Course number: 80240743

Deep Learning

Xiaolin Hu (胡晓林) & Jun Zhu (朱军)
Dept. of Computer Science and Technology
Tsinghua University

Last lecture review

Part 1

Part 2

Conv | Max pooling | Avg pooling | Full connection

ReLU layer | Sigmoid layer | Cross-entropy loss | …

Part 3

AlexNet | VGGNet | GoogLeNet | ResNet

Inception v2, v3 | DenseNet | …

Part 4

Adagrad → RMSProp → Adam

Early stop → Dropout → Data augmentation

Multiple Choice(multiple) Points: 1

Setting

Which of the following are used to obtain Inception-v2 from Inception-v1 (GoogLeNet)

A Factorize 5x5 conv to two 3x3 conv

B Factorize nxn conv to 1xn and nx1 conv

C Factorize nxn conv to 3xn and nx3 conv

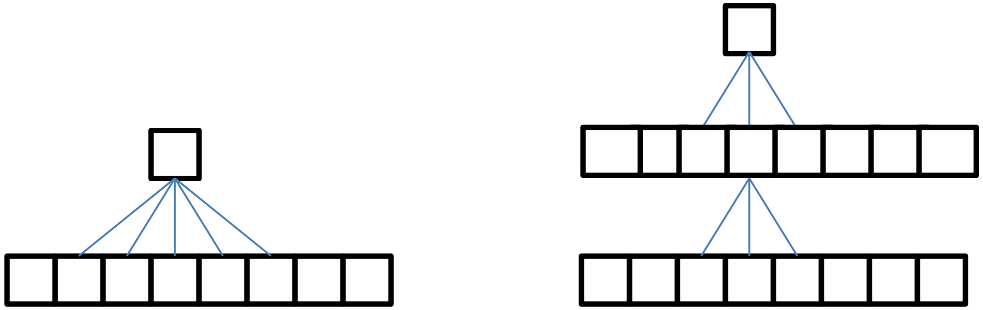
D Split every Inception module to 32 branches

Submit

Multiple Choice(single) Points: 1

Setting

If you factorize 5x5 conv to two 3x3 conv, does the RFs of the neurons change?



A Yes

B No

Submit

Presentation

Form groups of 2 and every group prepares a 5-minute presentation with slides for one of the following papers

- Hu, Shen, Sun (2018) Squeeze-and-Excitation Networks, CVPR
- Li, Wang, Hu, Yang (2019) Selective Kernel Networks, CVPR

Lecture 6: Applications of CNN in Computer Vision

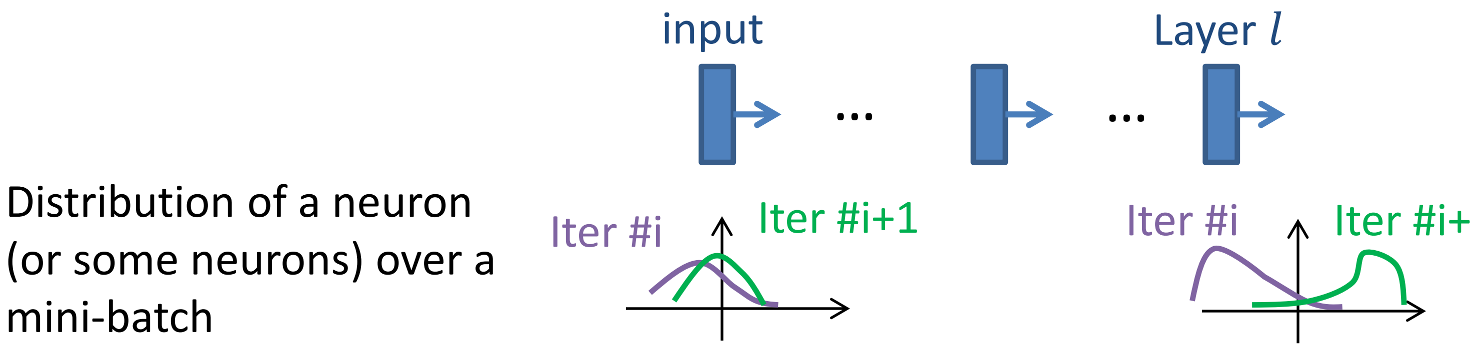
Xiaolin Hu
Dept. of Computer Science and Technology
Tsinghua University

Outline

1. Training techniques-III $\longrightarrow$ Batch normalization
2. Image classification
3. Object detection
4. Summary

“Internal covariate shift”

- Since we use SGD, the input mini-batches to the neural network at different iterations are different
- This *may* cause the distributions of the output of a layer to be different at different iterations



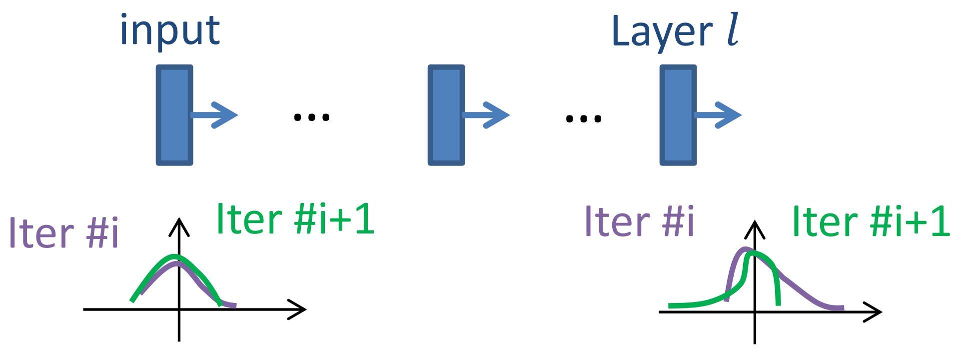
Internal Covariate Shift (ICS): The change in the distributions of internal nodes of a deep network, in the course of training

It *may* cause difficulty in optimization

Reduce ICS by normalization

Ioffe, Szegedy, 2015

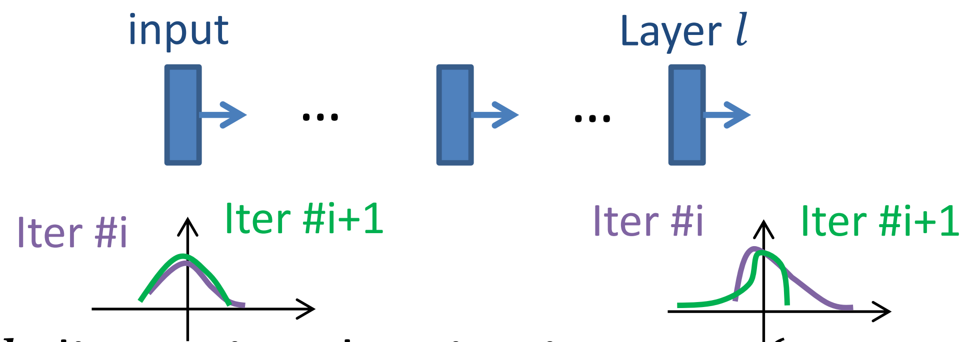
- We normalize each scalar feature independently, by making it have the mean of zero and the variance of 1



Reduce ICS by normalization

Ioffe, Szegedy, 2015

- We normalize each scalar feature independently, by making it have the mean of zero and the variance of 1



- Denote a d-dimensional activation $\boldsymbol{x} = (x_1, \dots, x_d)$, do normalization

$$\hat{x}_i = \frac{x_i - \mathrm{E}[x_i]}{\sqrt{\mathrm{Var}[x_i]}}$$

- Keep the representation ability of the layer

$$y_i = \gamma_i \hat{x}_i + \beta_i$$

If $\gamma_i = \sqrt{\mathrm{Var}[x_i]}$ and $\beta_i = \mathrm{E}[x_i]$, then we recover the original activations

Batch normalization

- Construct a new layer:
$$\boldsymbol{y}^{(n)} = BN_{\boldsymbol{\gamma},\boldsymbol{\beta}}\left(\boldsymbol{x}^{(n)}\right)$$
- Where $\boldsymbol{\gamma}, \boldsymbol{\beta}, \boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)} \in R^d$
- Forward pass
  - $\boldsymbol{\mu}_B = \frac{1}{m}\sum_{n=1}^{M} \boldsymbol{x}^{(n)}$
  - $\boldsymbol{\sigma}_B^2 = \frac{1}{m}\sum_{n=1}^{M}\left(\boldsymbol{x}^{(n)} - \boldsymbol{\mu}_B\right)^2$
  - $\widehat{\boldsymbol{x}}^{(n)} = \frac{\boldsymbol{x}^{(n)} - \boldsymbol{\mu}_B}{\sqrt{\boldsymbol{\sigma}_B^2 + \epsilon}}$
  - $\boldsymbol{y}^{(n)} = \boldsymbol{\gamma}\widehat{\boldsymbol{x}}^{(n)} + \boldsymbol{\beta}$

The arithmetic operations are elementwise

- What are needed to be computed in the backward pass?

During inference

- The normalization is neither necessary nor desirable during inference
- Once the network has been trained, we normalize

$$\widehat{\boldsymbol{x}} = \frac{\boldsymbol{x} - \mathrm{E}[\boldsymbol{x}]}{\sqrt{\mathrm{Var}[\boldsymbol{x}] + \epsilon}}$$

  where $\mathrm{E}[\boldsymbol{x}]$ and $\mathrm{Var}[\boldsymbol{x}]$ are measured over the entire training set

  - $\mathrm{E}[\boldsymbol{x}] = \mathrm{E}_B[\boldsymbol{\mu}_B]$
  - $\mathrm{Var}[\boldsymbol{x}] = \frac{m}{m-1}\mathrm{E}_B[\boldsymbol{\sigma}_B^2]$ where m is the number of mini-batches

- If you want to track the accuracy of a model as it trains, you can use the moving averages instead

Location in a network

- Often applied before the non-linearity of the previous layer
- The previous layer is always a linear transformation layer (fully-connected layer or convolutional layer)

$$\boldsymbol{y}^{(l)} = \boldsymbol{W}^{(l)}\boldsymbol{y}^{(l-1)} + \boldsymbol{b}^{(l)}$$

  – The bias term can be ignored because BN has a shift term that have the same effect. Therefore

$$\boldsymbol{y}^{(l)} = \mathrm{BN}\left(\boldsymbol{W}^{(l)}\boldsymbol{y}^{(l-1)}\right)$$

