Review

• Last lecture:
  • ImageNet challenge winners: going deeper
  • Representation learning
  • Fine-tuning
  • Programming tutorial

• Assignments (Canvas)
  • Lab assignment 2 due Monday

• Questions?
Today’s Topics

• Problem

• Applications

• PASCAL VOC detection challenge

• Faster R-CNN

• YOLO

• Programming Tutorial
Today’s Topics

• Problem

• Applications

• PASCAL VOC detection challenge

• Faster R-CNN

• YOLO

• Programming Tutorial
Problem Definition

- Localize with a bounding box object(s) of interest
Problem: **Semantic Object Detection**

- Localize with a bounding box every instance of an object from pre-specified categories

[Russakovsky et al; IJCV 2015]
Problem: **Salient** Object Detection

- Localize with a bounding box the salient object(s)

[A reasonably solved problem][1]

[Liu et al; CVPR 2007]
Object Detection vs Object Recognition

“How does (semantic) object detection differ from object recognition?”

• Extends object recognition of assigning labels to images by also indicating each object’s location with rectangular coordinates (which, in turn, requires differences for both model architectures and loss functions)

• Must learn an object’s appearance rather than only its image context;
  • e.g., giraffes are often photographed in savannah-like landscapes
Today’s Topics

• Problem

• Applications

• PASCAL VOC detection challenge

• Faster R-CNN

• YOLO

• Programming Tutorial
Social Media

Face detection
(e.g., Facebook)
Banking

Mobile check deposit
(e.g., Bank of America)
Transportation

License Plate Detection (e.g., AllGoVision)
Construction Safety

Pedestrian Detection
(e.g., Blaxtair)

Counting

Counting Fish (e.g., SalmonSoft)
http://www.wecountfish.com/?page_id=143

Business Traffic Analytics
Can you think of any other potential applications?
Today’s Topics

• Problem

• Applications

• PASCAL VOC detection challenge

• Faster R-CNN

• YOLO

• Programming Tutorial
VOC Dataset

• **Goal**: locate all instances of 20 object categories with BBs
• **Dataset**: 11,530 images collected from Flickr and annotated by annotators at University of Leeds

Dataset location: [http://host.robots.ox.ac.uk/pascal/VOC/index.html](http://host.robots.ox.ac.uk/pascal/VOC/index.html)

Single Object Evaluation

Ground Truth:

Algorithm:

Evaluation Measure

Score
Single Object Evaluation: Intersection Over Union

Ground Truth:

Algorithm:

\[
\frac{|A \cap B|}{|A \cup B|}
\]

Score
Single Object Evaluation: Intersection Over Union

Ground Truth:

Algorithm:

Then, threshold: e.g., 50% or greater means correct detection!
Overall Evaluation Basics: Precision

- For each object class (e.g., cat, dog, ...), compute precision: fraction of correct detections from all detections using 0.5 IoU threshold

[Russakovsky et al; IJCV 2015]
https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173
Overall Evaluation Basics: Precision

• For each object class (e.g., cat, dog, ...), compute precision: fraction of correct detections from all detections using IoU threshold (e.g., 0.5)

[Russakovsky et al; IJCV 2015]
https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173
Overall Evaluation Score: mAP

• For each object class (e.g., cat, dog, ...), compute Average Precision (AP)
  • Vary IoU threshold in order to create a precision-recall curve, and then compute
    compute area under the curve

• Then, compute mean AP across all classes

[Russakovsky et al; IJCV 2015]
https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173
Today’s Topics

• Problem

• Applications

• PASCAL VOC detection challenge

• Faster R-CNN

• YOLO

• Programming Tutorial
Historical Context: R-CNN Methods

- 1847: Gradient descent
- 1945: First programmable machine
- 1950: Turing test
- 1956: AI
- 1957: Perceptron
- 1959: Machine learning
- 1960s: Neural networks with effective learning strategy
- 1980: Neocognitron
- 1986: Backpropagation for CNNs
- 1989: MNIST, LeNet
- 1998: ImageNet
- 2009: AlexNet
- 2014: R-CNN, Fast R-CNN, Faster R-CNN
Why Faster R-CNN?

