CSE 152B: Computer Vision II

Manmohan Chandraker

Lecture 16: Object Detection
Virtual classrooms

• Virtual lectures on Zoom
  – Only host shares the screen
  – Keep video off and microphone muted
  – But please do speak up (remember to unmute!)

• Virtual interactions on Zoom
  – Ask and answer plenty of questions
  – “Raise hand” feature on Zoom when you wish to speak
  – Post questions on chat window
  – TA will help keep track of raised hands and chat window

• Lectures recorded and upload on Kaltura
  – Available under “Media Gallery” on Canvas
Course details

• Class webpage:

• Instructor email:
  – mkchandraker@eng.ucsd.edu

• TA: Rui Zhu
  – Emails: rzhu@eng.ucsd.edu

• Aim is to learn together, discuss and have fun!
Announcements

• Homework 3 released
  – Due date: Jun 4, at 4pm
  – Easier than HW2, but get started early!

• Final exams: Wed, Jun 10, at 7pm
  – Open notes
  – Logistics to be announced (have means to write ready)
  – Need to be logged-in on Zoom
Recap
Semantic Segmentation

Global Reasoning ↔ Locally accurate Boundaries

Figure from CityScapes Dataset
Fully-Convolutional Network

Fully-connected layer with $k$ units = Convolution layer with $k$ filters of size that covers input

Given 500 x 500 image, slide FCN with stride 32 to get 10 x 10 output.

We want a segmentation output at image resolution.

CSE 152B, SP20: Manmohan Chandraker
Combine Global and Local Information

Fuse coarse semantic information with local appearance

Similar to: skip connections.

[Long et al., FCNs for Semantic Segmentation]
Refinement: Unpooling followed by convolution

- Store 2x2 pooling index with 2 bits, as opposed to floating point feature map
- Significant memory reduction at inference time

[Badrinarayanan et al., SegNet]
Wide Receptive Fields

- Most networks have similar encoders (inspired by classification networks)
- Downsample to save memory and obtain large receptive fields
- Consider not downsampling features, but still achieve large receptive fields
  - Once downsampled, signal might be lost for small objects
  - Hard to recover by subsequent layers during training
- Dilated convolutions
  - Maintain spatial resolution along with a large receptive field
  - No pooling or subsampling
  - A module specifically for dense prediction
Dilated Convolutions

- Regular convolution:
  \[(F \ast k)(p) = \sum_{s+t=p} F(s) k(t).\]

- Dilated convolution:
  \[(F \ast_l k)(p) = \sum_{s+l\cdot t=p} F(s) k(t).\]

- Sometimes called dilated filter
- Filter is the same, convolution type changes

CSE 152B, SP20: Manmohan Chandraker
Dilated Residual Network

- Goals: Exploit the power of residual networks with advantages of dilation
  - Once training signal lost by downsampling, hard to recover it
  - Preserve spatial resolution of feature maps
  - Provide training signals that densely cover the input field
  - Backpropagation can now learn to preserve small but salient details
  - Also beneficial when classification network is transferred to other tasks

- ResNet uses stride 2 to downsample from block $G_k$ to $G_{k+1}$
- Do not use stride to downsample, maintain resolution
- Dilate subsequent layers by another factor of 2 to maintain receptive field

[Yu et al., Dilated Residual Networks]
Dilated Residual Network

- State-of-the-art performance on classification, segmentation
- High resolution features, classification network easily used for segmentation
- But two problems with DRN:
  - Memory consumption
  - Gridding artifacts
Dilated Residual Network: Memory

- State-of-the-art performance on classification, segmentation
- High resolution features, classification network easily used for segmentation
- But two problems with DRN:
  - Memory consumption
  - Gridding artifacts
- Dilation preserves number of parameters, but feature maps are bigger
- In practice, only $G_4$ and $G_5$ use dilation
- Output produced at 8-times smaller resolution