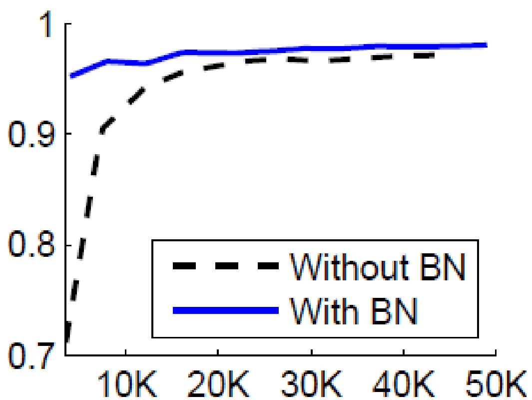
BN in CNN

- As said before, BN is applied after the convolutional layer
- Different elements of the same feature map are normalized in the same way
- Normalize all activations in a mini-batch, over all locations
  - Suppose the mini-batch size is M, and the feature map size is $P \times Q$, then the mean and variance are calculated across $M \cdot P \cdot Q$ elements
  - Learn a pair of parameters γ_i and β_i per feature map, rather than per activation
- The inference procedure is modified similarly, so that during inference BN applies the same linear transformation to each activation in a given feature map

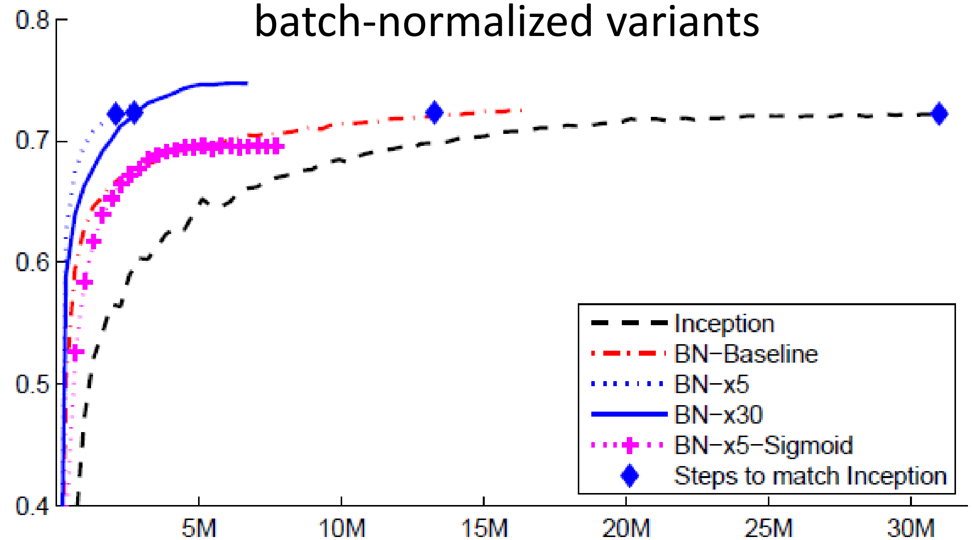
Advantages

- BN enables higher learning rates
- BN regularizes the model
  - The training network no longer produce deterministic values for a given training example
  - Dropout may not be needed (but see (Li, Chen, Hu, Yang, CVPR 2019))

Test accuracy on the MNIST



Single crop validation accuracy of Inception and its batch-normalized variants



X-axis: The number of training steps

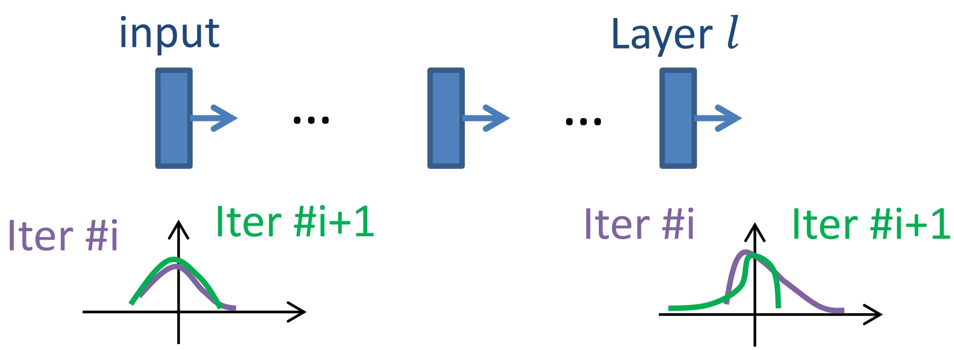
But, the advantage is controversial

- Santurkar, Tsipras, Ilyas, Madry (2018) How does batch normalization help optimization? NeurIPS

Summary of Part 1

Motivation

Reduce "Internal covariate shift"

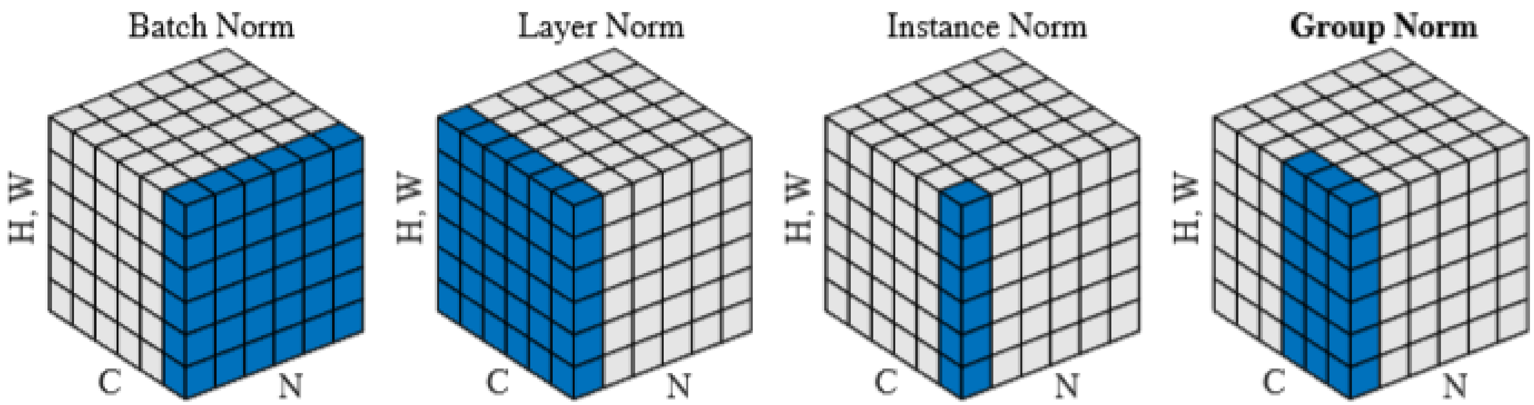


Method

$$\hat{x}_i = \frac{x_i - \mathrm{E}[x_i]}{\sqrt{\mathrm{Var}[x_i]}} \qquad y_i = \gamma_i \hat{x}_i + \beta_i$$

The reason of the good performance is controversial!

Other normalization techniques*



Wu, He, arXiv:1803.08494v3

All of them share the same linear transformation form, but do normalization over different dimensions

- Batch norm is usually used in CNN
- Layer norm is usually used in RNN
- Instance norm is usually used in image stylization
- Group norm is used in CNN to deal with small batchsize

More discussion: https://www.cnblogs.com/LXP-Never/p/11566064.html

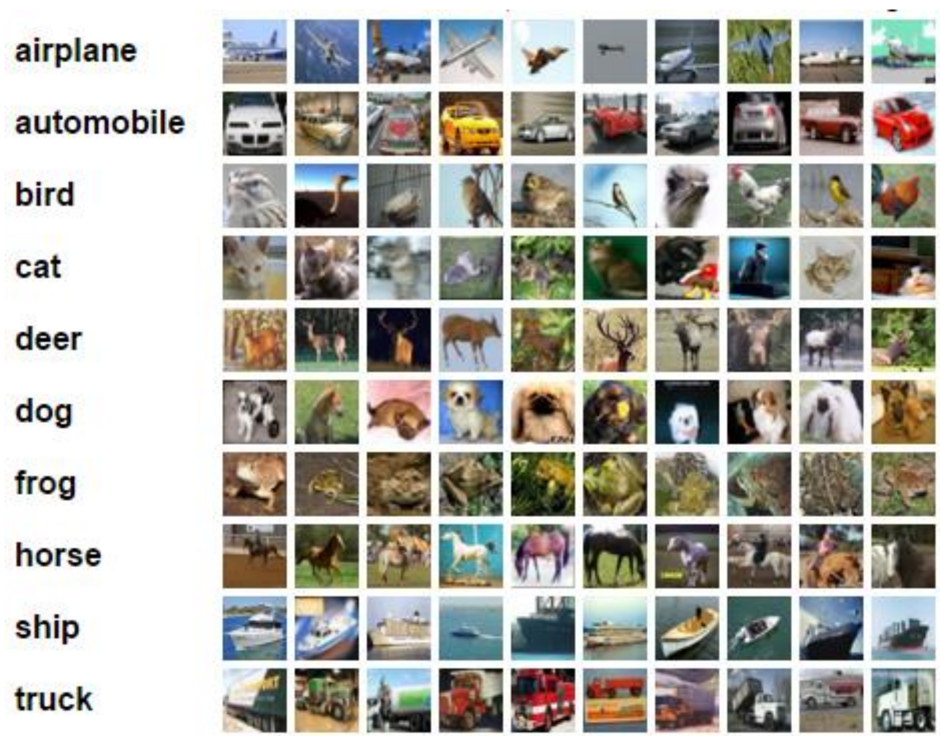
Outline

1. Training techniques-III
2. Image classification
3. Object detection
4. Summary

General image classification

CIFAR-10 & CIFAR100

- 50,000 training images and 10,000 test images
- 32x32 colour images



ImageNet competition (ILSVRC)

