Deepmind - Advanced models for CV

Thursday, 25 February 2021 12:53

- Object detection = classification + localization

  RGB image → Class label, Object bounding box

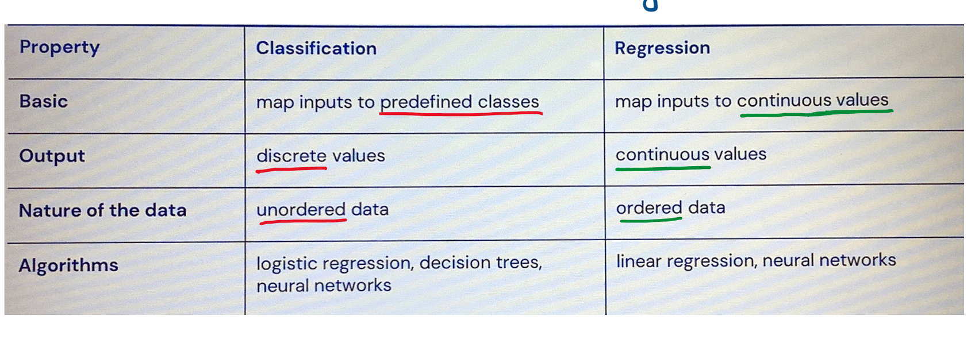
  training: ① Softmax + cross-entropy for classification

  $$\ell_{CE}(f_{sm}(x), t) = -\sum_{j=1}^{k} t_j \log[f_{sm}(x_j)] = -\sum_{j=1}^{k} t_j \left[x_j - \log \sum_{\ell=1}^{k} e^{x_\ell}\right]$$

  assign data points to categories, discrete output

  ② Quadratic loss for regression (bounding box prediction)

  $$\ell_2(x, t) = \| t - x \|^2 \quad ; t \text{- ground truth}$$



Property	Classification	Regression
Basic	map inputs to predefined classes	map inputs to continuous values
Output	discrete values	continuous values
Nature of the data	unordered data	ordered data
Algorithms	logistic regression, decision trees, neural networks	linear regression, neural networks

  we can convert regression into classification by:

  discretising the output values (one-hot label) → refine through regression (regrission in local area)

- Case study 1: Faster R-CNN

  two-stage detector: 1. Identify good candidate boxes.
  2. Classify and refine.

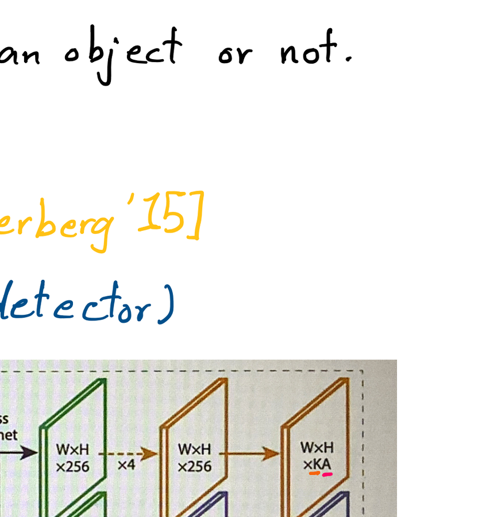
  ① Identify good candidate bboxes

  a. Discretise bbox space (x_c, y_c, h, w)
     1. anchor points for (x_c, y_c)
     2. scales and ratios for (h, w)

  b. n candidates per anchor.

  c. predict objectness score for each bbox.

  d. sort and keep top k

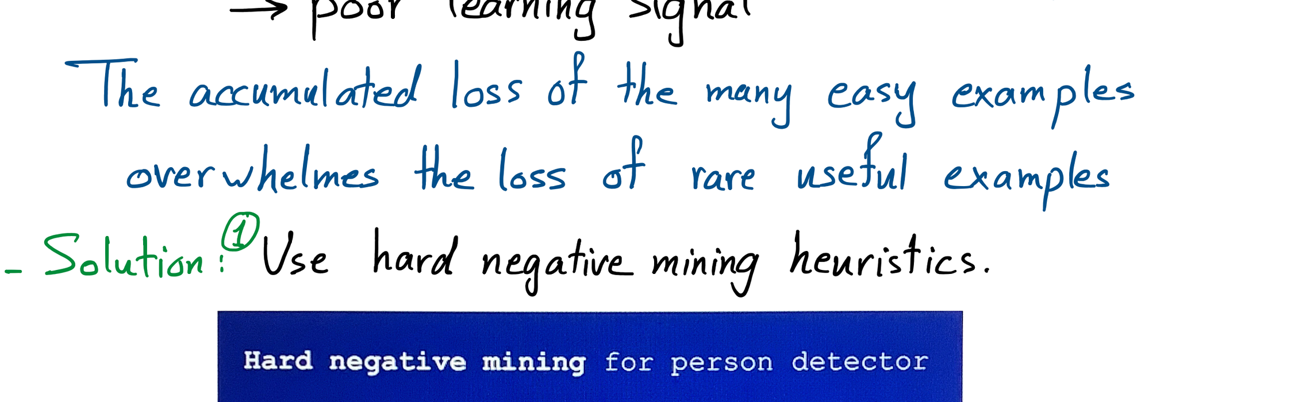


  objectness: check whether is there an object or not.

