec{a}_1 & \vec{a}_2 & \dots & \vec{a}_n \end{bmatrix}_{m\times n} \begin{bmatrix} d_1 & 0 & 0 & 0 \\ 0 & d_2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & d_n \end{bmatrix}_{n\times n} = \begin{bmatrix} d_1\vec{a}_1 & d_2\vec{a}_2 & \dots & d_n\vec{a}_n \end{bmatrix}_{m\times n}$$



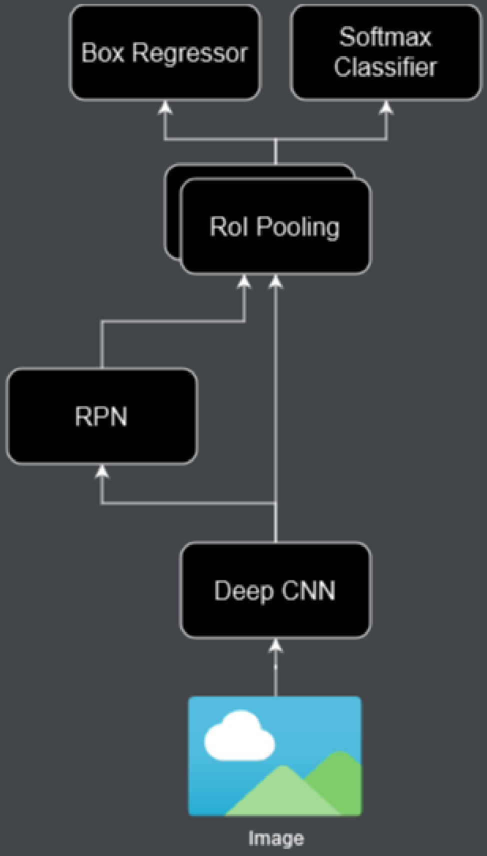




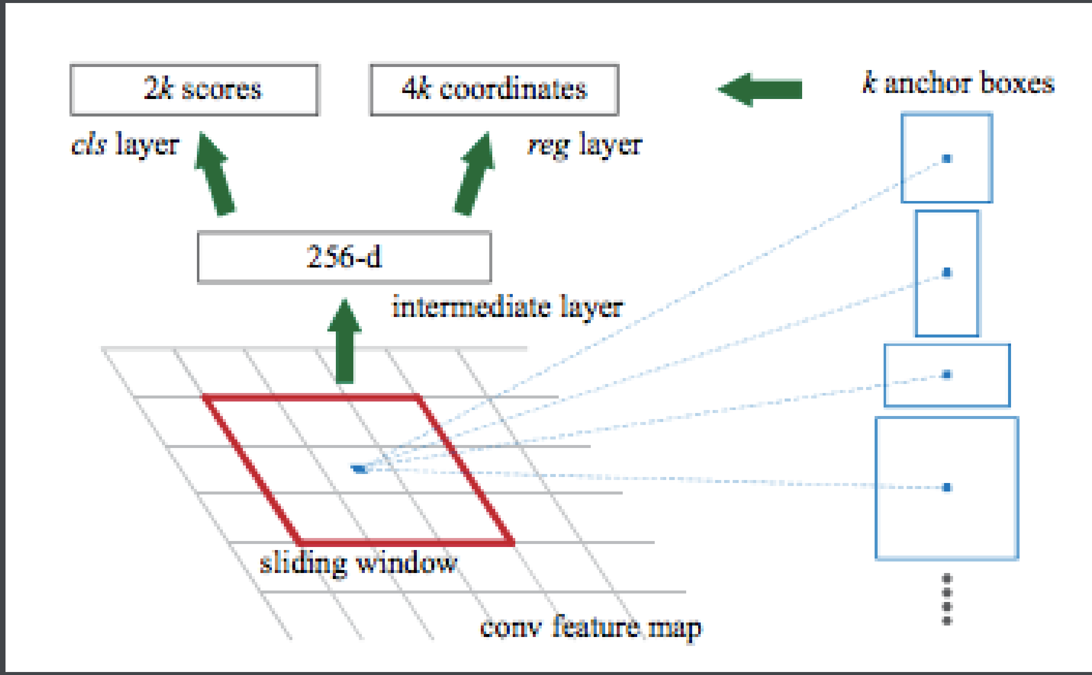
Fast R-CNN

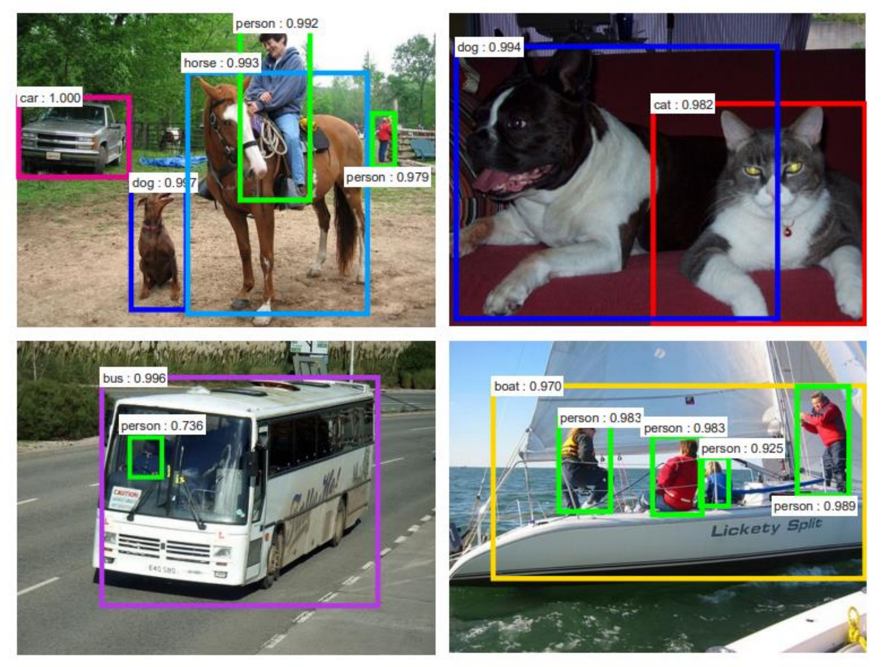


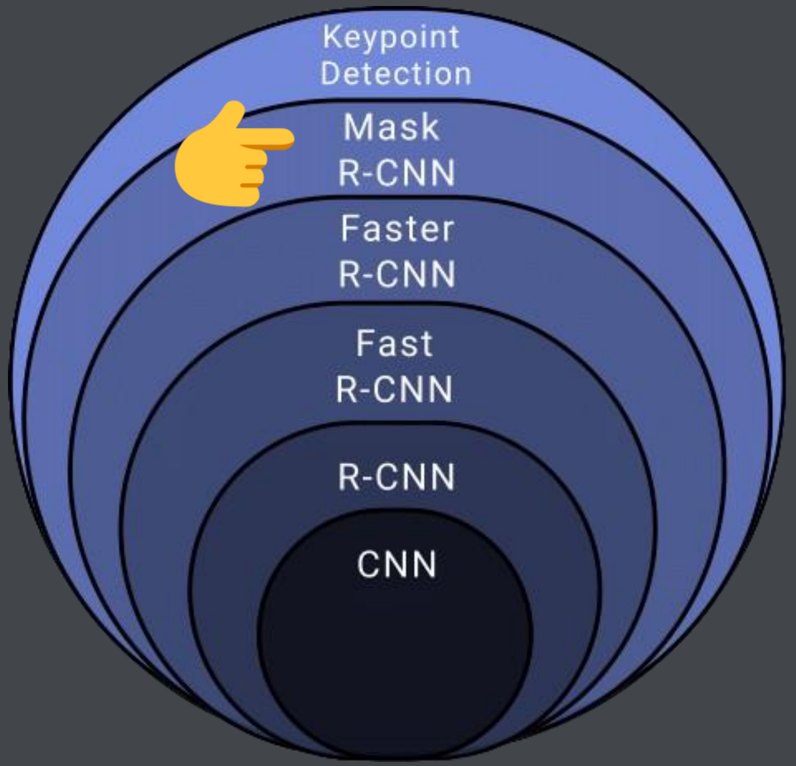
Faster R-CNN

