Advanced Techniques for CNNs

Elena Congeduti, 21-11-2024

Announcement

Small change to the schedule: Wednesday 4th of December, Lecture 8

Week	Monday	Tuesday	Wednesday	Thursday	Friday
2.1	Lecture 1 Lab 1		Lecture 2 Lab 2	Lecture 3 Lab 3	
2.2	Lecture 4 Lab 4		Lecture 5 Lab 5	Lecture 6 Lab 6 Release - Assignment 1	
2.3	Lecture - Good Practices Lab - Assignment 1		Buffer - Q&A Lab - Assignment 1	Lecture 7 Lab 7	
2.4	Deadline - Assignment 1 Lab - Assignment 1 feedback Release - Assignment 2		Lecture 8 Lab 8	Lecture - Revision Assignment 1 Lab - Assignment 2	
2.5	Lecture 9 Lab 9		Buffer - Q&A Lab - Assignment 2	Buffer - Q&A Lab - Assignment 2	
2.6	Deadline - Assignment 2		Exam 18:30 - 21:30		
Christmas Break					

Legend
Lecture
Regular Lab
Assignments
Deadlines/Exam
Buffer - Q&A

Lecture’s Agenda

- Bandits Exercise
- Modern Convolutional Architectures
- Residual Networks
- Transfer Learning
- Learning tasks: object detection, semantic segmentation

Part of the slides and examples have been developed by students from 2023-2024 edition of the course

Exercise

Multi-armed bandit

Possible reward y: 0, +1 , +100

Bandit x: 0, 1

$$x^{(1)}, x^{(2)}, \ldots x^{(10)} = 0, 1, 1, 1, 0, 1, 0, 1, 0, 0$$
$$y^{(1)}, y^{(2)}, \ldots y^{(10)} = 1, 0, 0, 0, 1, 0, 0, 100, 0, 0$$

Neural network setup

1. What is the learning task? What is the input and output?
2. How many input, hidden and output units?
3. Which is the last layer activation function?
4. What is the loss function?

Multiclass Classification

Training set $\{x^{(i)}, y^{(i)}\}_{i=1,\ldots,n}$ of i.i.d input/output pairs, $y^{(i)}$ is a class $\{1,\ldots,K\}$

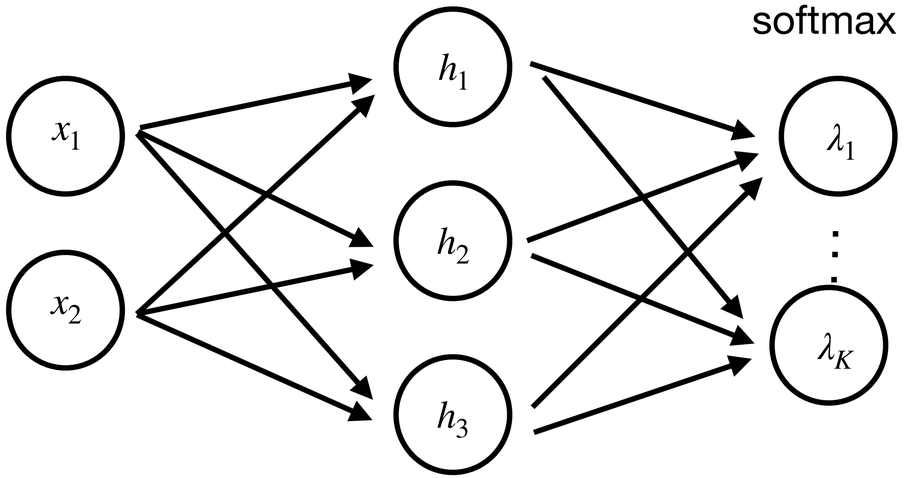
- One-hot vectors $y = (0,\ldots,0,1,0,\ldots,0)$
- K output features
- λ_k is the probability assigned to class k
- Softmax last layer:

$$\lambda_k = \text{softmax}(z_1, \ldots, z_K) = \frac{e^{z_k}}{\sum_{k'} e^{z'_k}}$$

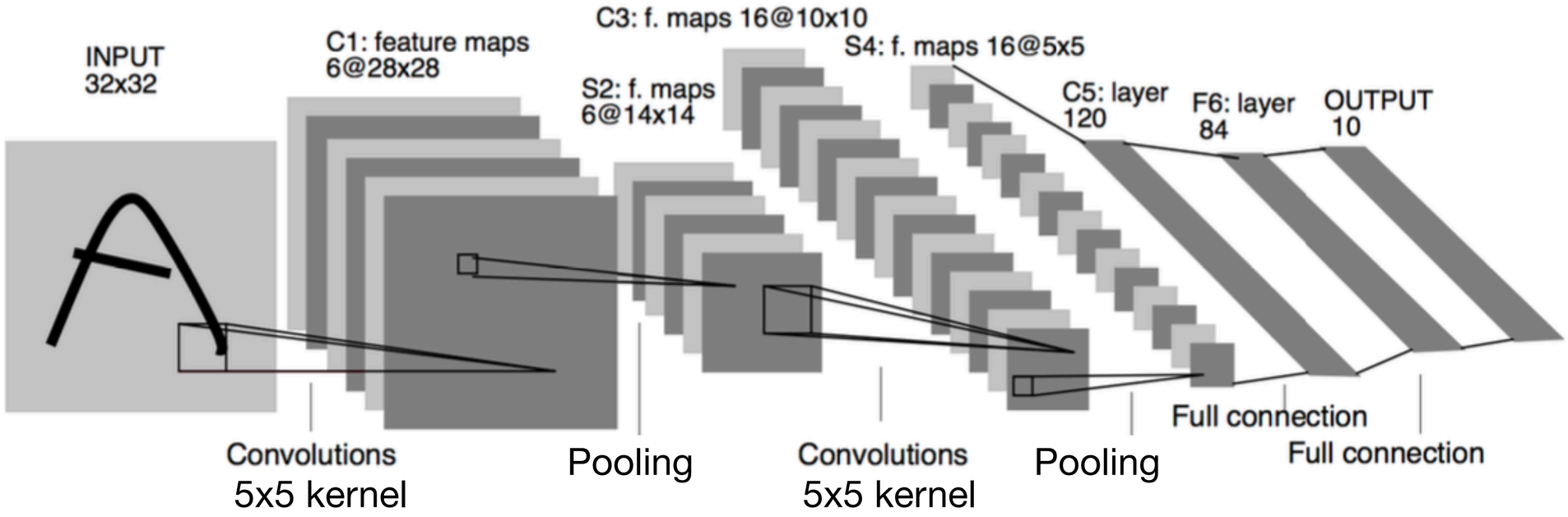
- Cross entropy loss

$$L_{CE}(\theta) = \frac{1}{n} \sum_{i=1}^{n} \underbrace{\sum_{k=1}^{K} - y_k^{(i)} \log(\lambda_k^{(i)})}_{l^{(i)}}$$

- The predicted class predicted $\hat{y} = argmax_k\{\lambda_k\}$

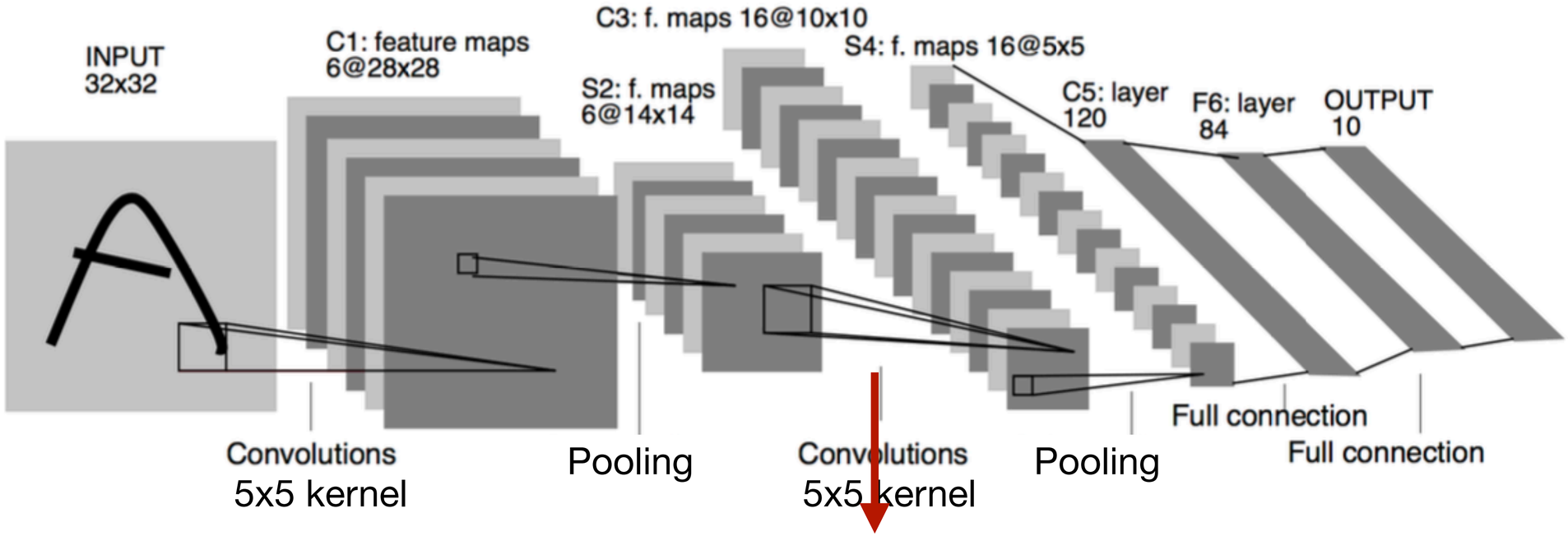


CNN Architecture



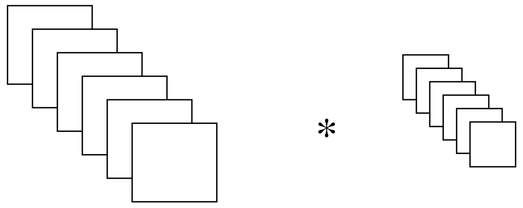
LeNet - 5 (1998)

CNN Architecture



One feature map

(out of 16)

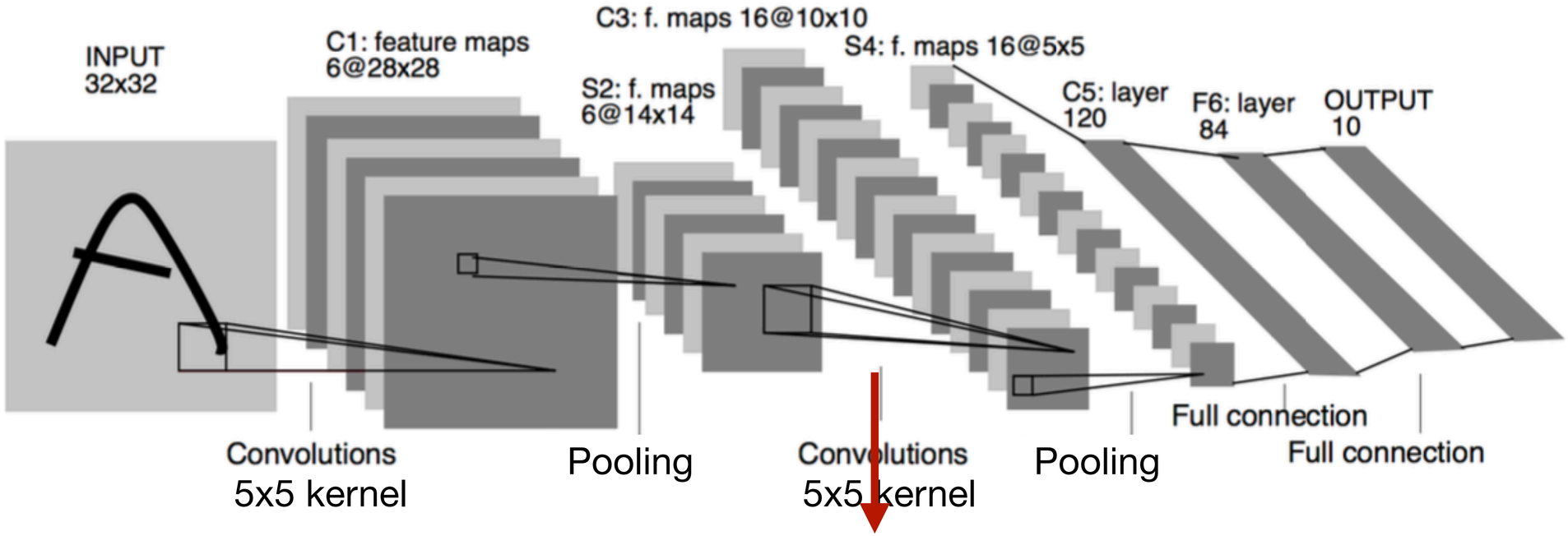


6 input channels

*

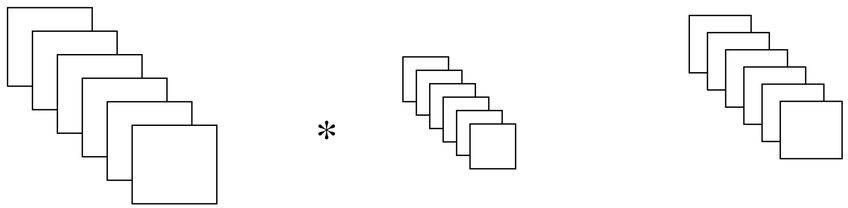
2d-kernels

CNN Architecture



One feature map

(out of 16)

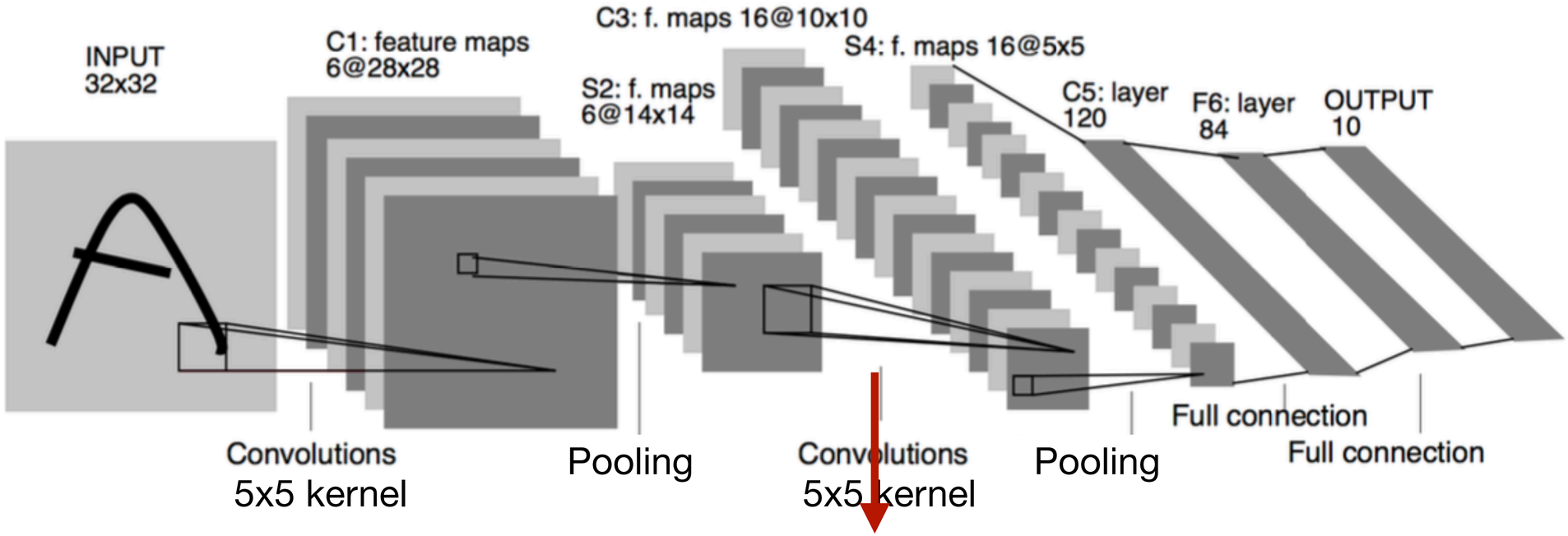


6 input channels

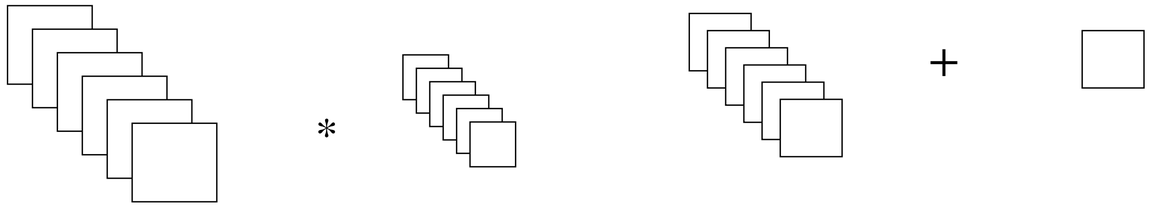
*

2d-kernels

CNN Architecture



One feature map
(out of 16)