  ② Refine through regression MLP.

  Spatial Transformer Networks [Jaderberg '15]

- Case study 2: Retina Net (one-stage detector)



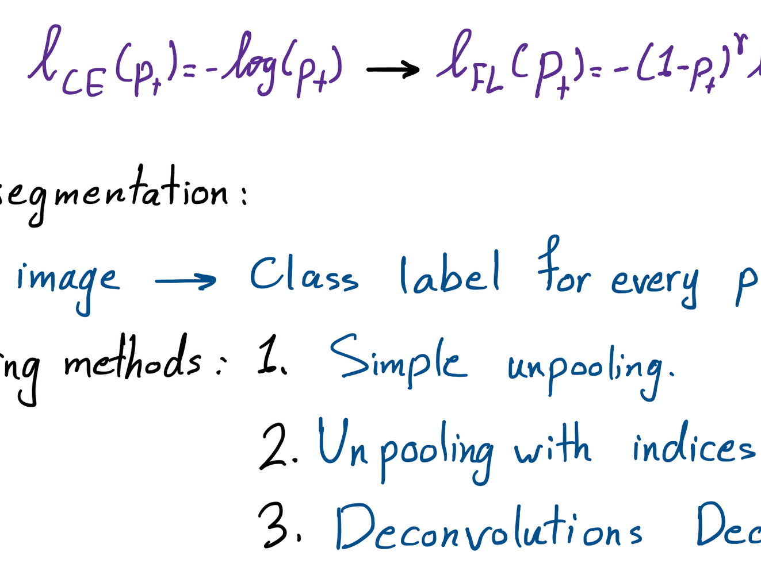
  k: number of classes, A: number of anchors.

  - Problem: most of the candidate bboxes are easy negatives; (belong to background)

    → poor learning signal

    The accumulated loss of the many easy examples overwhelmes the loss of rare useful examples

  - Solution: ① Use hard negative mining heuristics.



Hard negative mining for person detector

1. Get set of positive examples
2. Get a random subset of negative examples (full set is too big)
3. Train detector
4. Test on unseen images
5. Identify false positive examples (a person was detected where there was none) = **hard negatives**; add them to the training set
6. Repeat from 3.

  ② Retina Net uses Focal Loss (FL)

  $$\ell_{CE}(p_t) = -\log(p_t) \rightarrow \ell_{FL}(p_t) = -(1-p_t)^{\gamma} \log(p_t)$$

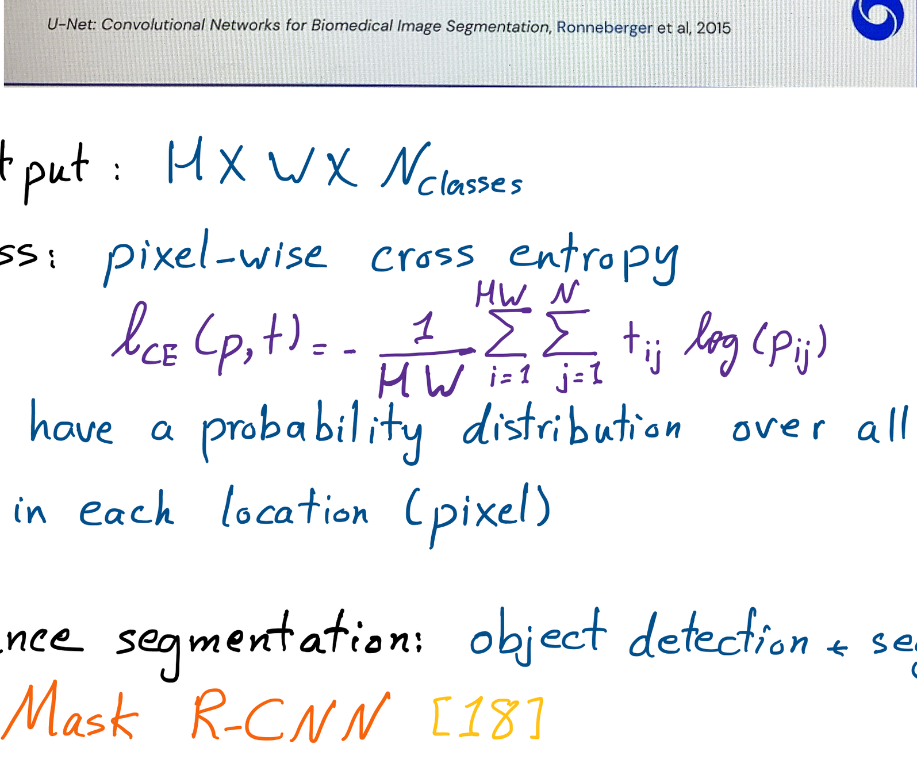
- Semantic segmentation:

  RGB image → Class label for every pixel

  Up sampling methods: 1. Simple unpooling.
  2. Unpooling with indices SegNet
  3. Deconvolutions DeconvNet.