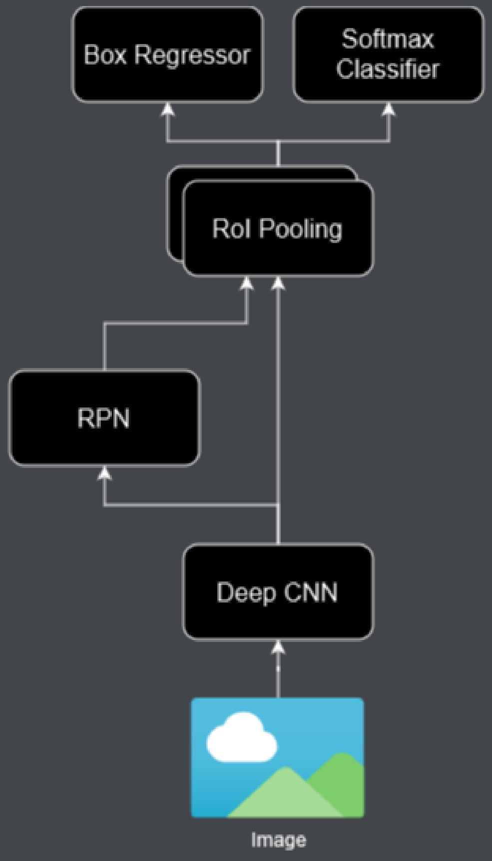

Box Regressor
Softmax Classifier
RoI Pooling
RPN
Deep CNN
Image

Faster R-CNN

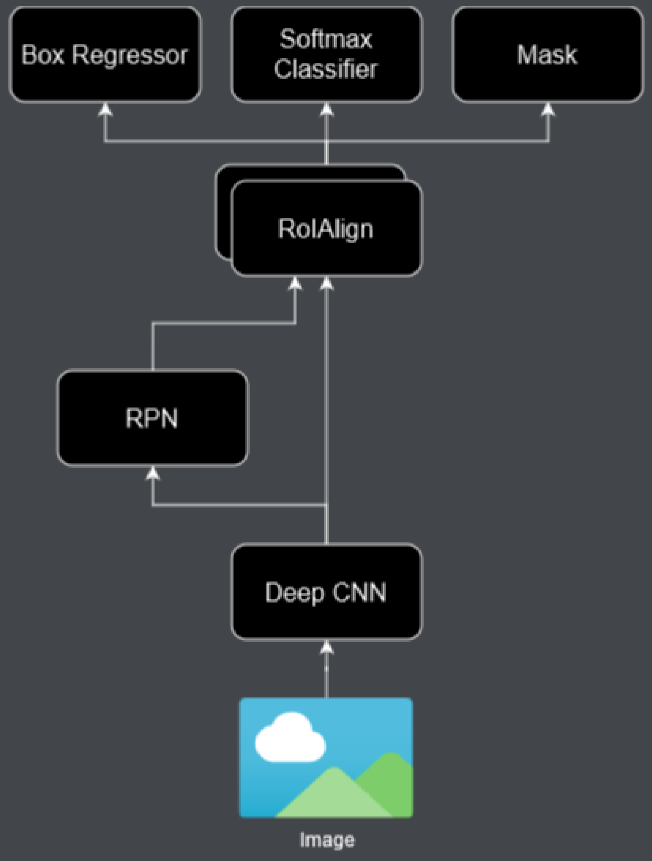

2k scores
4k coordinates
k anchor boxes
cls layer
reg layer
256-d
intermediate layer
sliding window
conv feature map


person : 0.992
horse : 0.993
car : 1.000
person : 0.979
dog : 0.994
cat : 0.982
bus : 0.996
person : 0.736
boat : 0.970
person : 0.983
person : 0.983
person : 0.925
person : 0.989
Lickety Split

