Deep Learning for Image Analysis - Object detection

Thomas Walter, PhD

Centre for Computational Biology (CBIO)
MINES Paris-Tech, PSL Research University
Institut Curie, PSL Research University
INSERM U900

Overview

Overview

Classification, segmentation and detection

- Image Classification: assign a label to each image.
- Semantic image Segmentation: partition the image, i.e. assign a label to each pixel.
- Object detection: detect instances of semantic objects of certain classes in images [Ren et al., 2017, Zhao et al., 2019].
- Object detection has thus two components:
  - *Object localization:* to determine where objects are located in a given image
  - *Object classification:* which category each object belongs to
- (Multiple) instance Segmentation: combines the objectives of object detection and segmentation.

Object detection: example

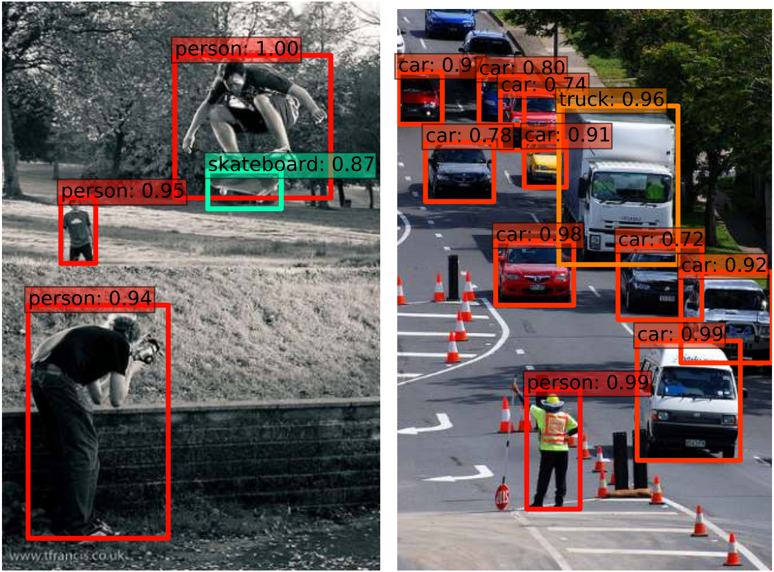

Figure: Object detection in action. Image taken from [Liu et al., 2016]

Object detection: example

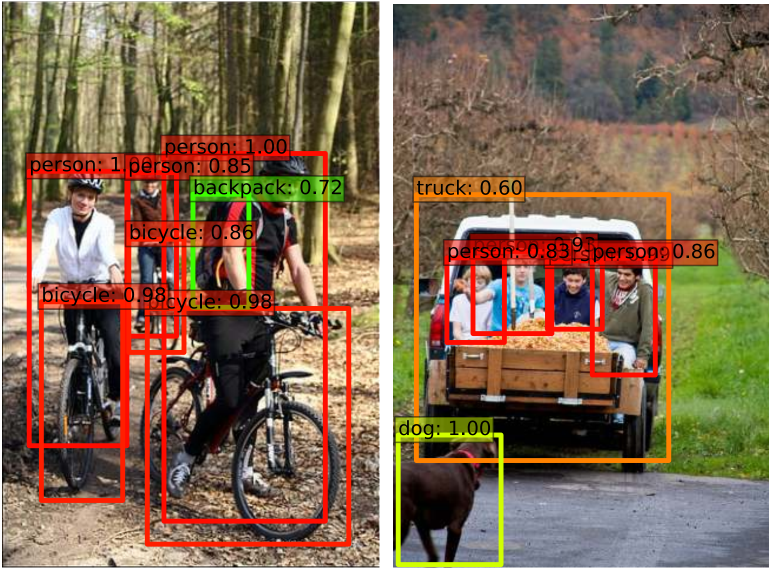

Figure: Object detection in action. Image taken from [Liu et al., 2016]

Object detection: example

Figure: Object detection in action. Image taken from [Liu et al., 2016]

Early approaches: the sliding window approach

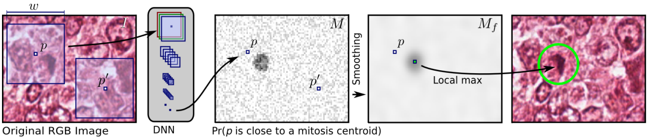

Figure: Mitosis detection in stained tissue sections [Cireșan et al., 2013].

- Approach was the winner of a mitosis detection challenge [Cireșan et al., 2013].
- Fixed size sliding window approach: each crop is presented to a CNN.
- The posterior probability is stored as an image value.
- Local maxima of this probability map indicate the presence of an object.
- Special case of object detection: the size of the objects was known before.

