Object Detection

CS 6384 Computer Vision
Professor Yu Xiang
The University of Texas at Dallas
Object Detection

• Localize objects in images and classify them

Why using bounding boxes?
• Easy to store
  • \((x, y, w, h)\): box center with width, height
  • \((x_1, y_1, x_2, y_2)\): top left corner and bottom right corner

• Easy for image processing
  • Crop a region

Wikipedia
Object Detection

• Localization + Classification

Input Image → Localization → Crop → Classifier → Dog
Localization: Sliding Window

• Select a window with a fixed size

• Scan the input image with the window (bounding box)

• How to deal with different object scales and aspect ratios?
  • Use boxes with different aspect ratios
  • Image pyramid

https://cvexplained.wordpress.com/tag/sliding-windows/
Localization: Region Proposal

- Leverage methods that can generate regions with high likelihood of containing objects
  - E.g., bottom-up segmentation methods, using edges

Selective Search, Sande et al., ICCV’11

Edge Boxes. Zitnick & Dollar, ECCV’14
Classification: Features

- Traditional methods: Hand-crafted features
- Deep learning methods: learned features in the network

Viola and Jones: rectangle features

Dadal & Triggs: Histograms of Oriented Gradients
Classification: Classifiers

• Traditional methods
  • AdaBoost
  • Support vector machines (SVMs)

Viola and Jones: AdaBoost

Felzenszwalb et al: SVM
Object detection with discriminatively trained part-based models. TPAMI, 2009.

• Deep learning methods
  • Neural networks
R-CNN

1. Input image
2. Extract region proposals (~2k)
3. Compute CNN features
4. Classify regions

Selective Search

SVM

Rich feature hierarchies for accurate object detection and semantic segmentation. Girshick et al., CVPR, 2014
Rich feature hierarchies for accurate object detection and semantic segmentation. Girshick et al., CVPR, 2014

