Week 3 Object detection

Object localization

- Classification with localization
- Detection

Localization?

add bounding box of output layers:

$$b_x, b_y, b_h, b_w$$

we give a output form as:

$$y = [\mathrm{pc}, b_x, b_y, b_h, b_w, c_1, c_2, c_3]^T$$

where pc stand for wether there is some object in the picture

loss function:

$$\begin{aligned} \mathrm{L}(\hat{y}, y) = \\ (\hat{y}_1 - y_1)^2 + (\hat{y}_2 - y_2)^2 + \ldots + (\hat{y}_8 - y_8)^2, \quad \text{if} \quad y_1 = 1; \\ (\hat{y}_1 - y_1)^2, \quad \text{if} \quad y_1 = 0; \end{aligned}$$

Landmark Detection

pick some points important, called `landmark`, of the object that we want to detect.

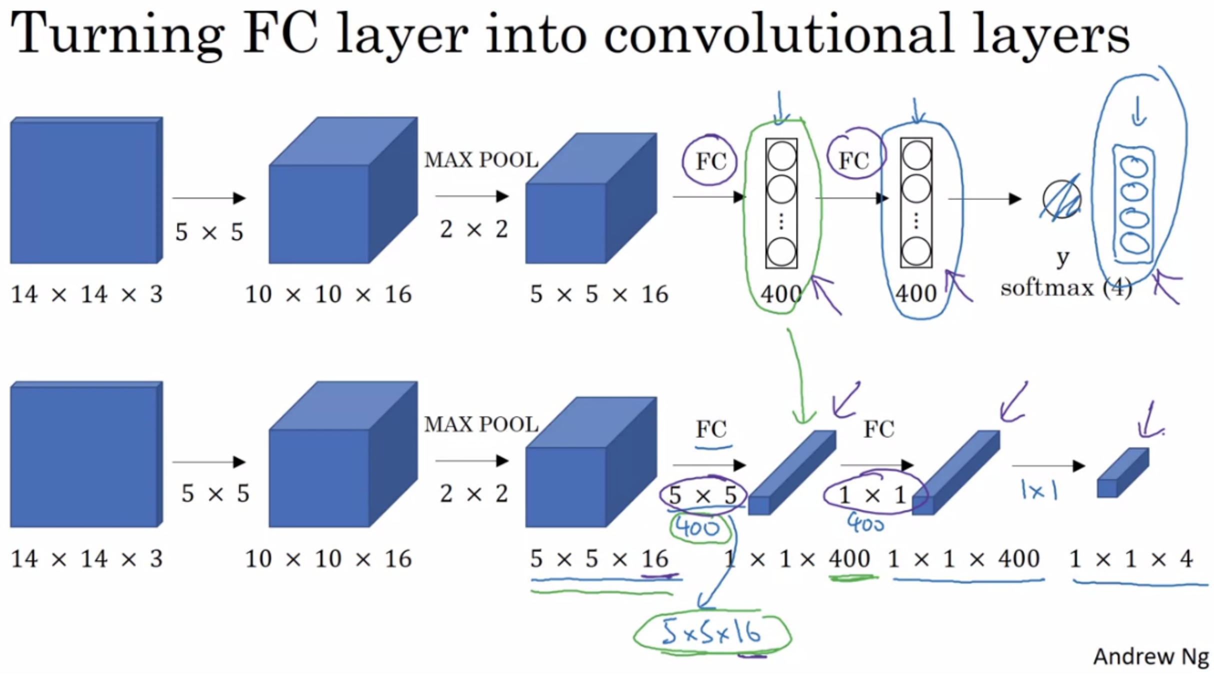
Object Detection

Sliding windows detection : small rectanguler region

- runnning slide windows throughout the picture
- using a larger window to do the same thing
- when there is once the proba of successful detection, we stop because we have already detected the location
- this is in fact very slow in this era