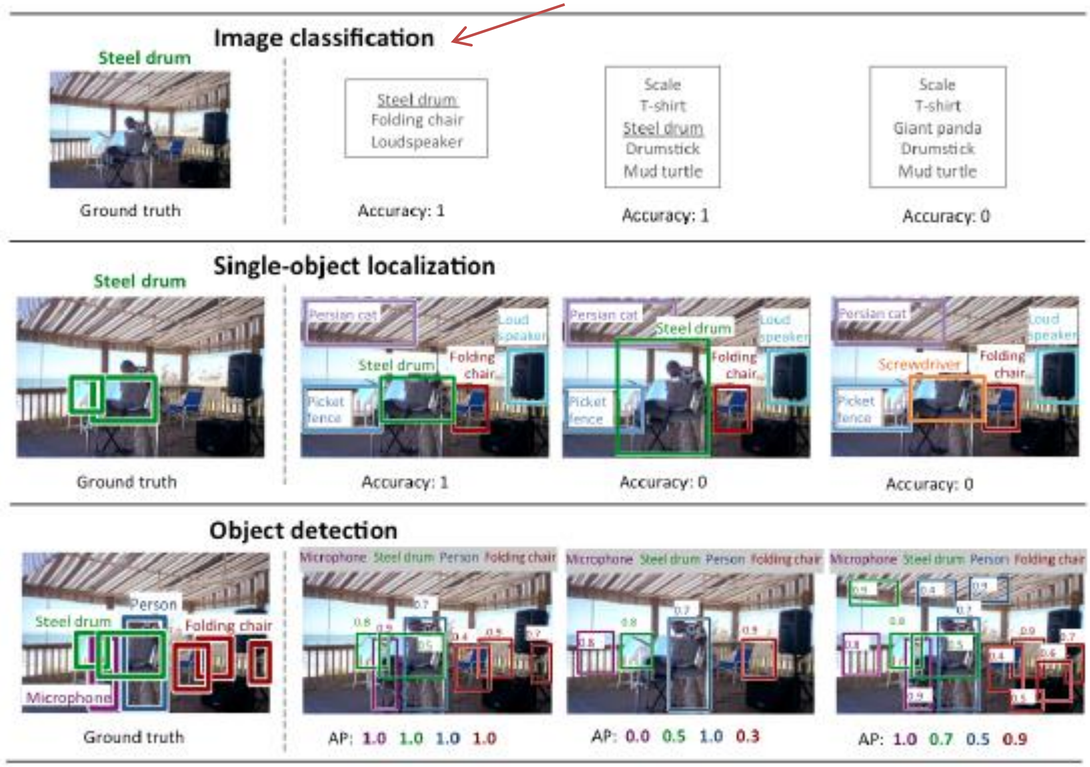
1M training images

Tasks

2010-

2011-

2013-



Top-1

Top-5 (preferred)

Two human expert: 5.1%, 12%

The first column shows the ground truth labeling on an example image, and the next three show three sample outputs with the corresponding evaluation score.

Russakovsky, et al., 2014

Specific image classification




Face verification

Coo d'Este   Melina Kanakaredes

Elijah Wood   Stefano Gabbana

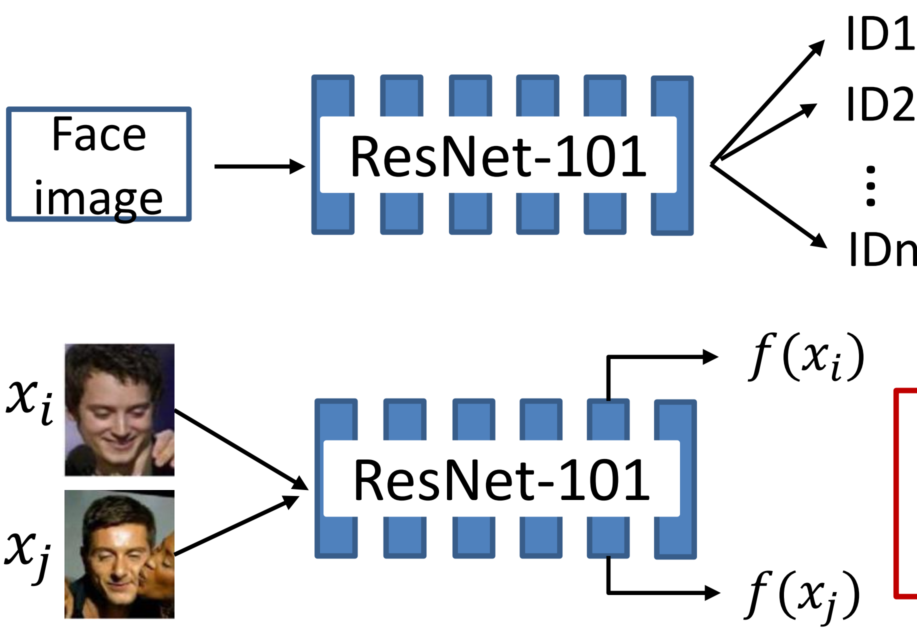
Jim O'Brien   Jim O'Brien

Model	Accuracy (%)
DeepFace (2014)	97.25
DeepID (2014)	97.45
DeepID2 (2014)	99.15
DeepID2+ (2014)	99.47
DeepID3 (2014)	99.53
FaceNet (2015)	99.63

- The task is actually different from image classification

Multi-class classification?

- Can you use a multi-class neural network-based classifier for this task and how?
  - Usually, pretrain such a model, then use the features from the model to do the verification with another model



There exist other methods for discrimination

threshold

If $\left\|f(x_i) - f(x_j)\right\|_2 < \alpha$, same;
Otherwise, different

Motivation

- Any problem with this pipeline?
  - It's not end-to-end. The multi-class classification task is not directly related to discrimination between two images
- Solution: design a loss function for discrimination
  - Option 1: Combine the two losses together → DeepID2

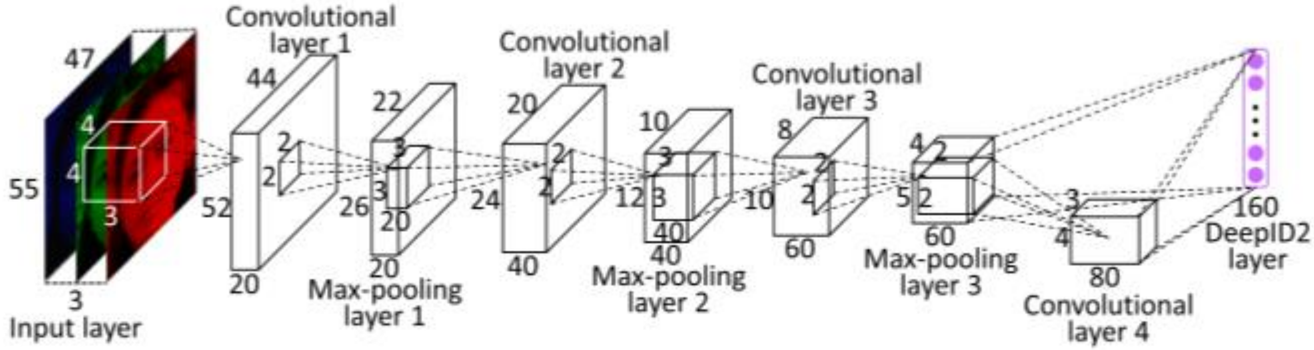
    There are problems for training the classification model

    - Too many classes
      - Calculating the normalization term in softmax is expensive
      - Every softmax output is very small
    - Too few samples in each class

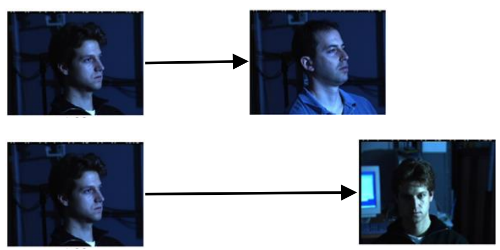
  - Option 2: Use the discrimination loss alone → FaceNet

DeepID2

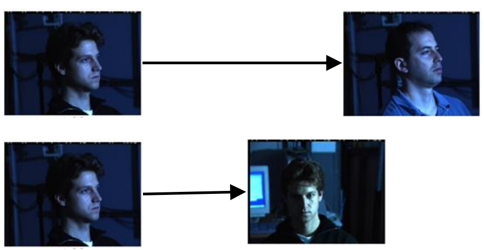
Sun, Wang, Tang, 2014, arXiv:1406,4773v1



- Contrastive loss



After learning



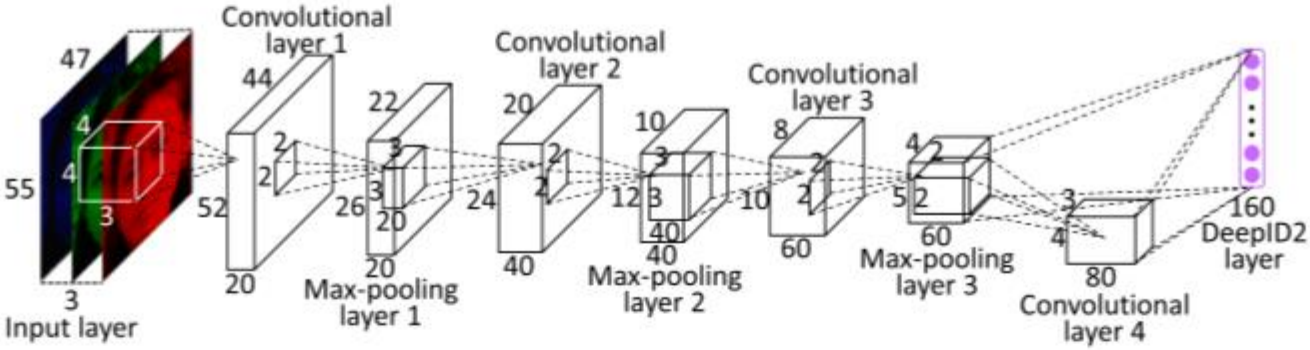
Goal:

$$\left|\left|f(x_i) - f(x_j)\right|\right|_2^2 \text{ is small as possible if } i, j \text{ belong to the same identity;}$$

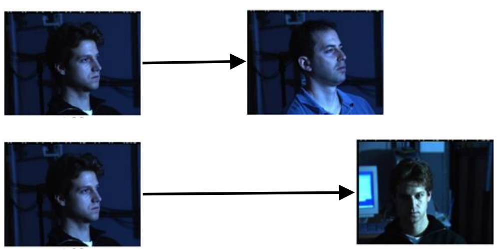
$$\left|\left|f(x_i) - f(x_j)\right|\right|_2^2 > \alpha \text{ otherwise, where } \alpha > 0$$

DeepID2

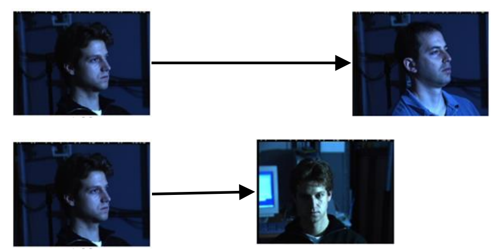
Sun, Wang, Tang, 2014, arXiv:1406,4773v1