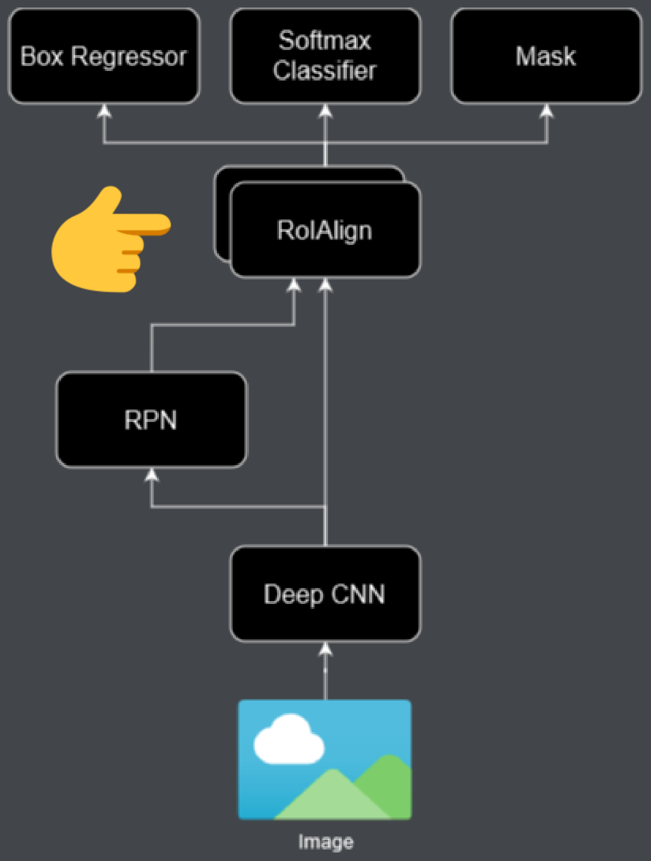




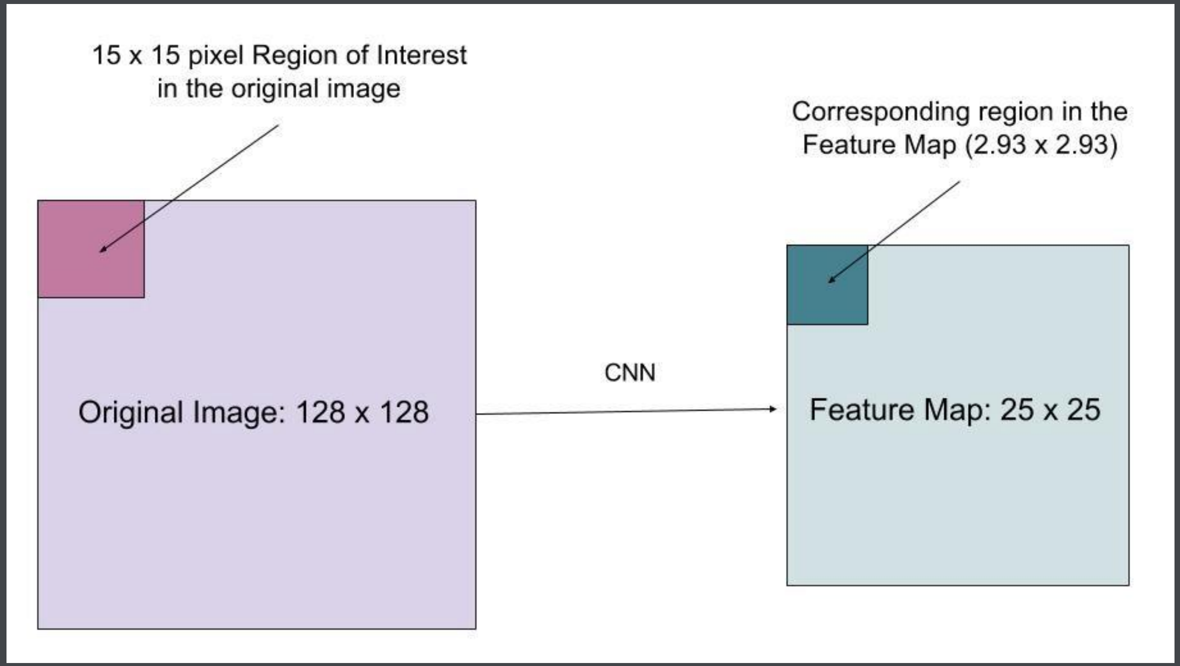
Faster R-CNN

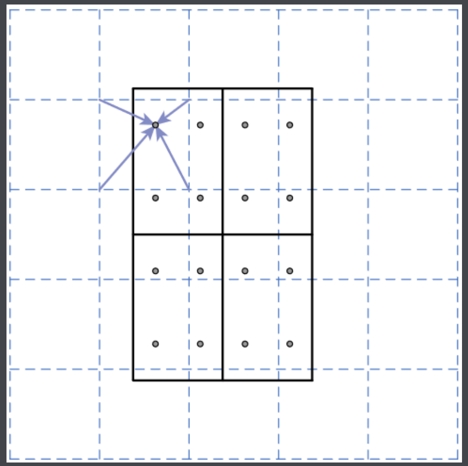


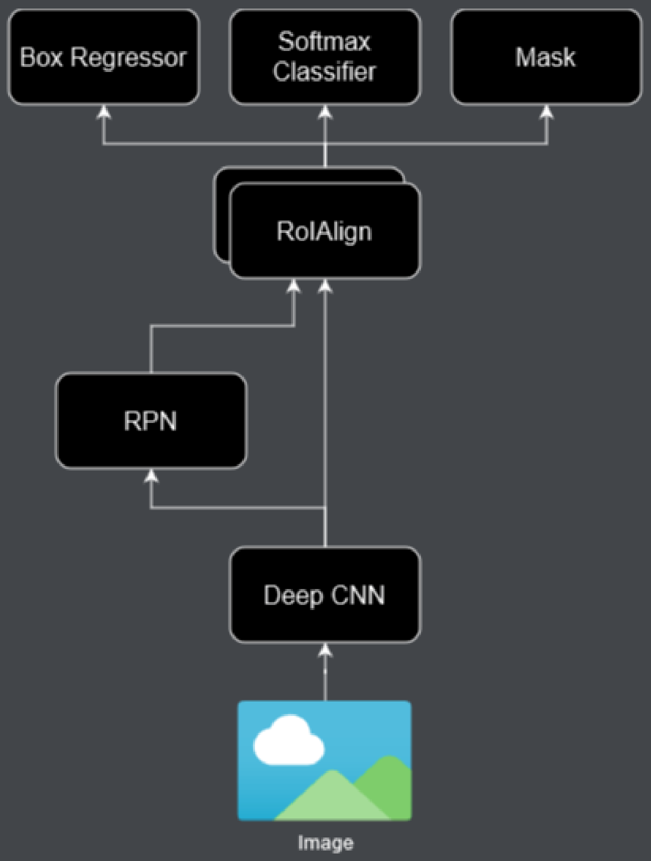
Mask R-CNN


Box Regressor
Softmax Classifier
Mask
RoIAlign
RPN
Deep CNN
Image

Mask R-CNN

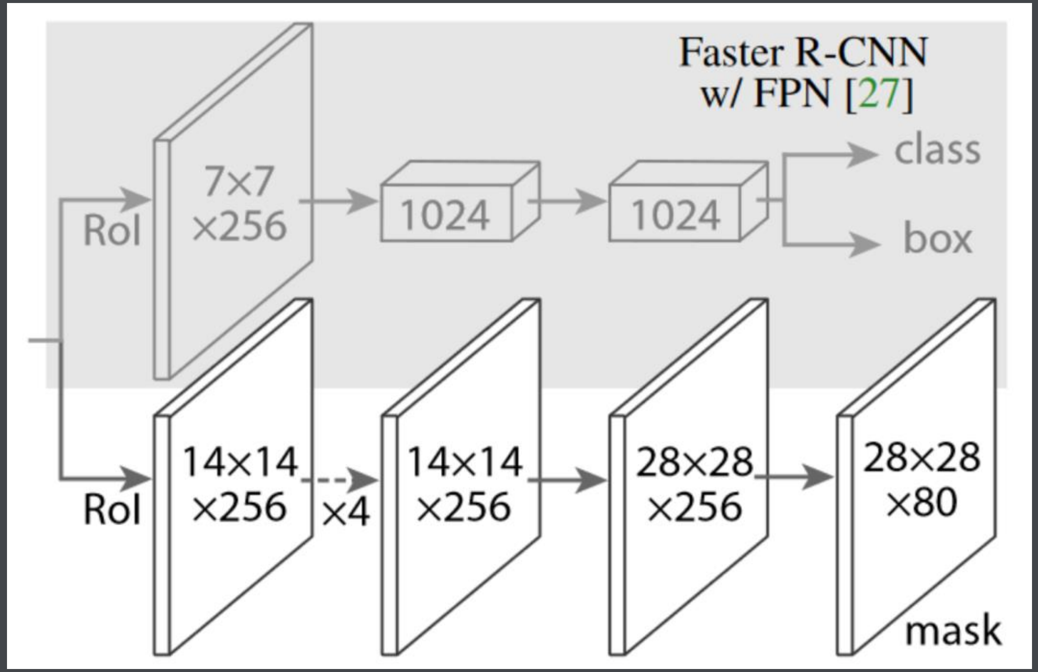

15 x 15 pixel Region of Interest in the original image
Corresponding region in the Feature Map (2.93 x 2.93)
Original Image: 128 x 128
CNN
Feature Map: 25 x 25