ResNet

Dilated ResNet
Dilated Residual Network: Degridding

- Gridding artifacts sometimes observed with DRN
  - Frequency in features maps exceeds sampling rate of filter
  - Filters with higher dilation are at “lower frequency”
Dilated Residual Network: Degridding

- Degridding solutions
  - Replace max-pooling in preceding layer with convolutional layers
  - Add convolution layers at end of network, with progressively lower dilation
  - Remove skip connections in newly added layers
    - Skip connections can propagate gridding artifacts from previous layers
Achieving More Context
Importance of Context: Issue with FCN

- Use of co-occurring visual statistics is crucial for semantic segmentation
  - Mismatched classes: predict car on water
  - Confusion classes: predict same object as skyscraper and building
  - Inconspicuous classes: miss the pillow since cannot correlate with bed

- It seems not enough context information is being learned
But a deep ResNet should encode sufficient context

- Theoretically, ResNet has receptive field that covers entire image
- Empirical size of receptive field can be much smaller
More Context Information

- But a deep ResNet should encode sufficient context
  - Theoretically, ResNet has receptive field that covers entire image
  - Empirical size of receptive field can be much smaller

- A data-driven way to determine empirical receptive field for a unit
  - Consider top K images that cause maximum activation for a unit
  - Slide a small occluder on each of the K images
  - A region is important if there is significant change in the activation
  - Empirical receptive field: average of discrepancy maps for the K images

---

[Zhou et al., ICLR 2015]
More Context Information

- But a deep ResNet should encode sufficient context
  - Theoretically, ResNet has receptive field that covers entire image
  - Empirical size of receptive field can be much smaller

- A data-driven way to determine empirical receptive field for a unit
  - Consider top K images that cause maximum activation for a unit
  - Slide a small occluder on each of the K images
  - A region is important if there is significant change in the activation
  - Empirical receptive field: average of discrepancy maps for the K images

![Images](a) Original Image  (b) Activation map  (c) Theoretical RF  (d) Empirical RF

CSE 152B, SP20: Manmohan Chandraker

[Zhou et al., ICLR 2015]
More Context Information

- But a deep ResNet should encode sufficient context
  - Theoretically, ResNet has receptive field that covers entire image
  - Empirical size of receptive field can be much smaller

- A data-driven way to determine empirical receptive field for a unit
  - Consider top K images that cause maximum activation for a unit
  - Slide a small occluder on each of the K images
  - A region is important if there is significant change in the activation
  - Empirical receptive field: average of discrepancy maps for the K images

<table>
<thead>
<tr>
<th></th>
<th>pool1</th>
<th>pool2</th>
<th>conv3</th>
<th>conv4</th>
<th>pool5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theoretic size</td>
<td>19</td>
<td>67</td>
<td>99</td>
<td>131</td>
<td>195</td>
</tr>
<tr>
<td>ImageNet-CNN actual size</td>
<td>17.9±1.6</td>
<td>36.7±5.4</td>
<td>51.1±9.9</td>
<td>60.4±16.0</td>
<td>70.3±21.6</td>
</tr>
</tbody>
</table>

[(Zhou et al., ICLR 2015)]
• Need a mechanism to explicitly encode context information

• Option 1: Global average pooling
More Context Information: Global Pooling

- Need a mechanism to explicitly encode context information

- Option 1: Global average pooling
  - Obtain context vector from feature maps in a layer
  - Unpool (replicate) to feature dimensions and concatenate
  - L2-normalization before concatenation for stable training
More Context Information: Global Pooling

- Need a mechanism to explicitly encode context information

- Option 1: Global average pooling
  - Obtain context vector from feature maps in a layer
  - Unpool (replicate) to feature dimensions and concatenate
  - L2-normalization before concatenation for stable training
  - Issue: spatial relationships may be lost in global pooling

