# Detection and Segmentation

## CS60010: Deep Learning

Abir Das

IIT Kharagpur

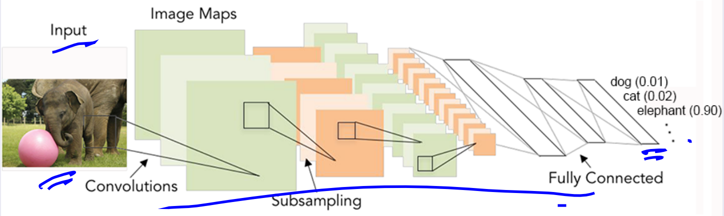
Mar 08, 2021

# Agenda

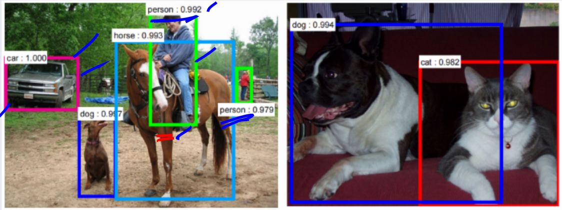
To get introduced to two important tasks of computer vision - detection and segmentation along with deep neural network's application in these areas in recent years.

# From Classification to Detection

## Classification



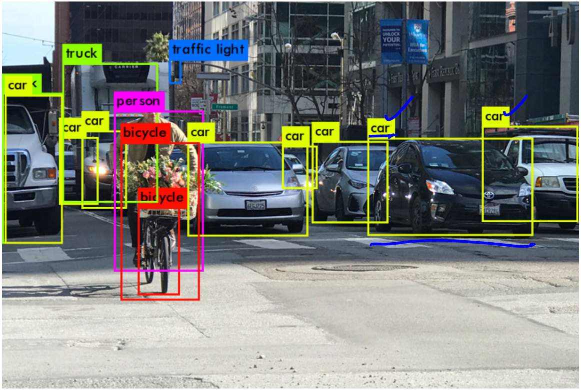

## Detection



# Challenges of Object Detection

- Simultaneous recognition and localization
- Images may contain objects from more than one class and multiple instances of the same class
- Evaluation

G.T.



# Localization and Detection

| Classification | Classification + Localization | Object Detection |
| --- | --- | --- |

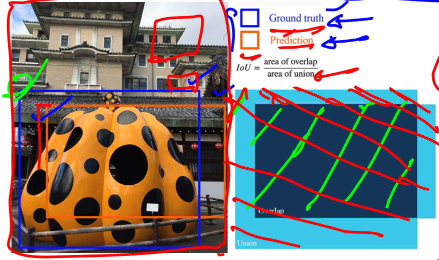
| CAT | CAT | CAT, DOG, DUCK |
| --- | --- | --- |

Single object

Multiple objects

# Evaluation

- At test time 3 things are predicted:- Bounding box coordinates, Bounding box class label, Confidence score
- Performance is measured in terms of IoU (Intersection over Union)




(0,1)

(x,y,w,h)

- According to PASCAL criterion,
  - a detection is correct if $\boxed{IoU > 0.5}$
  - For multiple detections only one is considered **true positive**

$\boxed{IoU > 0.7}$

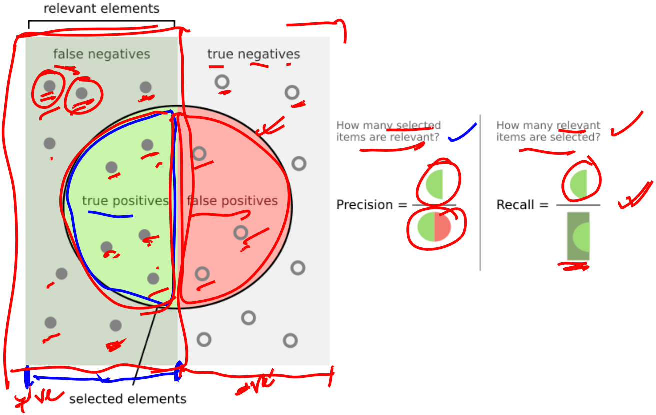
10 → 6

4 → 4 F.P.

by the (decreasing) confidence output. Multiple detections of the same object in an image were considered false detections e.g. 5 detections of a single object counted as 1 correct detection and 4 false detections—it was the responsibility of the participant's system to filter multiple detections from its output.