Named after the proposed technique: Region proposals with CNN features


Idea: test a “manageable” number of image regions with diverse properties (e.g., scales, aspect ratios) if the target object type is located there very fast
The single model performs two tasks:

1. proposes image regions and then
2. classifies category per region

Same image representation shared for both subnetworks:
- 2 architectures tested: VGG16 and ZF model
- input to subnetworks is last convolutional layer

Architecture: Region Proposal Network

**Input:** convolutional feature map from pretrained model

**Step 1:** 3 x 3 convolutional filter applied to identify candidate proposals (recall, filter in the middle of an architecture maps to a larger input space, aka receptive field)

https://www.deeplearningbook.org/contents/convnets.html
Step 2: candidate regions of multiple scales and aspect ratios are supported efficiently using $k = 9$ anchor boxes parameterized relative to the feature map region of interest per convolutional operation (3 scales and 3 aspect ratios)

Idea: images, feature maps, and filters all are a single size by using a pyramid of anchors (more efficient!)

Prior work’s approach to support multiple scales

Architecture: Region Proposal Network

**Step 2:** candidate regions of multiple scales and aspect ratios are supported efficiently using $k = 9$ anchor boxes parameterized relative to the feature map region of interest per convolutional operation (3 scales and 3 aspect ratios).

Anchor boxes: each is a region proposal specializing in a particular location, shape, and size (centered on each pixel).
Architecture: Region Proposal Network

For each region, predict:

1. Probability of object/not object
2. Parameters to regress anchor box to GT box (center, width, and height)

(k independent regressors learned to support k anchor box dimensions)
Architecture: Region Proposal Refinement

Original region proposal with center \((p_x, p_y)\), width \((p_w)\), and height \((p_h)\) is refined using learned model parameters \((d_x, d_y, d_w, d_h)\).

Training: Region Proposal Multi-task Loss

- **Multi-task loss**: for each region proposal, sum classification and localization losses

- **GT positive**: anchors with IoU > 0.7 with GT (can be multiple anchors) or, when none, highest scoring one

- **GT negative**: non-positive anchors with IoU < 0.3 with GT

- Any non-assigned anchors ignored

Training: Region Proposal Multi-task Loss

Objective function sums classification and localization losses for each region proposal.

- Softmax scores
- Box coordinates \((x, y, w, h)\)
- True location \((x', y', w', h')\)
- True label

\[
\text{Total loss} = \text{Softmax loss} + \text{L1 loss}
\]
Training: Region Proposal Multi-task Loss

Objective function sums classification and localization losses for each region proposal.

- **Softmax scores**
- **Box coordinates** \((x, y, w, h)\)
- **True location** \((x', y', w', h')\)
- **Softmax loss**
- **L1 loss**
- **Total loss**

How many values are possible for the true labels?
Training: Region Proposal Multi-task Loss

Objective function sums classification and localization losses for each region proposal.
Training: Region Proposal Multi-task Loss

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

Less sensitive to outliers than SSE

True location for true class “u”

Predicted location for class u

Training: Region Proposal Network

- **Multi-task loss**: for each region proposal, sum classification and localization losses
- **GT positive**: anchors with IoU > 0.7 with GT (can be multiple anchors) or, when none, highest scoring one
- **GT negative**: non-positive anchors with IoU < 0.3 with GT
- **Any non-assigned anchors ignored**

What is relevance of the **regression loss** when no object is present (i.e., GT negative)?
- none; regression loss disabled in such cases

The single model performs two tasks:

1. proposes image regions and then
2. classifies category per region, where the classifier’s architecture matches that of the region proposal networks except it predicts an object’s category with its coordinates

Same image representation shared for both subnetworks:
- 2 architectures tested: VGG16 and ZF model
- input to subnetworks is last convolutional layer
Training: Region Classifier Multi-task Loss

Objective function sums classification and localization losses for each region proposal.
Training: Overall

1. Train RPN
2. Train Fast R-CNN using proposals from pretrained RPN
3. Fine-tune layers unique to RPN
4. Fine-tune the fully connected layers of Fast R-CNN

Limitations

- Still relatively slow; i.e., does not support real-time performance
Today’s Topics

• Problem

• Applications

• PASCAL VOC detection challenge

• Faster R-CNN

• YOLO

• Programming Tutorial
Why YOLO?