[Liu et al., ParseNet]
More Context Information: Pyramid Pooling

- Need a mechanism to explicitly encode context information

- Option 1: Global average pooling
  - Obtain context vector from feature maps in a layer
  - Unpool (replicate) to feature dimensions and concatenate
  - L2-normalization before concatenation for stable training
  - Issue: spatial relationships may be lost in global pooling

- Option 2: Pyramid pooling

Originally proposed for obtaining fixed-length representation from images of arbitrary size
More Context Information: Pyramid Pooling

- Role in segmentation: Extract both global and regional context
  - A hierarchical global prior, with information across scales and regions

- Fuse features in several different pyramid scales
  - Pool input feature map into hierarchy of context features
  - Do 1x1 convolution to ensure each pyramid level gets equal weight
  - Upsample to original resolution and concatenate

[Zhao et al., PSPNet]
Evaluation of Semantic Segmentation
Dense Multiclass Prediction

CSE 152B, SP20: Manmohan Chandraker

[Images from: Jeremy Jordan]
Pixel-wise Cross-Entropy Loss

- Can weight each output channel for class imbalance in training set (FCN)
- Can assign higher weight to pixels near the boundary (U-Net)

Pixel-wise loss is calculated as the log loss, summed over all possible classes

$$- \sum_{classes} y_{true} \log(y_{pred})$$

This scoring is repeated over all pixels and averaged

CSE 152B, SP20: Manmohan Chandraker

[Images from: Jeremy Jordan]
Semantic Segmentation Metrics

Confusion matrix: \( C_{ij} = \sum_{I \in D} |\{z \in I \text{ such that } S_{gt}^I(z) = i \text{ and } S_{ps}^I(z) = j\}| \)

Number of pixels with ground truth label \( i \): \( G_i = \sum_{j=1}^{L} C_{ij} \)

Number of pixels with prediction \( j \): \( P_j = \sum_{i} C_{ij} \)

Overall pixel accuracy: \( OP = \frac{\sum_{i=1}^{L} C_{ii}}{\sum_{i=1}^{L} G_i} \)

Per-class accuracy: \( PC = \frac{1}{L} \sum_{i=1}^{L} \frac{C_{ii}}{G_i} \)

Intersection-over-union: \( JI = \frac{1}{L} \sum_{i=1}^{L} \frac{C_{ii}}{G_i + P_i - C_{ii}} \)

CSE 152B, SP20: Manmohan Chandraker
Semantic Segmentation Metrics

Confusion matrix: \[ C_{ij} = \sum_{I \in D} |\{z \in I \text{ such that } S_{gt}^I(z) = i \text{ and } S_{ps}^I(z) = j\}| \]

\[
\begin{bmatrix}
6 & 4 \\
2 & 18
\end{bmatrix}
\]

Number of pixels with ground truth label \(i\): \[ G_i = \sum_{j=1}^{L} C_{ij} \]

\[ [10, 20] \]

Number of pixels with prediction \(j\): \[ P_j = \sum_i C_{ij} \]

\[ [8, 22] \]

Overall pixel accuracy: \[ OP = \frac{\sum_{i=1}^{L} C_{ii}}{\sum_{i=1}^{L} G_i} \]

\[ \frac{6 + 18}{10 + 20} = 0.8 \]

Per-class accuracy: \[ PC = \frac{1}{L} \sum_{i=1}^{L} \frac{C_{ii}}{G_i} \]

\[ \frac{1}{2} \left( \frac{6}{10} + \frac{18}{20} \right) = 0.75 \]

Intersection-over-union: \[ JI = \frac{1}{L} \sum_{i=1}^{L} \frac{C_{ii}}{G_i + P_i - C_{ii}} \]

\[ = 0.625 \]
Semantic Segmentation Metrics

Confusion matrix: \( C_{ij} = \sum_{l \in \mathcal{D}} |\{ z \in l \text{ such that } S_{gt}^{l}(z) = i \text{ and } S_{ps}^{l}(z) = j \}| \)

Number of pixels with ground truth label \( i \): \( G_i = \sum_{j=1}^{L} C_{ij} \)