- Contrastive loss



After learning

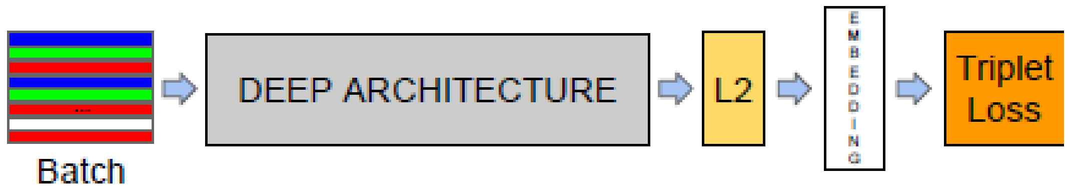


Loss:

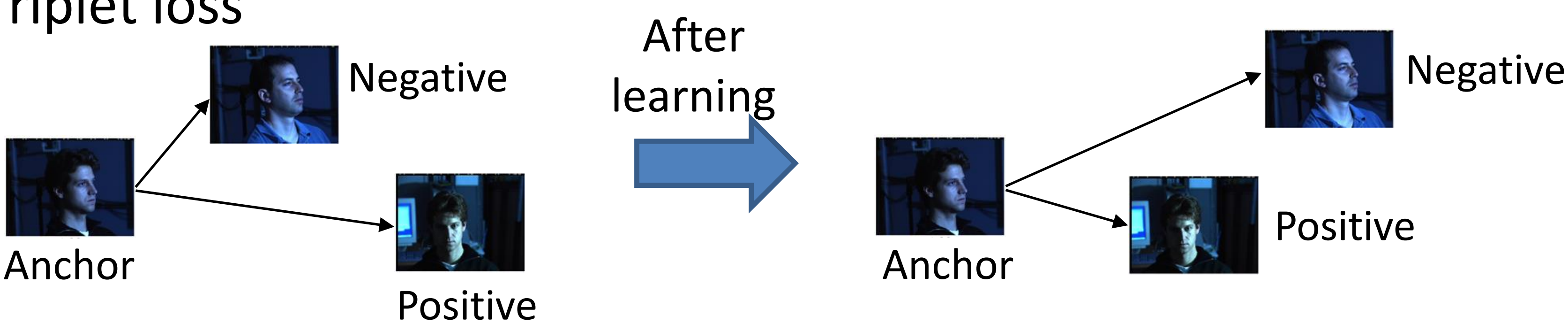
$$L(x_i, x_j, \underset{\substack{\uparrow \\ \text{label}}}{y_{ij}}, \theta) = \begin{cases} \frac{1}{2}\|f(x_i) - f(x_j)\|_2^2, & \text{if } y_{ij} = 1 \\ \frac{1}{2}\max\{0, \alpha - \|f(x_i) - f(x_j)\|_2^2\}, & \text{else.} \end{cases}$$

FaceNet

Schroff et al., CVPR 2015



- Triplet loss



Goal: $\left\| f(x_i^a) - f(x_i^p) \right\|_2^2 + \alpha < \left\| f(x_i^a) - f(x_i^n) \right\|_2^2$ where $\alpha > 0$

Loss: $\sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+$

It indicates at least 3 samples are needed in a minibatch

Any problem with this?

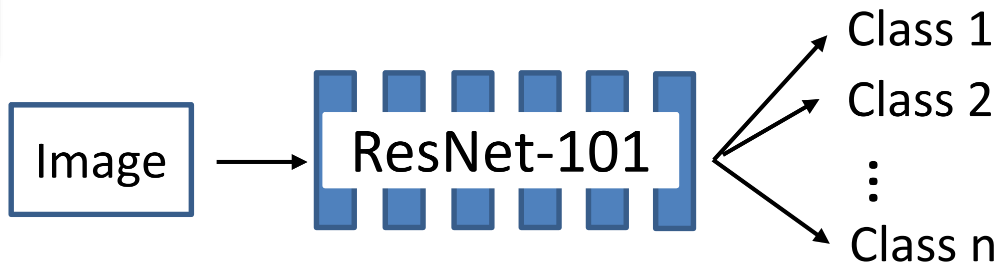
Other losses for face verification

- SphereFace:
  - Deep Hypersphere Embedding for Face Recognition, CVPR 2017
- NormFace:
  - L2 Hypersphere Embedding for Face Verification, ACM MM 2017
- ArcFace:
  - Additive Angular Margin Loss for Deep Face Recognition, CVPR 2019

Summary of Part 2

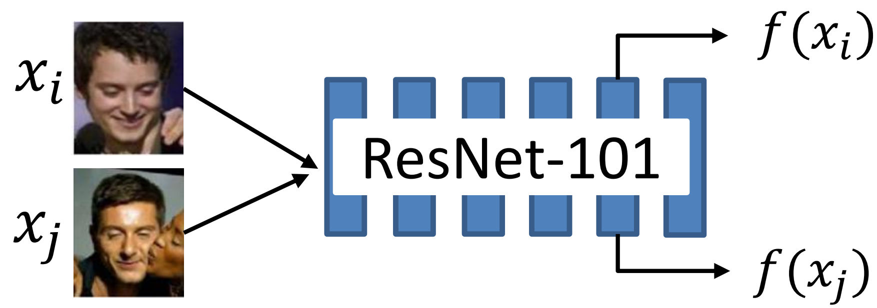
Multi-class classification

Q: Which class does this image belong to?

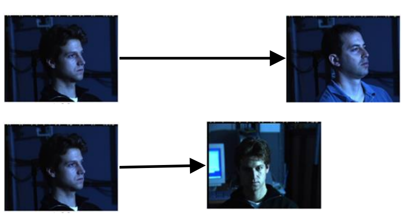


Verification

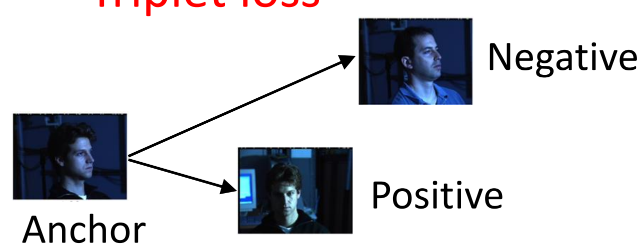
Q: Are the two images for the same person?



Contrastive loss



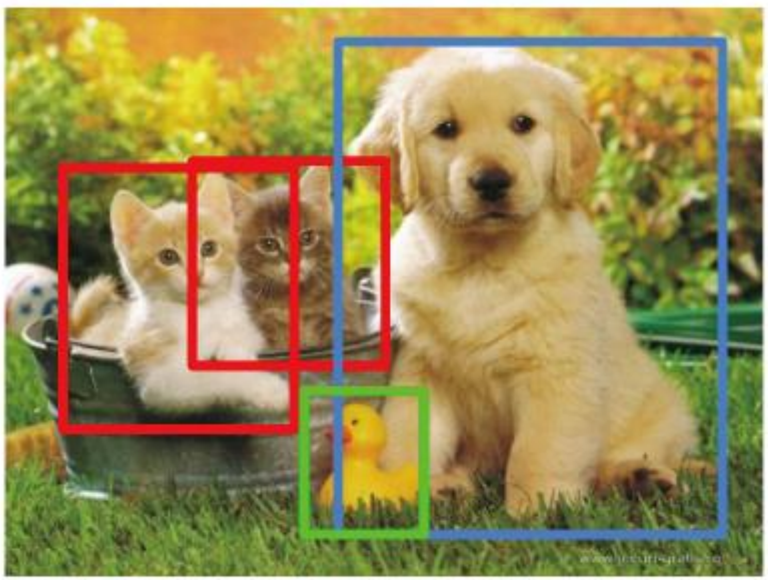
Triplet loss



Outline

1. Training techniques-III
2. Image classification
3. Object detection
4. Summary

Task



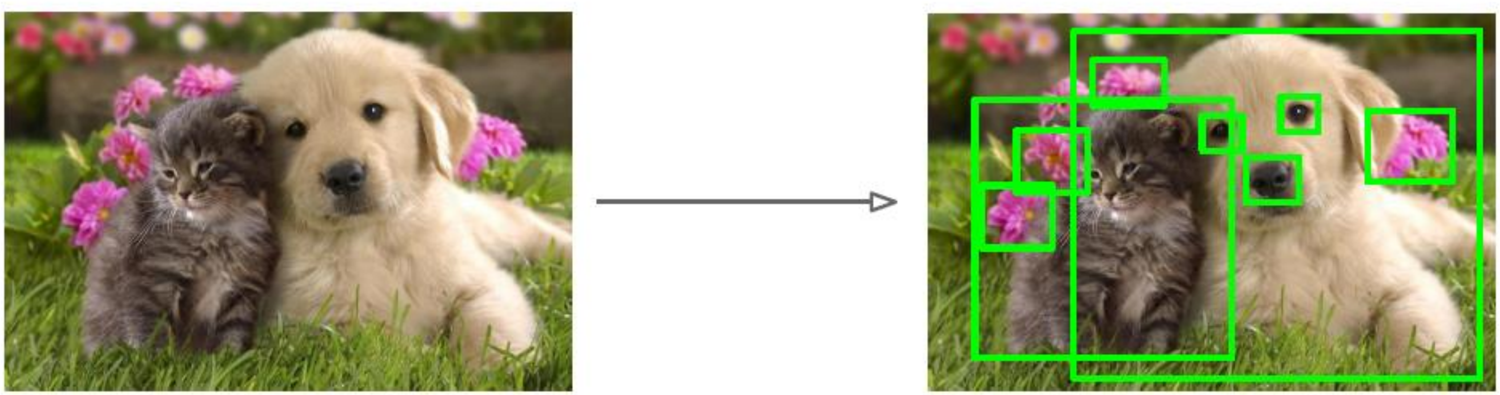
DOG, (x, y, w, h)
CAT, (x, y, w, h)
CAT, (x, y, w, h)
DUCK (x, y, w, h)

How would you do this?