Image Source

# Evaluation: Precision-Recall




- precision $= \frac{tp}{tp+fp}$
- recall $= \frac{tp}{tp+fn}$

Image Source

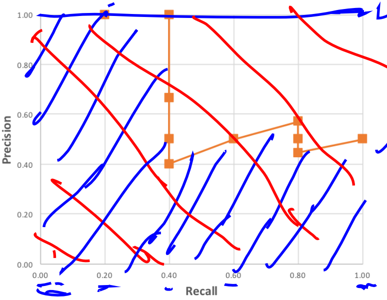
# Evaluation: Average Precision

5 detections: T-P

Lets consider an image with 5 apples where our detector provides 10 detections.

2/3 2/2 1/5 2/5 3/5 confidence score

| Rank | Correct | Precision | Recall |
|---|---|---|---|
| 1 | True Positive | 1.00 | 0.20 |
| 2 | True Positive | 1.00 | 0.40 |
| 3 | False Positive | 0.67 | 0.40 |
| 4 | False Positive | 0.50 | 0.40 |
| 5 | False Positive | 0.40 | 0.40 |
| 6 | True Positive | 0.50 | 0.60 |
| 7 | True Positive | 0.57 | 0.80 |
| 8 | False Positive | 0.50 | 0.80 |
| 9 | False Positive | 0.44 | 0.80 |
| 10 | True Positive | 0.50 | 1.00 |

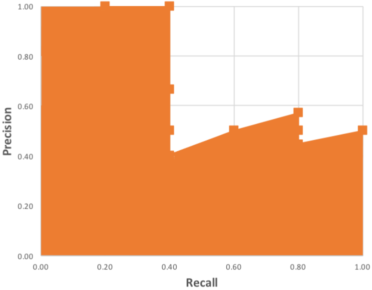



Source: This medium post

# Evaluation: Average Precision

Area under curve is a measure of performance. This gives the average precision of the detector.

| Rank | Correct | Precision | Recall |
|---|---|---|---|
| 1 | True Positive | 1.00 | 0.20 |
| 2 | True Positive | 1.00 | 0.40 |
| 3 | False Positive | 0.67 | 0.40 |
| 4 | False Positive | 0.50 | 0.40 |
| 5 | False Positive | 0.40 | 0.40 |
| 6 | True Positive | 0.50 | 0.60 |
| 7 | True Positive | 0.57 | 0.80 |
| 8 | False Positive | 0.50 | 0.80 |
| 9 | False Positive | 0.44 | 0.80 |
| 10 | True Positive | 0.50 | 1.00 |

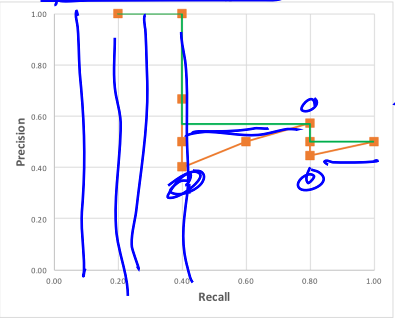



Source: This medium post

# Evaluation: mean Average Precision

A little more detail:

- The curve is made smooth from the zigzag pattern by finding the highest precision value at or to the right side of the recall values.
- Then the average is taken for 11 recall values (0, 0.1, 0.2, ... 1.0) - Average Precison (AP) → over recall values
- The mean average precision (mAP) is the mean of the average precisions (AP) for all classes of objects.



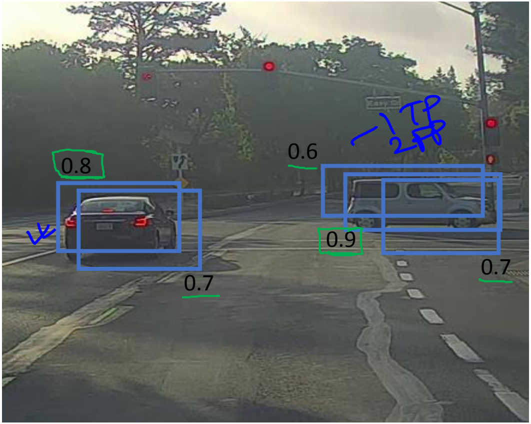

→ orange $AP_1$
→ Apple $AP_2$
→ cat $AP_3$

Source: This medium post

# Non-max Suppression

What to do if there are multiple detections of the same object? Can you think its effect on precision-recall?