Number of pixels with prediction \( j \): \( P_j = \sum_{i} C_{ij} \)

Overall pixel accuracy: \( OP = \frac{\sum_{i=1}^{L} C_{ii}}{\sum_{i=1}^{L} G_i} \) Biased in favor of large classes

Per-class accuracy: \( PC = \frac{1}{L} \sum_{i=1}^{L} \frac{C_{ii}}{G_i} \) Biased against background classes, boost all foreground classes

Intersection-over-union: \( JI = \frac{1}{L} \sum_{i=1}^{L} \frac{C_{ii}}{G_i + P_i - C_{ii}} \) Balances false positives and false negatives

CSE 152B, SP20: Manmohan Chandraker
## FCN Semantic Segmentation Results

Relative to prior state-of-the-art SDS:

- 30% relative improvement for mean IoU
- 286 times faster

<table>
<thead>
<tr>
<th></th>
<th>mean IU VOC2011 test</th>
<th>mean IU VOC2012 test</th>
<th>inference time</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-CNN [12]</td>
<td>47.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SDS [16]</td>
<td>52.6</td>
<td>51.6</td>
<td>~ 50 s</td>
</tr>
<tr>
<td>FCN-8s</td>
<td>62.7</td>
<td>62.2</td>
<td>~ 175 ms</td>
</tr>
</tbody>
</table>

[Long et al., FCNs for Semantic Segmentation]
Object Detection
Object Detection

[Ren et al., Faster R-CNN]
Object Detection

Object 1: \((x_1, y_1, w_1, h_1)\), dog
Object 2: \((x_2, y_2, w_2, h_2)\), cat

CSE 152B, SP20: Manmohan Chandraker
Object Detection

Object 1: \((x_1, y_1, w_1, h_1)\), dog
Object 2: \((x_2, y_2, w_2, h_2)\), cat
Object 3: \((x_3, y_3, w_3, h_3)\), dog
Object Detection

Need to handle outputs of variable lengths
Object Detection: Sliding Windows

Given new image:
1. Slide window
2. Score by classifier
Object Detection: Sliding Windows

Sliding Windows

- Need to consider windows at several different positions and scales
- Either use a simple feature extractor, or evaluate only few candidates
Region Proposals

- Find blob-like regions of image that might be objects
- Do not consider object type and tolerate high rate of false positives
**Step 1:** Train (or download) a classification model for ImageNet (AlexNet)
Step 2: Fine-tune model for detection
- Instead of 1000 ImageNet classes, want 20 object classes + background
- Throw away final fully-connected layer, reinitialize from scratch
- Keep training model using positive and negative regions from detection images
**Step 3: Extract features**
- Extract region proposals for all images
- For each region: warp to CNN input size, run forward through CNN, save pool5 features to disk
- Have a big hard drive: features are \( \sim 200 \text{GB} \) for PASCAL dataset!

[Slide from: Tamara Berg]  [Girshick et al., Rich feature hierarchies]
Step 4: Train one binary SVM per class to classify region features

R-CNN Training

Training image regions

Cached region features

Positive samples for cat SVM

Negative samples for cat SVM

[Slide from: Tamara Berg] [Girshick et al., Rich feature hierarchies]
R-CNN Training

**Step 4:** Train one binary SVM per class to classify region features

![Training image regions](image1.png)

![Cached region features](image2.png)

- Negative samples for dog SVM
- Positive samples for dog SVM

[Girshick et al., Rich feature hierarchies]
Step 5 (bbox regression): For each class, train a linear regression model to map from cached features to offsets to GT boxes to make up for “slightly wrong” proposals.

![Training image regions](image1)

![Cached region features](image2)

Regression targets (dx, dy, dw, dh)

Normalized coordinates

(0, 0, 0, 0) Proposal is good

(.25, 0, 0, 0) Proposal too far to left

(0, 0, -0.125, 0) Proposal too wide

[Girshick et al., Rich feature hierarchies]
R-CNN Regression

**Step 5 (bbox regression):** For each class, train a linear regression model to map from cached features to offsets to GT boxes to make up for “slightly wrong” proposals.