6 input channels

*

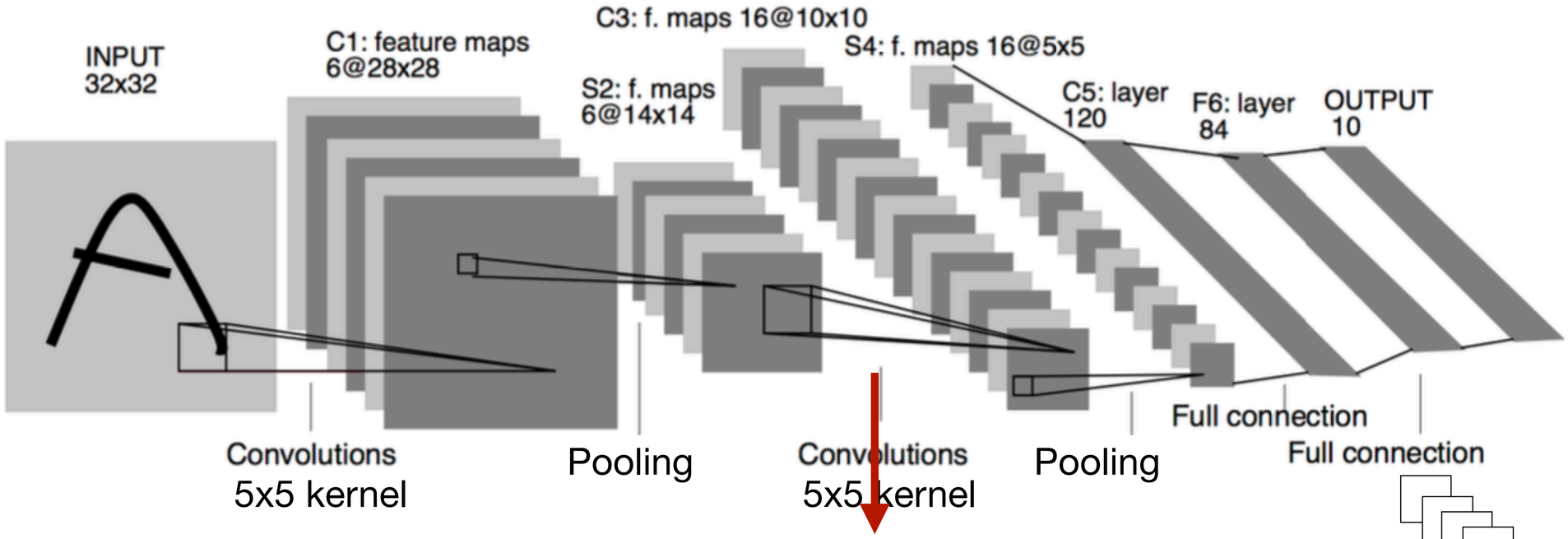
2d-kernels

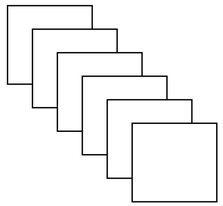
sum
over channels

\+

One output channel
(Out of 16)

CNN Architecture





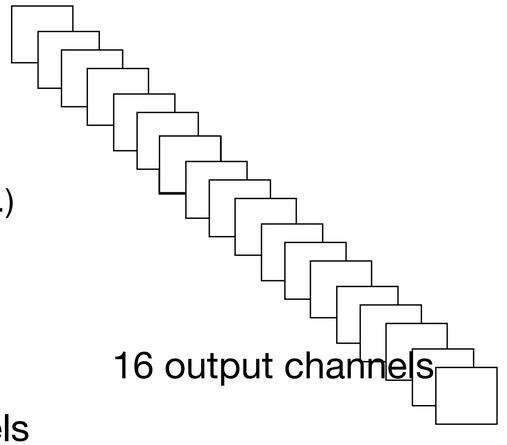
6 input channels

torch.conv2d(in_channels, out_channels,…)

2d-kernels

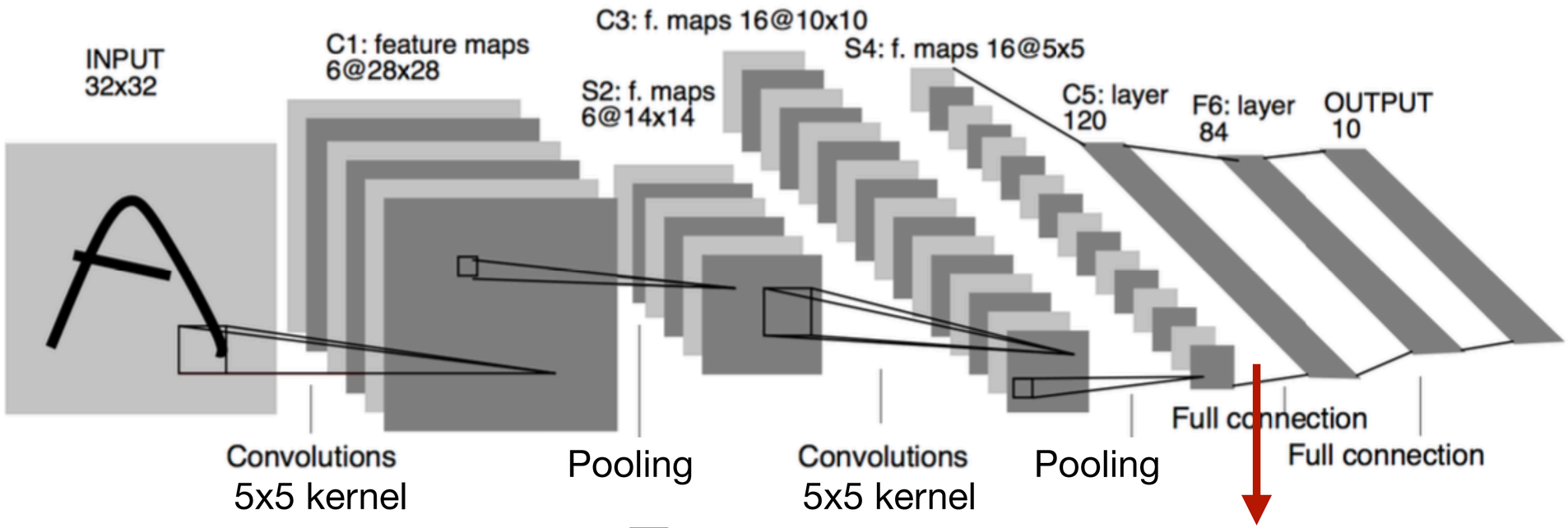
sum

over channels

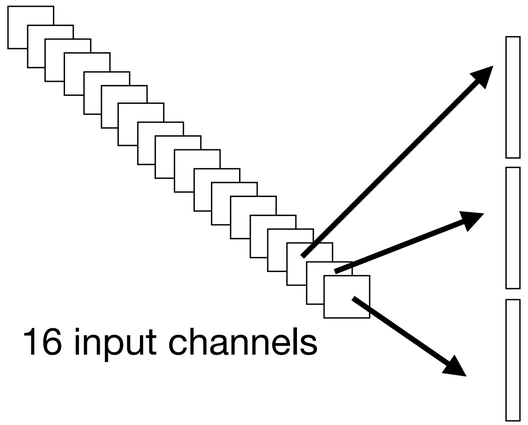


16 output channels

CNN Architecture

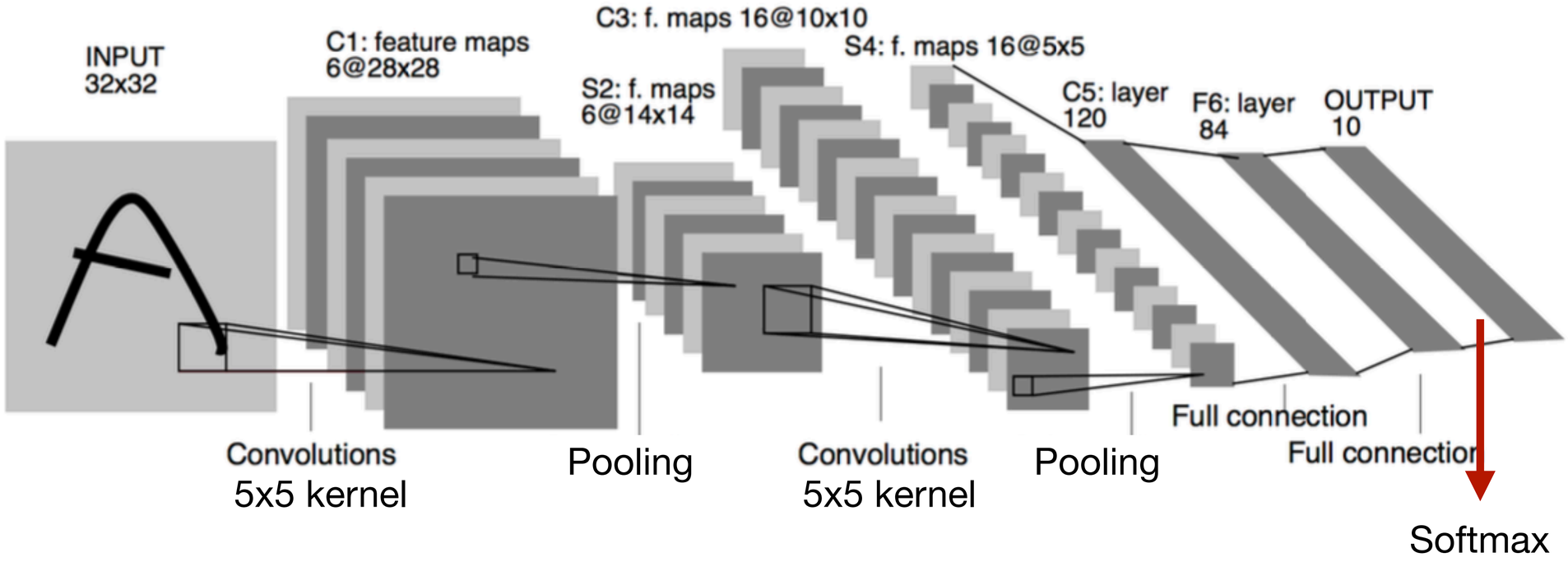


Flatten



400 = 16x5x5 input units

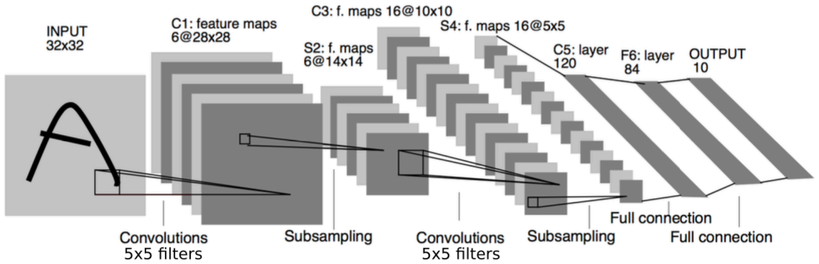
CNN Architecture



Convolutional Network Architectures

LeNet - 5

(1998)

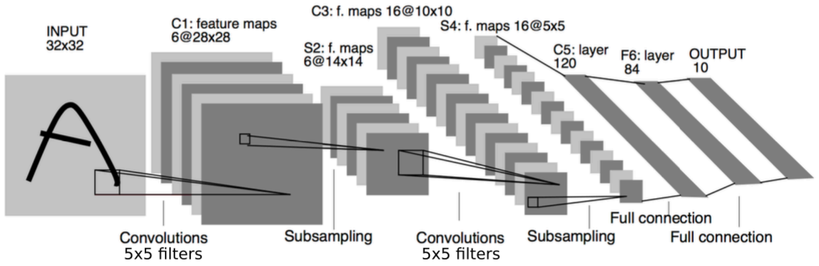


2 convolutional, 2 pooling, 3 fully connected layers

~ 60k parameters

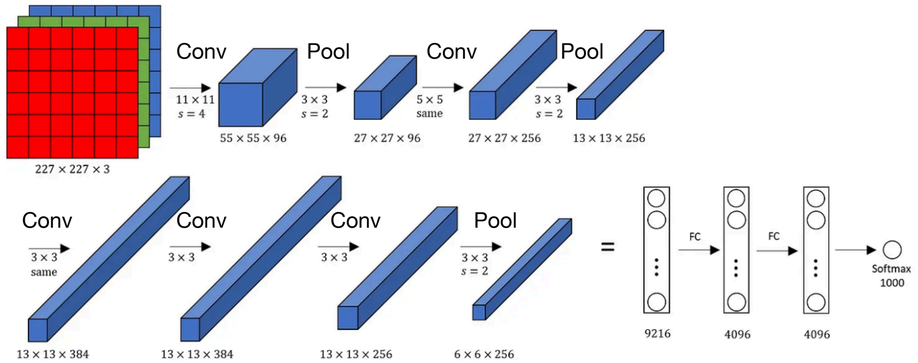
Convolutional Network Architectures

LeNet - 5
(1998)



2 convolutional, 2 pooling, 3 fully connected layers
~ 60k parameters

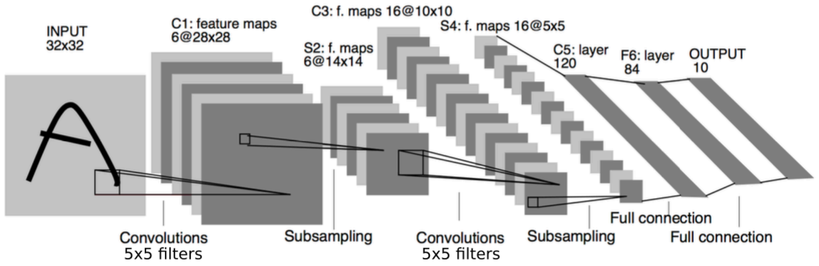
AlexNet
(2012)



5 convolutional, 3 pooling, 3 fully connected layers
~60M parameters

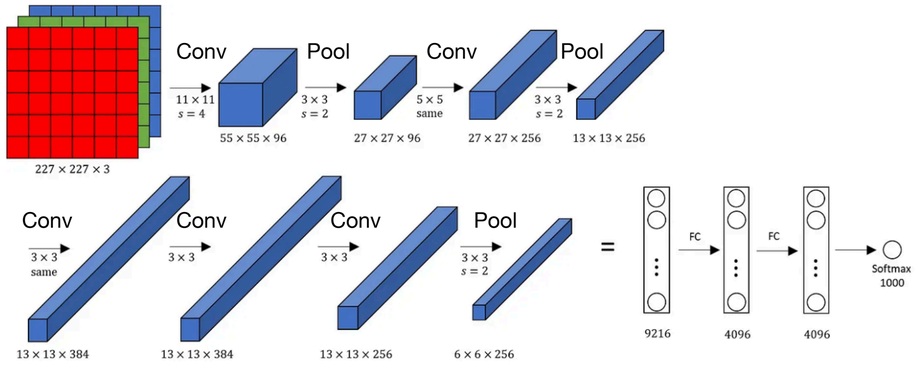
Convolutional Network Architectures

LeNet - 5
(1998)



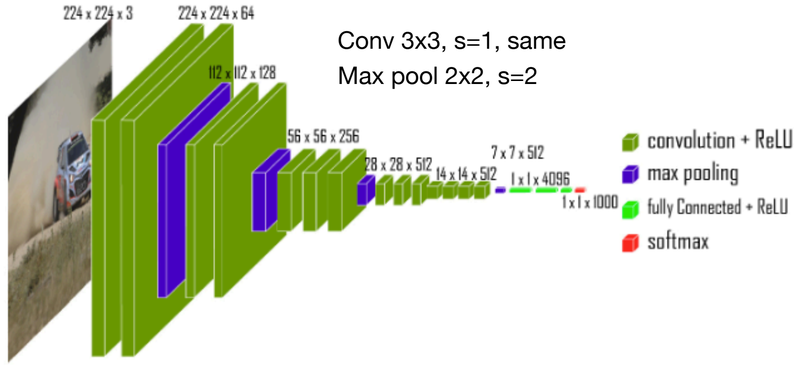
2 convolutional, 2 pooling, 3 fully connected layers
~ 60k parameters

AlexNet
(2012)



5 convolutional, 3 pooling, 3 fully connected layers
~60M parameters

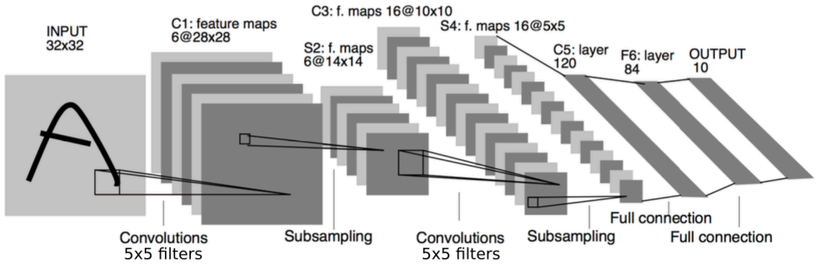
VGG - 16
(2015)



13 convolutional, 5 pooling, 3 fully connected layers
~138M parameters

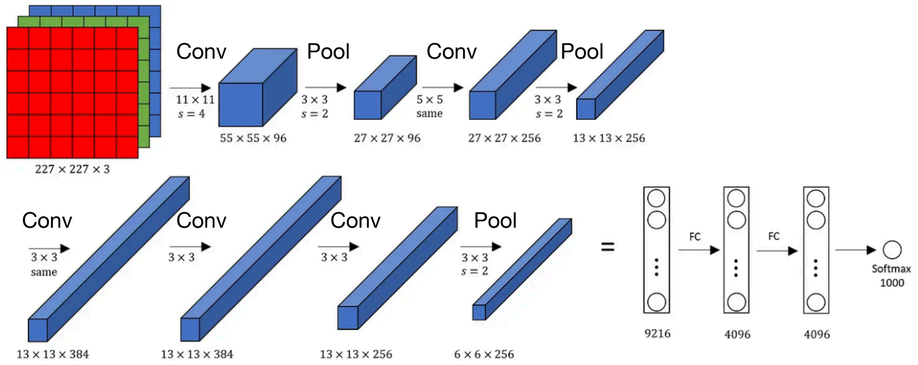
Convolutional Network Architectures

LeNet - 5
(1998)



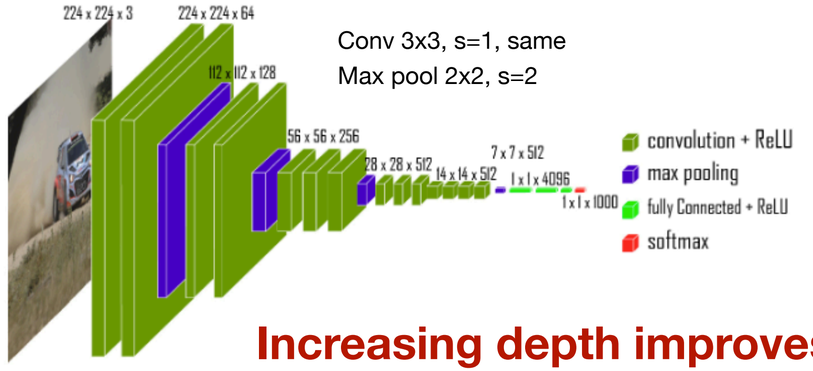
2 convolutional, 2 pooling, 2 fully connected layers
~ 60k parameters

AlexNet
(2012)