1 TP
1 FP




How many of them

2 TP and

5

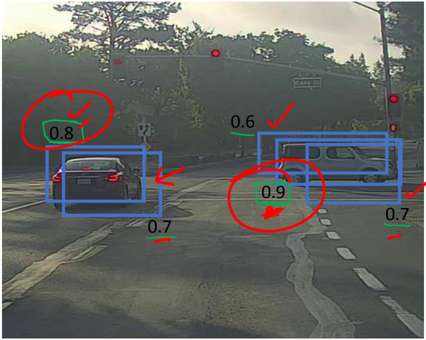
$\frac{2}{5}$

$\frac{2}{2+3}$

Source: deeplearning.ai

# Non-max Suppression

- Sort the predictions by the confidence scores
- Starting with the top score prediction, ignore any other prediction of the same class and high overlap (*e.g.*, IoU $> 0.5$) with the top ranked prediction
- Repeat the above step until all predictions are checked



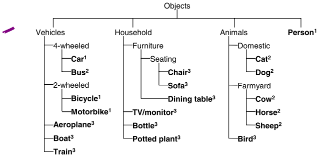

Source: deeplearning.ai

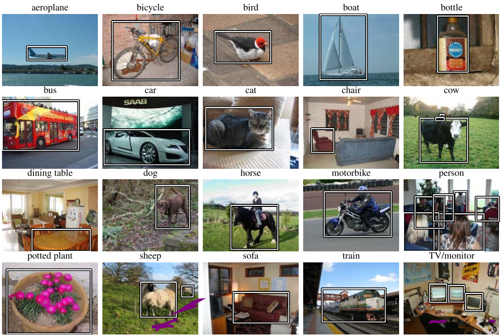
# Segmentation



Source: cs231n course, Stanford University

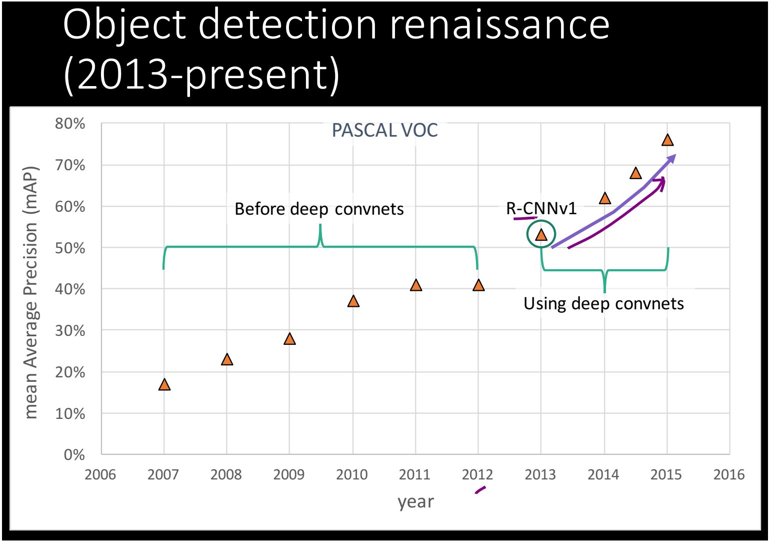
# PASCAL VOC




- Dataset size (by 2012): 11.5K training/val images, 27K bounding boxes, 7K segmentations

# PASCAL VOC




Source: ICCV '15, Fast R-CNN

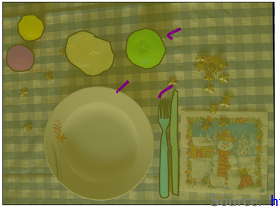
# COCO Dataset

COCO
Common Objects in Context

## What is COCO?

COCO is a large-scale object detection, segmentation, and captioning dataset. COCO has several features:

- Object segmentation
- Recognition in context
- Superpixel stuff segmentation
- 330K images (>200K labeled)
- 1.5 million object instances
- 80 object categories
- 91 stuff categories
- 5 captions per image
- 250,000 people with keypoints



Source: http://cocodataset.org

# COCO Tasks

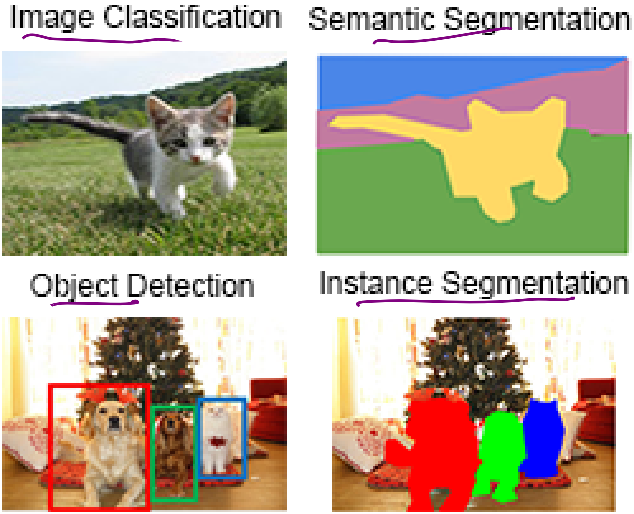


Image Classification
Semantic Segmentation
Object Detection
Instance Segmentation