<table>
<thead>
<tr>
<th>VOC 2007 test</th>
<th>aero</th>
<th>bike</th>
<th>bird</th>
<th>boat</th>
<th>bottle</th>
<th>bus</th>
<th>car</th>
<th>cat</th>
<th>chair</th>
<th>cow</th>
<th>table</th>
<th>dog</th>
<th>horse</th>
<th>mbike</th>
<th>person</th>
<th>plant</th>
<th>sheep</th>
<th>sofa</th>
<th>train</th>
<th>tv</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-CNN pool₅</td>
<td>51.8</td>
<td>60.2</td>
<td>36.4</td>
<td>27.8</td>
<td>23.2</td>
<td>52.8</td>
<td>60.6</td>
<td>49.2</td>
<td>18.3</td>
<td>47.8</td>
<td>44.3</td>
<td>40.8</td>
<td>56.8</td>
<td>58.7</td>
<td>42.4</td>
<td>23.4</td>
<td>46.1</td>
<td>36.7</td>
<td>51.3</td>
<td>55.7</td>
<td>44.2</td>
</tr>
<tr>
<td>R-CNN fc₆</td>
<td>59.3</td>
<td>61.8</td>
<td>43.1</td>
<td>34.0</td>
<td>25.1</td>
<td>53.1</td>
<td>60.6</td>
<td>52.8</td>
<td>21.7</td>
<td>47.8</td>
<td>42.7</td>
<td>47.8</td>
<td>52.5</td>
<td>58.5</td>
<td>44.6</td>
<td>25.6</td>
<td>48.3</td>
<td>34.0</td>
<td>53.1</td>
<td>58.0</td>
<td>46.2</td>
</tr>
<tr>
<td>R-CNN fc₇</td>
<td>57.6</td>
<td>57.9</td>
<td>38.5</td>
<td>31.8</td>
<td>23.7</td>
<td>51.2</td>
<td>58.9</td>
<td>51.4</td>
<td>20.0</td>
<td>50.5</td>
<td>40.9</td>
<td>46.0</td>
<td>51.6</td>
<td>55.9</td>
<td>43.3</td>
<td>23.3</td>
<td>48.1</td>
<td>35.3</td>
<td>51.0</td>
<td>57.4</td>
<td>44.7</td>
</tr>
<tr>
<td>R-CNN FT pool₅</td>
<td>58.2</td>
<td>63.3</td>
<td>37.9</td>
<td>27.6</td>
<td>26.1</td>
<td>54.1</td>
<td>66.9</td>
<td>51.4</td>
<td>26.7</td>
<td>55.5</td>
<td>43.4</td>
<td>43.1</td>
<td>57.7</td>
<td>59.0</td>
<td>45.8</td>
<td>28.1</td>
<td>50.8</td>
<td>40.6</td>
<td>53.1</td>
<td>56.4</td>
<td>47.3</td>
</tr>
<tr>
<td>R-CNN FT fc₆</td>
<td>63.5</td>
<td>66.0</td>
<td>47.9</td>
<td>37.7</td>
<td>29.9</td>
<td>62.5</td>
<td>70.2</td>
<td>60.2</td>
<td>32.0</td>
<td>57.9</td>
<td>47.0</td>
<td>53.5</td>
<td>60.1</td>
<td>64.2</td>
<td>52.2</td>
<td>31.3</td>
<td>55.0</td>
<td>50.0</td>
<td>57.7</td>
<td>63.0</td>
<td>53.1</td>
</tr>
<tr>
<td>R-CNN FT fc₇</td>
<td>64.2</td>
<td>69.7</td>
<td>50.0</td>
<td>41.9</td>
<td>32.0</td>
<td>62.6</td>
<td>71.0</td>
<td>60.7</td>
<td>32.7</td>
<td>58.5</td>
<td>46.5</td>
<td>56.1</td>
<td>60.6</td>
<td>66.8</td>
<td>54.2</td>
<td>31.5</td>
<td>52.8</td>
<td>48.9</td>
<td>57.9</td>
<td>64.7</td>
<td>54.2</td>
</tr>
<tr>
<td><strong>R-CNN FT fc₇ BB</strong></td>
<td><strong>68.1</strong></td>
<td><strong>72.8</strong></td>
<td><strong>56.8</strong></td>
<td><strong>43.0</strong></td>
<td><strong>36.8</strong></td>
<td><strong>66.3</strong></td>
<td><strong>74.2</strong></td>
<td><strong>67.6</strong></td>
<td><strong>34.4</strong></td>
<td><strong>63.5</strong></td>
<td><strong>54.5</strong></td>
<td><strong>61.2</strong></td>
<td><strong>69.1</strong></td>
<td><strong>68.6</strong></td>
<td><strong>58.7</strong></td>
<td><strong>33.4</strong></td>
<td><strong>62.9</strong></td>
<td><strong>51.1</strong></td>
<td><strong>62.5</strong></td>
<td><strong>64.8</strong></td>
<td><strong>58.5</strong></td>
</tr>
<tr>
<td>DPM v5 [20]</td>
<td>33.2</td>
<td>60.3</td>
<td>10.2</td>
<td>16.1</td>
<td>27.3</td>
<td>54.3</td>
<td>58.2</td>
<td>23.0</td>
<td>20.0</td>
<td>24.1</td>
<td>26.7</td>
<td>12.7</td>
<td>58.1</td>
<td>48.2</td>
<td>43.2</td>
<td>12.0</td>
<td>21.1</td>
<td>36.1</td>
<td>46.0</td>
<td>43.5</td>
<td>33.7</td>
</tr>
<tr>
<td>DPM ST [28]</td>
<td>23.8</td>
<td>58.2</td>
<td>10.5</td>
<td>8.5</td>
<td>27.1</td>
<td>50.4</td>
<td>52.0</td>
<td>7.3</td>
<td>19.2</td>
<td>22.8</td>
<td>18.1</td>
<td>8.0</td>
<td>55.9</td>
<td>44.8</td>
<td>32.4</td>
<td>13.3</td>
<td>15.9</td>
<td>22.8</td>
<td>46.2</td>
<td>44.9</td>
<td>29.1</td>
</tr>
<tr>
<td>DPM HSC [31]</td>
<td>32.2</td>
<td>58.3</td>
<td>11.5</td>
<td>16.3</td>
<td>30.6</td>
<td>49.9</td>
<td>54.8</td>
<td>23.5</td>
<td>21.5</td>
<td>27.7</td>
<td>34.0</td>
<td>13.7</td>
<td>58.1</td>
<td>51.6</td>
<td>39.9</td>
<td>12.4</td>
<td>23.5</td>
<td>34.4</td>
<td>47.4</td>
<td>45.2</td>
<td>34.3</td>
</tr>
</tbody>
</table>

