YOLO: You Only Look Once
Unified Real-Time Object Detection

Presenter: Liyang Zhong  Quan Zou
Outline

1. Review: R-CNN

2. YOLO:  -- Detection Procedure
          -- Network Design
          -- Training Part
          -- Experiments
R-CNN: Regions with CNN features

1. Input image
2. Extract region proposals (~2k)
3. Compute CNN features
4. Classify regions
Proposal + Classification
Shortcoming:

1. Slow, impossible for real-time detection

2. Hard to optimize
WHAT’S NEW

Regression
YOLO Features:

1. Extremely fast (45 frames per second)

2. Reason Globally on the Entire Image

3. Learn Generalizable Representations
Detection Procedure

https://docs.google.com/presentation/d/1kAa7NOamBt4calBU9iHgT8a86RRHz9Yz2oh4-GTdX6M/edit#slide=id.g151008b386_0_44
We split the image into an S*S grid
We split the image into an S*S grid

7*7 grid
Each cell predicts B boxes (x, y, w, h) and confidences of each box: P(Object)
Each cell predicts \( B \) boxes \((x,y,w,h)\) and confidences of each box: \( P(\text{Object}) \)
Each cell predicts \( B \) boxes \((x, y, w, h)\) and confidences of each box: \( P(\text{Object}) \)

\[
B = 2
\]
Each cell predicts B boxes (x, y, w, h) and confidences of each box: P(Object)
Each cell predicts B boxes \((x, y, w, h)\) and confidences of each box: \(P(\text{Object})\)
Each cell predicts boxes and confidences: $P(\text{Object})$
Each cell also predicts a class probability.
Conditioned on object: $P(\text{Car} \mid \text{Object})$

Eg.
Dog = 0.8
Cat = 0
Bike = 0
Then we combine the box and class predictions.

\[ P(\text{class}|\text{Object}) \times P(\text{Object}) = P(\text{class}) \]
Finally we do threshold detections and NMS
Each cell predicts:

- For each bounding box:
  - 4 coordinates (x, y, w, h)
  - 1 confidence value
- Some number of class probabilities

S * S * (B * 5 + C) tensor
Network
pretrain

Inference

stride = 2

https://zhuanlan.zhihu.com/p/24916786?refer=xiaoleimlnote
Train
During training, match example to the right cell
During training, match example to the right cell
Adjust that cell’s class prediction

Dog = 1
Cat = 0
Bike = 0
Look at that cell’s predicted boxes
Find the best one, adjust it, increase the confidence
Find the best one, adjust it, increase the confidence
Find the best one, adjust it, increase the confidence
Decrease the confidence of the other box
Decrease the confidence of the other box
Some cells don’t have any ground truth detections!
Some cells don’t have any ground truth detections!
Decrease the confidence of boxes boxes
Decrease the confidence of these boxes
Don’t adjust the class probabilities or coordinates
Loss Function (sum-squared error)

loss function:

\[ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij} \left[ (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij} \left[ (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right] \]

\[ + \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij} \left( C_i - \hat{C}_i \right)^2 \]

\[ + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij} \left( C_i - \hat{C}_i \right)^2 \]

\[ + \sum_{i=0}^{S^2} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \]  \hspace{1cm} (3)

We use sum-squared error because it is easy to optimize, however it does not perfectly align with our goal of maximizing average precision. It weights localization error equally with classification error which may not be ideal. Also, in every image many grid cells do not contain any object. This pushes the “confidence” scores of those cells towards zero, often overpowering the gradient from cells that do contain objects. This can lead to model instability, causing training to diverge early on.

To remedy this, we increase the loss from bounding box coordinate predictions and decrease the loss from confidence predictions for boxes that don’t contain objects. We use two parameters, \( \lambda_{\text{coord}} \) and \( \lambda_{\text{noobj}} \) to accomplish this. We set \( \lambda_{\text{coord}} = 5 \) and \( \lambda_{\text{noobj}} = .5 \).

\[ \lambda_{\text{coord}} = 5, \quad \lambda_{\text{noobj}} = 0.5 \]
Loss Function (sum-squared error)

loss function:

\[ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{i,j} \left[ (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \]

\[ + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{i,j} \left[ (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right] \]

\[ + \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{i,j} \left( C_i - \hat{C}_i \right)^2 \]

\[ + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{i,j} \left( C_i - \hat{C}_i \right)^2 \]

\[ + \sum_{i=0}^{S^2} \mathbb{1}_{i} \sum_{c \in \text{classes}} (\hat{p}_i(c) - \hat{\hat{p}}_i(c))^2 \]

Sum-squared error also equally weights errors in large boxes and small boxes. Our error metric should reflect that small deviations in large boxes matter less than in small boxes. To partially address this we predict the square root of the bounding box width and height instead of the width and height directly.

https://www.slideshare.net/TaegyunJeon1/pr12-you-only-look-once-yolo-unified-realtime-object-detection?from_action=save
Loss Function (sum-squared error)

loss function:

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

The jth bbox predictor in cell i is “responsible” for that prediction

If object appears in cell i

Note that the loss function only penalizes classification error if an object is present in that grid cell (hence the conditional class probability discussed earlier). It also only penalizes bounding box coordinate error if that predictor is “responsible” for the ground truth box (i.e. has the highest IOU of any predictor in that grid cell).
Experiments

• Datasets

• PASCAL VOC 2007 & VOC 2012

20 classes:
- Person: person
- Animal: bird, cat, cow, dog, horse, sheep
- Vehicle: aeroplane, bicycle, boat, bus, car, motorbike, train
- Indoor: bottle, chair, dining table, potted plant, sofa, tv/monitor

Train/validation/test: 9,963 images containing 24,640 annotated objects.