A simple idea: find regions that contain objects, then use a classifier to do object recognition

Region proposals

- Find "blobby" image regions that are likely to contain objects
  - Such regions are called *region proposals*, or *region of interest (RoI)*



Modified from: Fei-Fei Li & Andrej Karpathy & Justin Johnson

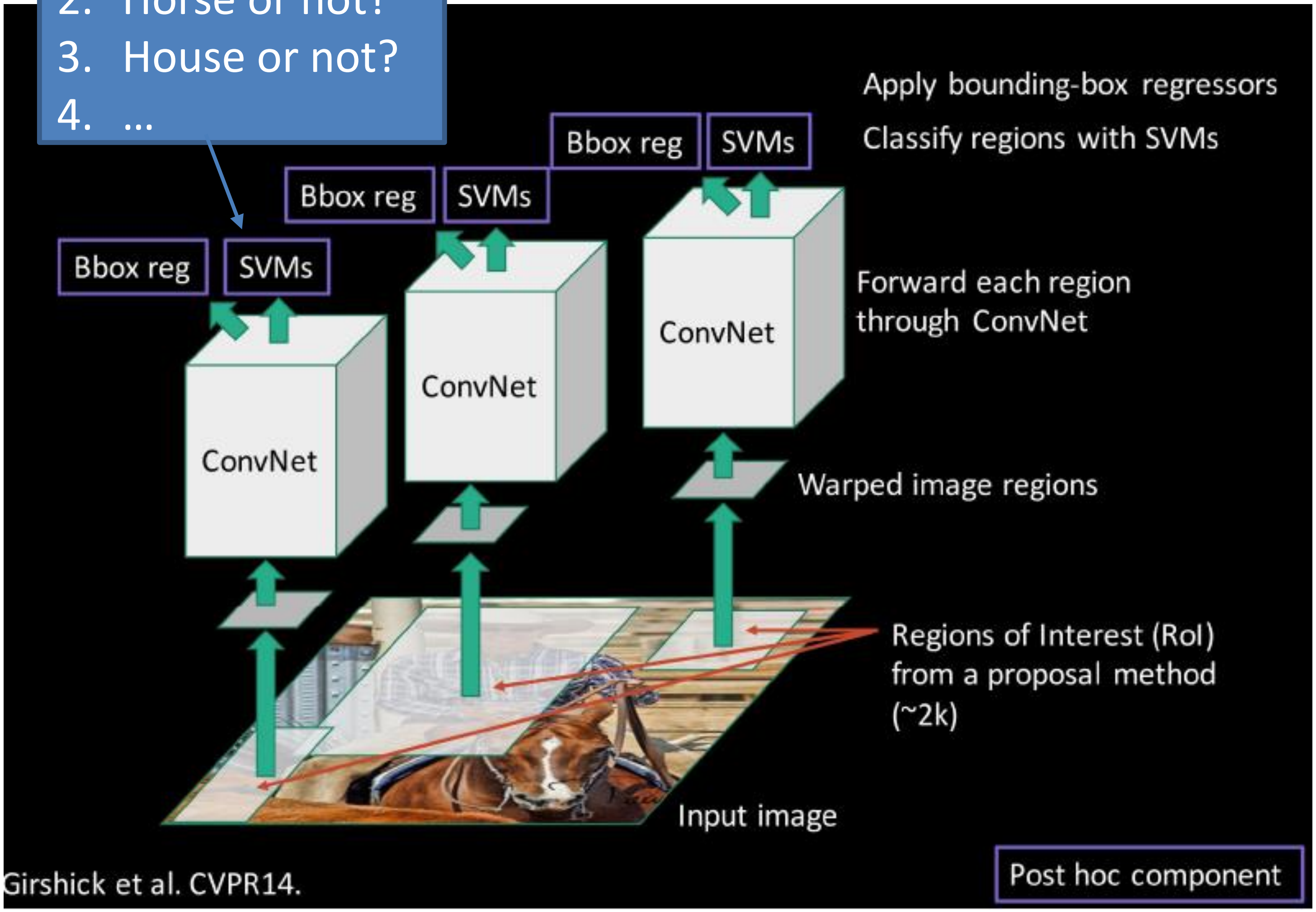
Region proposal: many choices

Method	Approach	Outputs Segments	Outputs Score	Control #proposals	Time (sec.)	Repea-tability	Recall Results	Detection Results
`Bing` [18]	Window scoring		✓	✓	0.2	★★★	★	·
`CPMC` [19]	Grouping	✓	✓	✓	250	-	★★	★
`EdgeBoxes` [20]	Window scoring		✓	✓	0.3	★★	★★★	★★★
`Endres` [21]	Grouping	✓	✓	✓	100	-	★★★	★★
`Geodesic` [22]	Grouping	✓		✓	1	★	★★★	★★
`MCG` [23]	Grouping	✓	✓	✓	30	★	★★★	★★★
`Objectness` [24]	Window scoring		✓	✓	3	·	★	·
`Rahtu` [25]	Window scoring		✓	✓	3	·	·	★
`RandomizedPrim's` [26]	Grouping	✓		✓	1	★	★	★★
`Rantalankila` [27]	Grouping	✓		✓	10	★★	·	★★
`Rigor` [28]	Grouping	✓		✓	10	★	★★	★★
`SelectiveSearch` [29]	Grouping	✓	✓	✓	10	★★	★★★	★★★
`Gaussian`				✓	0	·	·	★
`SlidingWindow`				✓	0	★★★	·	·
`Superpixels`		✓			1	★	·	·
`Uniform`				✓	0	·	·	·

Hosang et al, PAMI 2015

R-CNN

1. Person or not?
2. Horse or not?
3. House or not?
4. …



Slide credit: Ross Girschick

Steps

1. Train (or download) a classification model for ImageNet (AlexNet)
2. Fine-tune model for detection
3. Extract features from each region
4. Train one binary SVM per class to classify region features
5. For each class, train a linear regression model to map from cached features to offsets to GT boxes to make up for "slightly wrong" proposals

Slide credit: Fei-Fei Li & Andrej Karpathy & Justin Johnson

Any problem with R-CNN?

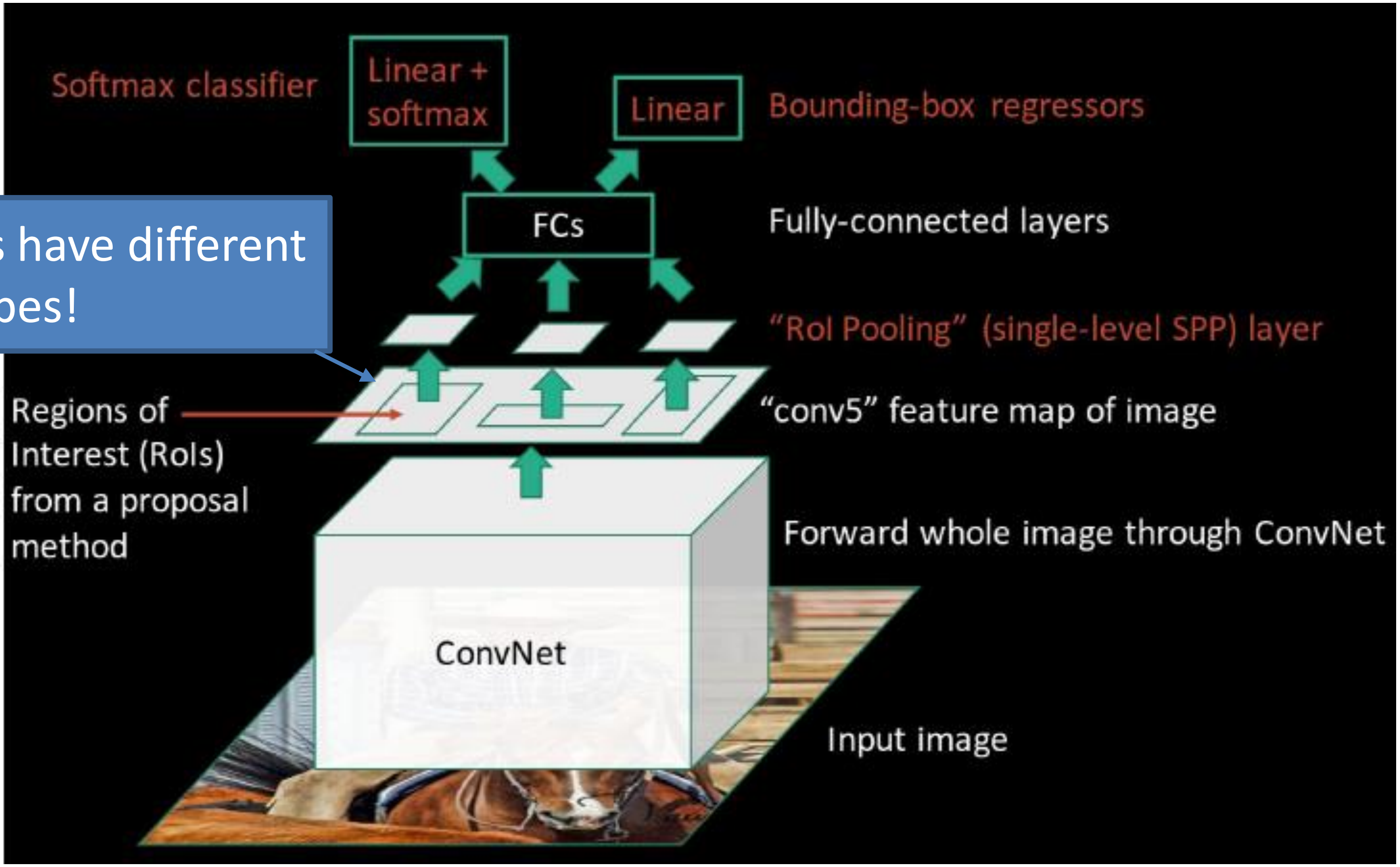
- Slow at test-time:
  - Need to run full forward pass of CNN for each region proposal
- SVMs and regressors are post-hoc:
  - CNN features not updated in response to SVMs and regressors

Do you have a better idea?

Let's run one forward pass of CNN on the entire image and map every ROI to the feature maps!