BB: bounding box regression

Features from AlexNet
Fast R-CNN

Fast R-CNN. Girshick, ICCV, 2015
**RoI Pooling**

Divide the mapping RoI into $H \times W$ grids

CNN

RoI mapping to feature map

$\mathbf{s} \times (x, y, h, w)$

$$s = \frac{1}{16}$$

Max pooling for each grid

$H \times W \times C$

$7 \times 7$

RoI pooling in Fast R-CNN

RoI

$(x, y, h, w)$
Bounding Box Regression

• Predict bounding box regression offset for K object classes

\[ t^k = (t^k_x, t^k_y, t^k_w, t^k_h) \]

\[ t_x = \frac{(G_x - P_x)}{P_w} \]
\[ t_y = \frac{(G_y - P_y)}{P_h} \]
\[ t_w = \log\left(\frac{G_w}{P_w}\right) \]
\[ t_h = \log\left(\frac{G_h}{P_h}\right). \]

\[ \hat{G}_x = P_w d_x(P) + P_x \]
\[ \hat{G}_y = P_h d_y(P) + P_y \]
\[ \hat{G}_w = P_w \exp(d_w(P)) \]
\[ \hat{G}_h = P_h \exp(d_h(P)). \]

G: ground truth, P: input RoI
Fast R-CNN

• Loss function

\[ L(p, u, t^u, v) = L_{\text{cls}}(p, u) + \lambda [u \geq 1] L_{\text{loc}}(t^u, v) \]

Softmax classification probabilities

\[ p = (p_0, \ldots, p_K) \]

Bounding box regress target

True class label

Bounding box regress prediction

\[ t^u = (t^u_x, t^u_y, t^u_w, t^u_h) \]

\[ L_{\text{loc}}(t^u, v) = \sum_{i \in \{x,y,w,h\}} \text{smooth}_{L_1}(t^u_i - v_i) \]

\[ \text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \]
## Fast R-CNN

<table>
<thead>
<tr>
<th></th>
<th>Fast R-CNN</th>
<th>R-CNN</th>
<th>SPPnet</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>S</td>
<td>M</td>
<td>L</td>
</tr>
<tr>
<td>train time (h)</td>
<td>1.2</td>
<td>2.0</td>
<td>9.5</td>
</tr>
<tr>
<td>train speedup</td>
<td>18.3×</td>
<td>14.0×</td>
<td>8.8×</td>
</tr>
<tr>
<td>test rate (s/im)</td>
<td>0.10</td>
<td>0.15</td>
<td>0.32</td>
</tr>
<tr>
<td>▶ with SVD</td>
<td>0.06</td>
<td>0.08</td>
<td>0.22</td>
</tr>
<tr>
<td>test speedup</td>
<td>98×</td>
<td>80×</td>
<td>146×</td>
</tr>
<tr>
<td>▶ with SVD</td>
<td>169×</td>
<td>150×</td>
<td>213×</td>
</tr>
<tr>
<td>VOC07 mAP</td>
<td>57.1</td>
<td>59.2</td>
<td><strong>66.9</strong></td>
</tr>
<tr>
<td>▶ with SVD</td>
<td>56.5</td>
<td>58.7</td>
<td>66.6</td>
</tr>
</tbody>
</table>

