**CS231n: CNNs for Visual Recognition** **Spring 2017**

# Lecture 11

*Course Coordinator: Prof. Fei-Fei Li* *Scribes: Akash Gupta*

**Semantic Segmentation** - Input an image and output a decision of category for every pixel in that image or predicting labels for every pixel.

NOTE - Does not differentiate between instances, only cares about pixels.

Attempts -

- Sliding Window - Extract patches and classify center pixel using a CNN. $->$ Computationally expensive.
- Fully Convolutional - Design a network as a bunch of conv. layers to make predictions for all pixels all at once! $->$ Computationally expensive as spatial size remains the same.
- Fully Conv with Downsampling and Upsampling to decrease computation.

**Types of Upsampling** -

1. Unpooling
   - Nearest Neighbor
   - Bed of Nails
2. Max Unpooling - Associated with each max pooling layer to unpool at same place where maxpooling took place. Helps preserve spatial information lost during max pooling.
3. Transpose Convolution - Learnable upsampling.
   - Multiply filter by scalar value in input feature map. For stride 2, for every one pixel movement the filter moves by 2 pixels and summing on overlaps. Stride gives ratio between movement in output and input. Also goes by different names - Deconvolution (bad since not inverse of convolution), Upconvolution, Fractionally strided convolution, Backward strided convolution.

Q) *Why do we sum in Transpose Convolution?*
Ans) Yeah summing can introduce differences in magnitude. Active area of research. Use filters 4x4 stride 2 or 2x2 stride 2.

**Classification + Localization** - In addition to predicting the class we want to predict the location of object a well. An assumption, is that we know there is going to be fixed number of objects.

1. Treat localization as a regression problem.
2. At the end, have 2 fully-connected layers, one for classification and other for regression.
3. At the time of training such network, we have 2 losses - Softmax loss and L2 loss/L1/smooth L1.

Q) *These losses have different units, do they dominate the gradient descent?*
Ans) Yeah, that is a problem and so to solve this take weighted sum of losses. This hyperparameter is different from other since here we are changing the value of loss itself and so is difficult to set than other hyperparams.

**Human Pose Estimation** - Represent pose as a set of fixed coordinates and train the CNN to output those coordinates and then apply L2 loss.

**Object Detection**- No knowledge of how many objects appear in the image and we want to draw a bounding box around every object.

- Object Detection as Regression? - Varying number of coordinates. Train separate CNNs. $->$ Intractable.
- Sliding Window - Take different crops and input to classification network which will output different classes + Background. $->$ Problem - How to choose the crops? Any size, any location, any aspect ratio. Need to apply CNN to huge number of locations and scales, very computationally expensive.
- Region Proposals/Regions of Interest (RoIs) - These networks gives fixed number of region proposals in the image and find "blobby" regions that are likely to contain objects. Ex - Selective Search $->$ High Recall.
- R-CNN - Given an input image, an RPN will ouput fixed number of region proposals and we will resize and run through a CNN to output a class and a regression for Region Proposal box. Region proposal from RPNs are not learnt.
- What kind of data? - Fully supervised - Images with bounding boxes marked with captions. Research is going on for when you don't have that data or noisy.
- R-CNNs are pretty slow ( 84 hours) and takes too much space to save region proposals.
- Fast R-CNN - First run the ConvNet on input image and then predict RoIs on this feature map. Then RoI pooling to change to fixed size and then perform classification. $->$ Exceptionally fast.
- Faster R-CNN - Insert RPN to predict region proposals from features. Train with 4 losses:
  1. RPN classify object/not object.
  2. RPN regress box coordinates.
  3. Final classification score (object classes)
  4. Final box coordinates.
- How to train Faster R-CNN since we don't have GT labels for Region Proposals? $->$ Bit hairy and dark magic for setting hyperparams. Works on the idea of threshold overlapping of RoIs with objects.

**Detection without proposals** -

- YOLO - You Only Look Once
  - Divide in a fixed size grid and for each grid predict many base bounding boxes and classification.
  - Output - $7 \times 7 \times (5 * B + C)$
  - Train using a big CNN.
  - Idea is to do object detection in a single shot.

**NOTE** - Region based tend to give higher accuracy but are slower when compared to SSD type methods.

**Dense Captioning** - Object detection + Image Captioning

**Instance Segmentation** - Predict locations of all the objects plus a segemntation mask for those objects or predicting which pixels belong to which instance of the object. Ex - Distinguishes between multiple instances of dog in an image in the segmentation mask.

- Mask R-CNN - Faster R-CNN with 2 branches, 1) Classification + bounding box coordinates, 2) semantic segmentation mini-network which will classify for each pixel in a Region proposal.
- Unifies all the above ideas.

Q) *How much training data do you need for this?*
Ans) Example- Microsoft COCO dataset - 200,000 images with 80 category labels and each image having 5 to 6 instances per image.