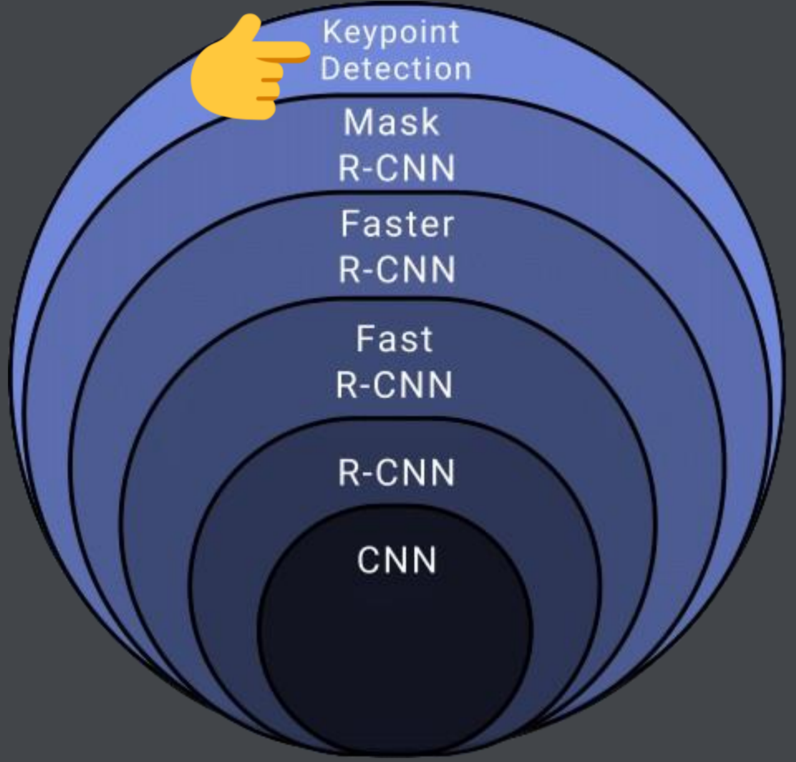


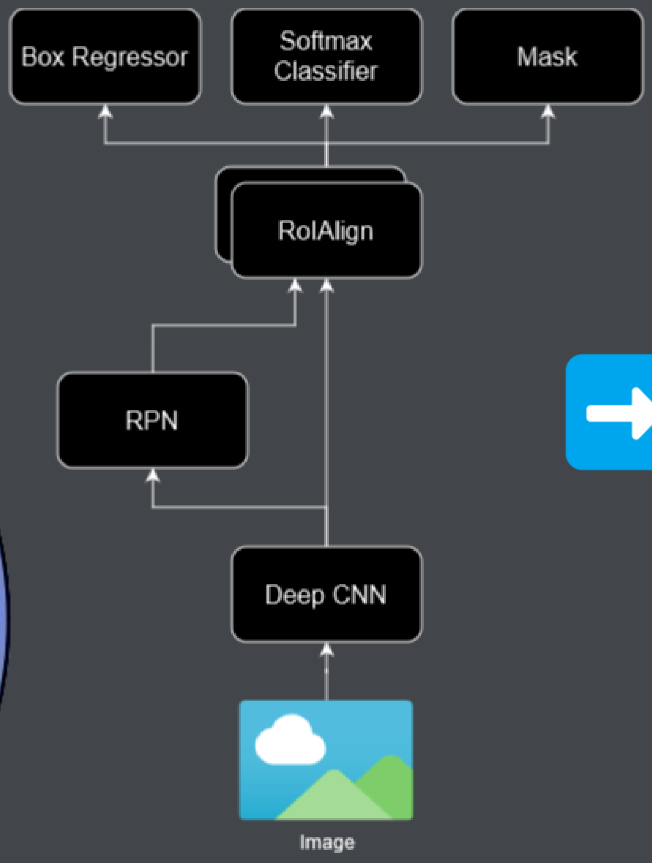

Box Regressor
Softmax Classifier
Mask
RoIAlign
RPN
Deep CNN
Image

Mask R-CNN

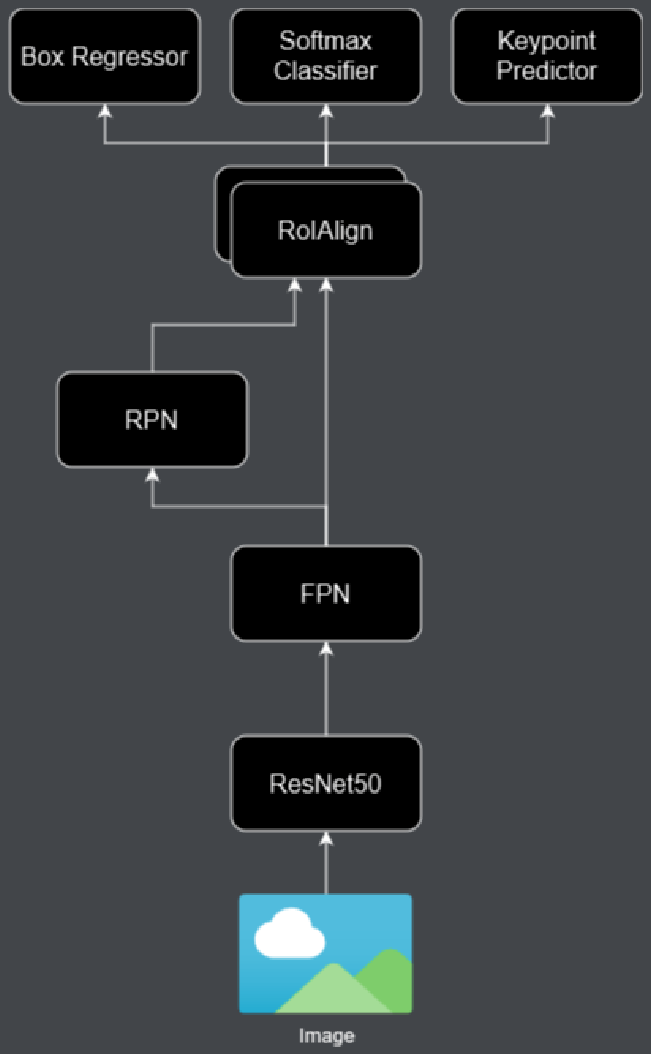

airplane 0.945
person 0.995
person 0.991
person 0.997
person 0.995


Faster R-CNN
w/ FPN [27]
RoI
7×7
×256
1024
1024
class
box
RoI
14×14
×256
×4
14×14
×256
28×28
×256
28×28
×80
mask