Named after the proposed technique: You Only Look Once


A CNN architecture that detects objects by looking at the entire image once, treating the problem as a regression problem.
Approach

1. Divide image into grid

2. For each grid cell, (1) locate (potentially multiple) objects and (2) predict a probability distribution for class labels (assuming an object is present)

3. For each grid cell, (1) locate (potentially multiple) objects
Approach: BB Prediction Per Grid Cell

1. What should $p_c$ equal if no object is present?
   - 0

2. What should $p_c$ equal if an object is present?
   - IoU between predicted and ground truth boxes

3. Although multiple BBs are predicted per grid cell, only the BB with the highest IoU to the GT is used. So why have multiple BBs per grid cell?
   - Encourage each BB predictor to specialize to different BB properties (e.g., sizes, aspect ratios, object category types)
Architecture

Input: RGB image resized to fixed input size
Output: 98 BB per image w/ class probabilities

Why is the output dimension 7 x 7 x 30?
- 7x7 grid x (2x5 BB values + 20 class values)
- 7x7 grid x 2 BBs = 98 BBs

Conv. Layer 7x7x64 + s-2
Maxpool Layer 2x2-s-2

Conv. Layer 3x3x192
Maxpool Layer 2x2-s-2

Conv. Layer 1x1x128
3x3x256
1x1x256
3x3x512
Maxpool Layer 2x2-s-2

Conv. Layers 1x1x256 \{3x3x512 \} \times 4
1x1x512
3x3x1024
Maxpool Layer 2x2-s-2

Conv. Layers 1x1x512 \{3x3x1024 \} \times 2
3x3x1024
3x3x1024

Conv. Layers 3x3x1024
Conv. Layers 3x3x1024
Conn. Layer 3x3x1024-s-2

Conn. Layer 3x3x1024-s-2

**Architecture**

Input: RGB image resized to fixed input size

Output: 98 BB per image w/ class probabilities (i.e., 7x7 grid x 2 BB per grid cell = 98 BB)

GoogleNet architecture designed for the ImageNet classification challenge slightly modified

A novelty of YOLO is it “reasons globally about the image when making predictions.” Where does YOLO get the global context?

- Deeper convolutional layers capture larger receptive fields in the image.
Training

1. Pretrain first 20 convolutional layers for ImageNet classification
2. Add a few layers that ends with a 7x7x30 layer
3. Train pretrained model for object detection on VOC

Training

- Repeat until stopping criterion met:
  1. **Forward pass**: propagate training data through model to make predictions
  2. **Error quantification**: measure dissatisfaction with a model’s predictions on training data
  3. **Backward pass**: using predicted output, calculate gradients backward to assign blame to each model parameter; account for weight sharing by using average of all connections for a parameter
  4. Update each parameter using calculated gradients

Key idea: how to quantify this?
Training: Multi-Part Loss Function

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

Penalizes imperfect positioning of object centers (for each BB)

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

Penalizes imperfect widths and heights (for each BB)

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

Pushes the BB confidence score to match the IoU between the prediction and GT

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

Punishes the network for predicting an object is present when no object is in the grid cell (i.e., pushes confidence to 0)

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

Penalty only for most confident BB predictor from j predictors when an object is present

Penalizes incorrect classifications (for each grid cell)

Penalty for classification occurs ONLY if an object is actually present in the grid cell

Number of grid cells; what does YOLO use?

Number of bounding box predictors per cell; what does YOLO use?
Training: Loss Function

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

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

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

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

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

Set coefficient of 0.5 to reduce loss from boxes that don’t contain objects to mitigate learning to push the confidence to 0 since most cells don’t contain objects.

Set coefficient of 5 to prioritize loss from BB shape predictions over misclassification predictions.

Square root used to weight a small deviation in large boxes less than for small boxes.
Today’s Topics

• Problem
• Applications
• PASCAL VOC detection challenge
• Faster R-CNN
• YOLO
• Programming Tutorial
Today’s Topics

• Problem

• Applications

• PASCAL VOC detection challenge

• Faster R-CNN

• YOLO

• Programming Tutorial
The End