## Classification + Localization

# Classification + Localization: Task

**Classification**: C classes

**Input:** Image

**Output:** Class label

**Evaluation metric:** Accuracy

→ CAT

**Localization**:

**Input:** Image

**Output**: Box in the image (x, y, w, h)

**Evaluation metric:** Intersection over Union


→ (x, y, w, h)

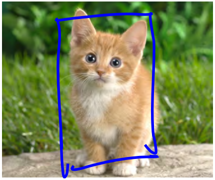
**Classification + Localization**: Do both

Source: cs231n course, Stanford University

# Classification + Localization

## Idea #1: Localization as Regression

**Input**: image



Neural Net

x, y, w, h

**Output**:
Box coordinates
(4 numbers)

**Correct output**:
box coordinates
(4 numbers)

**Loss**:
L2 distance

$(\hat{x}-x)^2 + (\hat{y}-y)^2 + \dots$

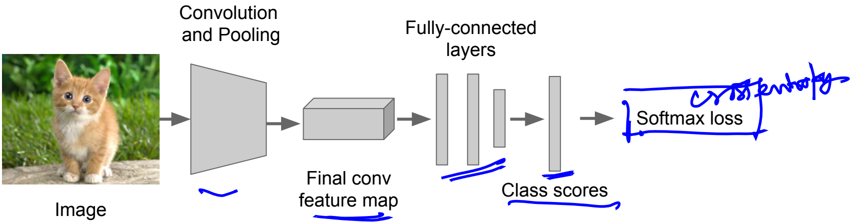
Only one object,
simpler than detection

Source: cs231n course, Stanford University

# Classification + Localization

## Simple Recipe for Classification + Localization

**Step 1**: Train (or download) a classification model (AlexNet, VGG, GoogLeNet)

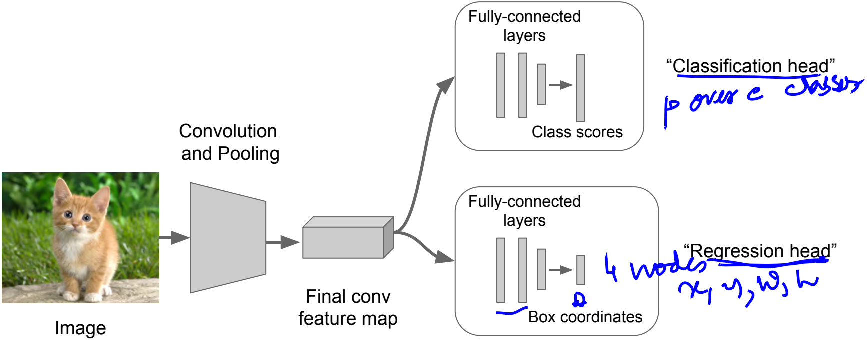



Source: cs231n course, Stanford University

# Classification + Localization

## Simple Recipe for Classification + Localization

**Step 2**: Attach new fully-connected "regression head" to the network

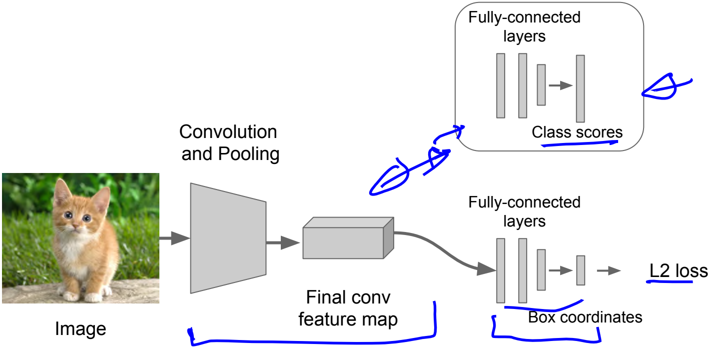



Source: cs231n course, Stanford University

