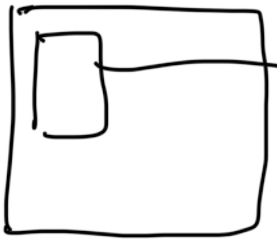
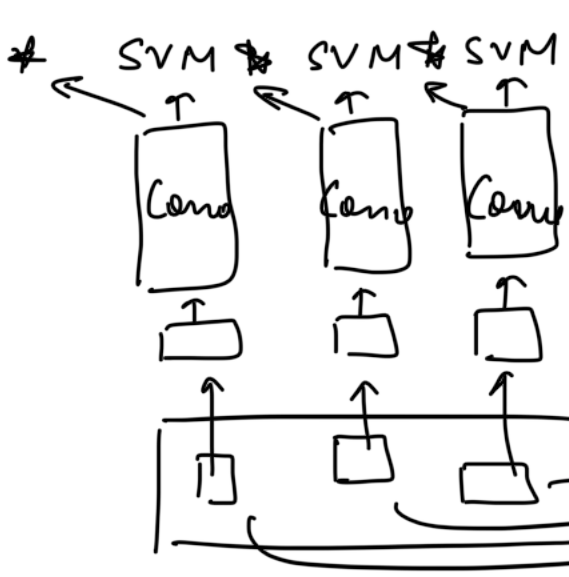
Week-10 DLP

L1 Object detec^n - Introduction

- Object detec^n → impt. task in CV.
  ↳ The task is to assign a label & a bounding box to all the objects that are present in the image.
- 
  Labels = [1, 2, 3]
  either from 1, 2 & 3.
- Basically, it will multiple classes of image classifica^n. The problem is too many positions and scales to test. If your classifier is fast enough, you can do it.
- R-CNN (Region-based ConvNet) - ConvNets are computationally demanding. But we can't test all positions and scales.
  ↳ Solu^n - We will only look at a subset of positions. Choosing them wisely.
- Region proposals → find blobs that will most likely to contain objects. The proposal is 'class-agnostic'.
- Process →
  1. input image
  2. extract reg. proposals (~2k)
  3. compute the CNN features
  4. classify regions

- slow RCNN



→ classify regions with SVMs.

→ each region goes through ConvNet

→ warpped image regions

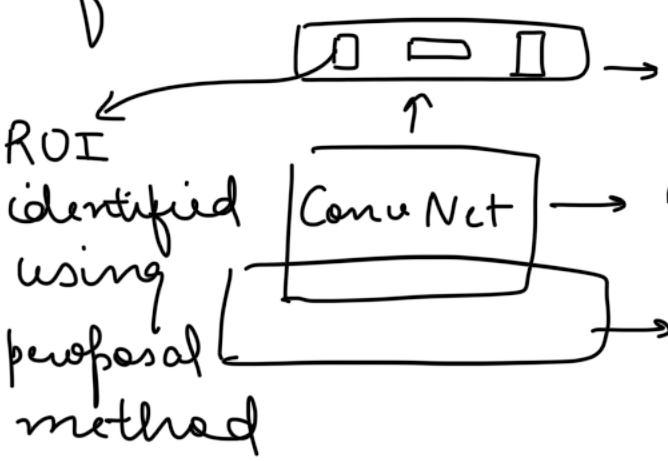
→ ROI (region of interest) from the above-mentioned proposal method (~2k)

* BBox reg. ⇒ apply bounding-box regressors.

- problems with slow R-CNN.
  - → logloss, hinge loss, squared loss.
  - → training is slow (84 h) + lot of disk space.
  - → inference (detect) is slow. current rate is 47s/image. It was later fixed by SPP-net.