Fast R-CNN

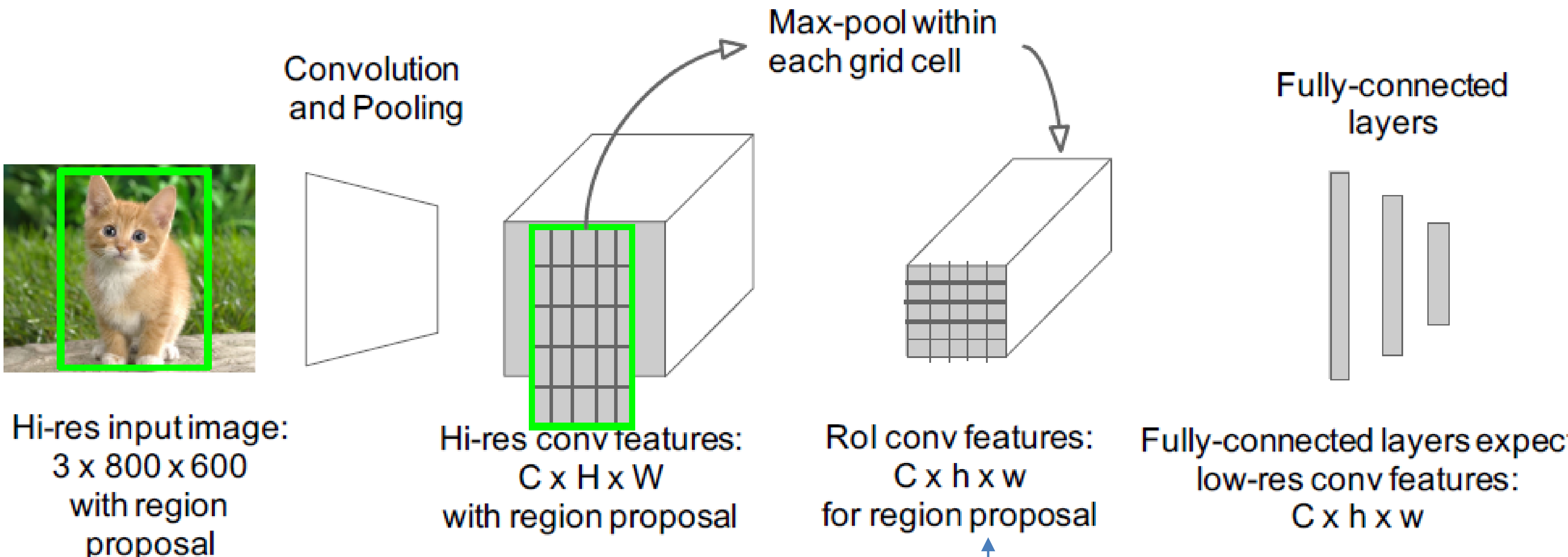
RoIs have different shapes!



Girshick, ICCV 2015

Slide credit: Ross Girschick

Region of interest pooling



The shapes of the resulting feature maps are the same now

Slide credit: Fei-Fei Li & Andrej Karpathy & Justin Johnson

Fast R-CNN Results

		R-CNN	Fast R-CNN
Faster!	Training Time:	84 hours	**9.5 hours**
	(Speedup)	1x	**8.8x**
FASTER!	Test time per image	47 seconds	**0.32 seconds**
	(Speedup)	1x	**146x**
Better!	mAP (VOC 2007)	66.0	**66.9**

Using VGG-16 CNN on Pascal VOC 2007 dataset

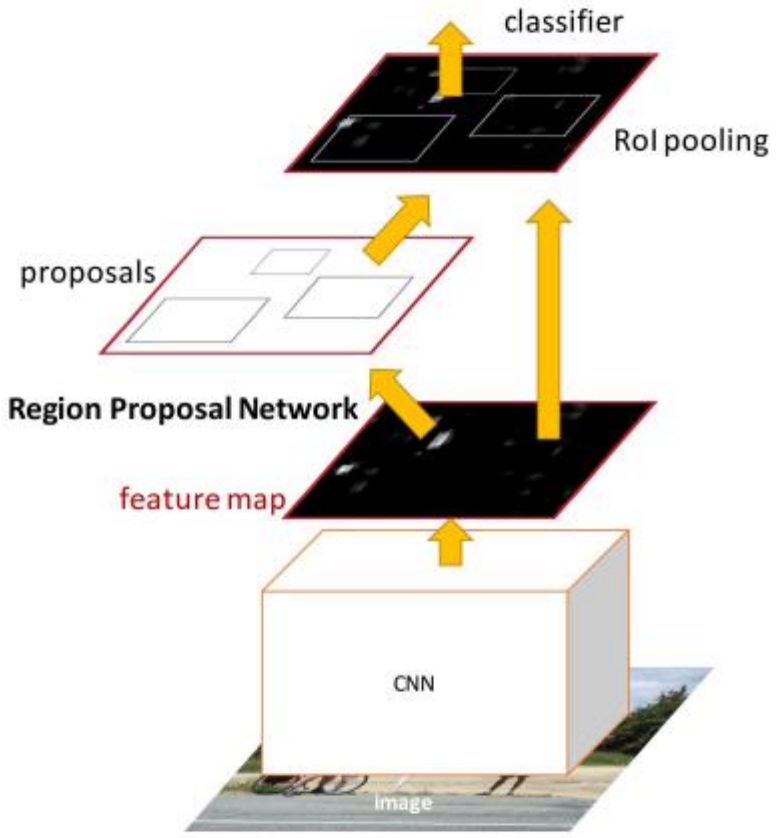
Slide credit: Fei-Fei Li & Andrej Karpathy & Justin Johnson

Question

Is there any problem with Fast R-CNN?

Faster R-CNN

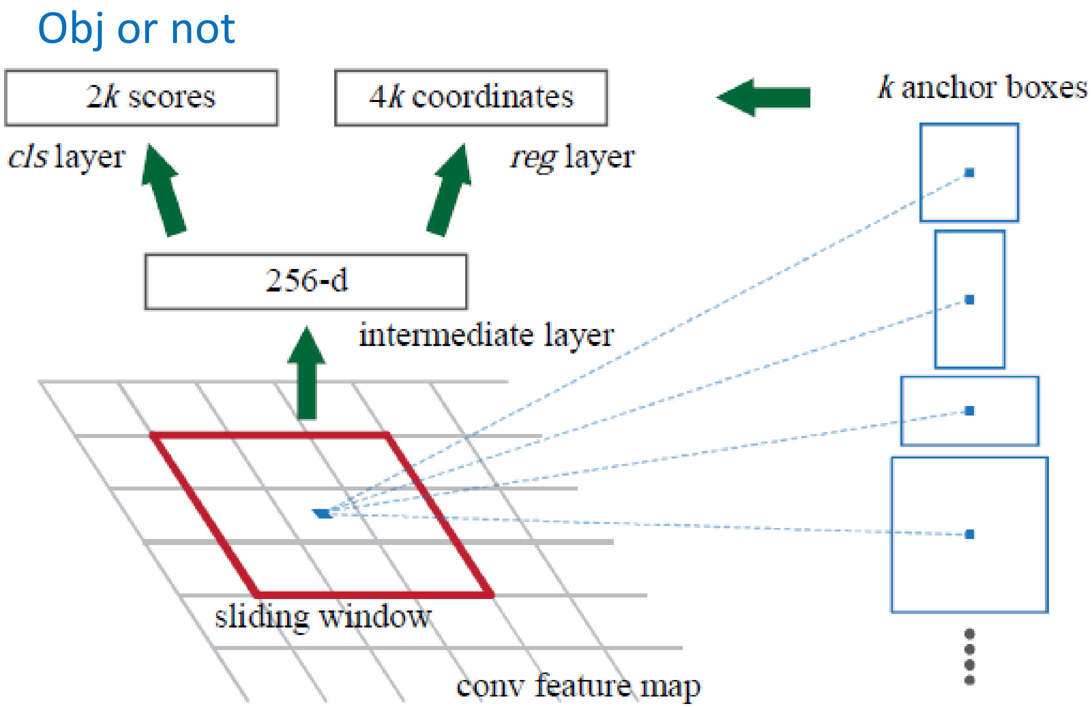
- Insert a Region Proposal Network (RPN) after the last convolutional layer
- RPN trained to produce region proposals directly; no need for external region proposals!
- After RPN, use RoI Pooling and an upstream classifier and bbox regressor just like Fast R-CNN



Ren et al., NIPS 2015

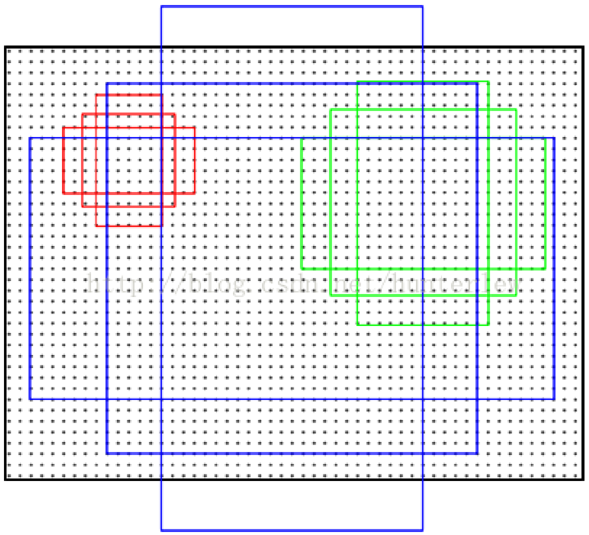
Slide credit: Ross Girschick

Region proposal network



Ren et al., NIPS 2015

At each location, k=9 boxes are generated in the input image



http://blog.csdn.NET/shenxiaolu1984/article/details/51152614

Faster R-CNN results

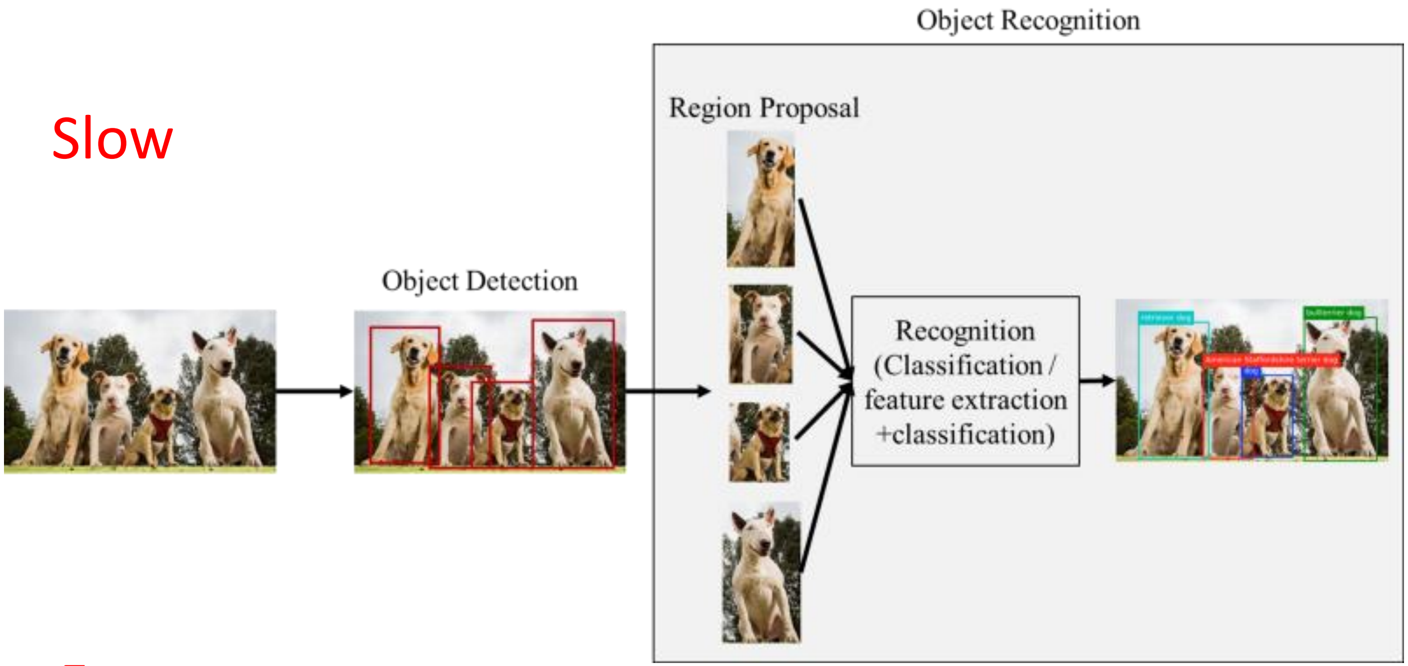
Table 2: Detection results on **PASCAL VOC 2007 test set**. The detector is Fast R-CNN and VGG-16. Training data: "07": VOC 2007 trainval, "07+12": union set of VOC 2007 trainval and VOC 2012 trainval. For RPN, the train-time proposals for Fast R-CNN are 2k. †: this was reported in [5]; using the repository provided by this paper, this number is higher (68.0±0.3 in six runs).