# Classification + Localization

## Simple Recipe for Classification + Localization

**Step 3**: Train the regression head only with SGD and L2 loss

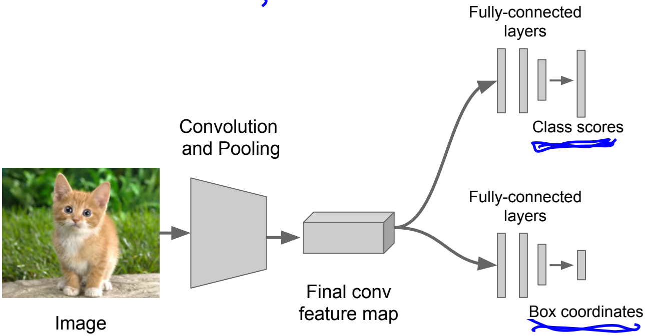



Source: cs231n course, Stanford University

# Classification + Localization

## Simple Recipe for Classification + Localization

**Step 4**: At test time use both heads



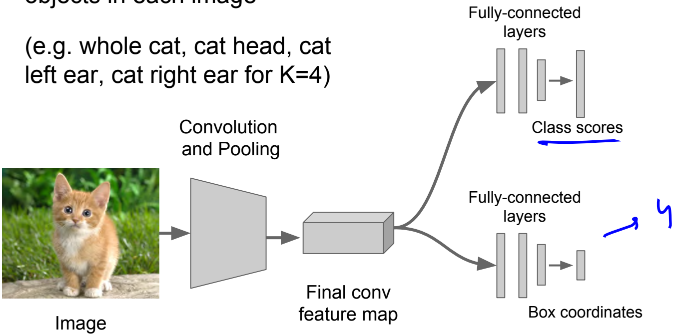

Source: cs231n course, Stanford University

## Classification + Localization

# Aside: Localizing multiple objects

Want to localize **exactly** K objects in each image

(e.g. whole cat, cat head, cat left ear, cat right ear for K=4)




K x 4 numbers
(one box per object)

Source: cs231n course, Stanford University

# Classification + Localization

## Aside: Human Pose Estimation

Represent a person by K joints

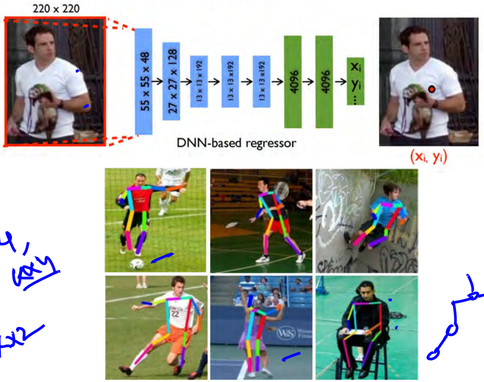
Kx2

Regress (x, y) for each joint from last fully-connected layer of AlexNet

(Details: Normalized coordinates, iterative refinement)

4, (x,y)

Kx2



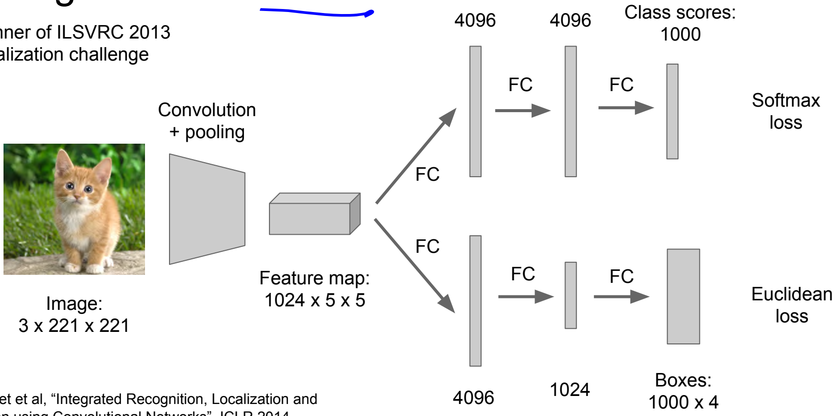

Toshev and Szegedy, "DeepPose: Human Pose Estimation via Deep Neural Networks", CVPR 2014

Source: cs231n course, Stanford University

# Classification + Localization

## Sliding Window: Overfeat

Winner of ILSVRC 2013
localization challenge

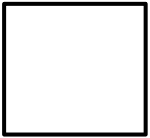



Sermanet et al, “Integrated Recognition, Localization and Detection using Convolutional Networks”, ICLR 2014