- Case study: U-NET

  - Encoder-decoder model
  - Skip connections to perserve details.



  Output: $H \times W \times N_{classes}$

  Loss: pixel-wise cross entropy

  $$\ell_{CE}(p, t) = -\frac{1}{HW} \sum_{i=1}^{HW} \sum_{j=1}^{N} t_{ij} \log(p_{ij})$$

  we have a probability distribution over all classes in each location (pixel)

- Instance segmentation: object detection + segmentation

  Mask R-CNN [18]

- Evaluation metrics:

  1. Classification:
     Accuracy: percentage of correct predictions.
     Top-1: top prediction is the correct class.
     Top-5: correct class in top-5 predictions.

  2. Object detection and segmentation:
     Intersection-over-Union (IoU) $J(P,T) = \frac{P \cap T}{P \cup T}$

- Transfer learning:

  Let $D = \{X, P(X)\}$ be a domain, and $T = \{Y, f(.)\}$ a task defined on this domain; given a source domain and task $\left(\frac{D_S}{T_S}\right)$ and a target domain and task $\left(\frac{D_T}{T_T}\right)$, reuse knowledge learnt by f_S in f_T.

  Taskonomy: Disentangling task transfer learning [Zamir '18]

  - Sim2Real is part of transfer learning:
    Use automatic domain randomization:
    data augmentation + hard negative mining

- Motion helps object recognition when learning to see.

  Natural data augmentation; translation; scale; 3D rotation; camera motion; light changes.

- Optical flow estimation:

  Pair of RGB images → Dense flow map, 2D translation displacements.

  - Case study: FlowNet
    - Encoder-decoder architecture.
    - Supervised learning.
    - loss: Euclidean distance

- Video models using 3D convolutions:

  Video as a volume:
  1. stack frames $T \times H \times W \times 3$
  2. apply 3D convolutions

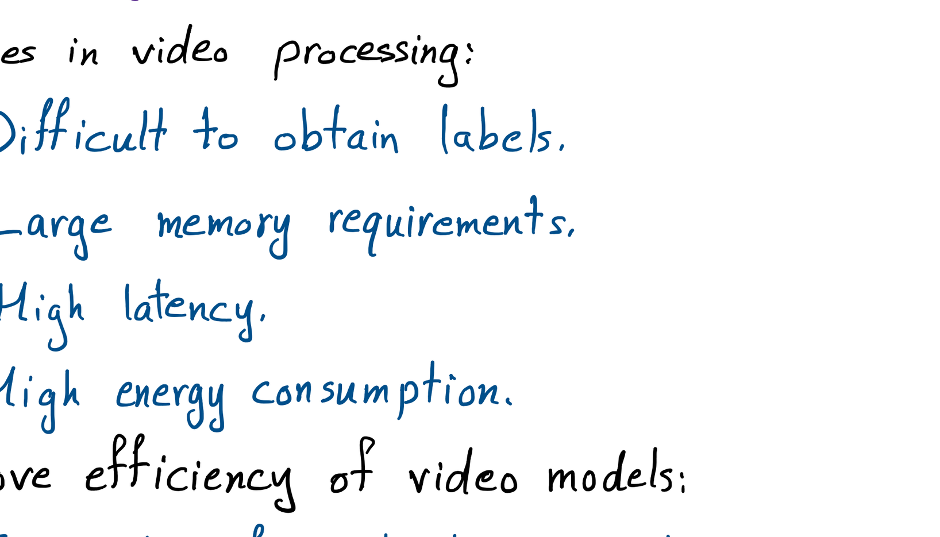
  The kernel slides across space and time to generate spatio-temporal feature map.

  - 3D convolutions are non-causal (we need $t+1$)
  - masked 3D convolutions are causal (for real-time) (put a mask on the weights of the filter that belong to the future)

- Action recognition:

  RGB video → action label one-hot

  - Case study: SlowFast



  - Transfer learning: inflating 2D kernels into 3D

    2D image kernel —tile along t→ 3D action classifier kernel

- Challenges in video processing:
  1. Difficult to obtain labels.
  2. Large memory requirements.
  3. High latency.
  4. High energy consumption.

- Improve efficiency of video models:
  1. Inspiration from biological systems.
  2. Maximize parallelism.
  3. Exploit redundancies in the visual data.

- Self-supervision - Metric learning:

  learn to predict distances between inputs given some similarity measure.

  1. Contrastive loss (margin loss)

  $$\ell(r_0, r_1, y) = y\, d(r_0, r_1)^2 + (1-y)(\max(0, m - d(r_0, r_1)))^2$$

  2. Triplet loss:

  $$\ell(r_a, r_p, r_n) = \max(0, m + d(r_a, r_p)^2 - d(r_a, r_n)^2)$$

  - better than contrastive loss
  - relative distance more meaningful than a margin.
  - hard negative mining to select informative triplets

    Sampling matters in Deep embedding [Wu '18]