Fast R-CNN

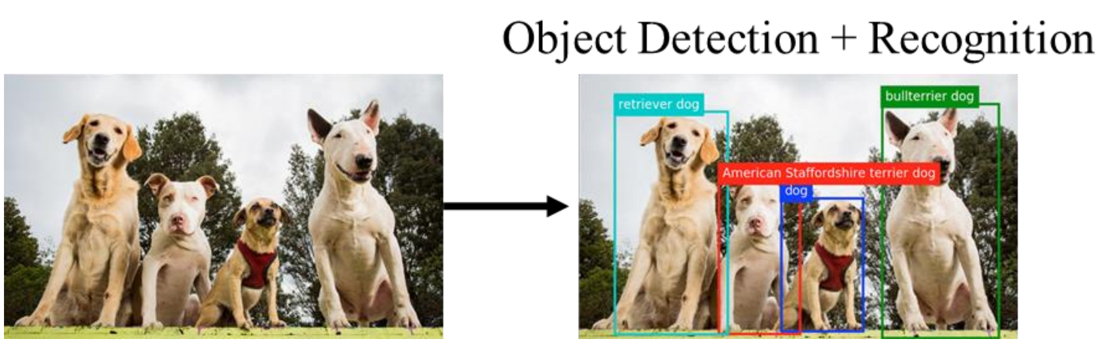
method	# proposals	data	mAP (%)	time (ms)
SS	2k	07	66.9†	1830
SS	2k	07+12	70.0	1830
RPN+VGG, unshared	300	07	68.5	342
RPN+VGG, shared	300	07	69.9	**198**
RPN+VGG, shared	300	07+12	**73.2**	**198**

Two-stage versus one-stage

Slow



Fast



How would you do this?

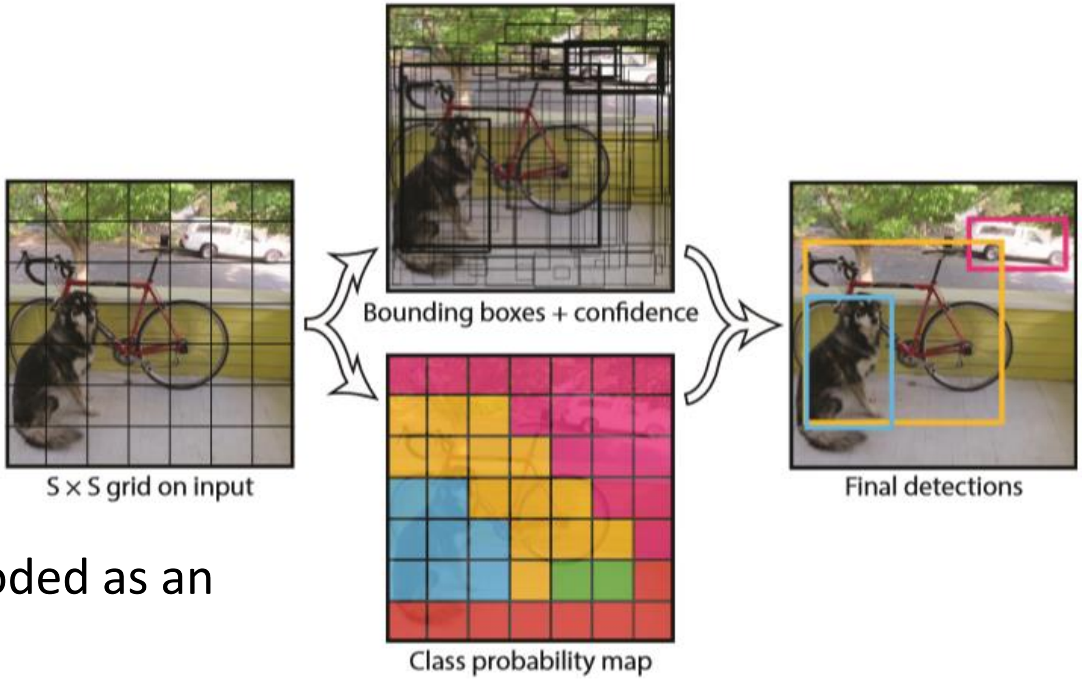
YOLO: You only look at once

Redmon, Divvala, Girshick, Farhadi, CVPR 2016

- It models detection as a regression problem

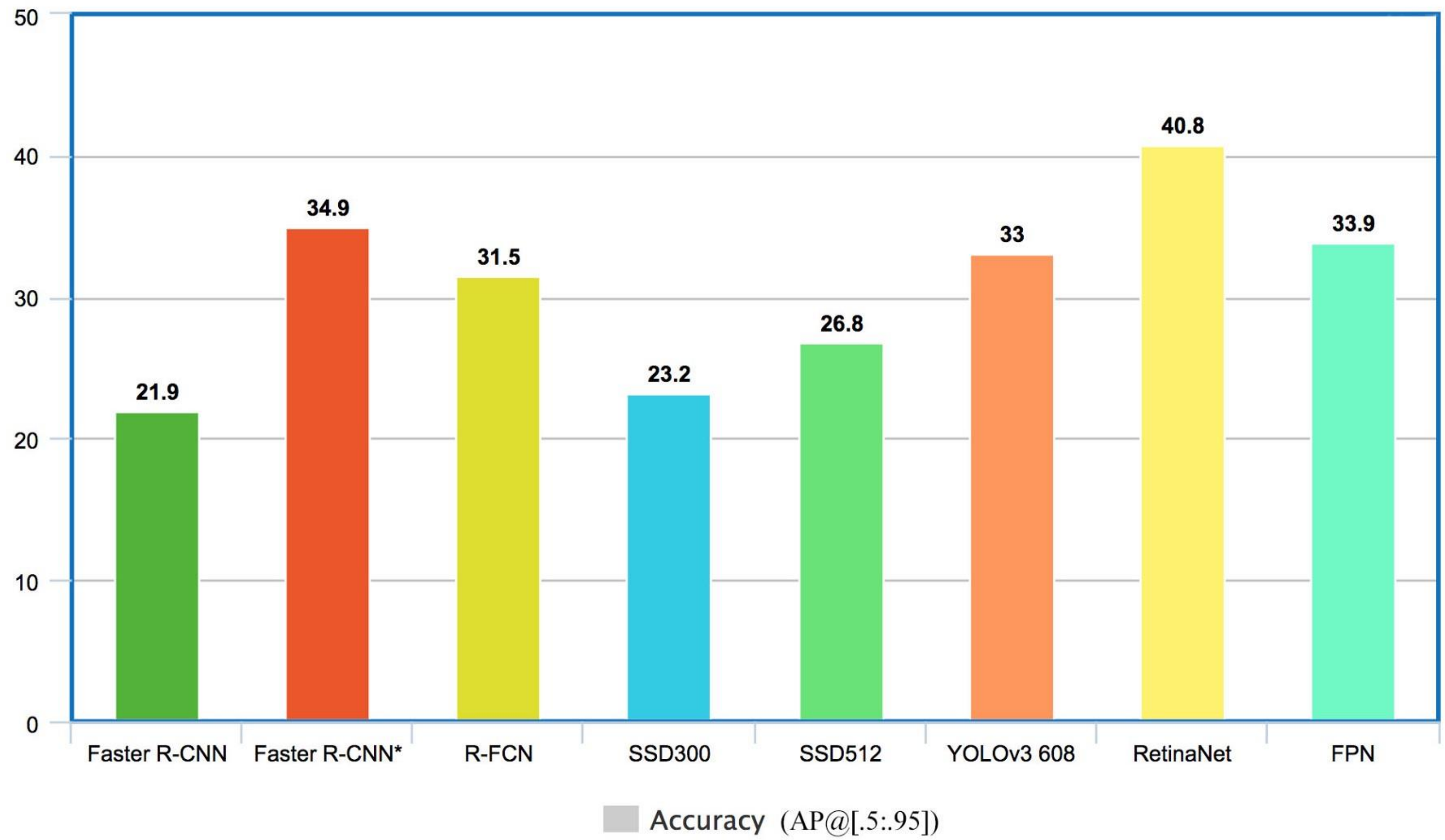
① Divide the image into an S×S grid
② For each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities