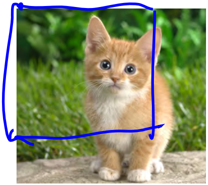
Source: cs231n course, Stanford University

# Classification + Localization

## Sliding Window: Overfeat



Network input:
3 x 221 x 221

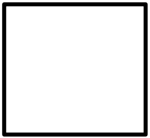


Larger image:
3 x 257 x 257

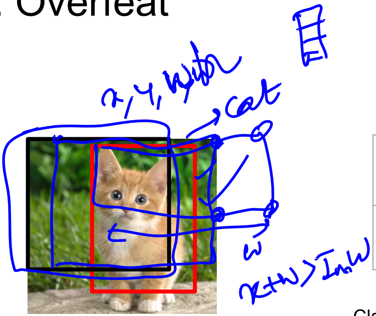
Source: cs231n course, Stanford University

# Classification + Localization

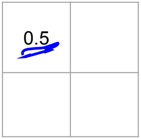
## Sliding Window: Overfeat



Network input:
3 x 221 x 221




Larger image:
3 x 257 x 257

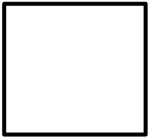



Classification scores:
P(cat)

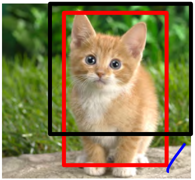
Source: cs231n course, Stanford University

# Classification + Localization

## Sliding Window: Overfeat



Network input:
3 x 221 x 221



Larger image:
3 x 257 x 257

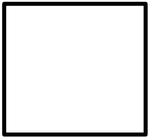
| | |
|---|---|
| 0.5 | 0.75 |
| | |

Classification scores:
P(cat)

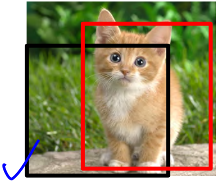
Source: cs231n course, Stanford University

# Classification + Localization

## Sliding Window: Overfeat



Network input:
3 x 221 x 221



Larger image:
3 x 257 x 257

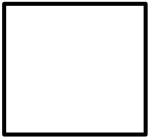
| | |
|---|---|
| 0.5 | 0.75 |
| 0.6 | |

Classification scores:
P(cat)

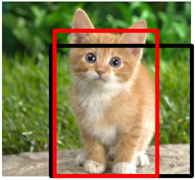
Source: cs231n course, Stanford University

# Classification + Localization

## Sliding Window: Overfeat



Network input:
3 x 221 x 221



Larger image:
3 x 257 x 257

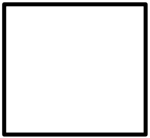
| | |
|---|---|
| 0.5 | 0.75 |
| 0.6 | 0.8 |

Classification scores:
P(cat)

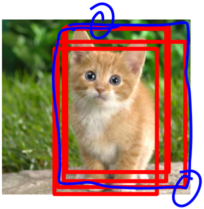
Source: cs231n course, Stanford University

# Classification + Localization

## Sliding Window: Overfeat



Network input:
3 x 221 x 221



Larger image:
3 x 257 x 257

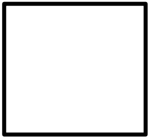
| | |
|---|---|
| 0.5 | 0.75 |
| 0.6 | 0.8 |

Classification scores:
P(cat)

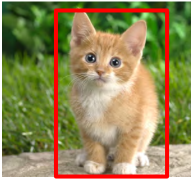
Source: cs231n course, Stanford University

# Classification + Localization

## Sliding Window: Overfeat



Network input:
3 x 221 x 221



Larger image:
3 x 257 x 257

Greedily merge boxes and scores (details in paper)

0.8

Classification score: P (cat)

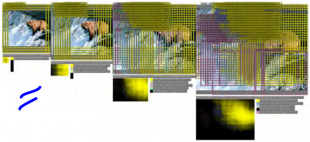
Source: cs231n course, Stanford University