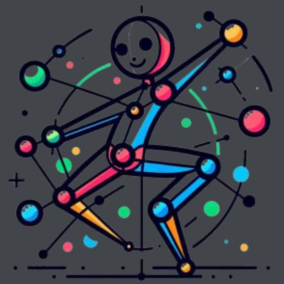


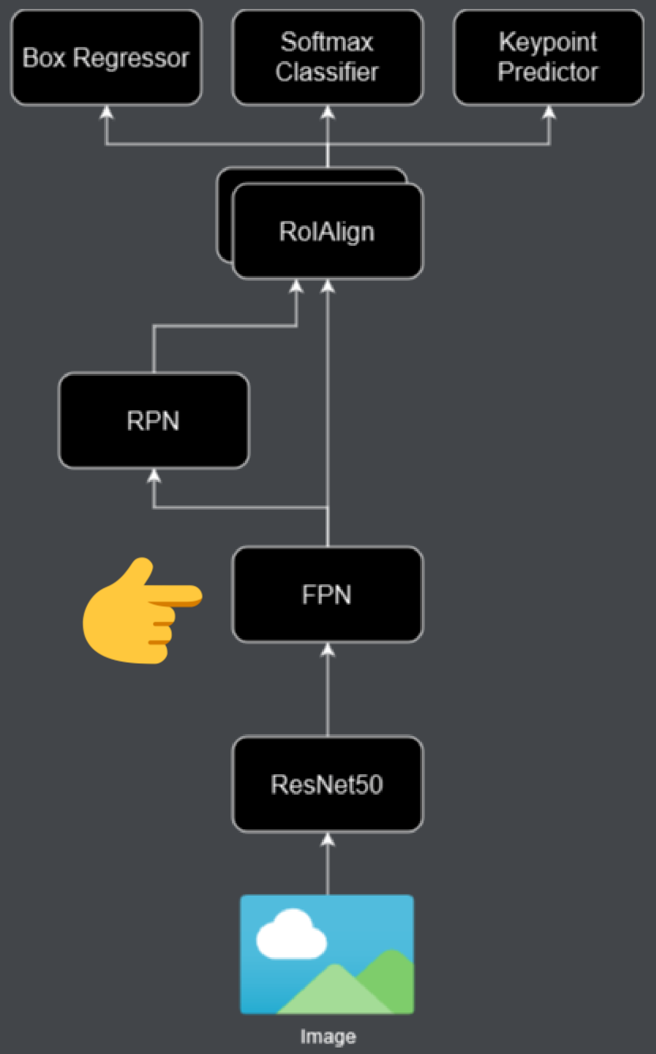


Mask R-CNN

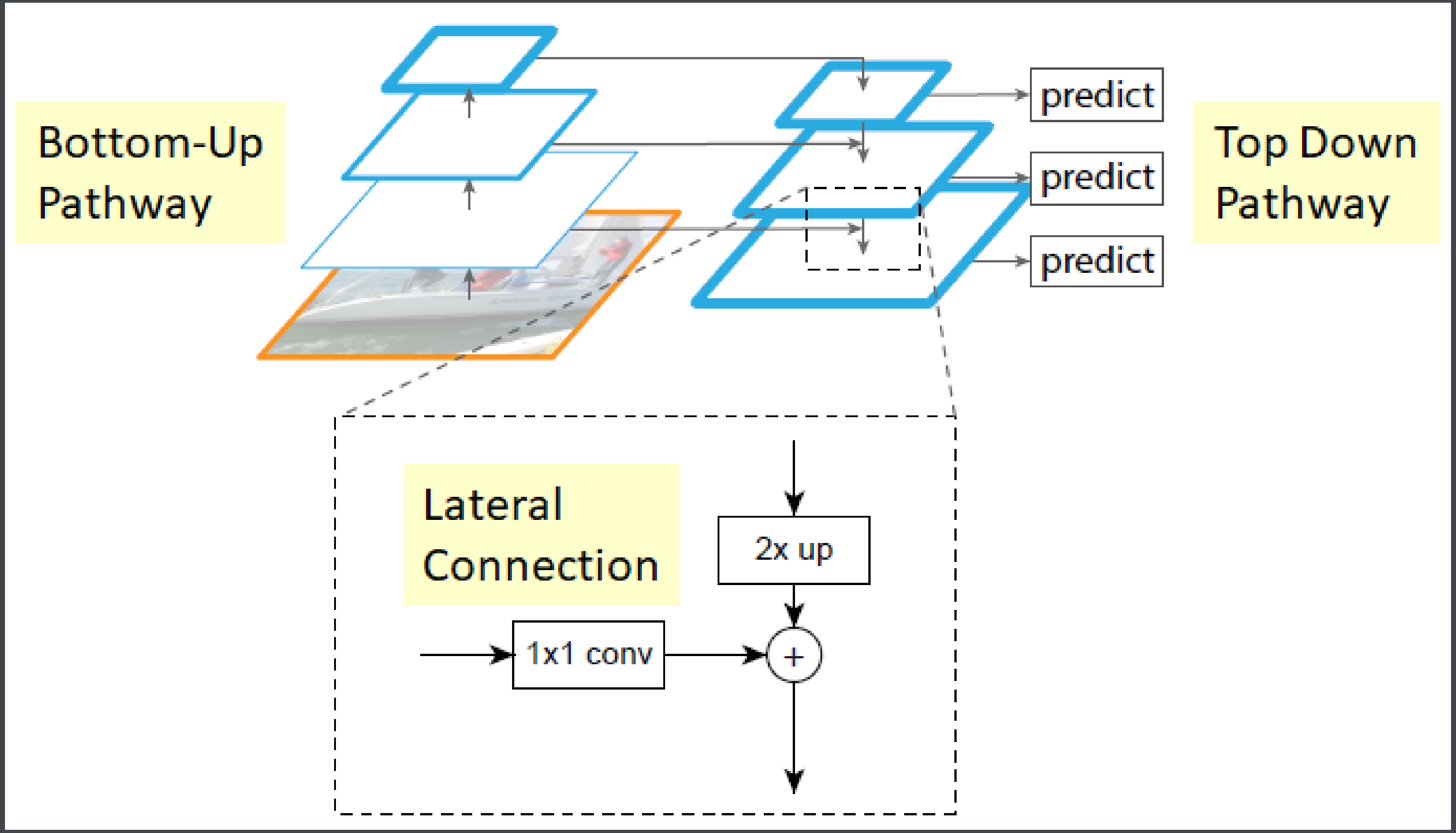


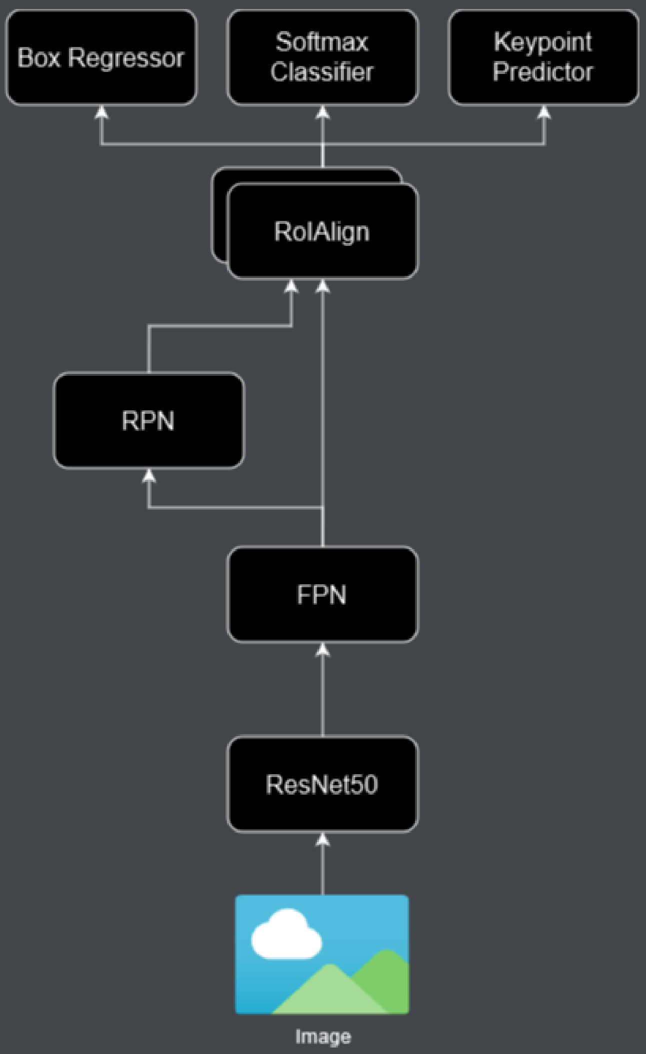
ResNet50 FPN R-CNN



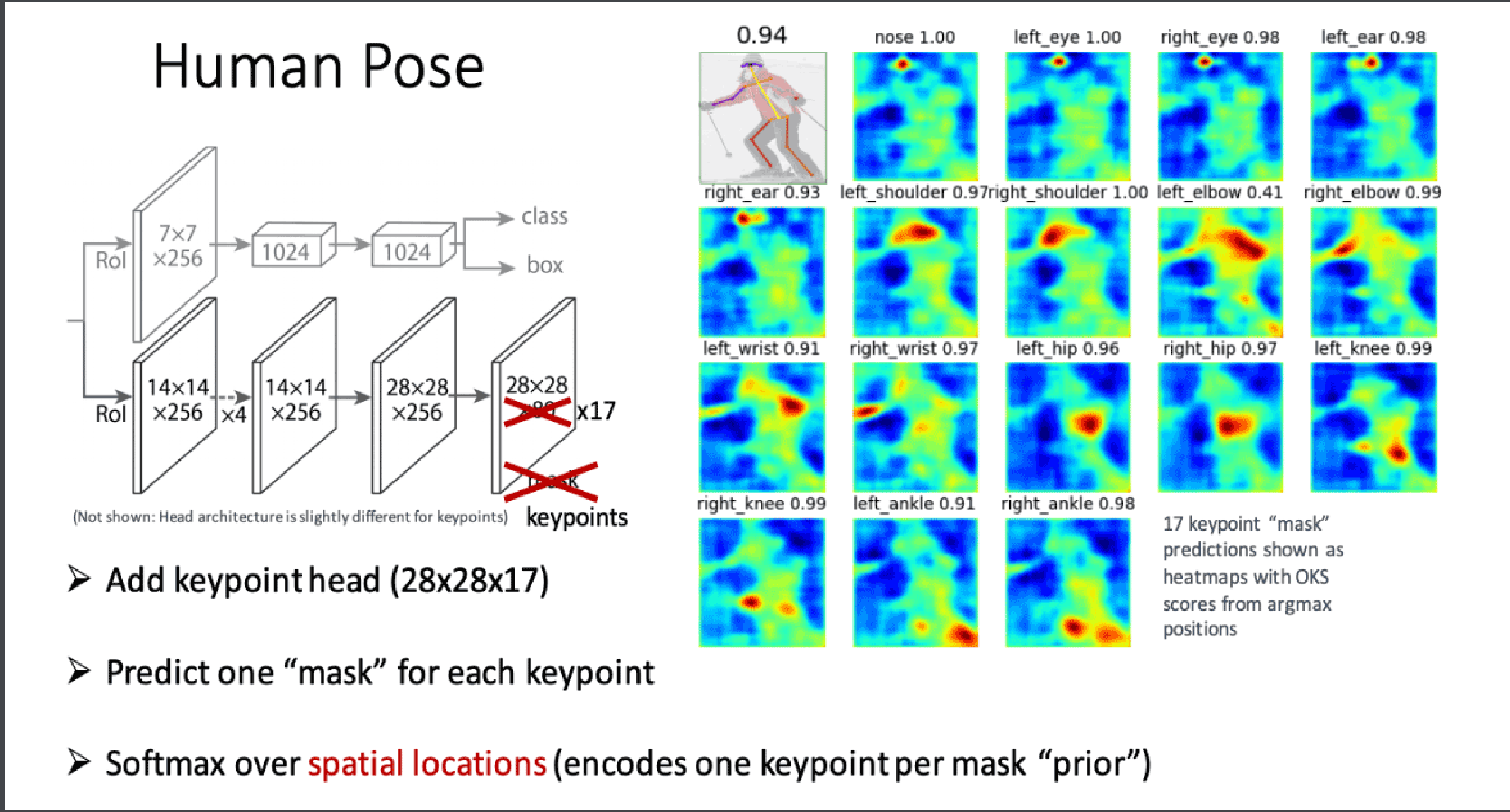


ResNet50 FPN R-CNN