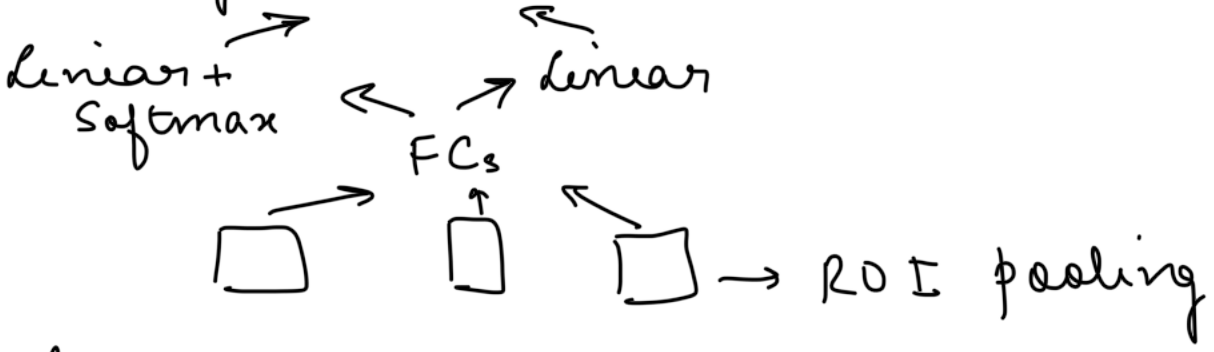
L2 Fast RCNN & Faster RCNN

- fast RCNN (test time)



- during training

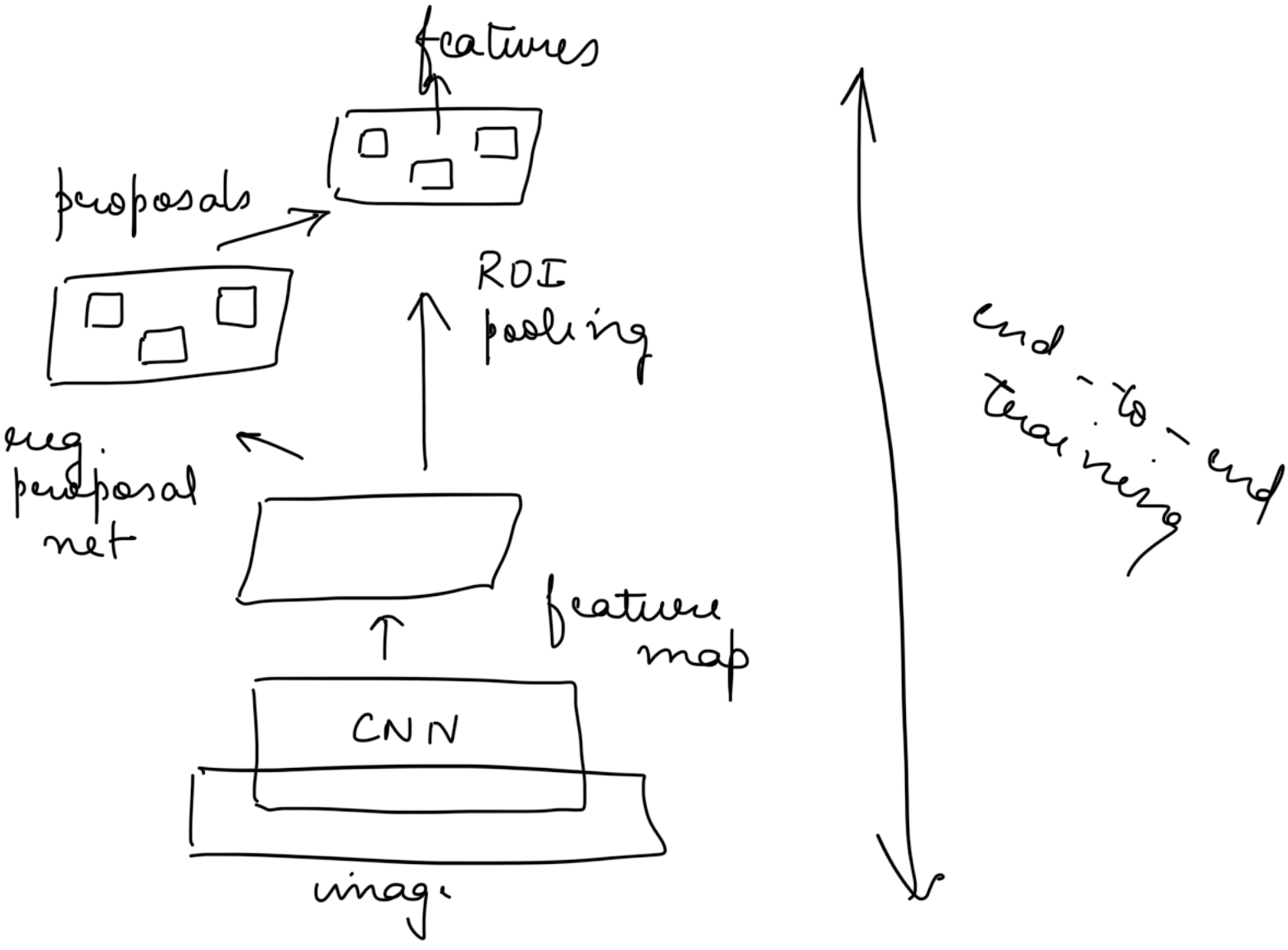
Log loss + Smooth L1 Loss ⇒ Multi-task loss