5 convolutional, 3 pooling, 3 fully connected layers
~60M parameters

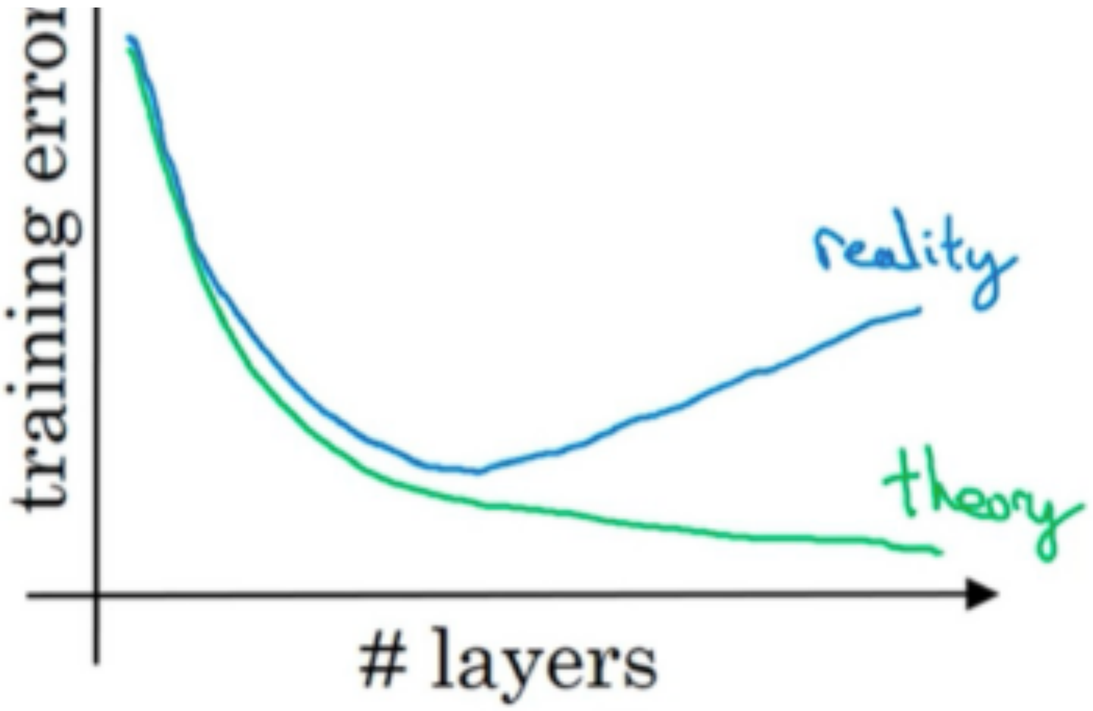
VGG - 16
(2015)



13 convolutional, 5 pooling, 3 fully connected layers
~138M parameters

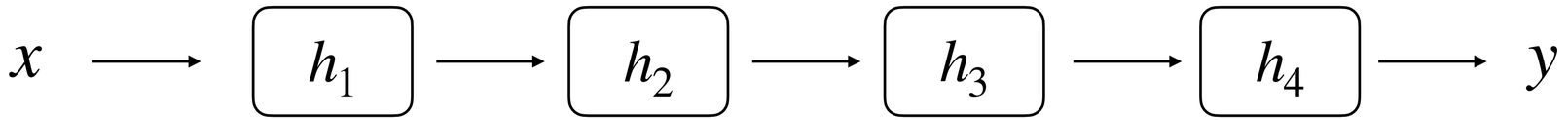
Increasing depth improves performance

Depth and Shattered Gradients

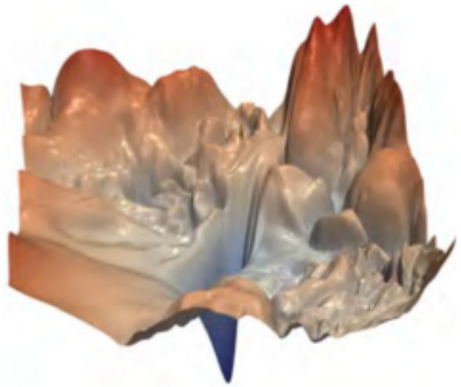


Increasing depth improves performance — up to a certain point!

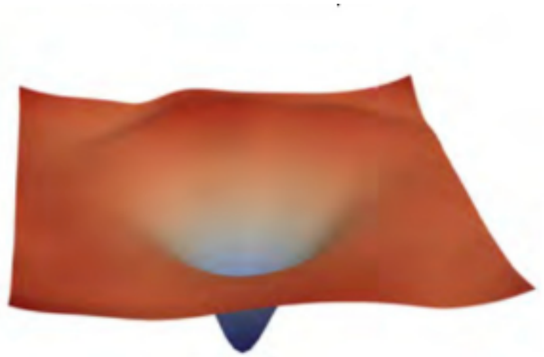
Depth and Shattered Gradients



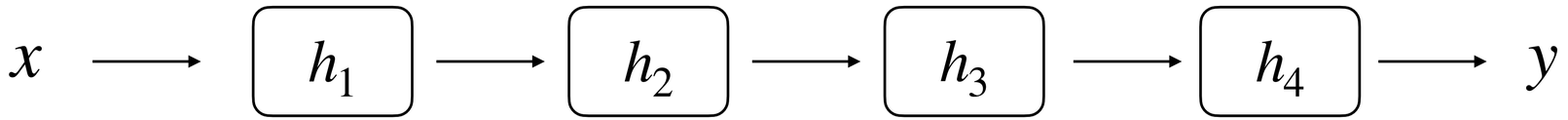
$$\frac{\partial L}{\partial h_1} = \frac{\partial L}{\partial h_4}\frac{\partial h_4}{\partial h_3}\frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial h_1}$$



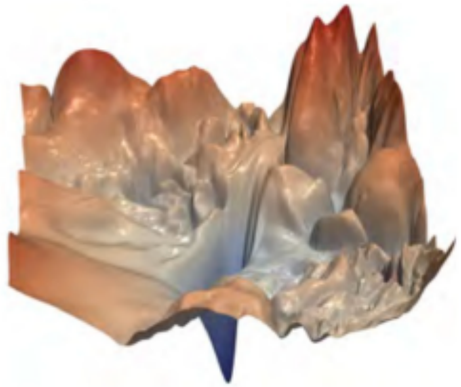
Shattered gradients

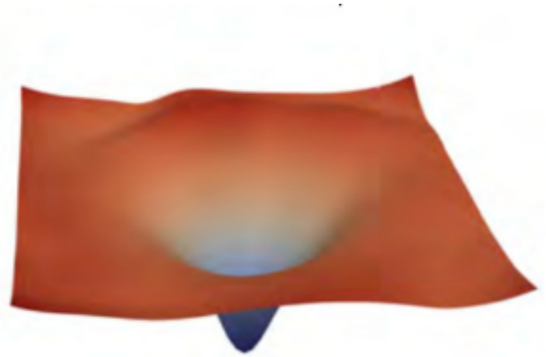


Depth and Shattered Gradients



$$\frac{\partial L}{\partial h_1} = \frac{\partial L}{\partial h_4}\frac{\partial h_4}{\partial h_3}\frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial h_1}$$

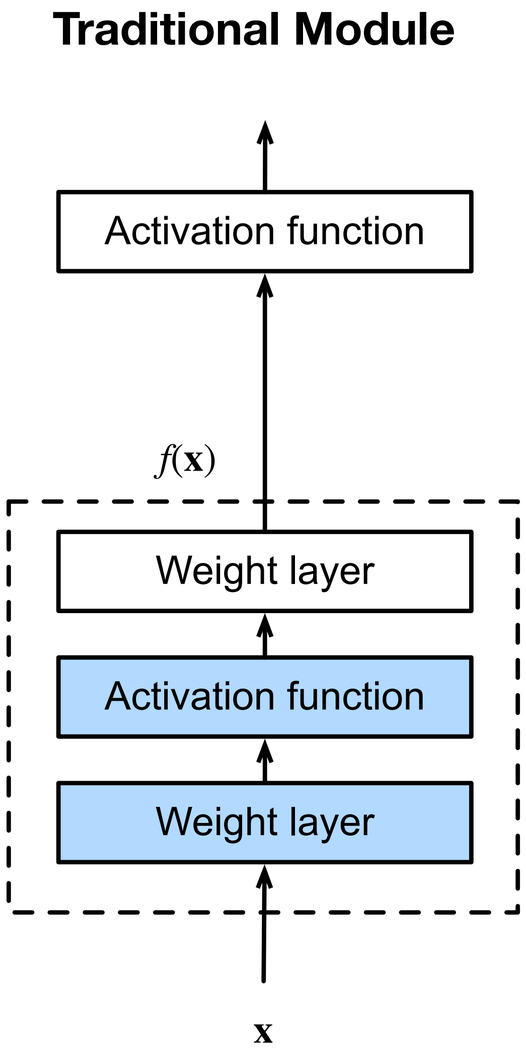




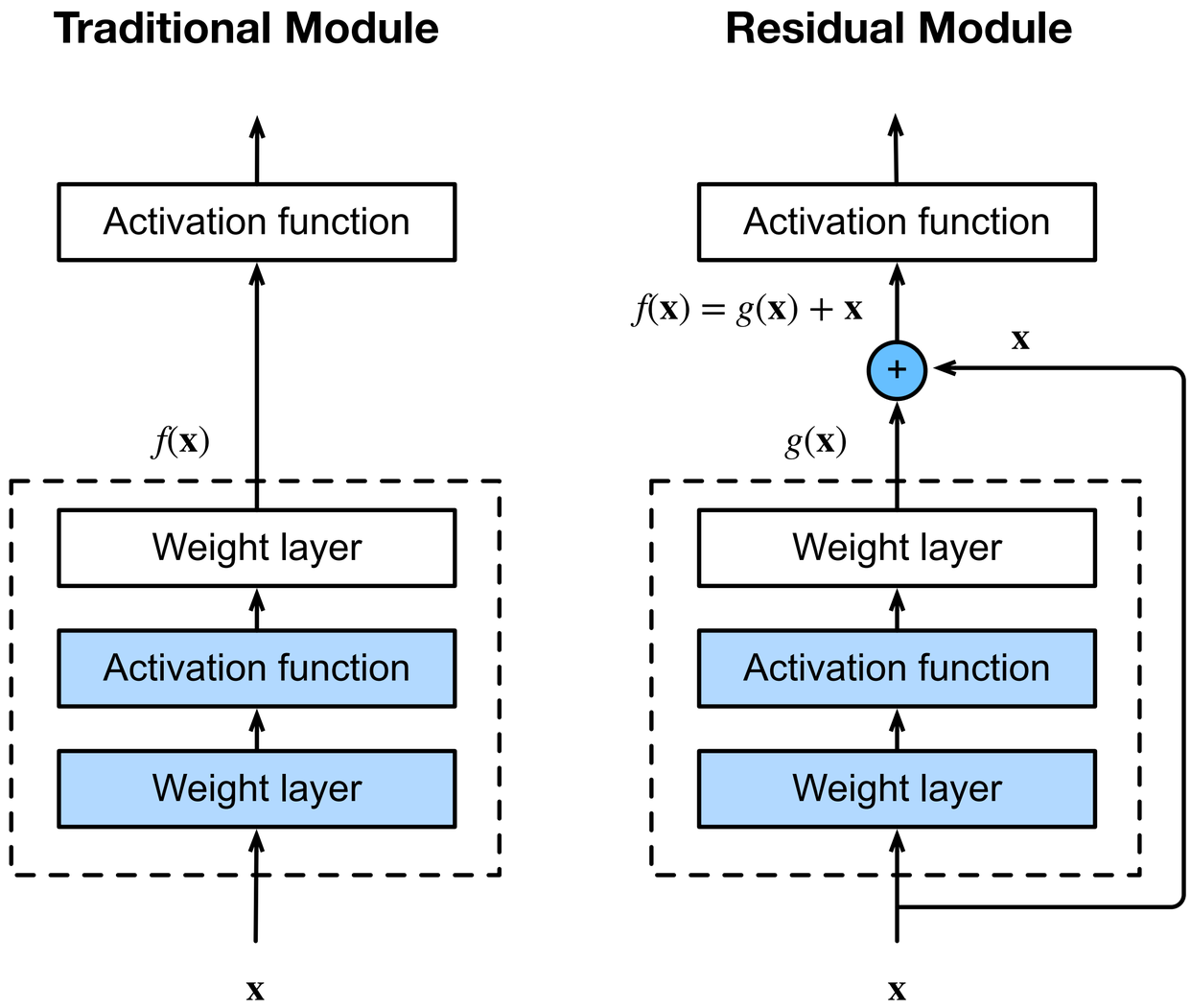
Shattered gradients

Can we somehow implement ‘shortcuts’ in the computation graph?