A milestone in object detection: R-CNN

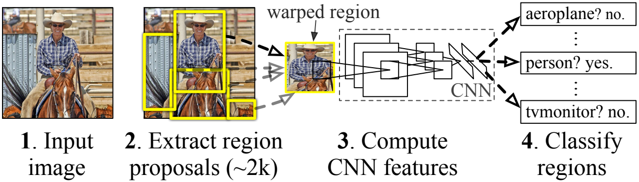

Figure: R-CNN strategy [Girshick et al., 2014]

.

- Starting from region proposals (by any method): ~ 2000 regions).
- Warping / Cropping of the selected regions into fixed resolution and extraction of a 4096-dimensional feature vector with a pretrained CNN.
- Classification with SVM (object types and background).
- Adjustment by bounding box regression
- Filtering with greedy non-maximum suppression (NMS): removal of regions with low overlap with a single object.

Drawbacks of R-CNN

- Fixed input size for the CNN: distortion and rescaling of images is necessary.
- Multi-stage pipeline (no end-to-end solution, which is globally optimal).
- Training is expensive in space and time, mainly due to the separate feature extraction step.
- Computationally expensive at prediction time, as many (overlapping) regions need to be classified.
- Sub-optimal region proposal step.

Fast R-CNN

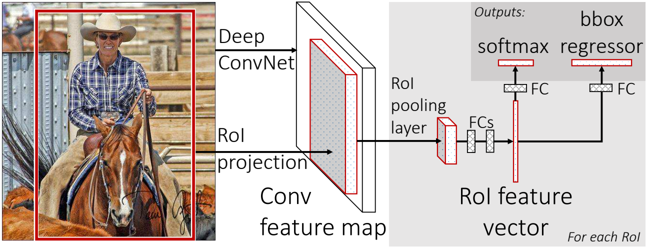

Figure: Fast R-CNN [Girshick, 2015]

.

- The entire image is processed by a neural network: generation of feature maps.
- Region are proposed by some algorithm (as before).
- To each region, a ROI pooling layer is applied.

Fast R-CNN: ROI pooling layer

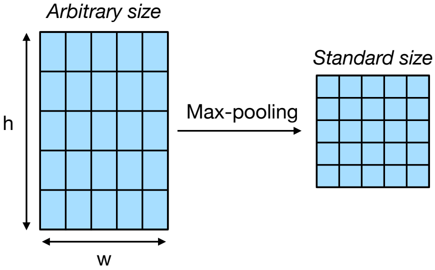

Figure: ROI pooling layer in Fast R-CNN

.

- Each region of arbitrary size $w \times h$ is divided into $W \times H$ tiles. W and H are fixed, where as w and h are arbitrary.
- For each tile, the maximum is calculated (max-pooling operation) in the feature map.
- The output (fixed size) can then be processed by dense layers.

Fast R-CNN: Output layer

- Two outputs:
  - Classification output (with a standard softmax layer)
  - Bounding box regression: prediction of position and extension offsets with respect to the original region proposal.
- The loss has thus two components: L_{class} which is the standard cross-entropy loss and L_{loc}, the localization loss (L_1 loss of the offsets with respect to the proposed regions).
- During training, the batch is constructed from many objects drawn from very few images. Feature maps do then not need to be recalculated.
- For the prediction, each class gets its own region proposal, that is processed individually with non-maxima suppression.

Faster R-CNN: motivation

- Fast R-CNN solves nearly all problem of R-CNN, and is end-to-end given a set of region proposals.
- The problem is that we still need to make region proposals to start with (time-consuming and two-stage algorithm).
- Faster R-CNN [Ren et al., 2017] trains a network called Region Proposal Network (RPN) to overcome this issue.

Faster R-CNN: Idea

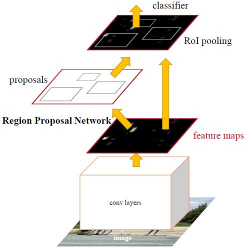

Figure: Faster R-CNN [Ren et al., 2017]

.

The idea is to share convolutional feature maps at test-time, i.e. to use the CNN feature maps calculated for the entire image for both region proposal and object classification [Ren et al., 2017].

Faster R-CNN: shared layers

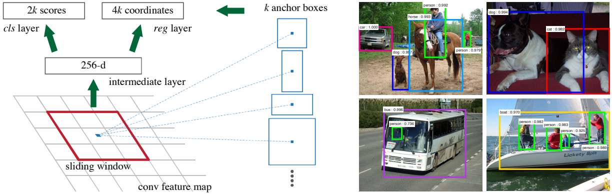

Figure: Faster R-CNN [Ren et al., 2017]

.