S: AlexNet, M: VGG, L: deep VGG

SVD for FCs layers

\[ W \approx U \Sigma_t V^T \]
Faster R-CNN

• A single network for object detection
  • Region proposal network
  • Classification network

Region Proposal Network

- 2k scores
- 4k coordinates
- k anchor boxes
- 256-d intermediate layer
- Sliding window
- Conv feature map

Classification

```
layer {
  name: "rpn_cls_score"
  type: "Convolution"
  bottom: "rpn/output"
  top: "rpn_cls_score"
  param { lr_mult: 1.0 }
  param { lr_mult: 2.0 }
  convolution_param {
    num_output: 18  # 2(bg/bg) * 9(anchors)
    kernel_size: 1 pad: 0 stride: 1
    weight_filler { type: "gaussian" std: 0.01 }
    bias_filler { type: "constant" value: 0 }
  }
}
```

Bounding box regression

```
layer {
  name: "rpn_bbox_pred"
  type: "Convolution"
  bottom: "rpn/output"
  top: "rpn_bbox_pred"
  param { lr_mult: 1.0 }
  param { lr_mult: 2.0 }
  convolution_param {
    num_output: 36  # 4 * 9(anchors)
    kernel_size: 1 pad: 0 stride: 1
    weight_filler { type: "gaussian" std: 0.01 }
    bias_filler { type: "constant" value: 0 }
  }
}
```

3x3 conv layer to 256-d

```
layer {
  name: "rpn_conv/3x3"
  type: "Convolution"
  bottom: "conv5"
  top: "rpn/output"
  param { lr_mult: 1.0 }
  param { lr_mult: 2.0 }
  convolution_param {
    num_output: 256
    kernel_size: 3 pad: 1 stride: 1
    weight_filler { type: "gaussian" std: 0.01 }
    bias_filler { type: "constant" value: 0 }
  }
}
```
Two stage vs One stage

• Two stage detection methods
  • Stage 1: generate region proposals
  • Stage 2: classify region proposals and refine their locations
  • E.g., R-CNN, Fast R-CNN, Faster R-CNN

• One stage detection methods
  • An end-to-end network for object detection
  • E.g., YOLO
YOLO

• Regress to bounding box locations and class probabilities

YOLO

• Regress to bounding box locations and class probabilities

- 24 convs
- 2 FCs

Fast YOLO: 9 convs, less filter

YOLO

- Training loss function

\[
\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{i,j}^{\text{obj}} \left[ (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{i,j}^{\text{obj}} \left[ (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right] \\
+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{i,j}^{\text{obj}} \left( C_i - \hat{C}_i \right)^2 \\
+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{i,j}^{\text{noobj}} \left( C_i - \hat{C}_i \right)^2 \\
+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left( p_i(c) - \hat{p}_i(c) \right)^2
\]

- \( \mathbb{1}_{i,j}^{\text{obj}} \): jth bounding box from cell i “responsible” for the prediction with the highest current IOU with the ground truth

- \( \mathbb{1}_{i}^{\text{obj}} \): Object in cell i

\( \lambda_{\text{coord}} = 5 \quad \lambda_{\text{noobj}} = .5 \)

Non-maximum Suppression

- Keep the box with the highest confidence/score
- Compute IoU between this box and other boxes
- Suppress boxes with IoU > threshold

\[
\text{IoU} = \frac{\text{Intersection}}{\text{Union}}
\]