Train a regressor to find offset function $d_i$ that reaches closest to regression target:

$$\mathcal{L}_{\text{reg}} = \sum_{i \in \{x, y, w, h\}} (t_i - d_i(p))^2 + \lambda \|w\|^2$$

Regression targets:

- $t_x = (g_x - p_x)/p_w$
- $t_y = (g_y - p_y)/p_h$
- $t_w = \log(g_w/p_w)$
- $t_h = \log(g_h/p_h)$
R-CNN Improvements

- Significant gain over pre-CNN methods
- Bounding box regression helps
- Deeper feature extractor leads to large gain
Issues with R-CNN

• Very expensive for inference
  • Full forward pass of CNN for each region proposal

• Classification and regression are disjoint from feature extraction
  • CNN features not updated together with detection

• Complex training pipeline

• Need mechanism to “connect” CNN features to classifier and regressor
Fast R-CNN

- Extract convolutional features just once for whole image
- Need method to share computation of convolutional layers between proposals for an image
- Allows several advantages
  - Faster inference
  - Connects CNN to classifier
  - Easier training
- Proposals still from selective search

CSE 152B, SP20: Manmohan Chandraker

[Girshick, Fast R-CNN]
Fast R-CNN: RoI Pooling

Hi-res input image: 3 x 800 x 600 with region proposal

Convolution and Pooling

Hi-res conv features: C x H x W with region proposal

Problem: Fully-connected layers expect fixed size low-res features: C x h x w

Fully-connected layers
Fast R-CNN: RoI Pooling

Hi-res input image: 3 x 800 x 600 with region proposal

Hi-res conv features: C x H x W with region proposal

Project region proposal onto conv feature map

Convolution and Pooling

Problem: Fully-connected layers expect fixed size low-res features: C x h x w
Fast R-CNN: RoI Pooling

Hi-res input image: 3 x 800 x 600 with region proposal

Hi-res conv features: C x H x W with region proposal

Divide projected region into h x w grid

Convolution and Pooling

Fully-connected layers

Problem: Fully-connected layers expect fixed size low-res features: C x h x w
Fast R-CNN: RoI Pooling

Hi-res input image: 3 x 800 x 600 with region proposal

Hi-res conv features: C x H x W with region proposal

RoI conv features: C x h x w for region proposal

Problem: Fully-connected layers expect fixed size low-res features: C x h x w

RoI Pooling

\[ y_{rj} = x_{i^*(r,j)} \]  (RoI r, output j)

\[ i^*(r,j) = \arg\max_{i \in \mathcal{R}(r,j)} x_i' \]

Pooling region
**Fast R-CNN: RoI Pooling**

Hi-res input image: 3 x 800 x 600 with region proposal

- **Convolution and Pooling**
  - Hi-res conv features: \( C \times H \times W \) with region proposal
  - RoI conv features: \( C \times h \times w \) for region proposal

**RoI Pooling**

\[
y_{rj} = x_{i^*(r,j)}
\]

\[
i^*(r,j) = \arg\max_{i \in R(r,j)} x_i
\]

Pooling region

**Problem:** Fully-connected layers expect fixed size low-res features: \( C \times h \times w \)

**Backpropagation**

\[
\frac{\partial L}{\partial x_i} = \sum_r \sum_j [i = i^*(r,j)] \frac{\partial L}{\partial y_{rj}}
\]

Gradient accumulated if \( i \) is the index chosen for max pooling
## Fast R-CNN: Improvements over R-CNN