These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor



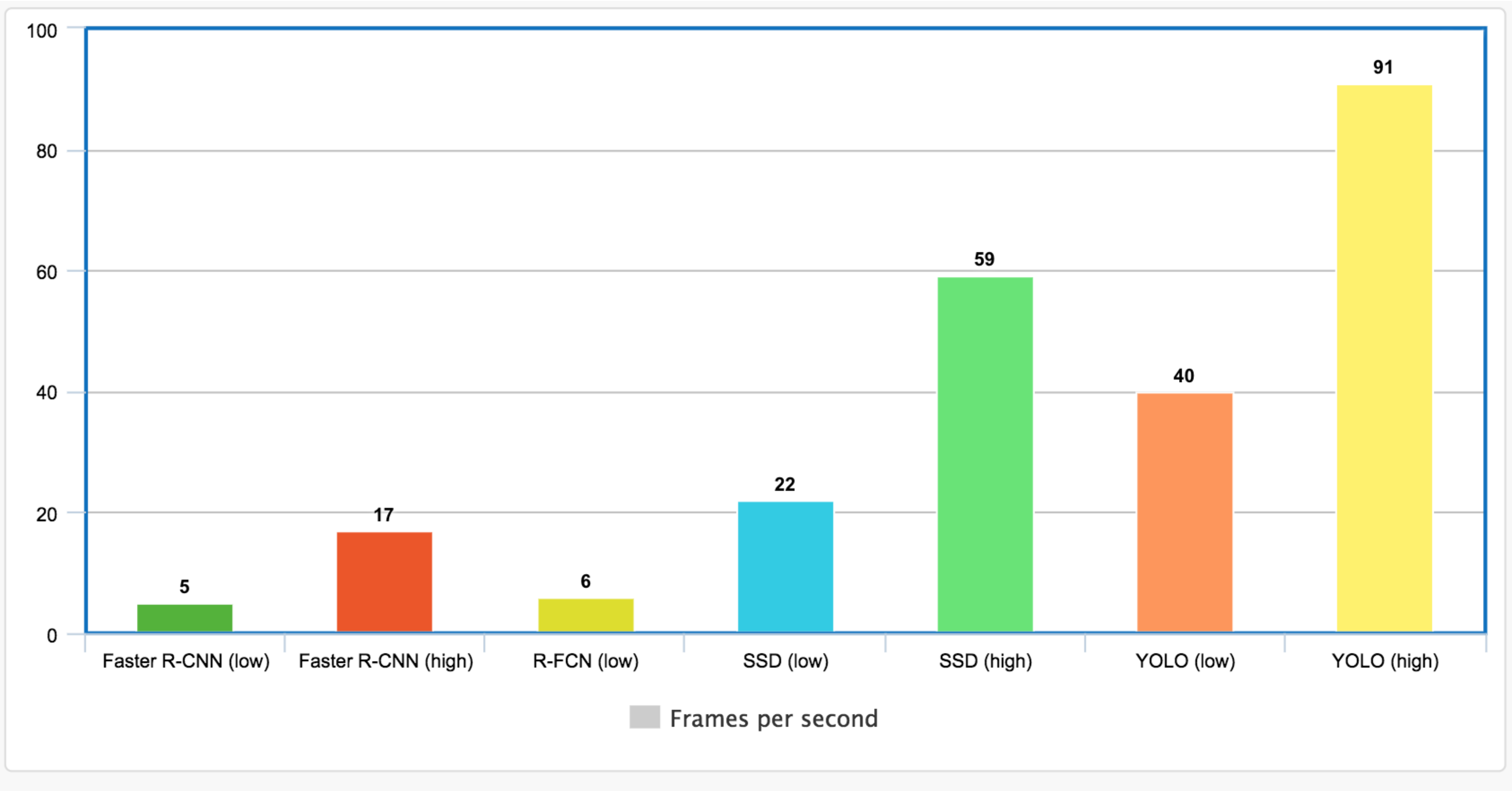
- It predicts the centers of ground truth boxes in each grid
- Every class prob corresponds to a bounding box centered in that cell

Performance comparison



https://github.com/yehengchen/Object-Detection-and-Tracking/blob/master/Two-stage%20vs%20One-stage%20Detectors.md

Performance comparison



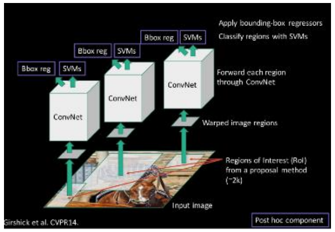
https://github.com/yehengchen/Object-Detection-and-Tracking/blob/master/Two-stage%20vs%20One-stage%20Detectors.md

Summary of Part 3

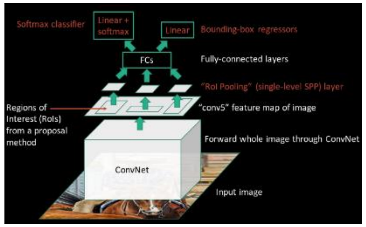
Two-stage

One-stage

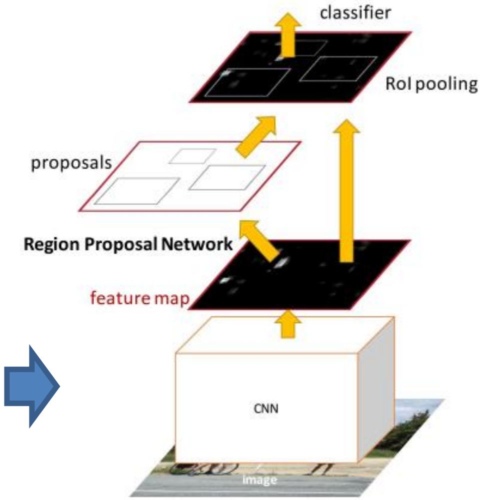
R-CNN



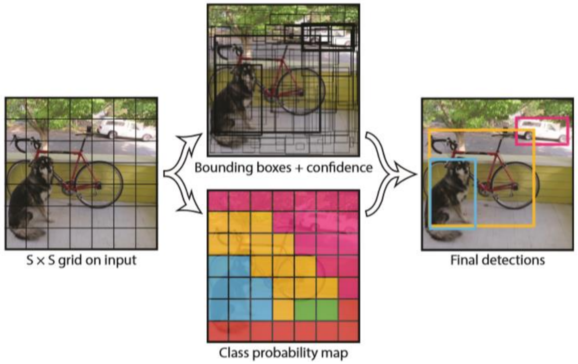
Fast R-CNN



Faster R-CNN



YOLO



Multiple Choice(single) Points: 1

Setting

How many samples at least are needed in a minibatch for minimizing the contrastive loss?

A 1

B 2

C 3

D 4

Submit

Multiple Choice(single) Points: 1

Setting

For training the FaceNet, a triplet loss is used. After training, the distance between the features of an anchor sample and a positive sample should be

A reduced

B increased

C the same

Submit

Multiple Choice(multiple) Points: 1

Setting

Which of the following generate RoIs using external models?

A R-CNN

B Fast R-CNN

C Faster R-CNN

D YOLO

Submit

Multiple Choice(multiple) Points: 1

Setting

Faster R-CNN uses an RPN to generate proposals. What predictions does RPN make at every location?

A Coordinates of bounding boxes

B Whether the bounding boxes contain objects or not

C Class labels of the objects

Submit

Multiple Choice(single) Points: 1

Setting

Which method is an one-stage detection method?

A R-CNN

B Fast-RCNN

C Faster-RCNN

D YOLO

Submit

Outline

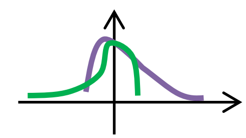
1. Training techniques-III
2. Image classification
3. Object detection
4. Summary

Summary of this lecture

Knowledge

Part 1

$$\hat{x}_i = \frac{x_i - \mathrm{E}[x_i]}{\sqrt{\mathrm{Var}[x_i]}} \qquad y_i = \gamma_i \hat{x}_i + \beta_i$$



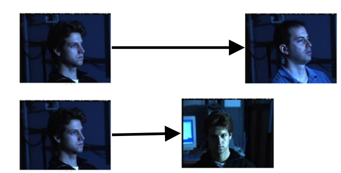
Part 2

Multi-class classification

Face verification

Triplet loss

Contrastive loss



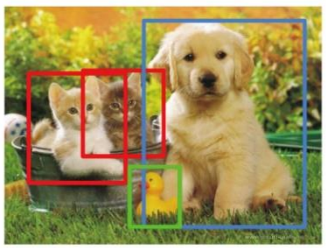
Part 3

R-CNN → Fast R-CNN → Faster R-CNN → YOLO

Summary of this lecture

Capability and value

- Simple ideas, influential works

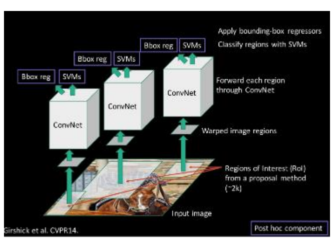


1. Predict lots of RoIs
2. Do classification for each RoI

Google scholar citations (till Nov 1, 2020):
14949

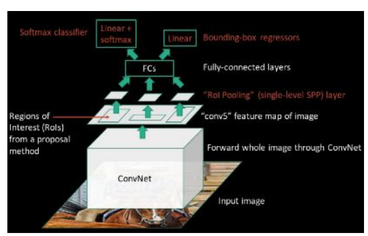
- Find problems, solve problems

R-CNN



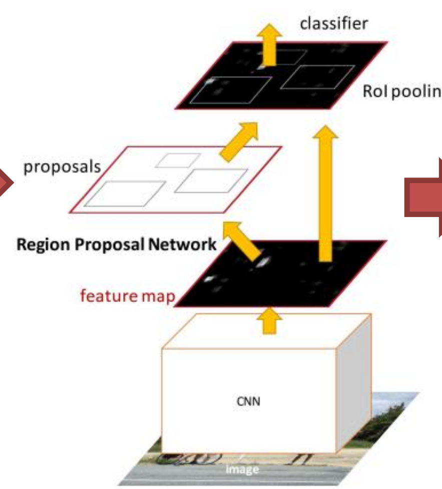
Too many feedforward passes

Fast R-CNN



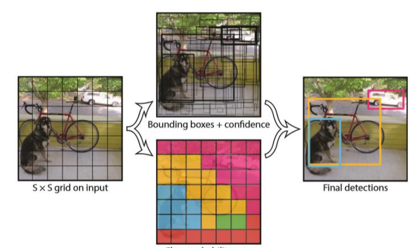
External models for RoI generation

Faster R-CNN



2 stages are slow

YOLO



Accuracy is low

Recommended reading

- Schroff, Kalenichenko, Philbin (2015)
  FaceNet: A Unified Embedding for Face Recognition and Clustering
  CVPR
- Girshick, Donahue, Darrell, Malik (2014)
  Rich feature hierarchies for accurate object detection and semantic segmentation
  CVPR
- Girshick (2015)
  Fast R-CNN
  ICCV
- Ren, He, Girshick, Sun (2015)
  Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks
  NIPS

Recommended reading

- Redmon, Divvala, Girshick, Farhadi (2016)
  You Only Look Once: Unified, Real-Time Object Detection
  CVPR

Prepare for the next lecture

- Form groups of 2 and every group prepares a 5-minute presentation with slides for the following paper
  - Santurkar, Tsipras, Ilyas, Madry, “How does batch normalization help optimization?” NeurIPS 2018