- First, we run the image through convolutional layers of a CNN and obtain feature maps that will serve both the region proposal and the object classification.
- We now "slide" a small $n \times n$ network over the common feature map. In practice, this is implemented as a convolutional layer, followed by $1D$-convolutions.
- The size can be relatively small (in [Ren et al., 2017], it is 3×3); the receptive field is much larger.

Faster R-CNN: Region proposal network (RPN)

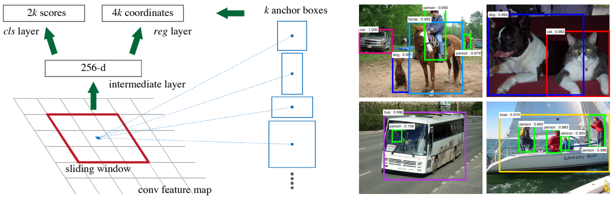

Figure: Faster R-CNN [Ren et al., 2017]

.

- This small network completes the Region Proposal Network (RPN).
- The RPN outputs a set of rectangular object proposals, each with an objectness score.
- For this, we define k anchor regions (defined by scale and aspect ratio) at each sliding-window location.

Faster R-CNN: Region proposal network (RPN)

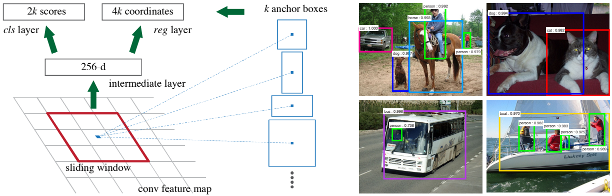

Figure: Faster R-CNN [Ren et al., 2017]

.

- For each sliding windows location and each of the k anchors, we predict:
  - Objectness (object yes/no) of the anchor.
  - Width, height and offset w.r.t. the anchor.
- During training, an anchor is considered to be positive if the $IoU > 0.7$ or if the IoU is maximal among all anchors and none is larger than 0.7. Negative anchors have $IoU < 0.3$.

Faster R-CNN: Region proposal network (RPN)

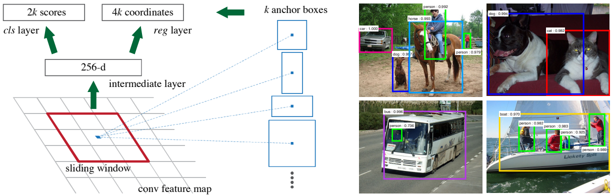

Figure: Faster R-CNN [Ren et al., 2017]

- We define a combined loss (as for Fast R-CNN), as sum of the classification loss and bounding box regression loss.
- Classification loss: cross entropy for a binary classifier, indicating whether the region contains an object or not.
- Regression loss compares for each region proposal its offsets to the anchors with the offsets of the ground truth box.

Faster R-CNN: Training

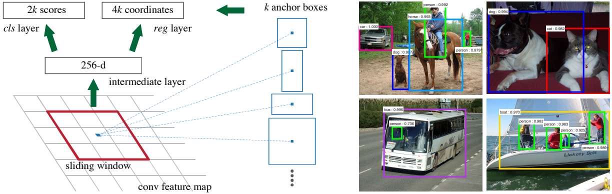

Figure: Faster R-CNN [Ren et al., 2017]

- We now simply apply a Fast R-CNN to the region proposals provided by the RPN.
- The shared layers are trained in an alternating scheme.
- After this initial training, the shared layers are frozen and the separate layers are trained end-to-end.

YOLO: You only look once

- Problem of most detection systems:
  - First: region proposals
  - Second: classification of all region proposals individually
  - Consequently: the best performing methods are very slow and not applicable in real-time
- YOLO [Redmon et al., 2016]: end-to-end strategy that only uses one forward-pass of an image for object detection.

YOLO: principle

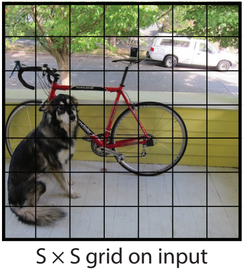

Figure: YOLO: the image is partitioned by an $S \times S$ grid.

- First, the image is divided into a $S \times S$ grid of cells.
- For each of these cells, we will then predict:
  - Location and size of B different boxes
  - A confidence score that the box contains an object.
  - The class of the object in each of the B boxes.

YOLO: principle

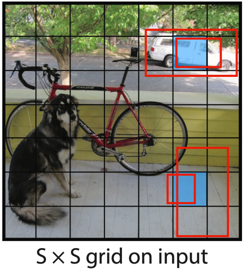

Figure: YOLO: the image is partitioned by an $S \times S$ grid.