Residual Connections



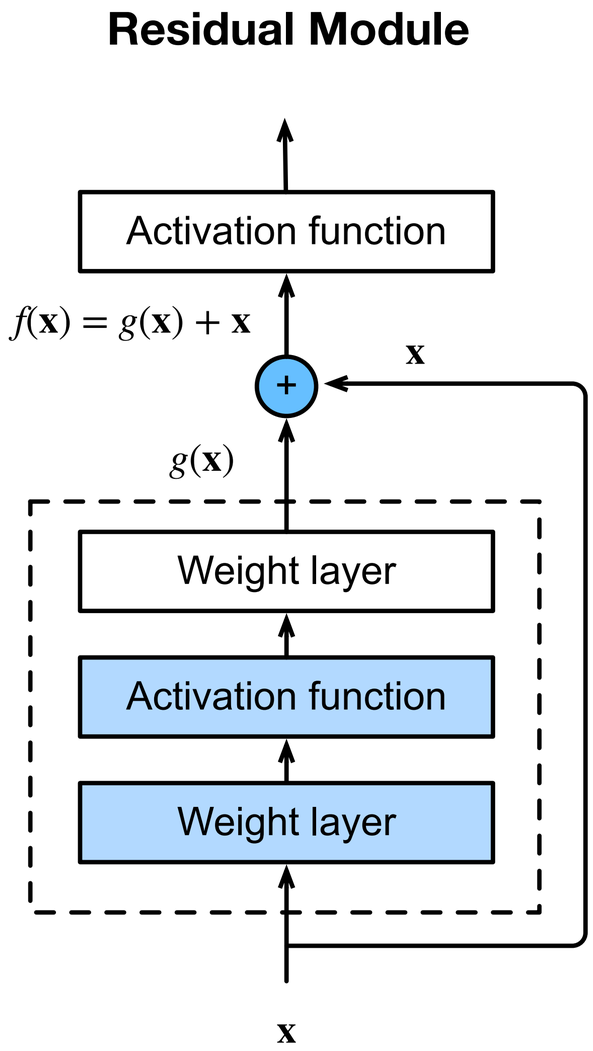
Residual Connections



Residual Connections

$$f(x) = g(x) + x$$

$$\frac{\partial f}{\partial x} = \frac{\partial g}{\partial x} + 1$$

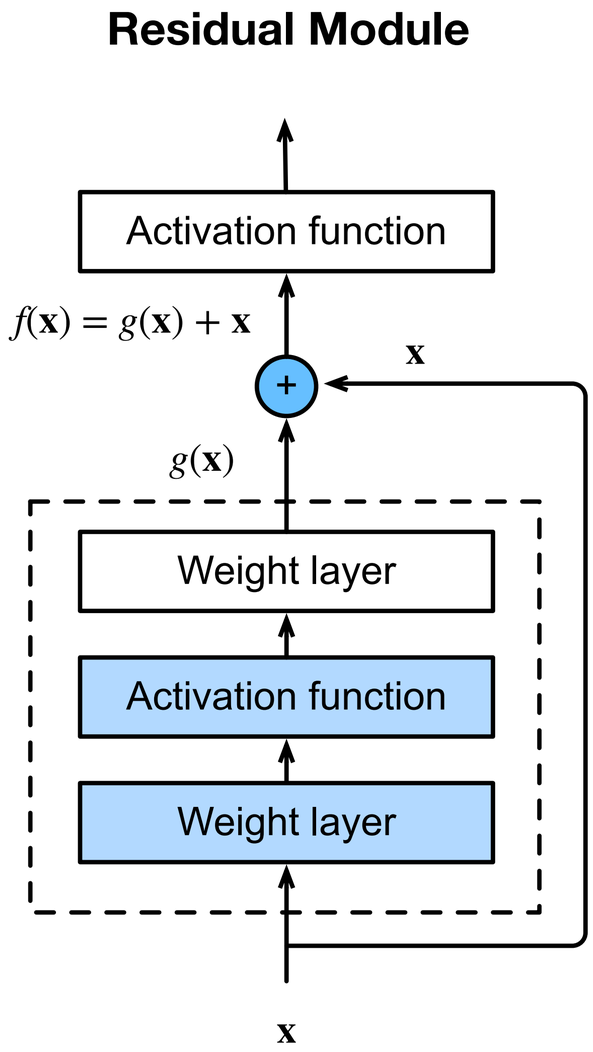


Residual Connections

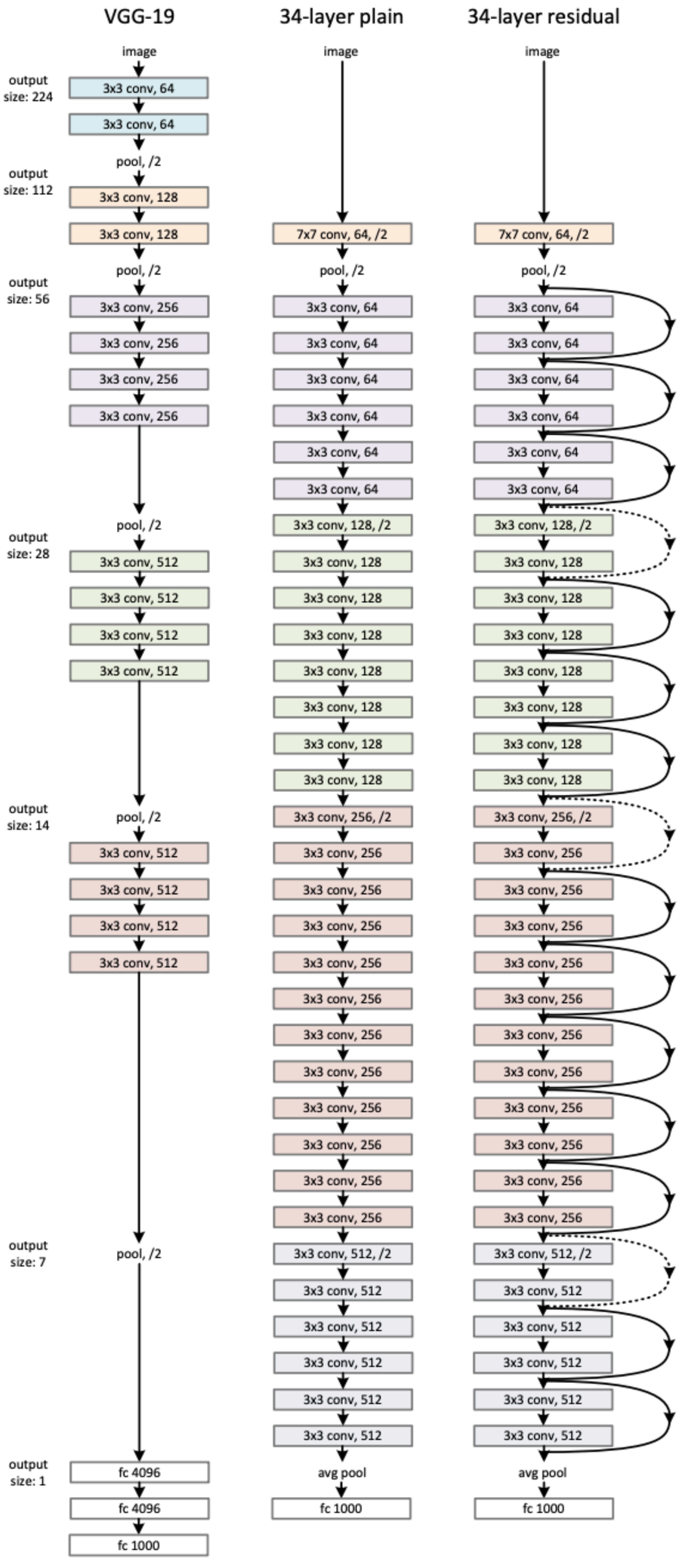
$$f(x) = g(x) + x$$

$$\frac{\partial f}{\partial x} = \frac{\partial g}{\partial x} + \boxed{1}$$

The gradient doesn't even when $\frac{\partial g}{\partial x} \approx 0$



ResNet-50


VGG-19
34-layer plain
34-layer residual
image
output size: 224
3x3 conv, 64
3x3 conv, 64
pool, /2
output size: 112
3x3 conv, 128
3x3 conv, 128
pool, /2
output size: 56
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
pool, /2
output size: 28
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
pool, /2
output size: 14
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
pool, /2
output size: 7
fc 4096
fc 4096
fc 1000
output size: 1
7x7 conv, 64, /2
pool, /2
3x3 conv, 128, /2
3x3 conv, 256, /2
3x3 conv, 512, /2
avg pool
fc 1000

Classification: Aliens vs Predators

Classification: Aliens vs Predators

Collecting a sufficiently large training set with labelled images to train from scratch the ResNet

Classification: Aliens vs Predators

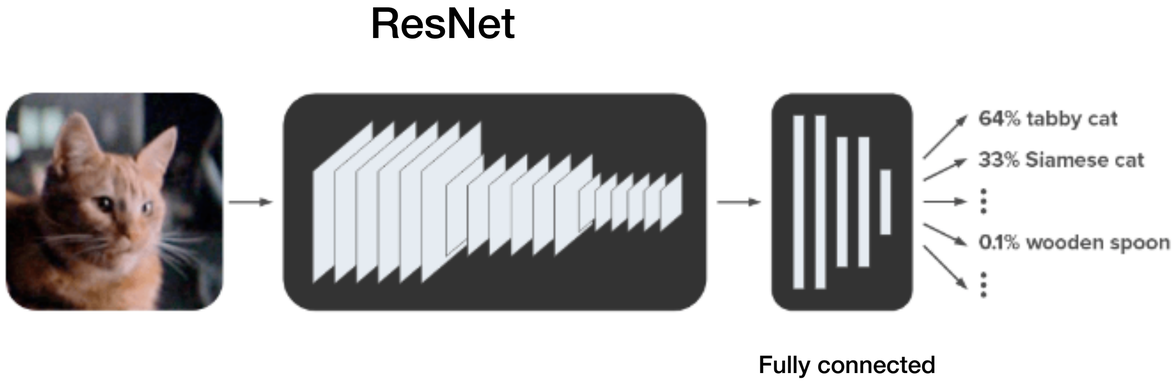
ImageNet

1.2 million images

1000 object categories

~~Collecting a sufficiently large training set with labelled images to train from scratch the ResNet~~

Pre-trained Model



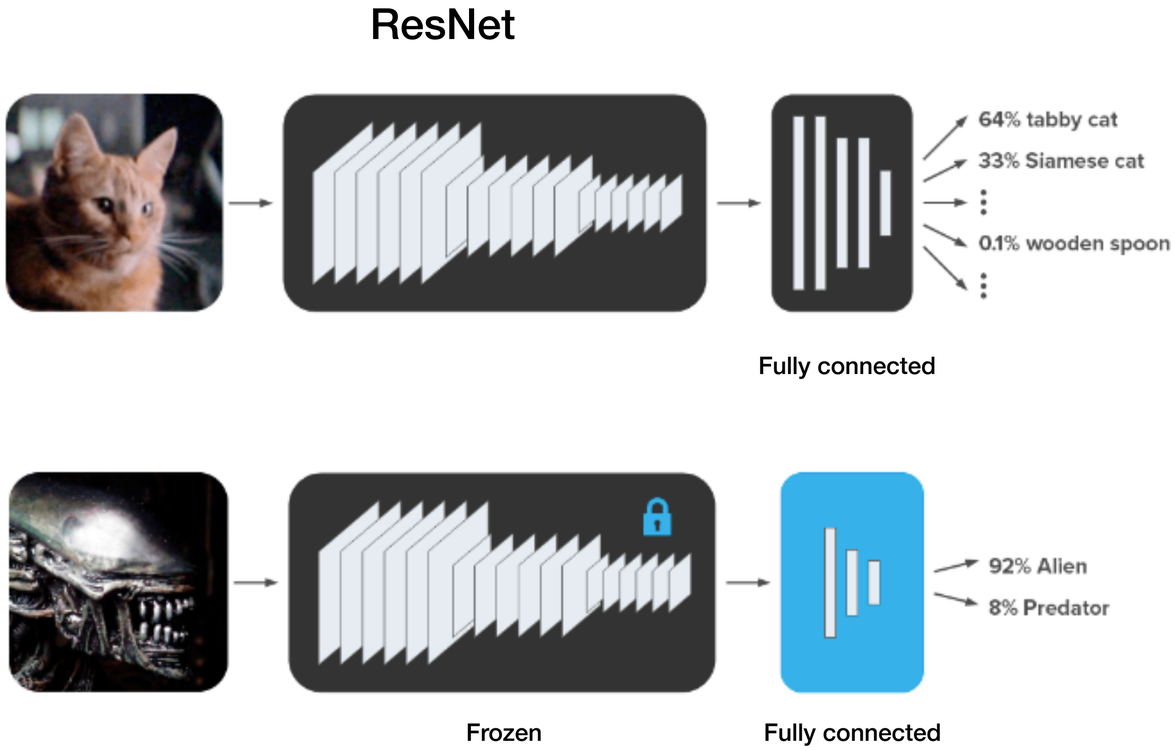
ImageNet

1.2 million images

1000 object categories

Import the parameters of the ResNet model pre-trained over ImageNet

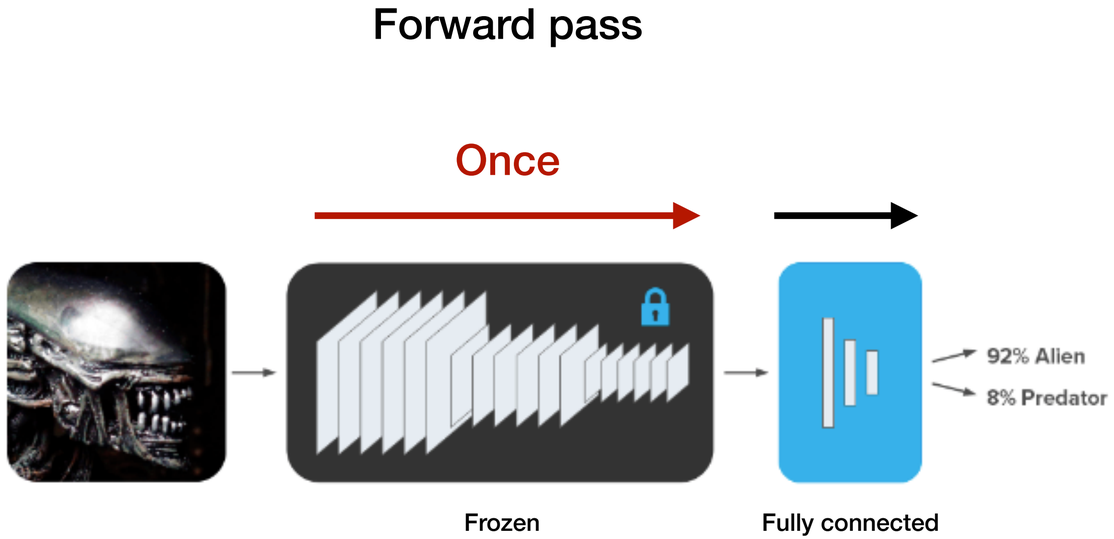
Fixed Feature Extractor



Import the parameters of the ResNet model pre-trained over ImageNet

- Freeze the weights of the convolutional layers and replace the last fully connected layers

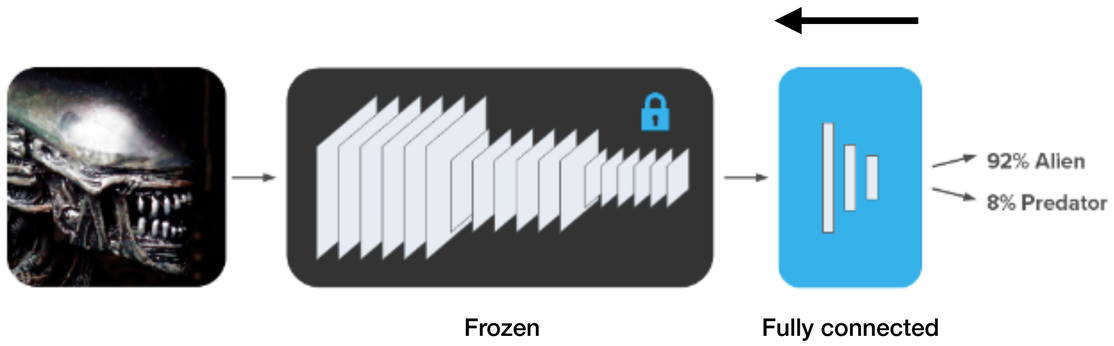
Fixed Feature Extractor



Import the parameters of the ResNet model pre-trained over ImageNet

- Freeze the weights of the convolutional layers and replace the last fully connected layers

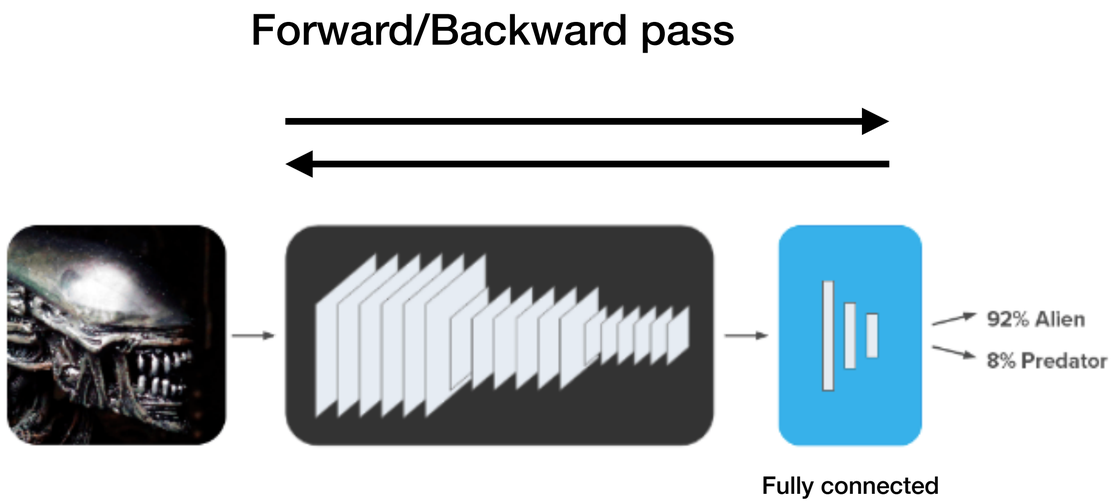
Fixed Feature Extractor



Import the parameters of the ResNet model pre-trained over ImageNet

- Freeze the weights of the convolutional layers and replace the last fully connected layers

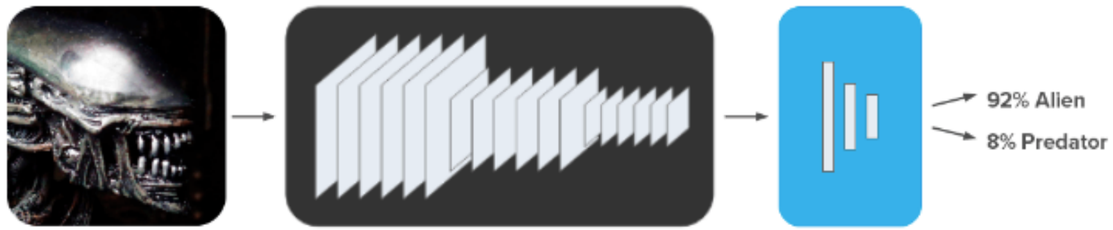
Finetuning



Import the parameters of the ResNet model pre-trained over ImageNet

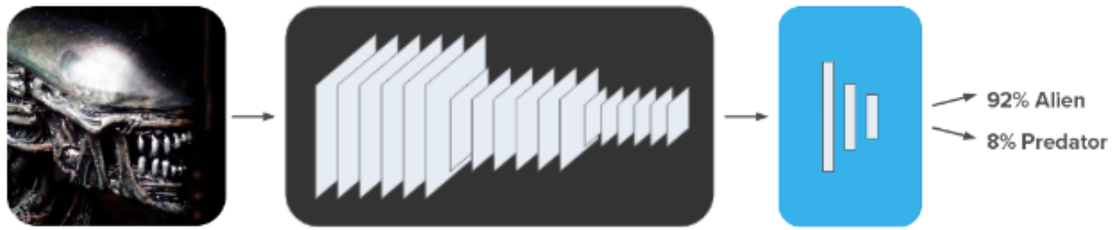
- Freeze the weights of the convolutional layers and replace the last fully connected layers
- Replace last fully connected layers and finetune the entire network

Transfer Learning: Questions



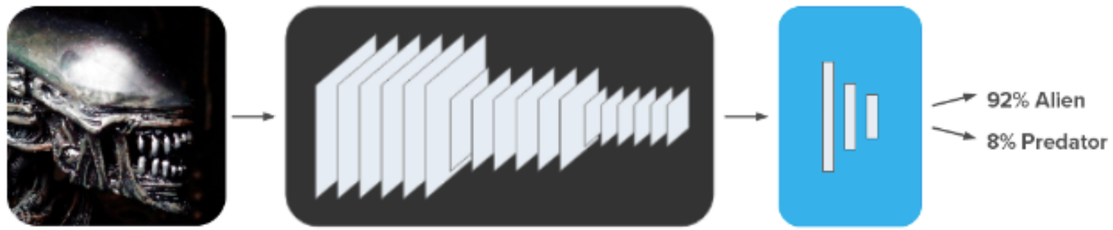
If we have small dataset of aliens and predators should we use fine-tuning or freezing the parameters?

Transfer Learning: Questions



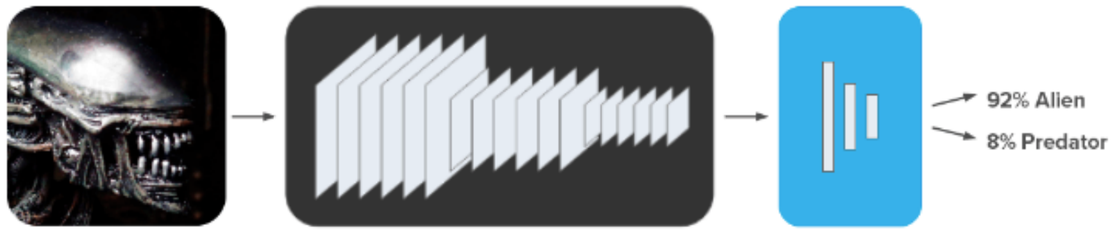
If we have small dataset of aliens and predators should we use fine-tuning or **freezing the parameters**?

Transfer Learning: Questions



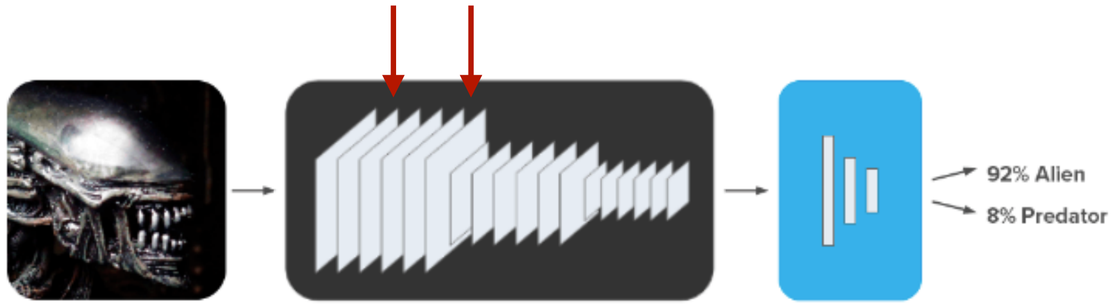
If we use finetuning, should we choose a large or small learning rate?