# Classification + Localization

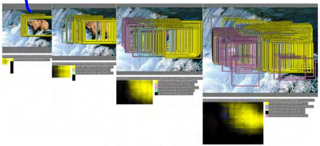
## Sliding Window: Overfeat

In practice use many sliding window locations and multiple scales

Window positions + score maps



Box regression outputs

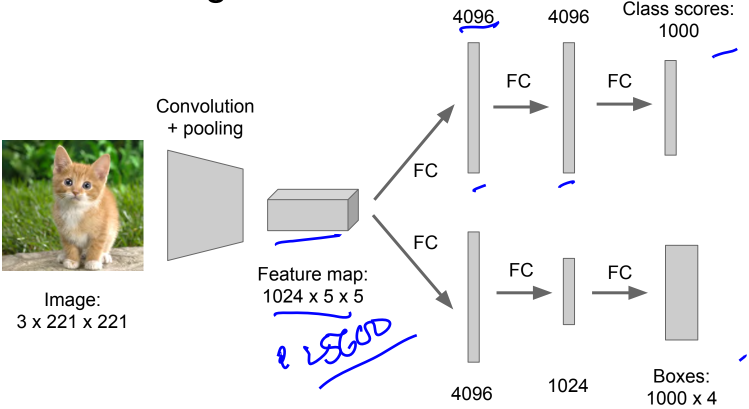


Final Predictions

Source: cs231n course, Stanford University

# Classification + Localization

## Efficient Sliding Window: Overfeat

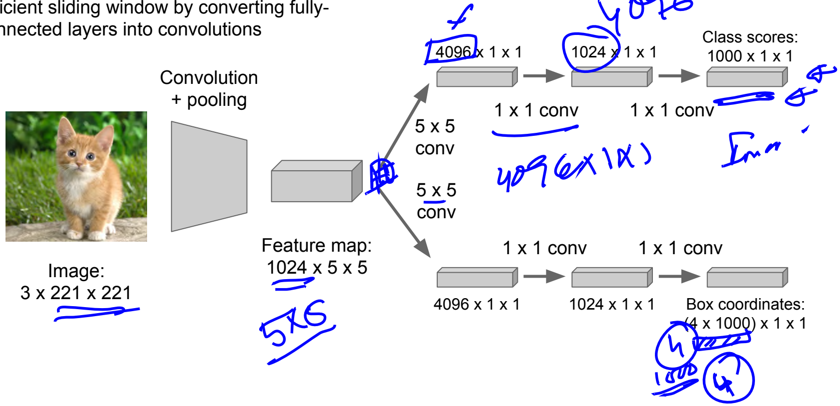



Source: cs231n course, Stanford University

# Classification + Localization

## Efficient Sliding Window: Overfeat

Efficient sliding window by converting fully-connected layers into convolutions



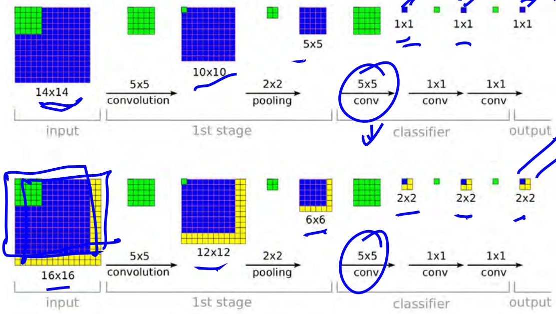

Source: cs231n course, Stanford University

# Classification + Localization

## Efficient Sliding Window: Overfeat

**Training time:** Small image, 1 x 1 classifier output

**Test time:** Larger image, 2 x 2 classifier output, only extra compute at yellow regions




Sermanet et al, "Integrated Recognition, Localization and Detection using Convolutional Networks", ICLR 2014

Source: cs231n course, Stanford University

# Classification + Localization

## ImageNet Classification + Localization


**AlexNet**: Localization method not published

**Overfeat**: Multiscale convolutional regression with box merging

**VGG**: Same as Overfeat, but fewer scales and locations; simpler method, gains all due to deeper features

**ResNet:** Different localization method (RPN) and much deeper features

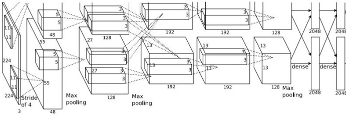
Source: cs231n course, Stanford University

# Detection as Regression

- In detection you don't know the number of objects present
- So, it is problematic to address detection as regression
- How many output neurons to put?

