أكاديمية كاوست
KAUST ACADEMY

# Day 3 (part 2): Computer Vision Summary

Note: Do Not depend entirely on this and study from the official slides.

Contributed by: Hassan Mohammed Nasr

# Object Detection

- Moving beyond is "there a cat" to "where are the cats"
    - Input: RGB Image
    - Output: A set of detected objects. Each with:
        - Class Label. : from fixed known set of categories
        - Bounding box : Four numbers defining the rectangle location (x,y,w,h)
    - Challenge: unlike classification (output always 1) detection requires predicting a variable number of outputs per image. Moreover for detection we need higher image resolution.

# Evaluation Metric

- Intersection over Union (IoU): is a metric to measure the overlap between the predicted box and the ground truth box. It tells us if the box is correct or not.

$$IoU = \frac{\text{Area of intersection}}{\text{Area of Union}}$$

- IoU = 1 : perfect match
- IoU = 0 : no overlap
- IoU > 0.5 : usually considered correct

# Naive Approach

- if we know that there is only one object in the image, we can treat it as a regression problem.

  Method: Fine tune a pre trined model, and add a fully connected layer (box head) that output 4 numbers for the box coordinates.
  Loss: weighted sum of softmax and l2 loss $L_{total} = L_{cls} + L_{reg}$

  - Problem: can't handle images with multiple objects because the network has a fixed number of outputs
    - Solution? : Slide a window over crops of the image and run a CNN classifier on each crop
      - Problem: Can work, BUT computationally impossible to do as we can have over 58 million boxes for only one small image
        - Solution: Use fast, heuristics algorithms to find a small set of boxes that are likely to cover all objects

          The R-CNN Family

# Two Stage Detectors (The R-CNN Family)

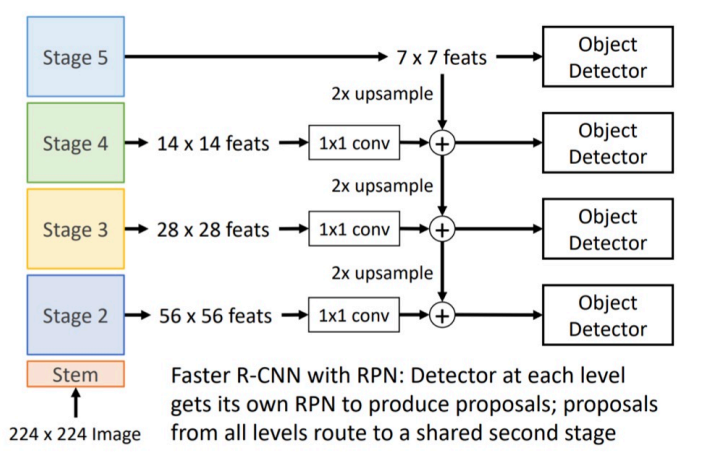
- Stage 1: Find Potential boxes using "selective search" based on heuristics (not learnable)
- Stage 2: Run Detector on these proposals and classify them

| Model | Description | Problem Solved |
| --- | --- | --- |
| • R-CNN | Use selective on the image to generate ≈ 2000 boxes (proposals) then run pass each box to CNN individually | Faster than the Naive Approach |
| • Fast R-CNN | Rob pooling: Runs the image through the CNN once to get feature map, then crop the features (proposals) from the feature map instead of raw pixels | Uses CNN first then selective search, making it faster to process |
| • Faster R-CNN | RPN(Region Proposal Network) Replace the selective with a small neural network that learns to propose boxes ( is this region represent an object?) | Replaced selective search:Every thing is now learnable |

# Closer look onto Faster R-CNN Family

- Concept: Make CNN do proposals
- Insert RPN to predict proposals from features before passing it to detection head
- Loss: Faster RCNN train using 4 types of loss:
  - RPN Classification : Binary classification, predicts is this box an object or not
  - RPN Regression: predict transform from anchor box to proposal box (tries to adjust the box so it can represent object)
  - Object Classification: Multi classification, classify the proposals as background/ object class
  - Object Regression: predict transform from proposal box to object box (final bounding box)

- Because we need to detect objects with different scales, we use something called feature pyramid network, which we pass different scales for the same object use different detectors for each scale but we use top down connections that feed information from high level features back down to lower level features




- Problem: Object detectors often output many overlapping detections ( multiple bounding boxes for same object)
- Solution: Non Max Suppression (NMS) keep only boxes with IoU bigger than threshold then pick the highest scoring box

# Single Stage Detectors (YOLO: you only look once)

- Concept: Discard the separate "proposal" step, it treats detections as single stage
- Used for real time
  - Step 1 : Divide the input image into an SxS grids ( 7 x7 )
  - Step 2: If the center of an object falls into a specific grid cell, that cell is responsible for detecting it
  - Step 3: Each grid cell predicts B Bounding Boxes (2) and a Class Probability vector (e.g., "Dog", "Car") . Each box includes a Confidence Score (How confident is the model that this box actually contains an object?)
  - Step 4: In Training We compare the predicted boxes to the Ground Truth. The box with the Highest IoU is the "winner" and learns to predict that object. In testing We get many duplicate boxes. We apply Non-Max Suppression (NMS) to remove overlaps and keep only the best unique detections .
    - Loss: YOLO uses 3 losses
      - Classification
      - Confidence
      - Regression / localization
- Limitations : Fixed input size, difficult to detect small objects and in yolo v1 each grid cell can predict only one class
  - Solution: predict class per bounding boxes not per cell
- Instance Segmentation : Use Faster RCNN to do instance segmentation, but simply add Mask Branch at the end after fixing misalignment (in shape) issues ensuring pixel level masks