Transfer Learning: Questions



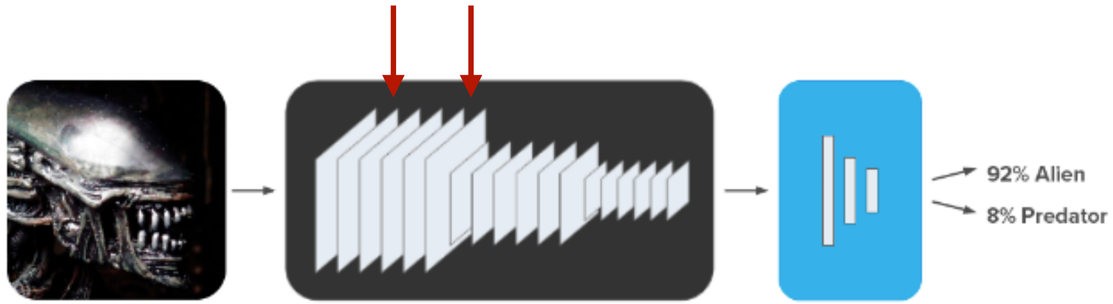
If we use finetuning, should we choose a large or **small** learning rate?

Transfer Learning: Questions



Can we freeze a number of chosen hidden layers and treat the choice of hidden layers to freeze as an hyperparameter?

Transfer Learning: Questions

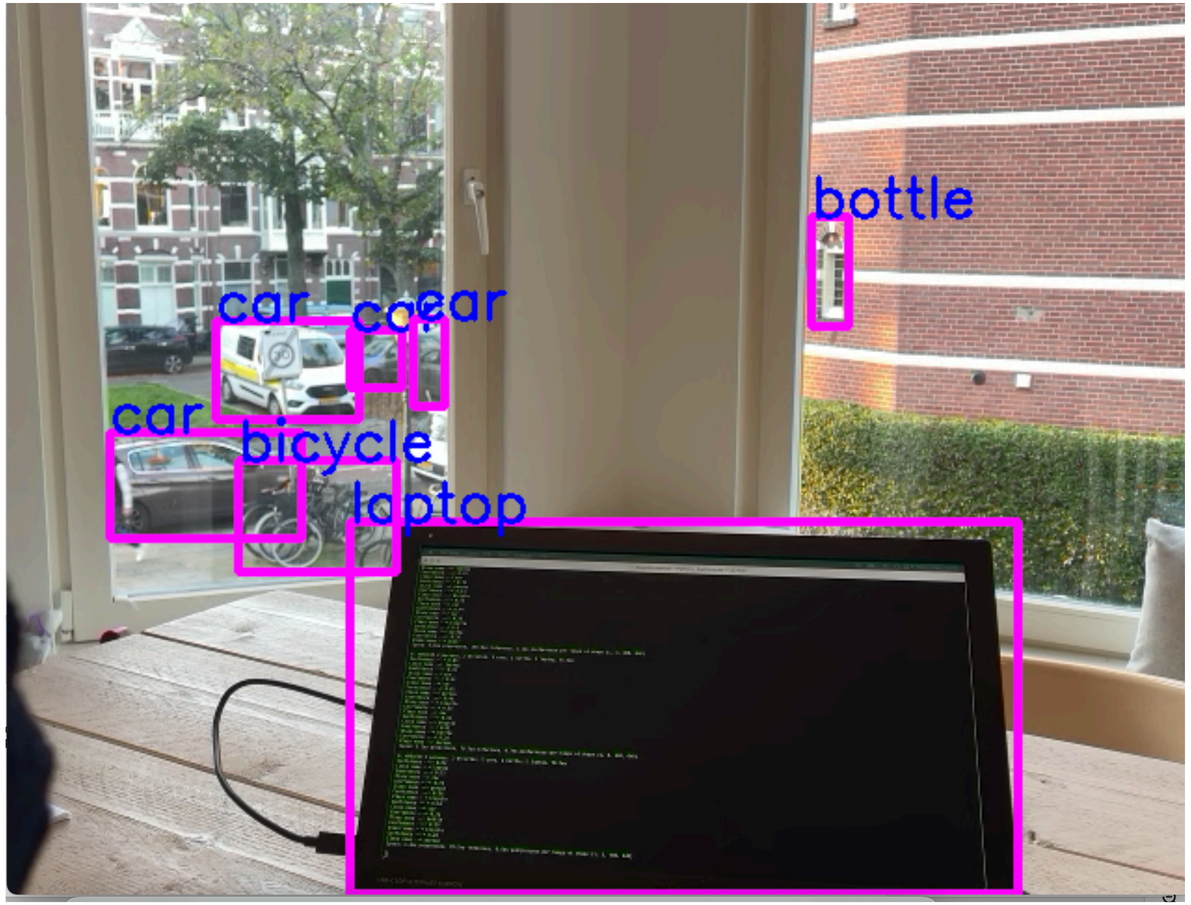


Can we freeze a number of chosen hidden layers and treat the choice of hidden layers to freeze as an hyperparameter?

No

Learning Tasks: Object Detection

Recognize and localize multiple objects in an image



Object Detection

Recognize and localize multiple objects in an image

Classification

Class labels “sofa”, “chair”, “armchair”

Softmax output $\lambda_{sofa}, \lambda_{chair}, \lambda_{armchair}$

Classification loss

Object Detection

Recognize and **localize** multiple objects in an image

Classification

Localization

Bounding box $[x_{min}, x_{max}, y_{min}, y_{max}]$

Output $[\hat{x}_{min}, \hat{x}_{max}, \hat{y}_{min}, \hat{y}_{max}]$

Regression loss

Object Detection

Recognize and localize multiple objects in an image

Classification

Localization

Class labels and bounding boxes “Sofa”, $[x_{min}, x_{max}, y_{min}, y_{max}]$

Output $[\lambda_{sofa}, \lambda_{chair}, \lambda_{table}, \hat{x}_{min}, \hat{x}_{max}, \hat{y}_{min}, \hat{y}_{max}]$

Combination of classification + regression loss

Object Detection

Recognize and localize multiple objects in an image

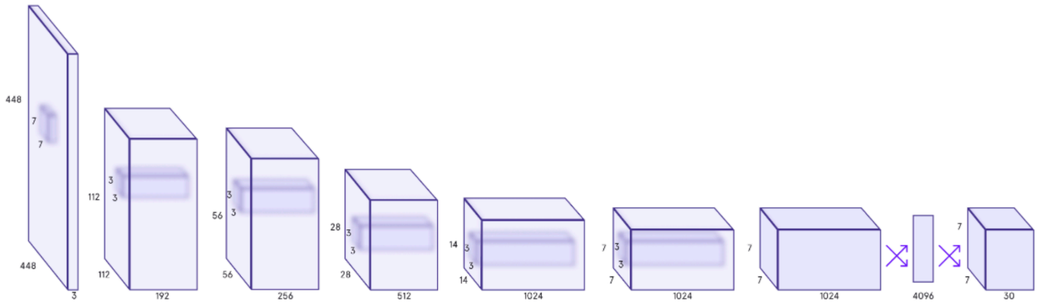
Classification

Localization

Detection

YOLO and Fast R-CNN* (idea)

Recognize and localize multiple objects in an image



Class labels

Sofa

Table

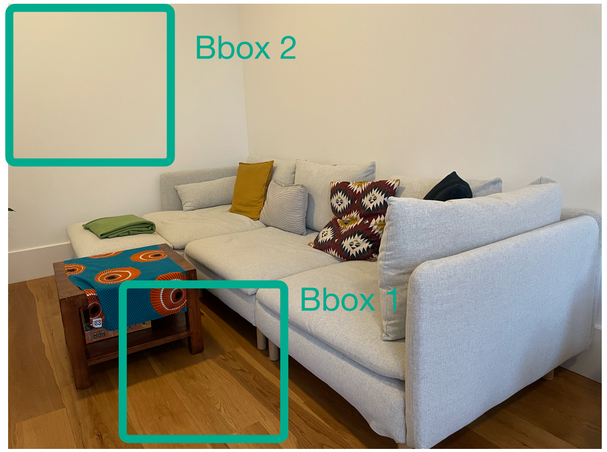
Chair

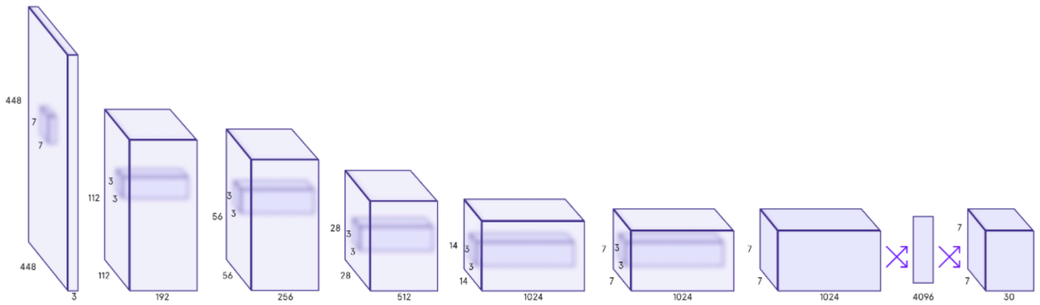
Background

*Redmon et al. You Only Look Once: Unified real-time object detection (2015); Girshick, R. Fast r-cnn (2015).

YOLO and Fast R-CNN (idea)

Recognize and localize multiple objects in an image





Class labels

Sofa

Table

Chair

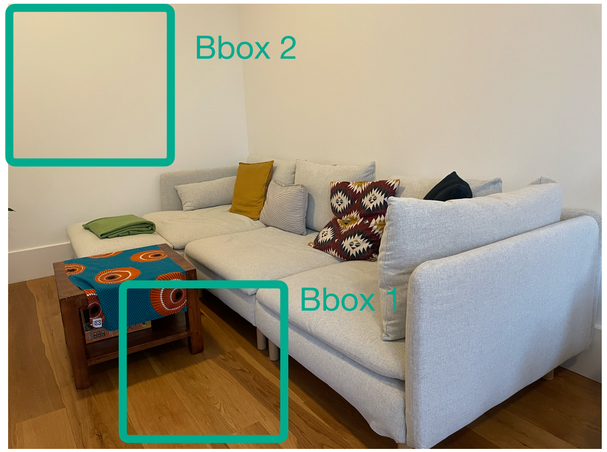
Background

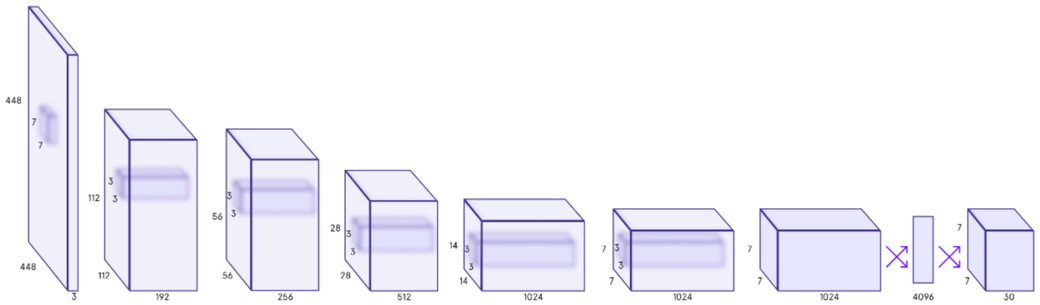
Output $[\lambda, \lambda_{sofa}, \lambda_{chair}, \lambda_{table}, \hat{x}_{min}, \hat{x}_{max}, \hat{y}_{min}, \hat{y}_{max}, \ldots]$

Bbox 1 Bbox 2

YOLO and Fast R-CNN (idea)

Recognize and localize multiple objects in an image





Class labels

Sofa

Table

Chair

Background

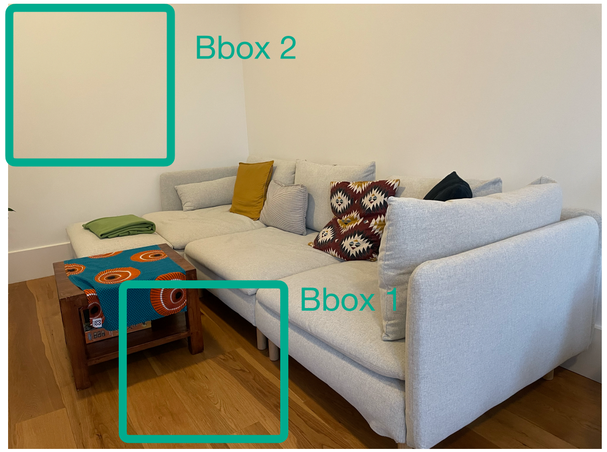
Output $[\lambda, \lambda_{sofa}, \lambda_{chair}, \lambda_{table}, \hat{x}_{min}, \hat{x}_{max}, \hat{y}_{min}, \hat{y}_{max}, \ldots]$

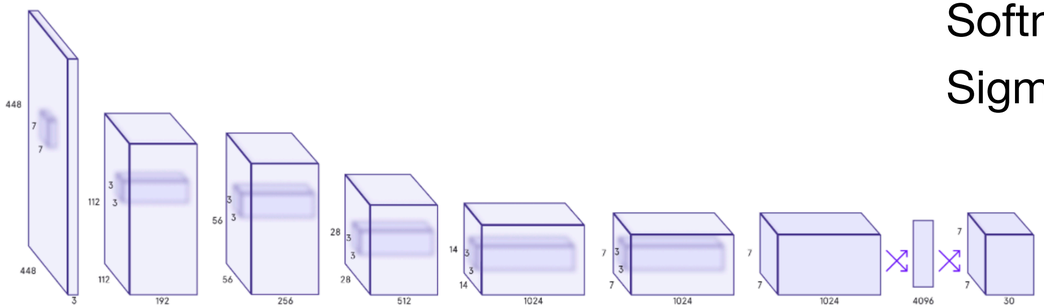
Bbox 1 Bbox 2

Confidence score: probability that an object is present in bbox 1, if 0 then is background

YOLO and Fast R-CNN (idea)

Recognize and localize multiple objects in an image





Softmax/
Sigmoid

Sigmoid + Scaling

λ
λ_{sofa}
λ_{chair}
λ_{table}
Bbox 1

λ
λ_{sofa}
λ_{chair}
λ_{table}
Bbox 2

$\hat{x}_{min}$,
$\hat{x}_{max}$,
$\hat{y}_{min}$,
$\hat{y}_{max}$
Bbox 1

$\hat{x}_{min}$,
$\hat{x}_{max}$,
$\hat{y}_{min}$,
$\hat{y}_{max}$
Bbox 2

Class labels

Sofa

Table

Chair

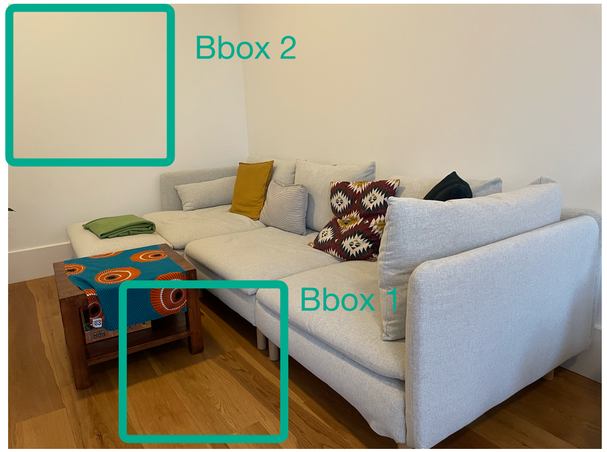
Background

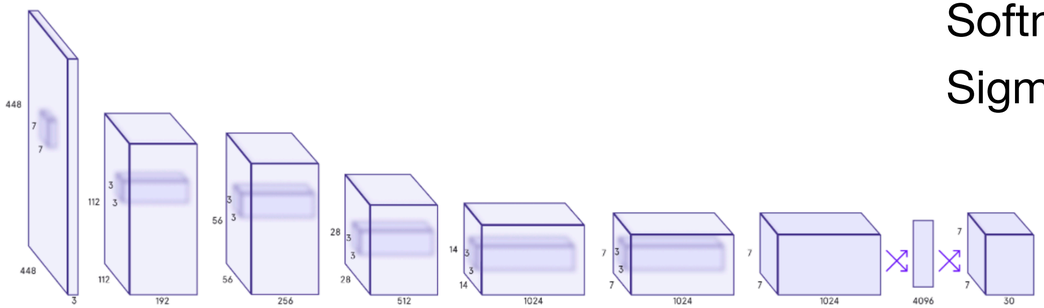
Output $[\lambda, \lambda_{sofa}, \lambda_{chair}, \lambda_{table}, \hat{x}_{min}, \hat{x}_{max}, \hat{y}_{min}, \hat{y}_{max}, \ldots]$

Bbox 1 Bbox 2

YOLO and Fast R-CNN (idea)

Recognize and localize multiple objects in an image





Softmax/
Sigmoid

Sigmoid + Scaling

λ
λ_{sofa}
λ_{chair}
λ_{table} Classification
λ Loss
λ_{sofa} (CE, SE, L1)
λ_{chair}
λ_{table}

+

$\hat{x}_{min}$,
$\hat{x}_{max}$,
$\hat{y}_{min}$, Regression
$\hat{y}_{max}$ Loss
$\hat{x}_{min}$, (SE, L1)
$\hat{x}_{max}$,
$\hat{y}_{min}$,
$\hat{y}_{max}$