<table>
<thead>
<tr>
<th></th>
<th>R-CNN</th>
<th>Fast R-CNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training Time:</td>
<td>84 hours</td>
<td>9.5 hours</td>
</tr>
<tr>
<td>Test time per image</td>
<td>47 seconds</td>
<td>0.32 seconds</td>
</tr>
<tr>
<td>mAP (VOC 2007)</td>
<td>66.0</td>
<td>66.9</td>
</tr>
</tbody>
</table>

Using VGG-16 CNN on Pascal VOC 2007 dataset
Faster R-CNN

- Proposals still from separate mechanism in Fast R-CNN
- Insert a **Region Proposal Network (RPN)** after last convolutional layer
- RPN trained to produce region proposals directly, no need for external region proposals!
- After RPN, use RoI Pooling and an upstream classifier and regressor just like Fast R-CNN

[Ren et al., Faster R-CNN]
Faster R-CNN: Region Proposal Network

- Slide a small window on the feature map
- Build a small network for:
  - classifying object or not-object
  - regressing bounding box locations
- Position of sliding window gives location information with respect to the image
- Box regression gives finer localization with respect to this sliding window
Faster R-CNN: Need for Multiscale Proposals

- Problem with region proposal using a convolutional network
  - Multiple scales of objects possible
- Possible solutions
  - Multiple scaled images
  - Filters of different sizes (usually together with above)
- But such approaches are time-consuming.
Faster R-CNN: Anchors

- Consider single object localization
  - Simply have a classification and regression head
- But now consider multiple objects, possibly overlapping
  - Regression for objects will interfere with each other
- Anchors: a set of reference positions on the feature map
  - Anchor box with high ground truth overlap responsible for regressing position
  - Determines reference and spatial extent for predicting object

Typically: 3 scales and 3 aspect ratios
Faster R-CNN: Anchors

• A cost-efficient way to achieve multi-scale outputs
  • Relies on image and feature maps at single scale
  • Feature computation is shared across anchor boxes at different scales

• Translation invariance
  • Use same convolutional RPN, anchor boxes for spatial localization
  • A translated object will lead to accordingly shifted proposals

Typically: 3 scales and 3 aspect ratios
Training RPN

- Place anchors uniformly across image, n boxes at every position (typically n = 9)
  - For 40 x 60 feature map, about 21k anchor boxes

- RPN output: 4n regression (x, y, w, h) and 2n classification (object, background)

- Non-maximum suppression to pick about 2k boxes
  - Remove boxes that overlap with others of higher score

- Labels for RPN classification: compute IoU of anchor boxes with ground truth
  - Assign IoU > 0.7 as object and IoU < 0.3 as background

- Bounding box regression
  - Displacement target: distance between centers of ground truth and anchor boxes
  - Size target: log ratio of anchor and ground truth dimensions

- Sample 256 anchors to form mini-batch for training

\[
L(\{(p_i\}, \{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*). \\
\]

Cross-entropy \quad Bounding box label \quad Smooth L1 \quad Regression target
Training Overall Detector

- Train CNN + RPN
  - Initialize with ImageNet pretrained weights
  - Train end-to-end for region proposal task
- Train CNN + Detector
  - Initialize with ImageNet pretrained weights
  - Use fixed proposals from above RPN
  - Train Fast R-CNN for detection task
- Fine-tune RPN
  - Use CNN from above step
  - Fine-tune RPN for proposal task
- Fine-tune detector
  - Keep CNN fixed
  - Fine-tune Fast R-CNN layers for detection task

- Subsequently, a joint training is also available with all four losses
  - RPN: classification (object or background), regression (anchor to proposal)
  - Fast R-CNN: classification (object category), regression (proposal to bounding box)
## Faster R-CNN Improvements

<table>
<thead>
<tr>
<th></th>
<th>R-CNN</th>
<th>Fast R-CNN</th>
<th>Faster R-CNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test time per image (with proposals)</td>
<td>50 seconds</td>
<td>2 seconds</td>
<td>0.2 seconds</td>
</tr>
<tr>
<td>mAP (VOC 2007)</td>
<td>66.0</td>
<td>66.9</td>
<td>66.9</td>
</tr>
</tbody>
</table>
Faster R-CNN Improvements