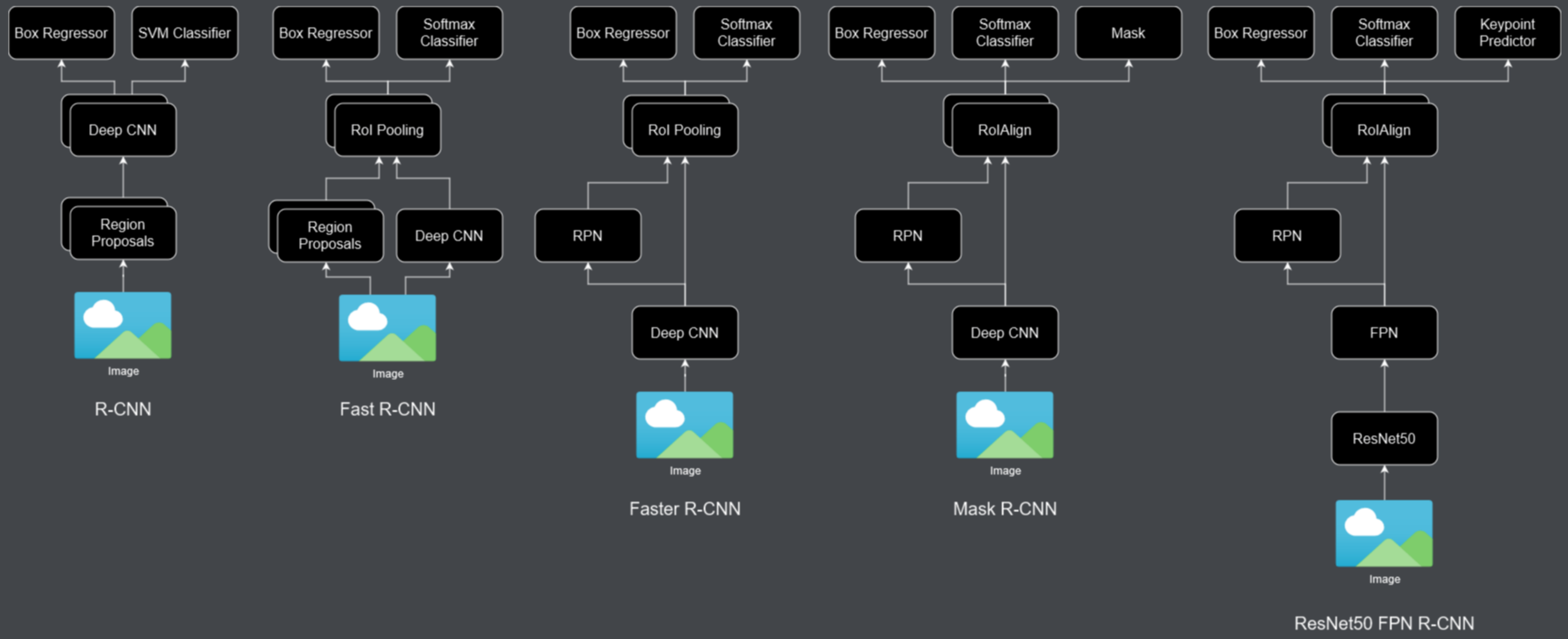




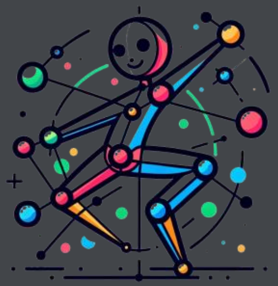
ResNet50 FPN R-CNN



Overview


Box Regressor
SVM Classifier
Deep CNN
Region Proposals
Image
R-CNN
Box Regressor
Softmax Classifier
RoI Pooling
Region Proposals
Deep CNN
Image
Fast R-CNN
Box Regressor
Softmax Classifier
RoI Pooling
RPN
Deep CNN
Image
Faster R-CNN
Box Regressor
Softmax Classifier
Mask
RoIAlign
RPN
Deep CNN
Image
Mask R-CNN
Box Regressor
Softmax Classifier
Keypoint Predictor
RoIAlign
RPN
FPN
ResNet50
Image
ResNet50 FPN R-CNN

