YOLO 9000
by Redmon and Farhadi

Cem Bulucu
21000987
Object Detection

- Objectives
- Locate
- Classify
Classification

• Single object
• Objective is to look at the image and determine what it is
• Is this a cat or a paw? Why not both?
Localization

• Objective is to figure out where the object is, separate it from the background
• Fixed number of objects
• Typically done by scanning the whole image with different sized windows
Object detection

- Multiple objects
- Objective is to both locate and classify all of them (or give a confidence score)
- Harder task
- Performance measures
  - recall = \(
  \frac{\text{# of correctly detected objects}}{\text{# of all objects in the reference map}}\)
  - precision = \(
  \frac{\text{# of correctly detected objects}}{\text{# of all detected objects}}\)
Previous work

• R-CNN
  • Take a classification model
  • Retrain the last FC layer
  • Generate potential bounding boxes
  • Run the classifier on all of them

• Deformable Part Models
  • Similar approach but classifier is run on evenly spaced intervals
Previous work – You Only Look Once

• Aims to combine the localization and classification as one regression problem
• Real-time object detection
• No window sliding
YOLO

• Extremely Fast (Fast YOLO is up to 155 fps)
• Single convolutional network
• Still accurate (twice the performance compared to other real-time methods)
YOLO - Method

• Divide image into SxS grid.
• If an object falls into a grid cell, that cell is responsible for object detection.
• Each cell predicts B bounding boxes (taken as 2), each consisting of 5 predictions including location and shape information, and confidence score.
• Each cell predicts a confidence score: \( \Pr(\text{Object}) \times \text{IOU}^{\text{truth}}_{\text{pred}} \)
• Each cell predicts C conditional class probabilities: \( \Pr(\text{Class}_i|\text{Object}) \)
• At test time, calculate class specific confidence scores for each box that is:
  \[ \Pr(\text{Class}_i|\text{Object}) \times \Pr(\text{Object}) \times \text{IOU}^{\text{truth}}_{\text{pred}} = \Pr(\text{Class}_i) \times \text{IOU}^{\text{truth}}_{\text{pred}} \]
YOLO - Design

- Implemented as a convolutional network
- Initial convolutional layers extract features (24 layers, Fast YOLO has 9)
- Fully connected layers predict output probabilities and coordinates (2 layers)
- Final output is: $S \times S \times (B \times 5 + C)$ tensor of predictions
YOLO - Training

- Linear activation in final layer, leaky ReLU at all others
- Only penalize classification error if there is an object at the grid cell
- Only penalize bounding box error if that predictor is “responsible”
- To avoid overfitting: dropout, random scaling
YOLO - Disadvantages

• Main source of error is localization
• Strong special constraints on bounding box predictions
  • Nearby objects might go unnoticed
  • Especially small objects in groups are problematic
• Does not generalize well to new aspect ratios
• Same error for small and large boxes
### YOLO 9000 – Better, Faster, Stronger

- Based on YOLO, first develop YOLOv2 (which is examined in most of the paper) then finally YOLO 9000 (Only examined at the end of the paper)
- A number of modifications to increase performance

<table>
<thead>
<tr>
<th>Feature</th>
<th>YOLO</th>
<th>YOLOv2</th>
</tr>
</thead>
<tbody>
<tr>
<td>batch norm?</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>hi-res classifier?</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>convolutional?</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>anchor boxes?</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>new network?</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>dimension priors?</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>location prediction?</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>passthrough?</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>multi-scale?</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>hi-res detector?</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

| VOC2007 mAP                      | 63.4 | 65.8  | 69.5  | 69.2  | 69.6  | 74.4  | 75.4  | 76.8  | 78.6  |


Batch Normalization

- Improvements in convergence
- No need for dropout anymore
- %2 improvement on mAP
High resolution classifier

• Original YOLO works with 224x224 resolution (448x448 for detection)
• Modification:
  • Tune the classification network with 448x448 resolution for 10 epochs
  • Then tune the resulting network on detection
• Profit: +4% mAP
Convolutional with anchor boxes