ImageNet Detection (mAP)

- NeoNet ensemble (2015): 53.57
- Faster R-CNN single (2015): 42.94
- GoogleNet ensemble (2014): 43.93
- NUS ensemble (2014): 37.21
- SPP ensemble (2014): 35.11
- UVA-Euvison (2013): 22.58
- Overfeat (2013): 19.4
Object Detection

Huang et al. Speed/Accuracy Tradeoffs for Modern Convolutional Object Detectors. CVPR 2017

CSE 152B, SP20: Manmohan Chandraker
Instance Segmentation

Image classification

Object detection

Semantic segmentation

Instance segmentation

[Lin et al., MS COCO]
Instance Segmentation

- Two choices:
  - Predict instance first, then classify
  - Segment first, then break into instances
  - Mask R-CNN is instance-first

- Decouple mask (FCN) and class prediction (Faster R-CNN)

[He et al., Mask R-CNN]
Mask R-CNN

- Human pose estimation with keypoints as one-hot masks
- Training (2 days on 8 GPUs) and inference (200ms) speed
- RoIAlign: quantization-free for better localization

[He et al., Mask R-CNN]
RoI-Pooling in Faster R-CNN

[Credit: deepsense.io]
RoI-Align in Mask R-CNN
Decoupled segmentation and classification

- Decoupling achieved with per-class binary masks (sigmoid)
- Alternative in FCN: multinomial masks (softmax)

\[ L = L_{cls} + L_{box} + L_{mask} \]

- Generate mask for every class without competition between classes
- Dedicated classification branch to predict class label to select output mask

<table>
<thead>
<tr>
<th></th>
<th>AP</th>
<th>AP_{50}</th>
<th>AP_{75}</th>
</tr>
</thead>
<tbody>
<tr>
<td>softmax</td>
<td>24.8</td>
<td>44.1</td>
<td>25.1</td>
</tr>
<tr>
<td>sigmoid</td>
<td>30.3</td>
<td>51.2</td>
<td>31.5</td>
</tr>
<tr>
<td></td>
<td>+5.5</td>
<td>+7.1</td>
<td>+6.4</td>
</tr>
</tbody>
</table>
Single-Shot Detector

- Two step process of Faster R-CNN can be expensive
  - Box classification and regression are done twice
  - RPN produces boxes, but they are used to pool features
  - Separate classifier then used for evaluation

- Key motivation for SSD: achieve good accuracy-speed trade-off
  - Use something similar to RPN for directly scoring anchors
  - Extract anchors at multiple scales to get accuracy with minimal overhead
Single-Shot Detector

- Standard VGG (or ResNet) base network

- Add further layers with progressively decreasing size
  - Used for predicting detections at multiple scales
  - Layers with wider receptive fields expected to detect larger objects

[Liu et al., SSD]
SSD: Default boxes (anchors)

- SSD uses default boxes, which are similar to anchors in Faster R-CNN
- Default boxes located at each cell on the feature map
  - Multiple boxes corresponding to different aspect ratios
- Default boxes across feature layers for better discretization of scale
SSD: Default boxes (anchors)

- Anchor boxes of different feature maps have different scales
  - Various feature maps responsible for detecting objects of different sizes
- Let $m$ be the number of feature maps used for prediction
- Let $s_k$ denote the scale of feature map $k$
- Define $s_{\text{min}} = 0.2$ and $s_{\text{max}} = 0.9$

$$s_k = s_{\text{min}} + \frac{s_{\text{max}} - s_{\text{min}}}{m - 1} (k - 1), \quad k \in [1, m]$$
SSD: Convolution prediction at multiple scales