📃 Agenda



- 🚀 Einleitung
- 🏔️ Datensatz
- 🏗️ Architektur
- 🏋️ Training
- ✅ Testing
- ♾️ Fazit

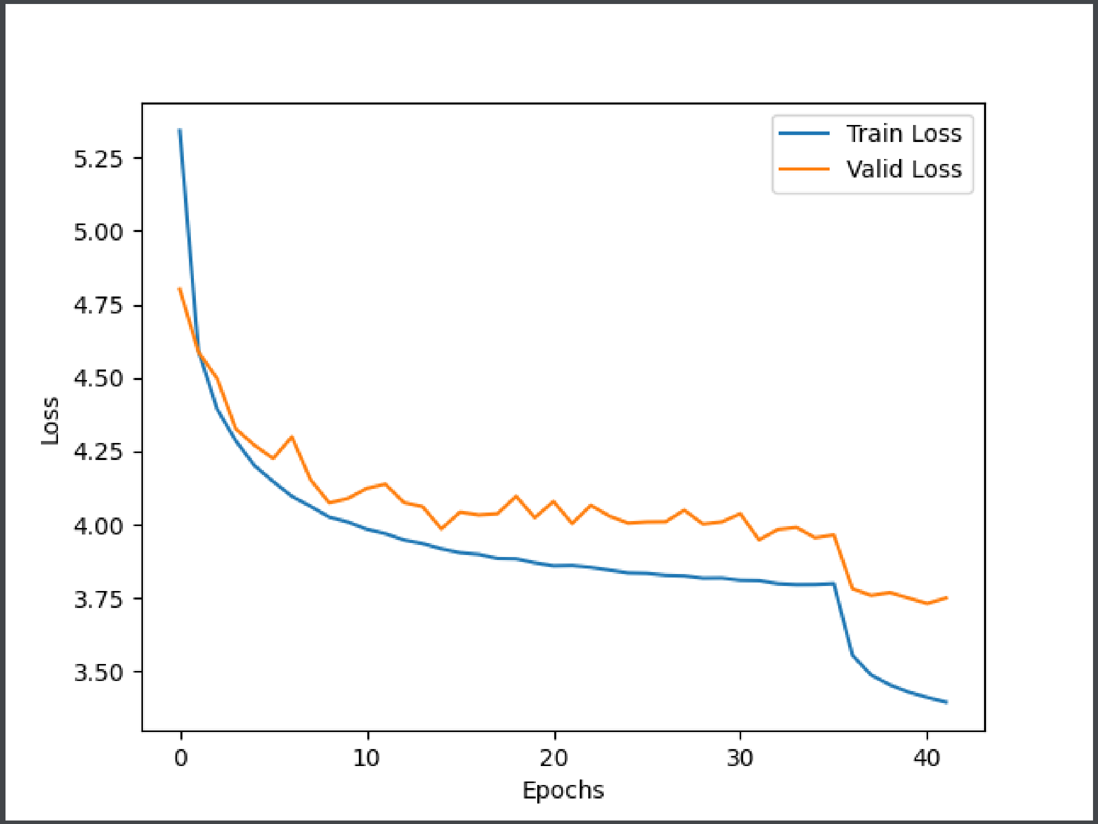
Training

```
EPOCHS = 42
BATCH_SIZE = 8
NUM_KEYPOINTS = 17

# OPTIMIZER
LEARN_RATE = 0.02
MOMENTUM = 0.9
WEIGHT_DECAY = 1e-4

# SCHEDULAR
MILESTONES = (36, 43)
GAMMA = 0.1

optimizer = SGD(params, lr=LEARN_RATE, momentum=MOMENTUM, weight_decay=WEIGHT_DECAY)
lr_scheduler = MultiStepLR(optimizer, milestones=MILESTONES, gamma=GAMMA)
```

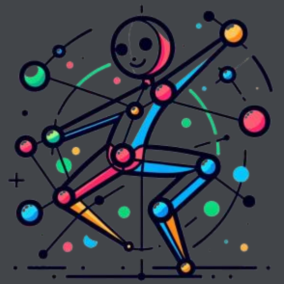


```python
def train_one_epoch(model, optimizer, train_loader, val_loader, device, epoch, status_bar,
print_freq=10, sched=None):
    # ...

    if epoch == 0:
        warmup_factor = 1.0 / 1000
        warmup_iters = min(1000, len(train_loader) - 1)

        sched = LinearLR(optimizer, start_factor=warmup_factor, total_iters=warmup_iters)

    # ...
```



📃 Agenda

- 🚀 Einleitung
- 🏔️ Datensatz
- 🏗️ Architektur
- 🏋️ Training
- ✅ Testing
- ♾️ Fazit

Testing

Metrik	IoU	Fläche	%
AP	0.50:0.95	all	65.5
AP	0.50	all	86.0
AP	0.75	all	71.2
AP	0.50:0.95	M	62.1
AP	0.50:0.95	L	72.0
AR	0.50:0.95	all	72.3
AR	0.50	all	90.9
AR	0.75	all	77.5
AR	0.50:0.95	M	67.8
AR	0.50:0.95	L	78.7

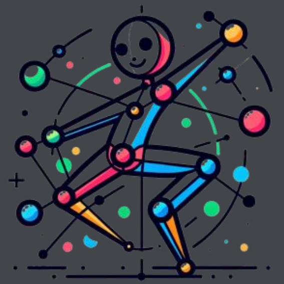
Tabelle 1: ***Keypoint Average Precision & Average Recall*** *mit 20 maximalen Erkunnungen pro Bild*

Metrik	IoU	Fläche	mD	%
AP	0.50:0.95	all	100	54.9
AP	0.50	all	100	82.5
AP	0.75	all	100	59.8
AP	0.50:0.95	S	100	37.7
AP	0.50:0.95	M	100	63.1
AP	0.50:0.95	L	100	70.8
AR	0.50:0.95	all	1	18.8
AR	0.50:0.95	all	10	55.8
AR	0.50:0.95	all	100	64.3
AR	0.50:0.95	S	100	49.4
AR	0.50:0.95	M	100	70.9
AR	0.50:0.95	L	100	78.8

Tabelle 2: ***Bounding Box Average Precision & Average Recall****. mD - maximale Erkennungen pro Bild*

📃 Agenda

- 🚀 Einleitung
- 🏔️ Datensatz
- 🏗️ Architektur
- 🏋️ Training
- ✅ Testing
- ♾️ Fazit



Fazit