Class labels

Sofa

Table

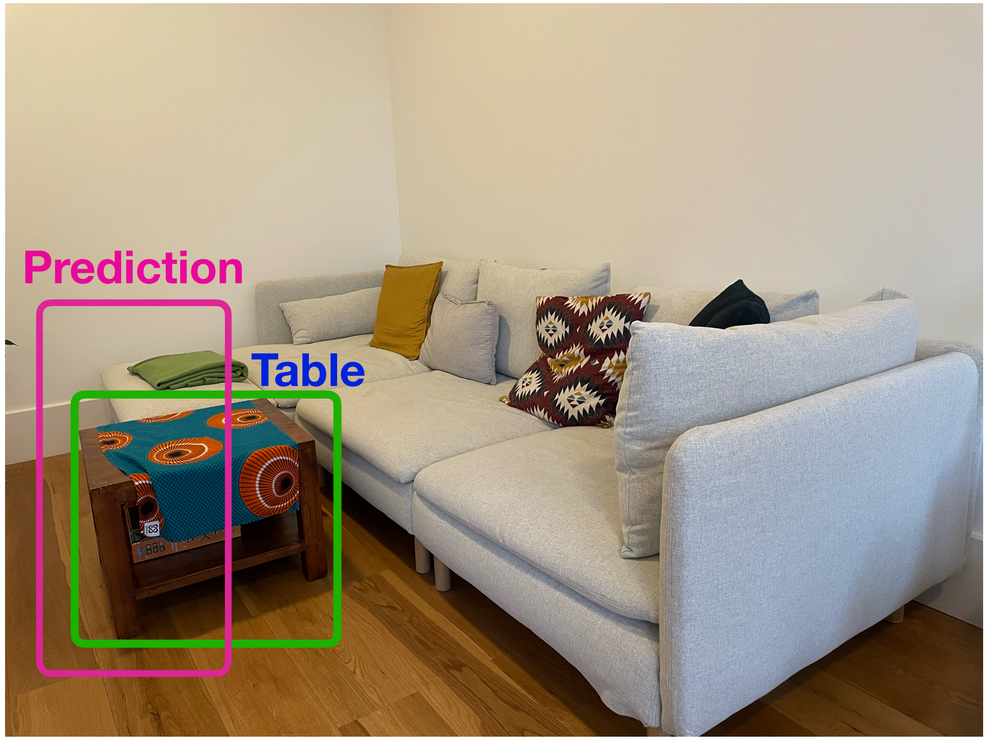
Chair

Background

Output $[\lambda, \lambda_{sofa}, \lambda_{chair}, \lambda_{table}, \hat{x}_{min}, \hat{x}_{max}, \hat{y}_{min}, \hat{y}_{max}, \ldots]$

Bbox 1 Bbox 2

Inference: Performance Metrics



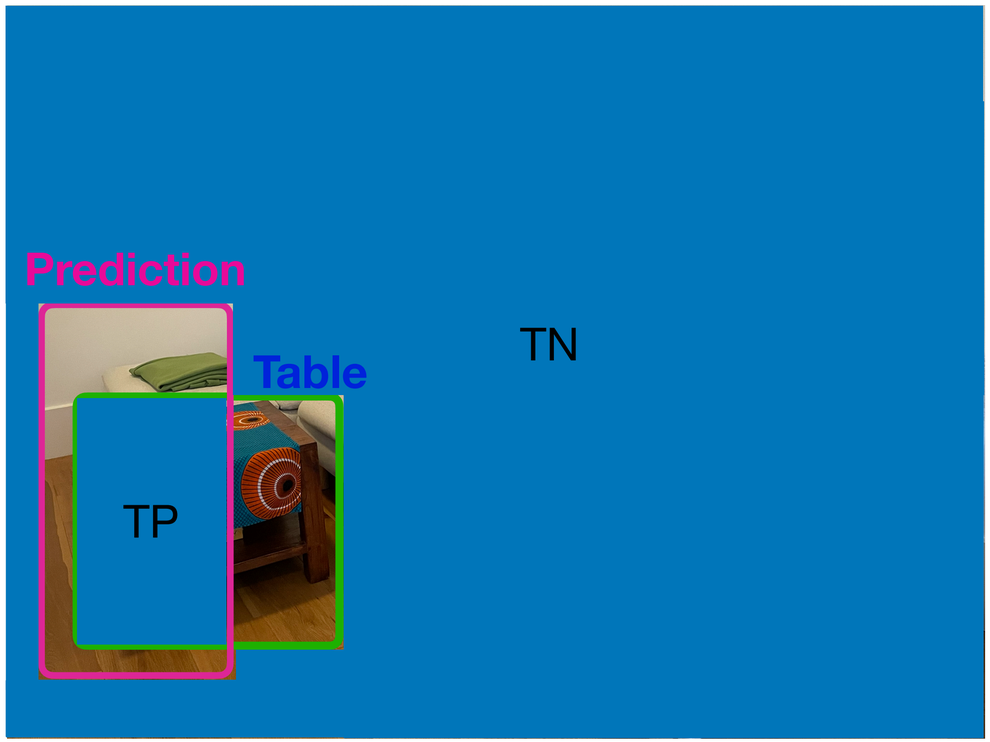
Accuracy (pixel-level)?

Inference: Performance Metrics

TP (true positive)

Accuracy (pixel-level)?

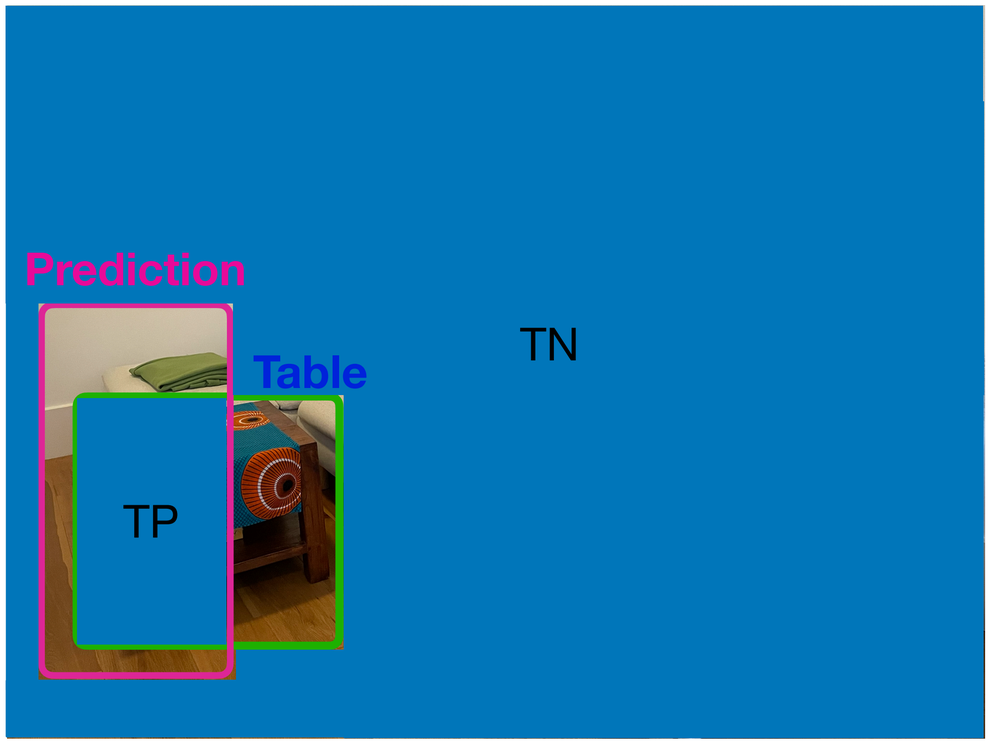
Inference: Performance Metrics

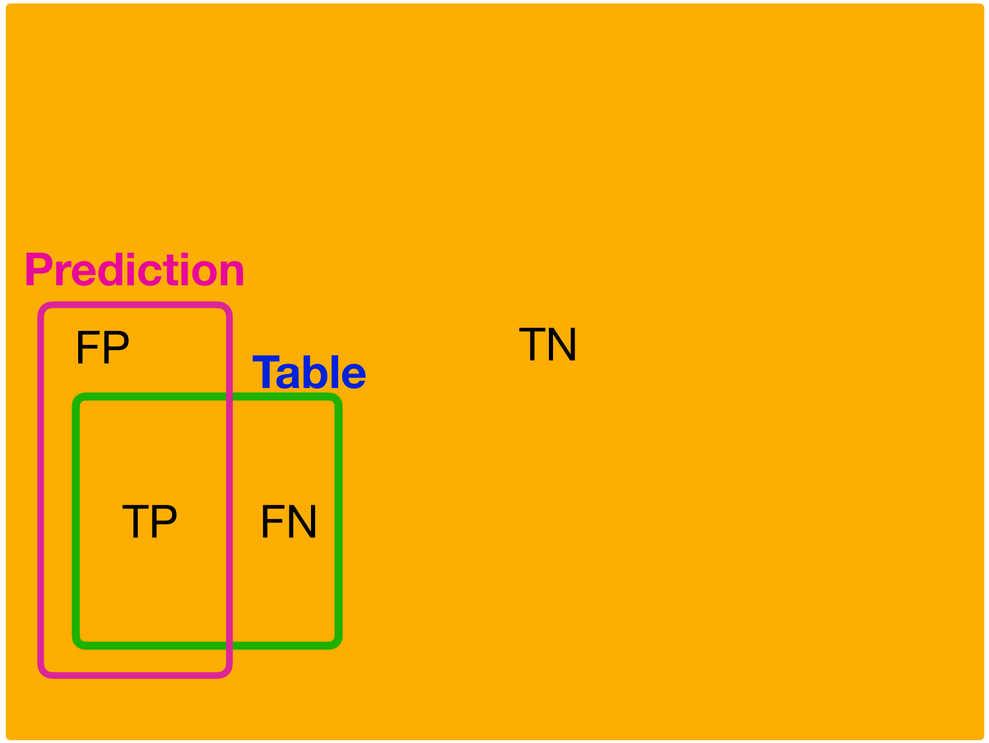


TP + TN (correctly classified)

Accuracy (pixel-level)?

Inference: Performance Metrics

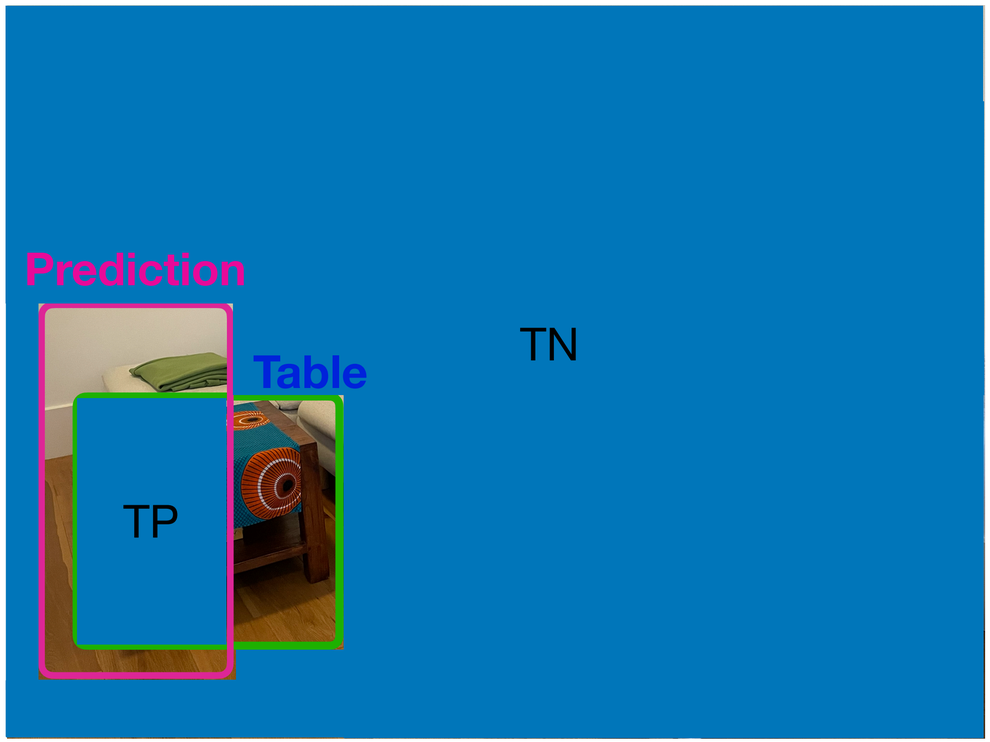


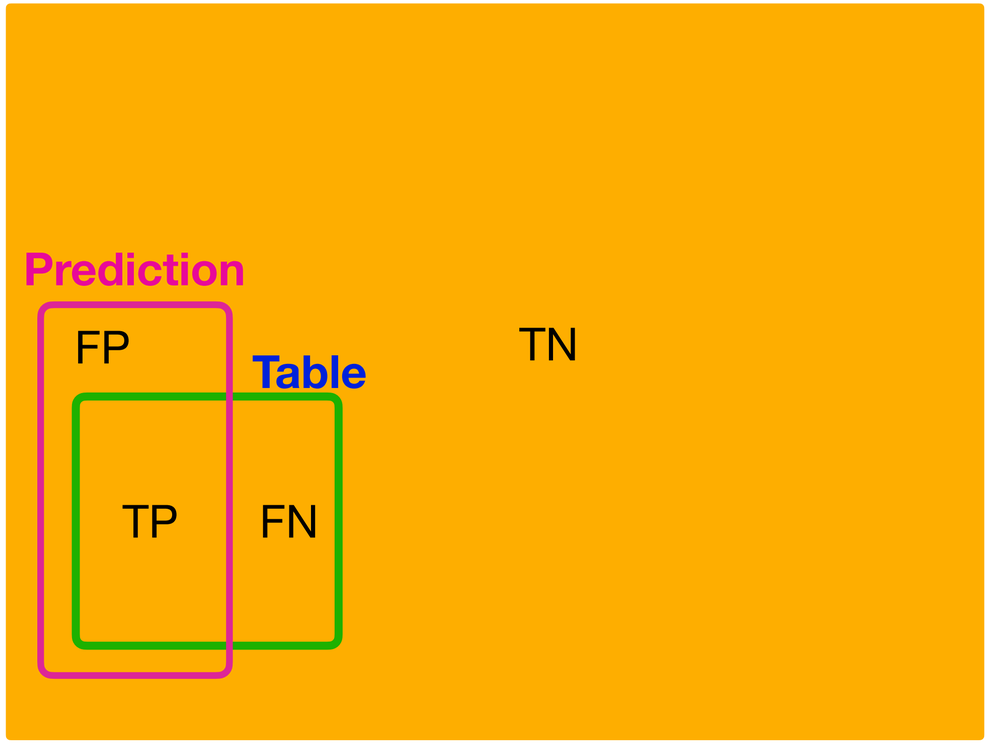


Accuracy (pixel-level)?

$$\frac{TP + TN}{TP + TN + FP + FN} =$$

Inference: Performance Metrics





~~Accuracy (pixel-level)~~

$$\frac{TP + TN}{TP + TN + FP + FN} =$$

Intersection over Union (IoU)

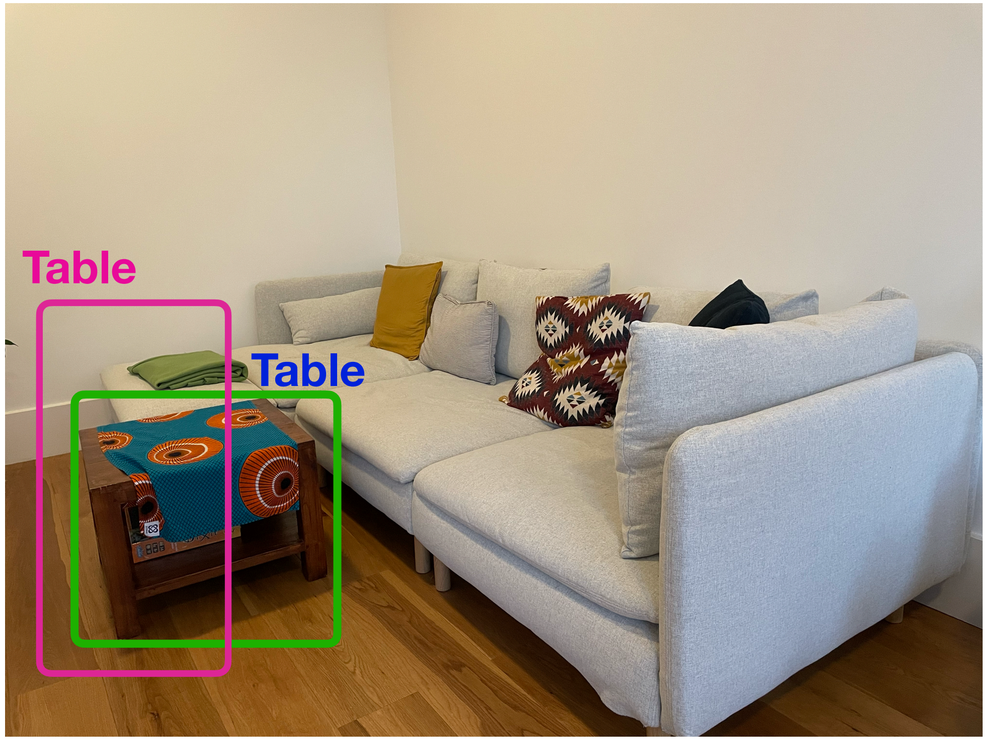
Intersection over union (IoU) =  $= \frac{TP}{FP+TP+FN}$

Intersection over Union (IoU)

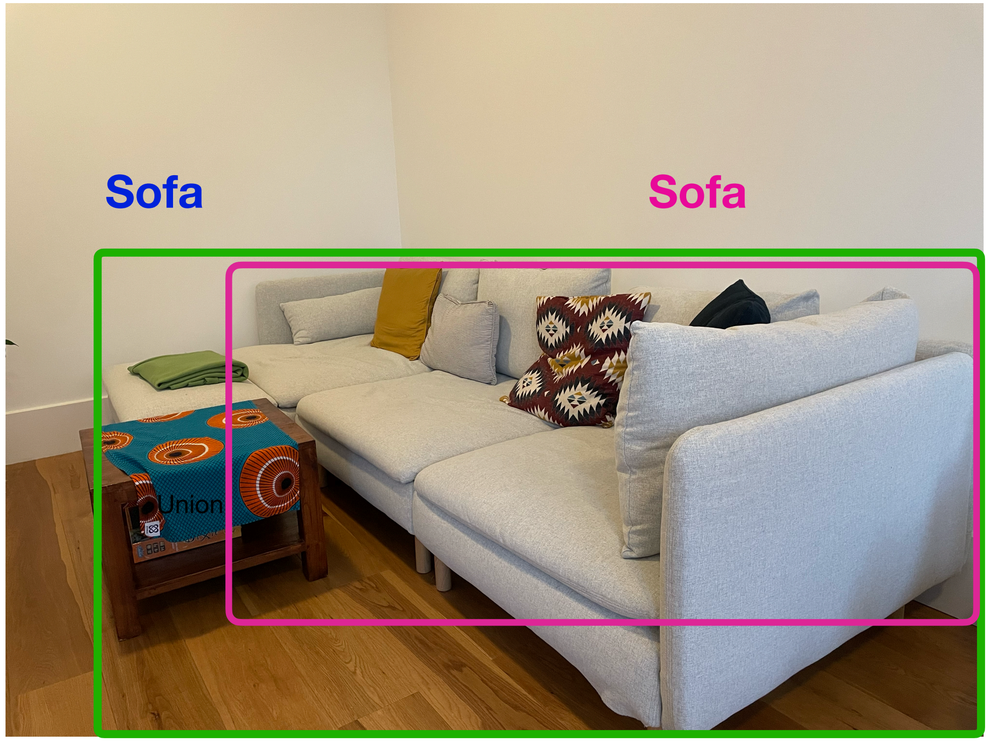
Intersection over union (IoU) =  $= \frac{TP}{FP+TP+FN}$