- On each feature map, two types of convolutional filters applied
  - $c$ filters for class prediction, where $c$ is the number of object categories
  - 4 filters for bounding box regression, for coordinates $x, y, w, h$
- With $k$ default boxes at every cell, there are $(c+4)k$ filters for each feature map
- Output for $m \times n$ feature map is a $m \times n \times (c+4)k$ map
  - Represents category label and regression offsets for each default box
SSD: Training

- Ground truth positives: match ground truth to each default box with IoU > 0.5
- Ground truth negatives: all other default boxes
SSD: Training

- Ground truth positives: match ground truth to each default box with IoU > 0.5
- Ground truth negatives: all other default boxes
- Loss: Combination of classification confidence and localization accuracy
- Classification loss is softmax for multi-class prediction
  \[ L_{conf}(x, c) = - \sum_{i \in Pos} x_{ij}^p \log(\hat{c}_i^p) - \sum_{i \in Neg} \log(\hat{c}_i^0) \]
  where \( \hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)} \)
  \( x_{ij}^p = 1 \) if default box \( i \) matches ground truth box \( j \) of class \( p \), otherwise 0
- Smooth L1 loss for localization
  \[ L_{loc}(x, l, g) = \sum_{i \in Pos} \sum_{m \in \{cx, cy, w, h\}} x_{ij}^k \text{smooth}_{L1}(l_i^m - \hat{g}_j^m) \]
  Offsets Predicted location Ground truth for box \( j \)
SSD: Training

- Ground truth positives: match ground truth to each default box with IoU > 0.5
- Ground truth negatives: all other default boxes
- Loss: Combination of classification confidence and localization accuracy
- Classification loss is softmax for multi-class prediction
  \[ L_{\text{conf}}(x, c) = - \sum_{i \in \text{Pos}} x^p_{ij} \log(\hat{c}^p_i) - \sum_{i \in \text{Neg}} \log(\hat{c}^0_i) \text{ where } \hat{c}^p_i = \frac{\exp(c^p_i)}{\sum_p \exp(c^p_i)} \]
  \[ x^p_{ij} = 1 \text{ if default box } i \text{ matches ground truth box } j \text{ of class } p, \text{ otherwise 0} \]
- Smooth L1 loss for localization
  \[ L_{\text{loc}}(x, l, g) = \sum_{i \in \text{Pos}} \sum_{m \in \{cx, cy, w, h\}} x^k_{ij} \text{smooth}_L^1(l^m_i - \hat{g}^m_j) \]
- Number of negatives far exceeds positives
  - Hard negative mining: choose default boxes with highest classification loss
SSD: Analysis

• Helps to use multiple layers for prediction

<table>
<thead>
<tr>
<th>Prediction source layers from:</th>
<th>mAP</th>
<th>use boundary boxes?</th>
<th># Boxes</th>
</tr>
</thead>
<tbody>
<tr>
<td>conv4_3 conv7 conv8_2 conv9_2 conv10_2 conv11_2</td>
<td>74.3</td>
<td>Yes</td>
<td>8732</td>
</tr>
<tr>
<td></td>
<td>74.6</td>
<td>Yes</td>
<td>8764</td>
</tr>
<tr>
<td></td>
<td>73.8</td>
<td>Yes</td>
<td>8942</td>
</tr>
<tr>
<td></td>
<td>70.7</td>
<td>Yes</td>
<td>9864</td>
</tr>
<tr>
<td></td>
<td>64.2</td>
<td>Yes</td>
<td>9025</td>
</tr>
<tr>
<td></td>
<td>62.4</td>
<td>Yes</td>
<td>8664</td>
</tr>
</tbody>
</table>

• In general, most errors in SSD are due to poorer localization
  • Explicit pooling of region proposals in Faster R-CNN prevents this
  • Possibly also cause for more confusion between similar categories in SSD

• SSD also tends to do better for larger objects compared to smaller ones
Object detection: Accuracy and speed

Huang et al. Speed/Accuracy Tradeoffs for Modern Convolutional Object Detectors. CVPR 2017

CSE 252C, SP20: Manmohan Chandraker