https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c
# YOLO

<table>
<thead>
<tr>
<th>Real-Time Detectors</th>
<th>Train</th>
<th>mAP</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>100Hz DPM [31]</td>
<td>2007</td>
<td>16.0</td>
<td>100</td>
</tr>
<tr>
<td>30Hz DPM [31]</td>
<td>2007</td>
<td>26.1</td>
<td>30</td>
</tr>
<tr>
<td>Fast YOLO</td>
<td>2007+2012</td>
<td>52.7</td>
<td>155</td>
</tr>
<tr>
<td>YOLO</td>
<td>2007+2012</td>
<td>63.4</td>
<td>45</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Less Than Real-Time</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Fastest DPM [38]</td>
<td>2007</td>
<td>30.4</td>
<td>15</td>
</tr>
<tr>
<td>R-CNN Minus R [20]</td>
<td>2007</td>
<td>53.5</td>
<td>6</td>
</tr>
<tr>
<td>Fast R-CNN [14]</td>
<td>2007+2012</td>
<td>70.0</td>
<td>0.5</td>
</tr>
<tr>
<td>Faster R-CNN VGG-16[28]</td>
<td>2007+2012</td>
<td>73.2</td>
<td>7</td>
</tr>
<tr>
<td>YOLO VGG-16</td>
<td>2007+2012</td>
<td>66.4</td>
<td>21</td>
</tr>
</tbody>
</table>

YOLOv2 and YOLOv3

**YOLOv2**
- Batch normalization (normalization of the layers' inputs by re-centering and re-scaling)
- High resolution classifier 416x416
- Convolutional with anchor boxes (remove FC layers)
- Dimension clustering to decide the anchor boxes
- Bounding box regression
- Multi-scale training (change input image size)

**YOLOv3**
- Binary cross-entropy loss for the class predictions
- Prediction across scales

YOLOv3: An Incremental Improvement
DTER

- Vision transformer-based object detection

End-to-End Object Detection with Transformers. Carion et al., ECCV, 2020
DTER

• Backbone

\[ x_{\text{img}} \in \mathbb{R}^{3 \times H_0 \times W_0} \quad \Rightarrow \quad f \in \mathbb{R}^{C \times H \times W} \]

\[ C = 2048 \quad H, W = \frac{H_0}{32}, \frac{W_0}{32} \]

• Encoder
  • 1x1 conv on f
  • HxW tokens with d-dimension each

End-to-End Object Detection with Transformers. Carion et al., ECCV, 2020
Transformer: Encoder

- Positional encoding
  - Make use the order of the sequence
  - With dimension $d_{\text{model}}$ for each input

$$PE_{(\text{pos}, 2i)} = \sin\left(\text{pos}/10000^{2i/d_{\text{model}}}\right)$$
$$PE_{(\text{pos}, 2i+1)} = \cos\left(\text{pos}/10000^{2i/d_{\text{model}}}\right)$$

Attention is all you need. Vaswani et al., NeurIPS'17
DTER

• Decoder
  • Decodes N object queries in parallel
  • Object queries: learned positional encodings (treat as weights in the network)
DTER

• Prediction heads
  • 3 FC layers
  • Box: normalized \((x, y, h, w)\)
  • Class: softmax prediction with the “no object” class

End-to-End Object Detection with Transformers. Carion et al., ECCV, 2020
DTER

• Training
  • bipartite matching between predicted and ground truth objects

Predicated boxes \( \hat{y} = \{\hat{y}_i\}_{i=1}^N \)
Ground truth boxes \( y = \{y_i\}_{i=1}^N \)
padded with non-object

Hungarian algorithm

\[
\mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}) = -\mathbb{1}_{\{c_i \neq \emptyset\}} \hat{p}_{\sigma(i)}(c_i) + \mathbb{1}_{\{c_i \neq \emptyset\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)})
\]

Hungarian loss
\[
\mathcal{L}_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^N \left[ -\log \hat{p}_{\sigma(i)}(c_i) + \mathbb{1}_{\{c_i \neq \emptyset\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)}) \right]
\]