20 classes. The train/val data has 11,530 images containing 27,450 ROI annotated objects and 6,929 segmentations.
Experiments

• Datasets

20 classes
Accurate object detection is slow!

<table>
<thead>
<tr>
<th>Method</th>
<th>Pascal 2007 mAP</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPM v5</td>
<td>33.7</td>
<td>0.07 FPS</td>
</tr>
</tbody>
</table>

Ref: https://pjreddie.com/publications/
Accurate object detection is slow!

<table>
<thead>
<tr>
<th></th>
<th>Pascal 2007 mAP</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPM v5</td>
<td>33.7</td>
<td>.07 FPS</td>
</tr>
<tr>
<td>R-CNN</td>
<td>66.0</td>
<td>.05 FPS</td>
</tr>
</tbody>
</table>

Ref: https://pjreddie.com/publications/
Accurate object detection is slow!

<table>
<thead>
<tr>
<th>Method</th>
<th>Pascal 2007 mAP</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPM v5</td>
<td>33.7</td>
<td>0.07 FPS</td>
</tr>
<tr>
<td>R-CNN</td>
<td>66.0</td>
<td>0.05 FPS</td>
</tr>
</tbody>
</table>

Ref: https://pjreddie.com/publications/

⅓ Mile, 1760 feet
Accurate object detection is slow!

<table>
<thead>
<tr>
<th></th>
<th>Pascal 2007 mAP</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPM v5</td>
<td>33.7</td>
<td>.07 FPS</td>
</tr>
<tr>
<td>R-CNN</td>
<td>66.0</td>
<td>.05 FPS</td>
</tr>
<tr>
<td>Fast R-CNN</td>
<td>70.0</td>
<td>.5 FPS</td>
</tr>
</tbody>
</table>

Ref: https://pjreddie.com/publications/
Accurate object detection is slow!

<table>
<thead>
<tr>
<th>Model</th>
<th>Pascal 2007 mAP</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPM v5</td>
<td>33.7</td>
<td>.07 FPS</td>
</tr>
<tr>
<td>R-CNN</td>
<td>66.0</td>
<td>.05 FPS</td>
</tr>
<tr>
<td>Fast R-CNN</td>
<td>70.0</td>
<td>.5 FPS</td>
</tr>
<tr>
<td>Faster R-CNN</td>
<td>73.2</td>
<td>7 FPS</td>
</tr>
</tbody>
</table>

Ref: https://pjreddie.com/publications/
Accurate object detection is slow!

<table>
<thead>
<tr>
<th>Method</th>
<th>Pascal 2007 mAP</th>
<th>Speed</th>
<th>FPS</th>
<th>ms/img</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPM v5</td>
<td>33.7</td>
<td>0.07</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>R-CNN</td>
<td>66.0</td>
<td>0.05</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Fast R-CNN</td>
<td>70.0</td>
<td>0.5</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Faster R-CNN</td>
<td>73.2</td>
<td>7</td>
<td>140</td>
<td></td>
</tr>
<tr>
<td>YOLO</td>
<td>63.4</td>
<td>45</td>
<td>22</td>
<td></td>
</tr>
</tbody>
</table>

Ref: https://pjreddie.com/publications/
Error Analysis

Figure 4: Error Analysis: Fast R-CNN vs. YOLO These charts show the percentage of localization and background errors in the top N detections for various categories (N = # objects in that category).
YOLO generalizes well to new domains (like art)

Ref: https://pjreddie.com/publications/
It outperforms methods like DPM and R-CNN when generalizing to person detection in artwork.
Strengths and Weaknesses

- **Strengths:**
  - Fast: 45fps, smaller version 155fps
  - End2end training
  - Background error is low
Strengths and Weaknesses

- Weaknesses:
  - Performance is lower than state-of-art
  - Makes more localization errors
Open Questions

- How to determine the number of cell, bounding box and the size of the box
- Why normalization x,y,w,h even all the input images have the same resolution?
Table 3: Detection frameworks on PASCAL VOC 2007. YOLOv2 is faster and more accurate than prior detection methods. It can also run at different resolutions for an easy tradeoff between speed and accuracy. Each YOLOv2 entry is actually the same trained model with the same weights, just evaluated at a different size. All timing information is on a GeForce GTX Titan X (original, not Pascal model).