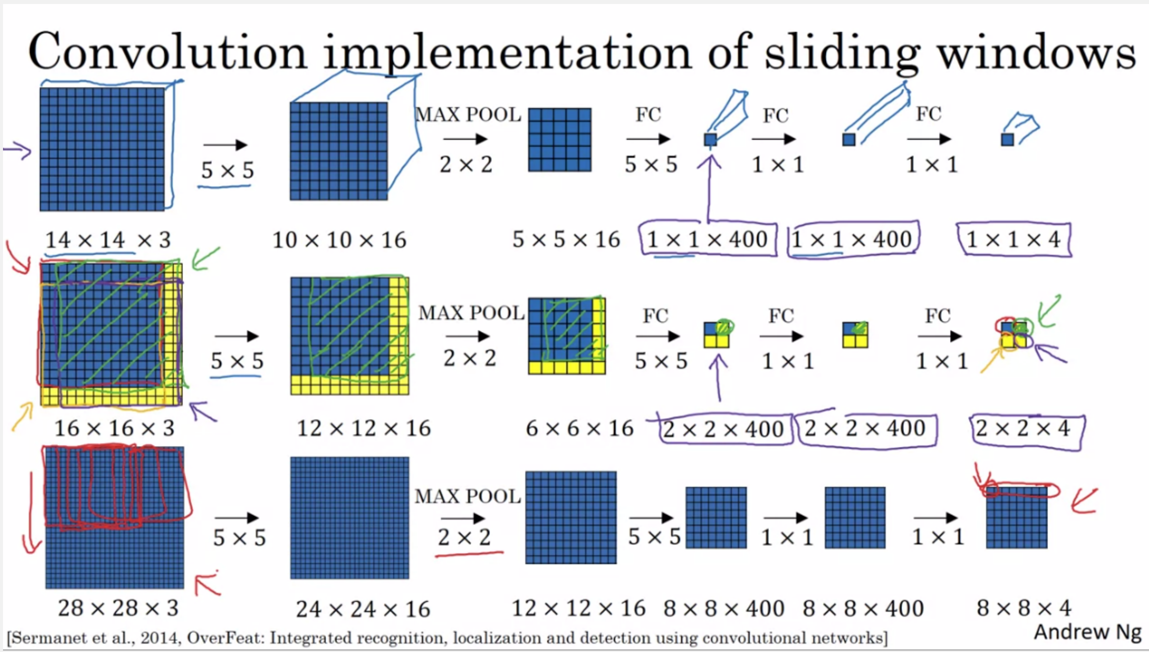
Convolutional Implementation of Slidding Windows

- Turning FC layer into convolutional layers
  for the last several FC layers in the end of CNN
    - idea: use the number of filter to stand for different neurals of fully connected layers



Convolution implemention of sliding windows

- First introduced by Sermanet et al., 2014

> OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks
>
> Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, Yann LeCun
>
> 
>

- instead of forcing you to run four propagation on four subsets of the input image independently.
- `Instead`, it combines all four into one form of computation and shares a lot of the computation in the regions of image that are common. So all four of the 14 by 14 patches we saw here.
- That's also why we addapate it to Conv models

Bounding Box Prediction

Output accurate bounding boxs

YOLO algorithme

`YOLO` stands for **You Only Look Once**

First introduced by Redmon et al.,2015

Use grid cells

- For each grid cell, we use the Conv Sliding Windows
- asign every object into a grid (only one grid is asigned)
- target output: $3 \times 3 \times 8$
- Specify the bounding box:
  - $b_h \& b_w$ could be bigger than 1.

Intersection Over Union

- What motivates the definition of IoU,
  `as a way to` evaluate whether or not your object localization algorithm is accurate or not.

- Evaluating object localization:

$$\text{Intersection Over Union (IoU)} = \frac{\text{size of intersection}}{\text{size of union}}$$

- "Correct" if IoU larger than ~0.5

Non-max Suppression

One of the problems of Object Detection as you've learned about this so far, is that your algorithm may find **multiple detections** of the same objects.

- differents grids will think that they have found the same object
- Those have smaller probability will be suppressed
- consider just the highlighted case of detection
- What we do exactly?
  - Each output prediction is

$$y = [p_c, b_x, b_y, b_h, b_w]^T$$
$$\text{where,} \quad c_1, c_2, c_3 \quad \text{are dismissed for instant}$$

  - Discard all boxes with $p_c \leq 0.6$
  - While there are any remaining boxes:
    - Pick the box with the largest p_c Output that as a prediction.
    - Discard any remaining box with $\text{IoU} \geq 0.5$ with the box output in the previous step

Anchor Boxes

Overlapping objects of differents types problem

pre-define two anchor boxes with different shapes

Previously:	**With two anchor boxes:**
Each object in training image is assigned to grid cell that contains that object's midpoint.	Each object in Training image is assigned to grid cell that contains object's midpoint and `anchor box` for the grid cell withe highest `IoU`.

Putting it together: YOLO Algorithme

1. Training set
    1. pedestrain
    2. cars
    3. motocycles
    - y is of size $3 \times 3 \times 2 \times 8$
        1. 3×3 stands for size of grid cells
        2. 2 stands for number of anchors
        3. 8 stands for number of single output
2. Making predictions
   use the out put of size $3 \times 3 \times 2 \times 8$ to detect where is the object and then use non-max suppression
3. Outputting the non-max supressed outputs
    - For each grid cell, get 2 predicted bounding boxes.
    - Get rid of low probability predictions.
    - For each class (pedastrain, car, motorcycle) **independently** use non-max suppression to generate final prediction.

Region Proposal: R-CNN

Segementation algorithme
find region that is high posible to have objects

Faster algorithms:

- R-CNN: Propose regions. Classify proposed regions one at a time. Output label + bounding box
- Fast R-CNN: Propose regions. Use convolutional implementationof sliding windows to classify all the proposed regions.
- Faster R-CNN: Use convolutional network to propose regions.

Citations:

- Girshik et. al,. 2013
- Girshik, 2015
- Ren et al., 2016