- First, the image is divided into a $S \times S$ grid of cells.
- For each of these cells, we will then predict:
  - Location and size of M different boxes.
  - A confidence score that the box contains an object.
  - The class of the object in each of the B boxes.

YOLO: each cell predicts boxes and confidences

- If the center of an object falls into one cell, that cell is responsible for the prediction of the object.
- Geometry and position of the bounding box:
  - Center (x, y) with respect to the origin of the cell.
  - The relative width and height: w, h (normalized by image width and height).
- The confidence score of a predicted bounding box $\hat{B}_i$ is defined as:

$$Conf_i = P(Obj)IOU(\hat{B}_i, B_i)$$

  where B_i is the ground truth bounding box.
- At test time the confidence values $Conf_i$ are predicted (together with the bounding box geometry).

YOLO: prediction of the class

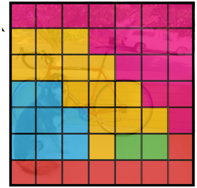

Figure: Class map prediction

- Each cell of the partition predicts also K class probabilities, conditioned on the presence of an object $P(C_i|obj)$.
- During prediction, this class probability is multiplied with the confidence score:

  $$P(C_k|obj)Conf = P(C_k|obj)P(obj)IOU(\hat{B}, B) = P(C_k)IOU(\hat{B}, B)$$

  This provides class-specific confidence scores for each of the predicted boxes.

YOLO: Output

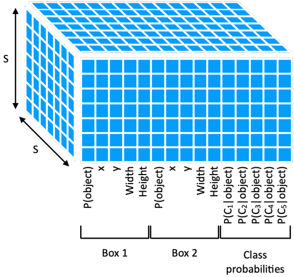

Figure: Output layer of YOLO for 5 classes

If we assume that our initial grid was 7×7, we predict two boxes per cell and that we have 20 classes, we obtain as output layer a tensor of dimension:

$$(7 \times 7) \times (2 \times 5 + 20) = 1470$$

This is the number of output variables we would predict.

YOLO: Examples

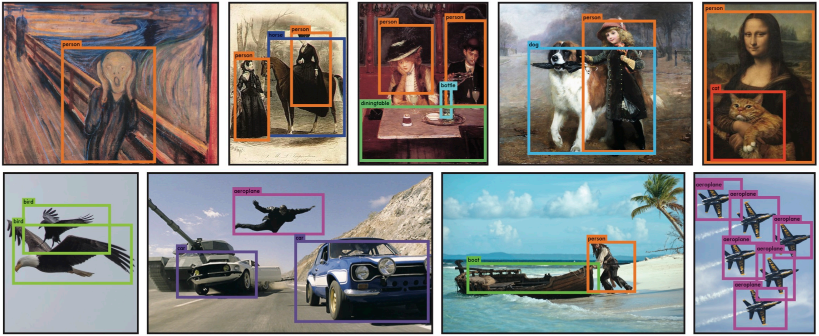

Figure: Examples for YOLO detections

The method is extremely fast. Localization is less precise than for Faster R-CNN.

Conclusion

- Object detection is a major challenge in Computer Vision with applications in biomedical image analysis, autonomous driving, industrial applications, etc.
- CNNs outperform most traditional methods by a large margin.
- Today, object detection is among the most stunning applications of Computer Vision.
- There are hundreds of methods, but the most important advances were achieved by R-CNN, Fast R-CNN, Faster R-CNN and YOLO.
- They can be combined with segmentation (Mask R-CNN).

References I

[Cireșan et al., 2013] Cireșan, D. C., Giusti, A., Gambardella, L. M., and Schmidhuber, J. (2013). Mitosis Detection in Breast Cancer Histology Images with Deep Neural Networks. In Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J. M., Mattern, F., Mitchell, J. C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M. Y., Weikum, G., Mori, K., Sakuma, I., Sato, Y., Barillot, C., and Navab, N., editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013*, volume 8150, pages 411–418. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Girshick, 2015] Girshick, R. (2015). Fast R-CNN. *arXiv:1504.08083 [cs]*.

References II

[Girshick et al., 2014] Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *arXiv:1311.2524 [cs]*.

[Liu et al., 2016] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. *arXiv:1512.02325 [cs]*, 9905:21–37.

[Redmon et al., 2016] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. *arXiv:1506.02640 [cs]*.

[Ren et al., 2017] Ren, S., He, K., Girshick, R., and Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149.

References III

[Zhao et al., 2019] Zhao, Z.-Q., Zheng, P., Xu, S.-t., and Wu, X. (2019). Object Detection with Deep Learning: A Review. *arXiv:1807.05511 [cs]*.