- YOLO predicts coordinates of bounding boxes using fully connected layers, but estimating offsets using region proposal network (RPN) like in Faster R-CNN is simpler.
- YOLOv2 removes the fully connected layer and adopts the above approach and use anchor boxes (windows with different sizes and ratios) to predict bounding boxes.
- Adjust the resolution so that a single grid cell in the middle.
- 69.5 mAP to 69.2mAP but 81% recall to 88% recall.
Dimension Clusters

- The problem with anchors is that they are hand-picked samples
- Run k-means on training set bounding boxes to create anchors
- To prevent larger boxes yielding higher errors, use a different distance metric: $d(\text{box, centroid}) = 1 - \text{IOU(\text{box, centroid})}$
Direct location prediction

- Another issue with anchors is instability, in RPNs the anchor box can be anywhere in the image, regardless of the grid cell predicting it
- Instead of predicting offsets, predict locations relative to location of grid cell
- 5 bounding boxes for each cell, and 5 values for each bounding box
Fine-Grained Features

• YOLO uses $13 \times 13$ features to predict detections
• Add a pass through layer that bypasses and brings $26 \times 26$ features like in ResNet, concatenate high and low resolution features
• $26 \times 26 \times N \rightarrow 13 \times 13 \times (4 \times N)$ (can be concatenated with $13 \times 13$ features)
• Detector runs with the concatenated features.
• Profit: +1%
Multi-scale Training

• No fully connected layers -> resizing possible
• Change network every few iterations
• Force network to learn to predict well for different dimensional inputs
• Effect of input size:
  • At low resolution, fair detection performance but runs fast
  • At high resolution, state-of-art detection and slower but still above real time speeds
Darknet-19

- New classification model
- 19 convolutional layers and 5 maxpooling layers
- 1x1 filters compress feature representation between 3x3 convolutions
Hierarchical classification - 1

- Use a tree structure that holds concepts, with root as general as object and leaves as specific as types of dogs or models of cars.
- Tree is constructed with tracing concepts in ImageNet, to the root “object” in WordNet (which is a graph).
Hierarchical classification - 2

- Root node -> $P(\text{object})$ = value from detection confidence using YOLOv2
- Predict conditional probabilities at each node of the tree like the ones below for terrier node:

\[
\begin{align*}
Pr(\text{Norfolk terrier}|\text{terrier})
\quad Pr(\text{Yorkshire terrier}|\text{terrier})
\quad Pr(\text{Bedlington terrier}|\text{terrier})
\end{align*}
\]

Then if a picture of a Norfolk terrier is encountered, its probability is calculated as:

\[
Pr(\text{Norfolk terrier}) = Pr(\text{Norfolk terrier}|\text{terrier})
\quad \ast Pr(\text{terrier}|\text{hunting dog})
\quad \ast \ldots \ast
\quad \ast Pr(\text{mammal}|Pr(\text{animal})
\quad \ast Pr(\text{animal}|\text{physical object})
\]
Dataset combination with WordTree

- Using the tree various datasets that contain general and specific classes can be combined as in the previous figure
  - COCO – general concepts (higher nodes)
  - ImageNet – specific concepts (lower nodes and leaves)
Joint Classification and Detection

- Using combined datasets (detection and classification) in training, YOLO 9000 is achieved
  - When a detection image is encountered, backprop as usual
  - When a classification image is encountered, backprop loss at the level of the label. Meaning do not penalize for more specific types than the label. Also to only backprop the classification loss find the bounding box that predicts highest probability for that class and calculate loss according to its tree.
YOLO 9000

• The joint dataset of COCO and ImageNet has 9418 classes in total
• Same architecture with YOLOv2, but only 3 anchor boxes are used
• Gets 16 mAP on the 156 object classes that it has never seen.
• Especially good at learning new animals
YOLOv2 and YOLO9000 Disadvantages

- YOLO9000 learns new animals well but struggles to learn other objects like humans or clothing.
- Not fast and powerful at the same time.
- Needs a lot of memory.
Conclusion

• YOLO was fast, fairly accurate
• New YOLO can be adjusted to give performance or speed according to the needs of the application
• With high resolutions it achieves the greater performance among compared methods, and it comes third in speed
• A great way to combine datasets for different tasks (classification and detection) is presented