Based on optimal assignment

End-to-End Object Detection with Transformers. Carion et al., ECCV, 2020
## DTER

<table>
<thead>
<tr>
<th>Model</th>
<th>GFLOPS/FPS</th>
<th>#params</th>
<th>AP</th>
<th>AP$_{50}$</th>
<th>AP$_{75}$</th>
<th>AP$_S$</th>
<th>AP$_M$</th>
<th>AP$_L$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faster RCNN-DC5</td>
<td>320/16</td>
<td>166M</td>
<td>39.0</td>
<td>60.5</td>
<td>42.3</td>
<td>21.4</td>
<td>43.5</td>
<td>52.5</td>
</tr>
<tr>
<td>Faster RCNN-FPN</td>
<td>180/26</td>
<td>42M</td>
<td>40.2</td>
<td>61.0</td>
<td>43.8</td>
<td>24.2</td>
<td>43.5</td>
<td>52.0</td>
</tr>
<tr>
<td>Faster RCNN-R101-FPN</td>
<td>246/20</td>
<td>60M</td>
<td>42.0</td>
<td>62.5</td>
<td>45.9</td>
<td>25.2</td>
<td>45.6</td>
<td>54.6</td>
</tr>
<tr>
<td>Faster RCNN-DC5+</td>
<td>320/16</td>
<td>166M</td>
<td>41.1</td>
<td>61.4</td>
<td>44.3</td>
<td>22.9</td>
<td>45.9</td>
<td>55.0</td>
</tr>
<tr>
<td>Faster RCNN-FPN+</td>
<td>180/26</td>
<td>42M</td>
<td>42.0</td>
<td>62.1</td>
<td>45.5</td>
<td>26.6</td>
<td>45.4</td>
<td>53.4</td>
</tr>
<tr>
<td>Faster RCNN-R101-FPN+</td>
<td>246/20</td>
<td>60M</td>
<td>44.0</td>
<td>63.9</td>
<td><strong>47.8</strong></td>
<td><strong>27.2</strong></td>
<td>48.1</td>
<td>56.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>GFLOPS/FPS</th>
<th>#params</th>
<th>AP</th>
<th>AP$_{50}$</th>
<th>AP$_{75}$</th>
<th>AP$_S$</th>
<th>AP$_M$</th>
<th>AP$_L$</th>
</tr>
</thead>
<tbody>
<tr>
<td>DETR</td>
<td>86/28</td>
<td>41M</td>
<td>42.0</td>
<td>62.4</td>
<td>44.2</td>
<td>20.5</td>
<td>45.8</td>
<td>61.1</td>
</tr>
<tr>
<td>DETR-DC5</td>
<td>187/12</td>
<td>41M</td>
<td>43.3</td>
<td>63.1</td>
<td>45.9</td>
<td>22.5</td>
<td>47.3</td>
<td>61.1</td>
</tr>
<tr>
<td>DETR-R101</td>
<td>152/20</td>
<td>60M</td>
<td>43.5</td>
<td>63.8</td>
<td>46.4</td>
<td>21.9</td>
<td>48.0</td>
<td>61.8</td>
</tr>
<tr>
<td>DETR-DC5-R101</td>
<td>253/10</td>
<td>60M</td>
<td><strong>44.9</strong></td>
<td><strong>64.7</strong></td>
<td><strong>47.7</strong></td>
<td><strong>23.7</strong></td>
<td><strong>49.5</strong></td>
<td><strong>62.3</strong></td>
</tr>
</tbody>
</table>

DC5: dilated C5 stage  
FPN: Feature pyramid networks

End-to-End Object Detection with Transformers. Carion et al., ECCV, 2020
Summary

• Two-stage detectors
  • R-CNN, Fast R-CNN, Faster R-CNN
  • Region proposal + classification
  • Good performance, slow

• One-stage detectors
  • YOLO, SSD
  • End-to-end network to regress to bounding boxes
  • Fast, comparable performance to two-stage detectors

• Transformer-based detectors
  • DTER
  • Attention-based set prediction, using object queries
Object Detection on COCO test-dev

https://paperswithcode.com/sota/object-detection-on-coco
Further Reading

• Viola–Jones object detection, 2001
  https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf

• Deformable part model, 2010,


• YOLO, 2015 https://arxiv.org/abs/1506.02640

• YOLOv2, 2016 https://arxiv.org/abs/1612.08242