If IoU > threshold (0.5 - 0.75) the object is considered correctly localized

Performance Metrics



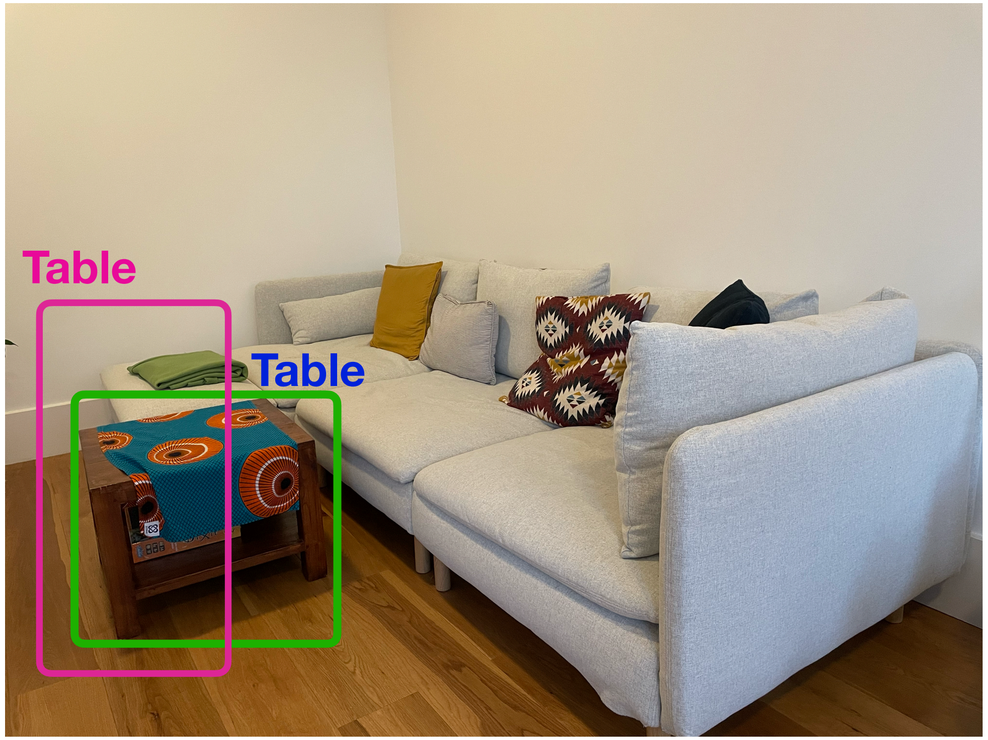
IoU<0.5



IoU>0.5

IoU threshold = 0.5

Performance Metrics

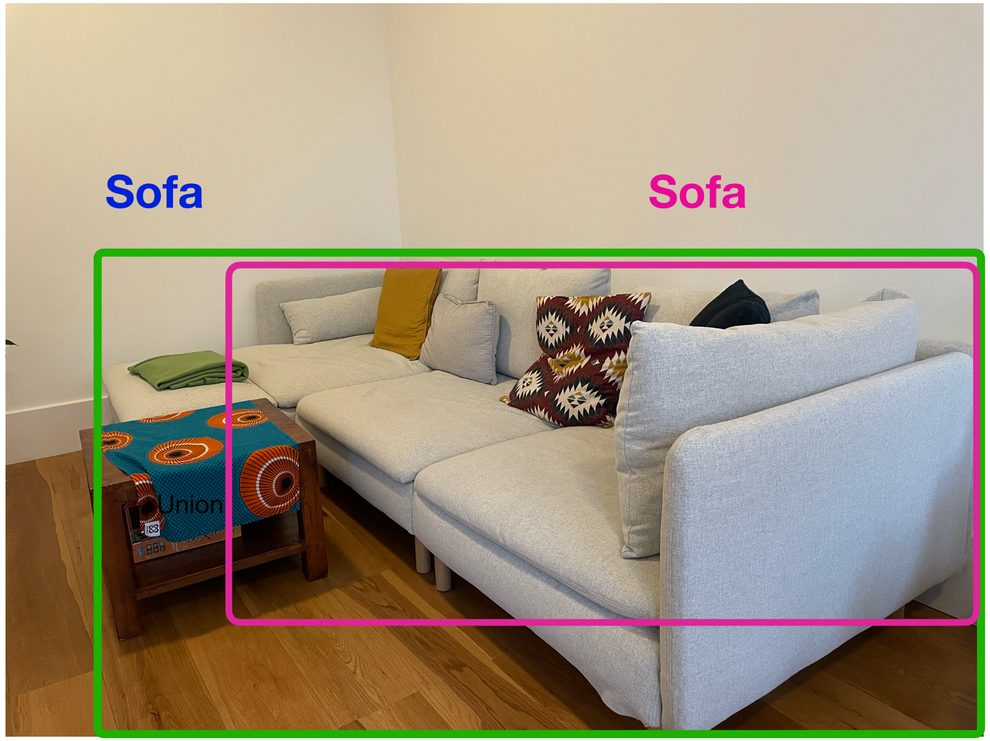


IoU<0.5

↓

Incorrectly localized

False positive



IoU>0.5

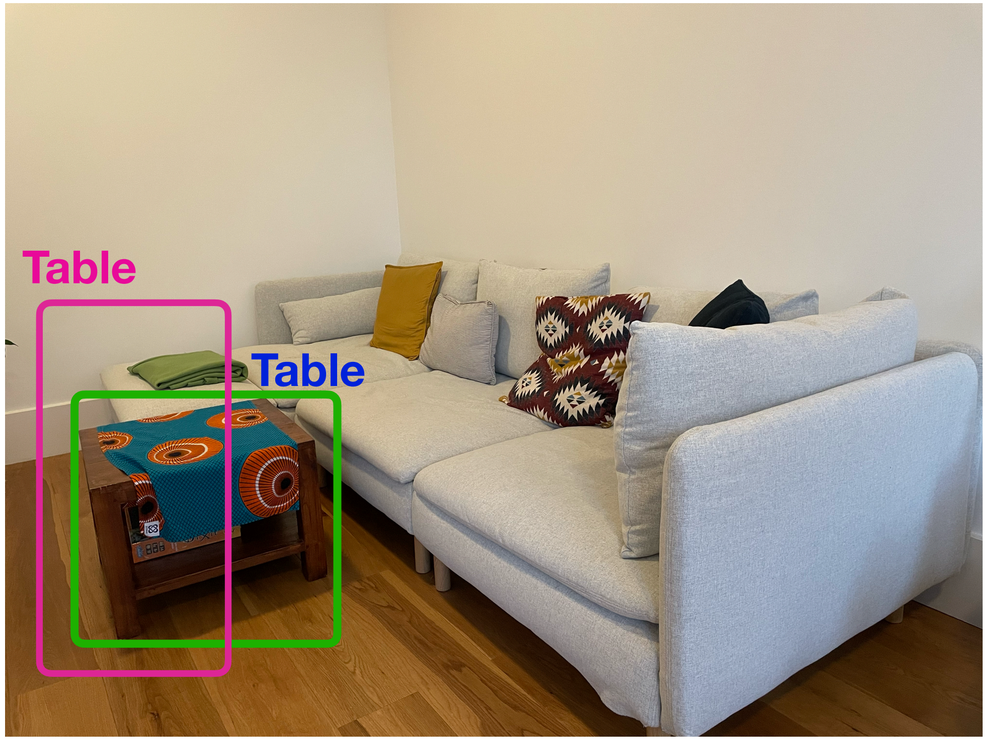
↓

Correctly localized

True positive

IoU threshold = 0.5

Performance Metrics



Class object: table

- precision = correctly localized tables/number of predicted tables
- recall = correctly localized tables/number of actual tables

IoU<0.5

↓

Incorrectly localized

False positive

IoU>0.5

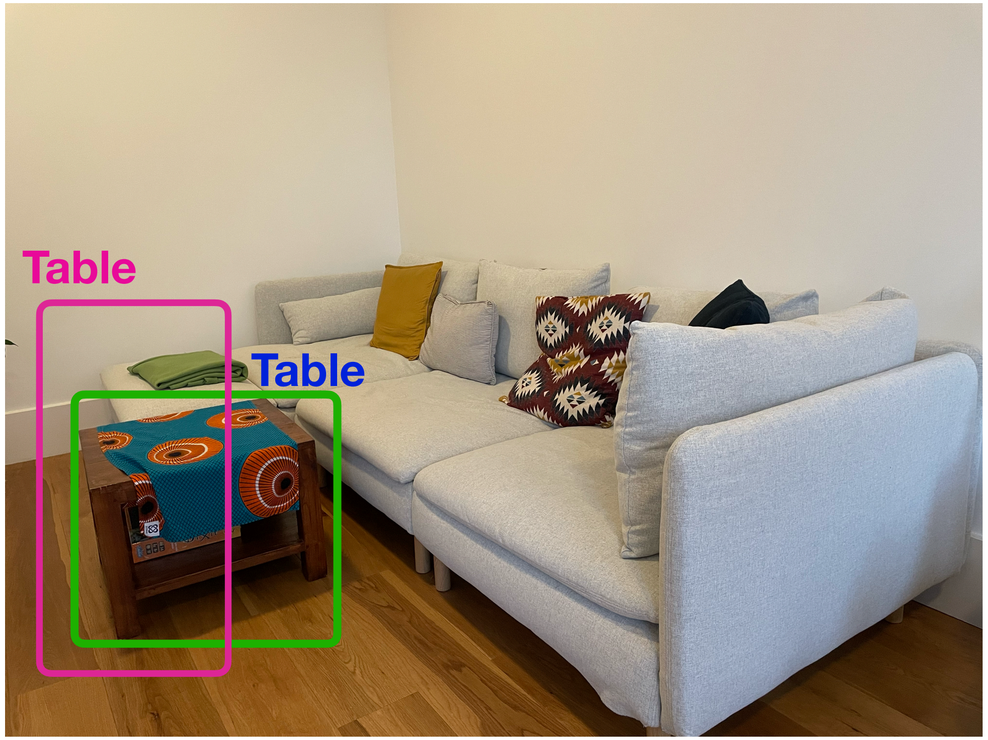
↓

Correctly localized

True positive

IoU threshold = 0.5

Performance Metrics: mean Average Precision



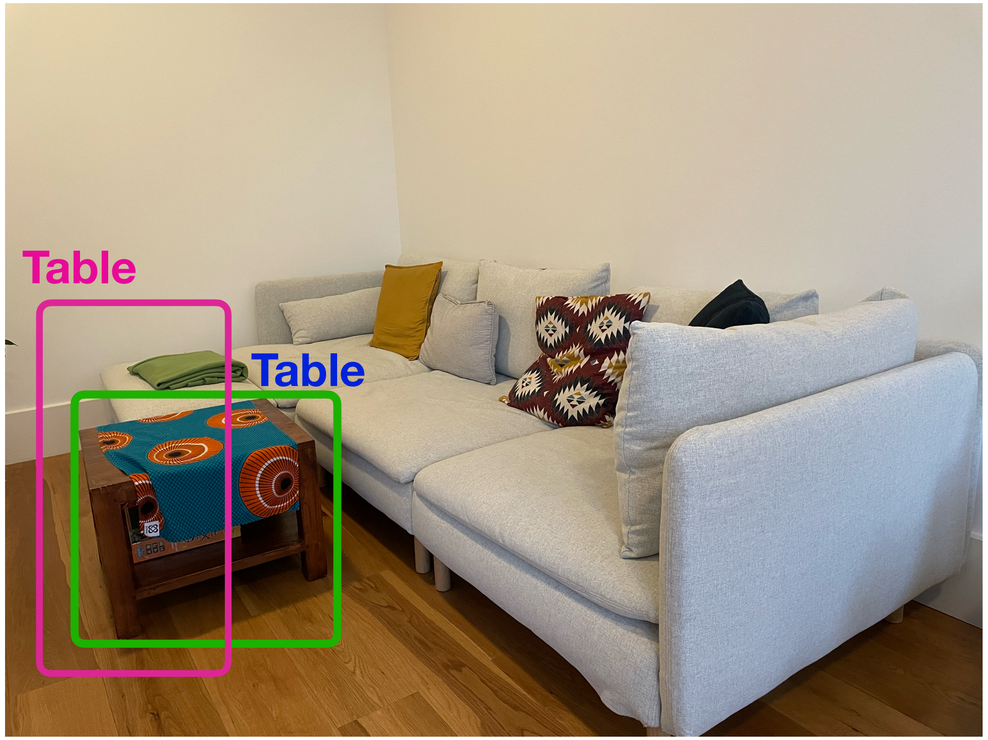
Class object: table

Rank the table detections according to the confidence score

Detection rank	Correct	Precision	Recall
$\lambda = 0.9$ 1	True		
$\lambda = 0.8$ 2	True		
$\lambda = 0.79$ 3	False		
4	False		
5	False		
6	True		

IoU threshold

Performance Metrics: mean Average Precision



Class object: table

Rank the table detections according to the confidence score

Detection rank	Correct	Precision	Recall
1	True		
2	True		
3	False		
4	False		
5	False		
6	True		

$$\text{precision}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i}$$

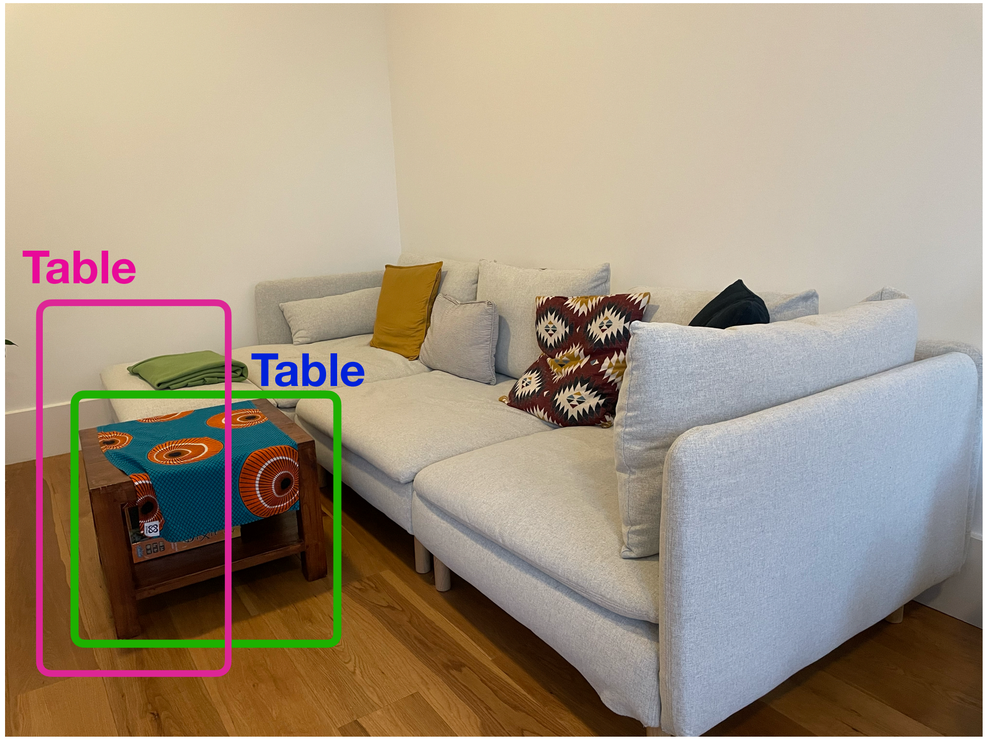
$$\text{recall}_i = \frac{\text{TP}_i}{\text{total number of ground truth tables}}$$

TP_i and FP_i are the cumulative true and false positive up to row i

Total number of ground truth tables = 5

IoU threshold

Performance Metrics: mean Average Precision



Class object: table

Rank the table detections according to the confidence score

Detection rank	Correct	Precision	Recall
1	True	1	0.2
2	True	?	?
3	False		
4	False		
5	False		
6	True		

$$\text{precision}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i}$$

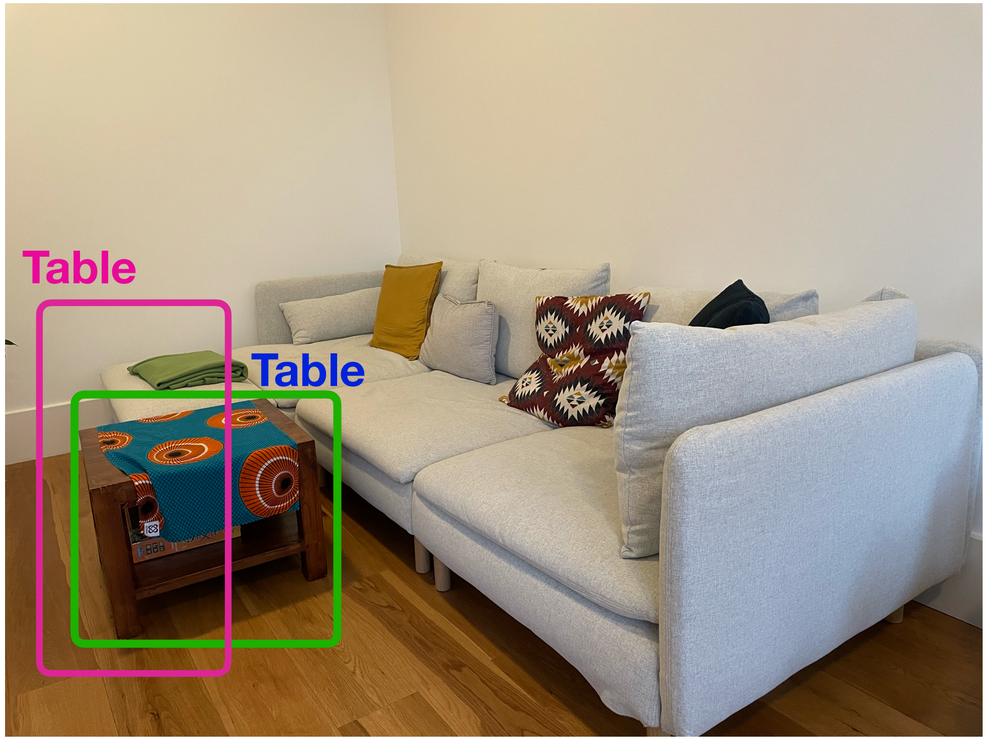
$$\text{recall}_i = \frac{\text{TP}_i}{\text{total number of ground truth tables}}$$