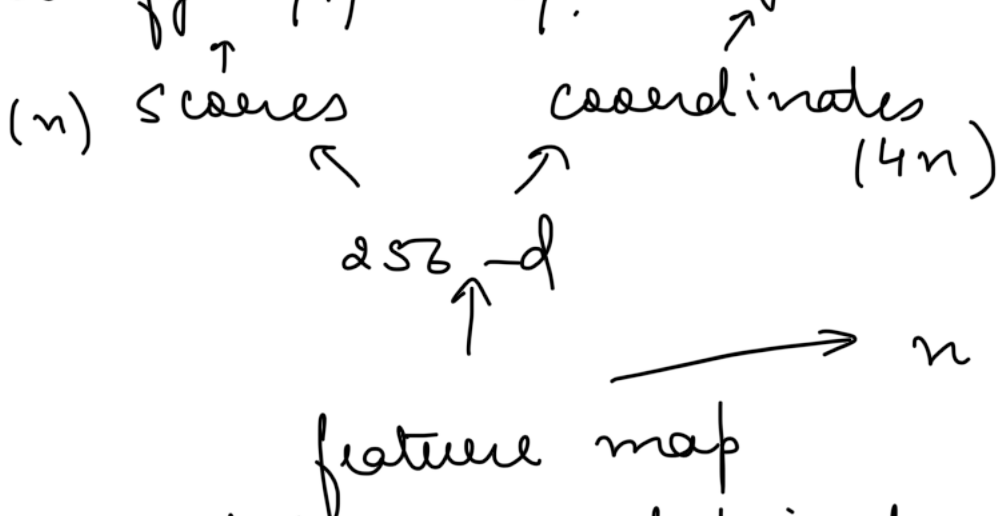
- Cons -

1. out of network region proposals. Selective search. 2s/image.

- Faster RCNN
  - → solely based on CNN.
  - → no external modules.
  - → each step is end-to-end.



- Region proposal network
  → slides a small window on the feature map.
  → pos^n of the window with ref. to image.
  classify obj./not obj. regress box locations



  → it is fully convolutional, it is trained end-to-end.
  → RPN shares convolutional features maps with the detec^n network.

L3 YOLO – You Only Look Once

- It is much advantageous over RCNN.
  1. a single neural network for localiza^n and classifica^n
  2. inference happens only once.
  3. looks at the entire image each time leading to less false positives.

- Steps –
  1. image is split into S×S cells.
  2. for each cell, generate B bounding boxes.
  3. for each bounding box, we have 5 predic^n → x, y, w, h, and confidence.

4. also, we do object classificaⁿ.

- each cell predicts a class probability conditioned on object being present.

Eg. P(Car | Object)

- $P(\text{Class} | \text{Object}) \times P(\text{Object}) = P(\text{Class})$

- Yolo architecture –

1. input → 448 × 448 × 3
2. 24 conv. layers
3. S = 7, B = 2, C = 20 (default)
4. Output is $S \times S \times (5B + C)$

L4 Object Detecⁿ Metrics & Experimental Results

- we use MAP (mean avg. precision)

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

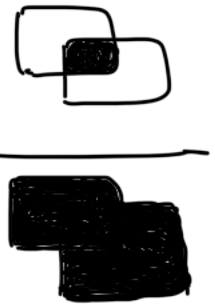
	Actual +	Actual −
Predicted +	TP	FP
Predicted −	FN	TN

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- precision – refers to finding TP out of all positive predictions

- recall - how to find the TP out of all predictions
- If IOU > certain threshold (0.5), the predic^{n} is TP. If IOU ≤ threshold, the predic^{n} is a FP.

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



- AP (Avg. Precision) - area under the precision - recall curve. Avg. of precision values across a range of recall values. This is c/d AP for that specific class.
- MAP - mean of the AP values across all object classes.
- error analysis -
  1. makes far less background errors
  2. but far more localiza^{n} errors.