# Detection as Classification

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

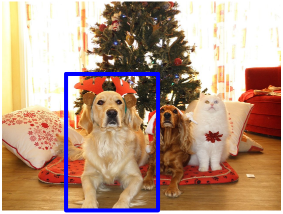


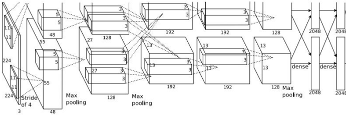

Dog? NO
Cat? NO
Background? YES

CS231n course, Stanford University

# Detection as Classification

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



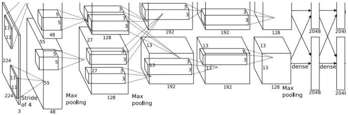



Dog? YES
Cat? NO
Background? NO

CS231n course, Stanford University

# Detection as Classification

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

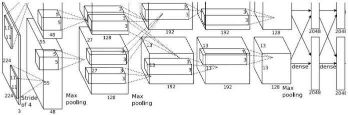



Dog? YES
Cat? NO
Background? NO

CS231n course, Stanford University

# Detection as Classification

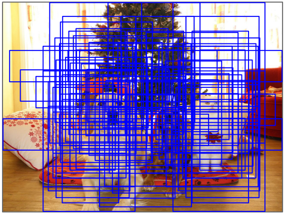
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background




Dog? NO
Cat? YES
Background? NO

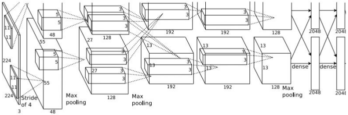
CS231n course, Stanford University

# Detection as Classification



Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

Problem: Need to apply CNN to huge number of locations, scales, and aspect ratios, very computationally expensive!



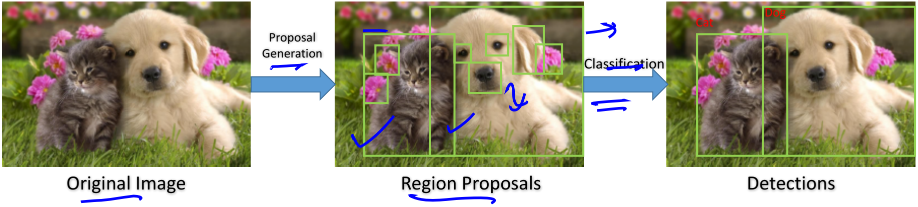

Dog? NO
Cat? YES
Background? NO

CS231n course, Stanford University

# Detection as Classification

- Need to apply CNN to huge number of locations, scales and aspect ratios
- If the classifier is fast enough, this is done. Pre Deep Learning approach.
- Deep learning classifiers, first get a tiny subset of possible positions. Only these are passed through the deep classifiers.
- The possible positions are called 'candidate proposals' or 'region proposals'.

# Detection with Region Proposals




- Generate and evaluate a few (much less than exhaustive search) region proposals
- Proposal mechanism can take advantage of low-level cues (*e.g.*, edges or connected components)
- Classifier can be slower but more powerful

# Selective Search

First level

Input Image

J Uijlings, K van de Sande, T Gevers, and A Smeulders, 'Selective Search for Object Recognition', IJCV 2013

# Selective Search

**Algorithm 1:** Hierarchical Grouping Algorithm

**Input**: (colour) image
**Output**: Set of object location hypotheses $L$

Obtain initial regions $R = \{r_1, \cdots, r_n\}$ using [13]
Initialise similarity set $S = \emptyset$
**foreach** *Neighbouring region pair* $(r_i, r_j)$ **do**
  Calculate similarity $s(r_i, r_j)$
  $S = S \cup s(r_i, r_j)$

**while** $S \neq \emptyset$ **do**
  Get highest similarity $s(r_i, r_j) = \max(S)$
  Merge corresponding regions $r_t = r_i \cup r_j$
  Remove similarities regarding $r_i$ : $S = S \setminus s(r_i, r_*)$
  Remove similarities regarding $r_j$ : $S = S \setminus s(r_*, r_j)$
  Calculate similarity set $S_t$ between $r_t$ and its neighbours
  $S = S \cup S_t$
  $R = R \cup r_t$

Extract object location boxes $L$ from all regions in $R$

$i,j$

J Uijlings, K van de Sande, T Gevers, and A Smeulders, 'Selective Search for Object Recognition', IJCV 2013

# EdgeBoxes

- Edgeboxes depend on a fast scoring/evaluating method for bounding boxes.
- First edges are extracted for the whole image and they are grouped according to their similarity
- The main idea of scoring boxes builds on the fact that edges tend to correspond to object boundaries and bounding boxes that tightly enclose a set of edges are likely to contain an object.
- Gets 75% recall with 800 boxes (vs 1400 for Selective Search) and is 40 times faster

C Zitnick and P Dollar, 'Edge Boxes: Locating Object Proposals from Edges', ECCV 2014

# Many Region Proposal Methods

| 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 | · | · | · |

J Hosang, R Benenson, P Dollar and B Schiele, 'What makes for effective detection proposals?', IEEE TPAMI 2016