TP_i and FP_i are the cumulative true and false positive up to row i

Total number of ground truth tables = 5

IoU threshold

Performance Metrics: mean Average Precision



Class object: table

Rank the table detections according to the confidence score

Detection rank	Correct	Precision	Recall
1	True	1	0.2
2	True	**1**	**0.4**
3	False	0.67	0.4
4	False	0.5	0.4
5	False	0.4	0.4
6	True	0.5	0.6

$$\text{precision}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i}$$

$$\text{recall}_i = \frac{\text{TP}_i}{\text{total number of ground truth tables}}$$

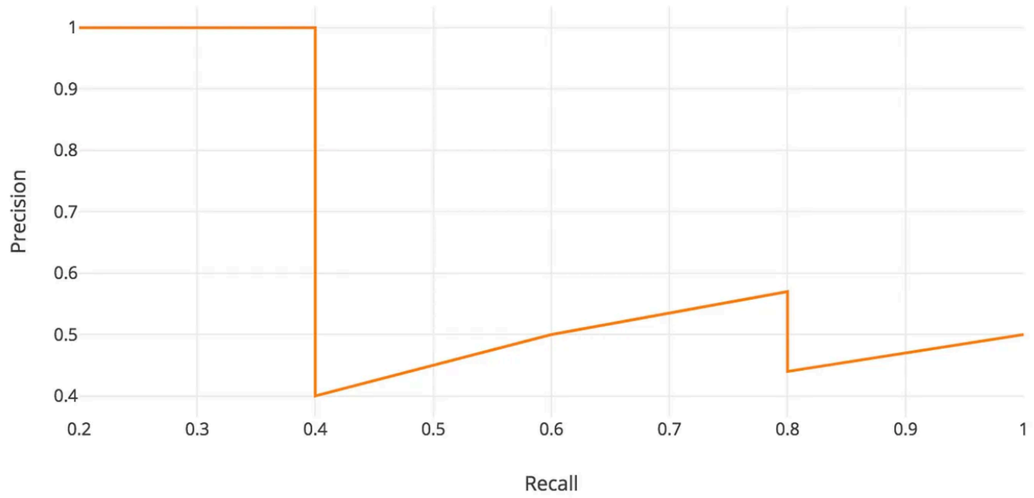
TP_i and FP_i are the cumulative true and false positive up to row i

Total number of ground truth tables = 5

IoU threshold

Performance Metrics: mean Average Precision

Precision-Recall curve



Class object: table

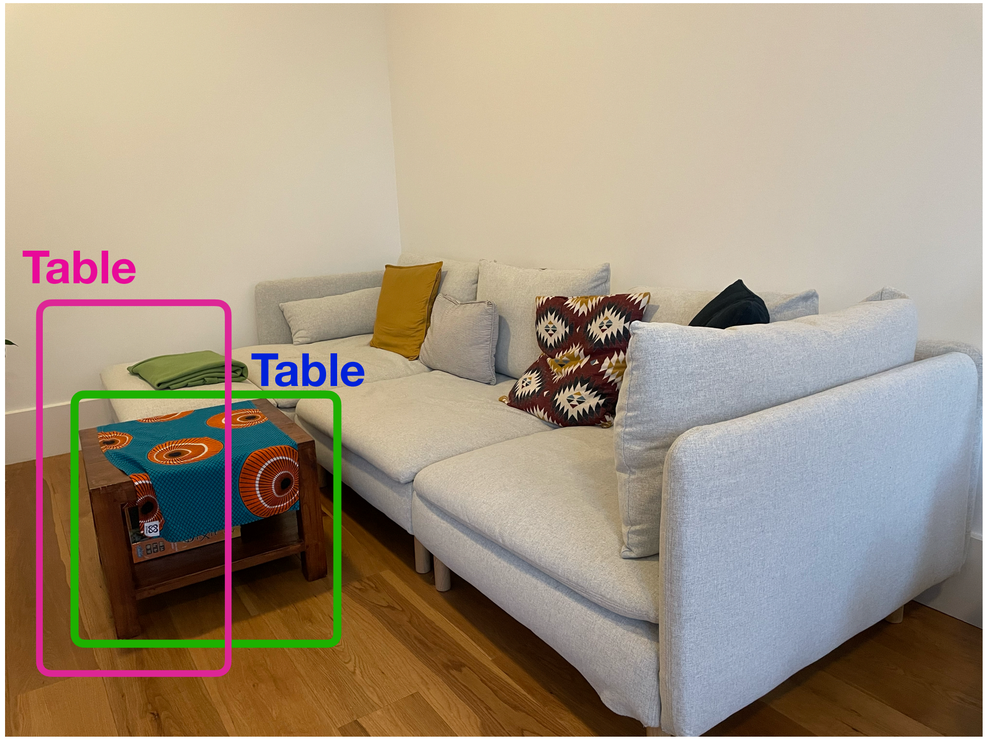
Rank the table detections according to the confidence score

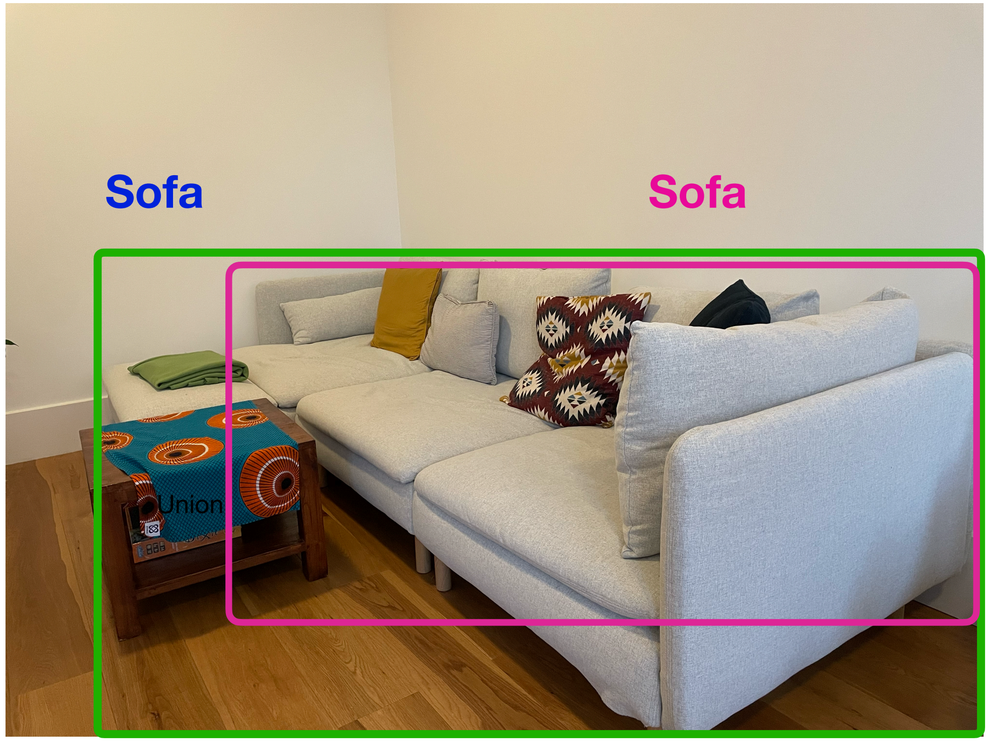
Detection rank	Correct	Precision	Recall
1	True	1	0.2
2	True	1	0.4
3	False	0.67	0.4
4	False	0.5	0.4
5	False	0.4	0.4
6	True	0.5	0.6

Average Precision (AP) = area under the precision-recall curve

IoU threshold

Performance Metrics: mean Average Precision





Average Precision (AP) = area under the precision-recall curve

Mean Average Precision (mAP)
mean over the object classes of the AP for a given IoU threshold or the average mAP over different thresholds [0.5, 0.55, … 0.9, 0.95]

Learning Tasks: Semantic Segmentation

Input image

Object detection

Semantic Segmentation

Input image

Object detection

Segmentation map/mask

Semantic Segmentation

Input image

Object detection

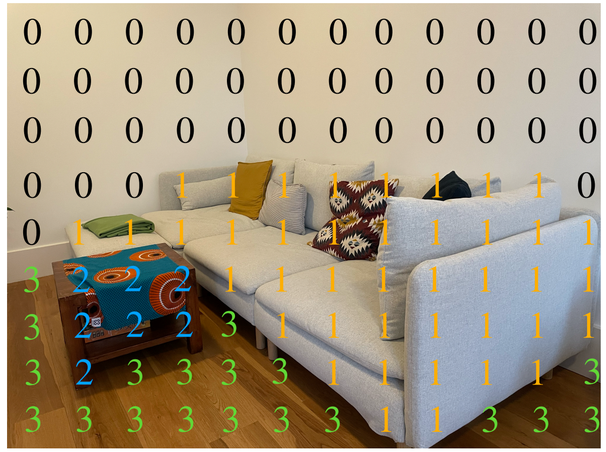
Segmentation map/mask

Per-pixel classification problem

Semantic Segmentation

Input image

Label for the classification task



Segmentation map/mask

Per-pixel classification problem

Classes

Wall 0

Sofa 1

Table 2

Floor 3

Semantic Segmentation

Input image

Label for the classification task

0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1
3	2	2	2	1	1	1	1	1	1	1	1
3	2	2	2	3	1	1	1	1	1	1	1
3	2	3	3	3	3	1	1	1	1	1	3
3	3	3	3	3	3	3	1	1	3	3	3

Segmentation map/mask

Per-pixel classification problem

Classes

Wall 0

Sofa 1

Table 2

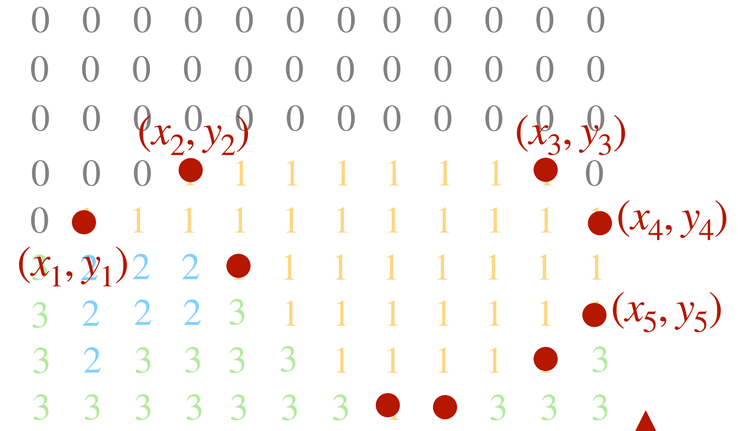
Floor 3

$$y = [[x_1, y_1, x_2, y_2, \ldots, x_1, y_1]_{wall}\ [x_1, y_1, x_2, y_2, \ldots, x_1, y_1]_{sofa} \ldots]$$

Semantic Segmentation

Input image

Label for the classification task



Segmentation map/mask

Per-pixel classification problem

Classes

Wall 0

Sofa 1

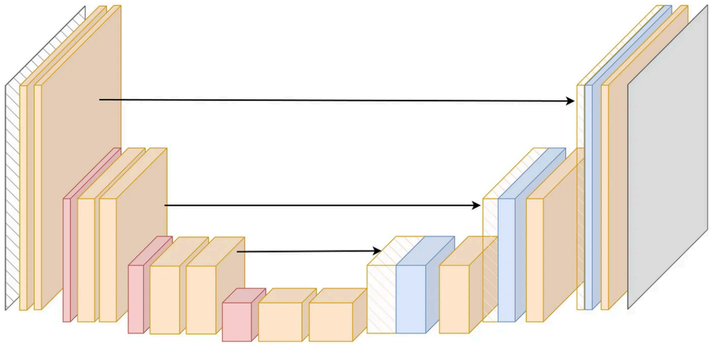
Table 2

Floor 3

$$y = [[x_1, y_1, x_2, y_2, \ldots, x_1, y_1]_{wall}\ [x_1, y_1, x_2, y_2, \ldots, x_1, y_1]_{sofa} \ldots]$$

U-Net* (idea)

Input image



0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1
3	2	2	2	1	1	1	1	1	1	1	1
3	2	2	2	3	1	1	1	1	1	1	1
3	2	3	3	3	3	1	1	1	1	1	3
3	3	3	3	3	3	3	1	1	3	3	3

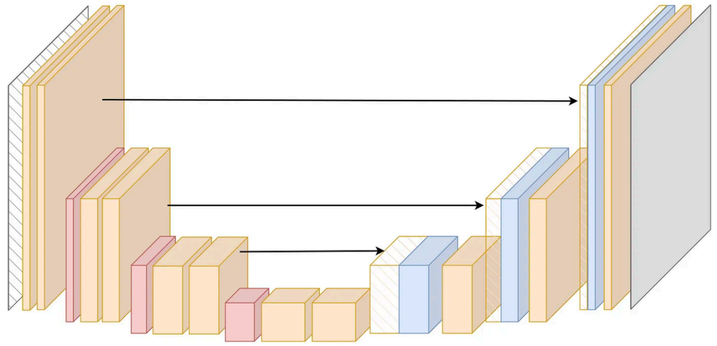
Segmentation map/mask

Output is an image!

*Ronneberger et al. U-Net: Convolutional Networks for Biomedical Image Segmentation (2015)

U-Net* (idea)

Input image



0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1
3	2	2	2	1	1	1	1	1	1	1	1
3	2	2	2	3	1	1	1	1	1	1	1
3	2	3	3	3	3	1	1	1	1	1	3
3	3	3	3	3	3	3	1	1	3	3	3

Segmentation map/mask

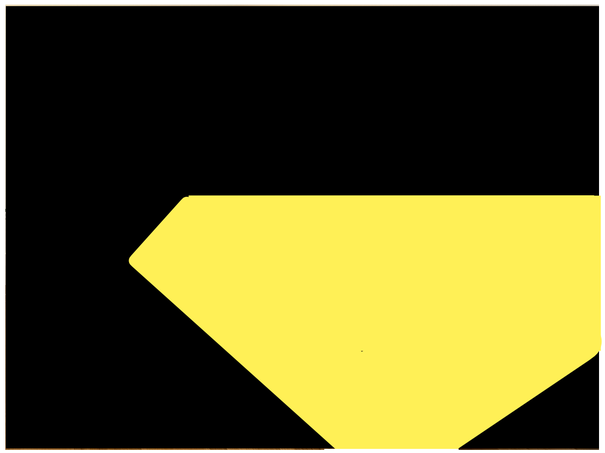
Output is an image!

Loss function: Cross entropy , IoU loss

*Ronneberger et al. U-Net: Convolutional Networks for Biomedical Image Segmentation (2015)

Inference: Performance Metrics

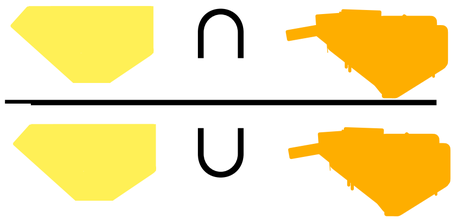
Input image



Output mask

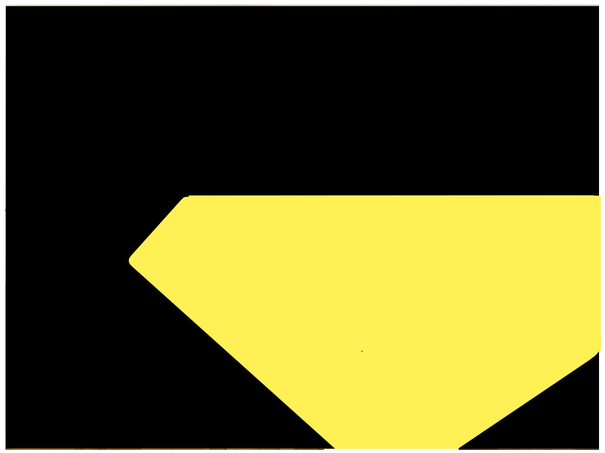
Segmentation map/mask

- IoU



Inference: Performance Metrics

Input image



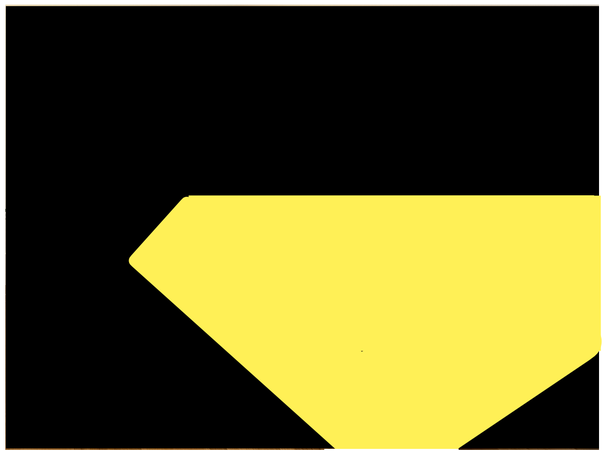
Output mask

Segmentation map/mask

- IoU
- Accuracy, Precision, Recall (pixel-wise)

Inference: Performance Metrics

Input image



Output mask

Segmentation map/mask

- IoU
- Accuracy, Precision, Recall (pixel-wise)
- F1 score or Dice coefficient $= 2\frac{\text{TP}}{2\text{TP} + \text{FP} + \text{FN}} = 2\frac{\cap}{+}$

Summary

Topics

- Convolution Neural Network architectures
- Advanced techniques: residual connections, transfer learning
- Object Detection, Semantic Segmentation

Reading material

- Understand Deep Learning - Chapter 11.1, 11.2
- Andrew NG, C4W2L02 - Classic Networks, Transfer Learning, Object Localization, Object Detection, Semantic Segmentation
- CS231n: Deep Learning for Computer Vision Stanford - Transfer Learning