Computer Vision – Lecture 12

Deep Learning III

17.06.2019

Bastian Leibe
Visual Computing Institute
RWTH Aachen University
http://www.vision.rwth-aachen.de/
leibe@vision.rwth-aachen.de
Course Outline

• Image Processing Basics
• Segmentation & Grouping
• Object Recognition & Categorization
   Sliding Window based Object Detection
• Local Features & Matching
• Deep Learning
   Convolutional Neural Networks (CNNs)
   Deep Learning Background
   CNNs for Object Detection
   CNNs for Semantic Segmentation
   CNNs for Matching
• 3D Reconstruction
Topics of This Lecture

• **CNN Architectures**
  - LeNet
  - AlexNet
  - VGGNet
  - GoogLeNet
  - ResNet

• **CNNs for Object Detection**
  - R-CNN
  - Fast R-CNN
  - Faster R-CNN
  - Mask R-CNN
  - YOLO / SSD

- Early convolutional architecture
  - 2 Convolutional layers, 2 pooling layers
  - Fully-connected NN layers for classification
  - Successfully used for handwritten digit recognition (MNIST)

ImageNet Challenge 2012

• ImageNet
  - ~14M labeled internet images
  - 20k classes
  - Human labels via Amazon Mechanical Turk

• Challenge (ILSVRC)
  - 1.2 million training images
  - 1000 classes
  - Goal: Predict ground-truth class within top-5 responses
  - Currently one of the top benchmarks in Computer Vision

[Deng et al., CVPR’09]
CNN Architectures: AlexNet (2012)

- Similar framework as LeNet, but
  - Bigger model (7 hidden layers, 650k units, 60M parameters)
  - More data ($10^6$ images instead of $10^3$)
  - GPU implementation
  - Better regularization and up-to-date tricks for training (Dropout)

ILSVRC 2012 Results

- AlexNet almost halved the error rate
  - 16.4% error (top-5) vs. 26.2% for the next best approach
  - A revolution in Computer Vision
  - Acquired by Google in Jan ‘13, deployed in Google+ in May ‘13
CNN Architectures: VGGNet (2014/15)

K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, ICLR 2015

Image source: Hirokatsu Kataoka
CNN Architectures: VGGNet (2014/15)

• Main ideas
  ➢ Deeper network
  ➢ Stacked convolutional layers with smaller filters (+ nonlinearity)
  ➢ Detailed evaluation of all components

• Results
  ➢ Improved ILSVRC top-5 error rate to 6.7%.
  ➢ 138M parameters (VGG16), most of those in the FC layers (102M)

<table>
<thead>
<tr>
<th>ConvNet Configuration</th>
<th>A</th>
<th>A-LRN</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>11 weight layers</td>
<td>11 weight layers</td>
<td>13 weight layers</td>
<td>16 weight layers</td>
<td>16 weight layers</td>
<td>19 weight layers</td>
<td></td>
</tr>
<tr>
<td>input (224 × 224 RGB image)</td>
<td>conv3-64</td>
<td>conv3-64</td>
<td>conv3-64</td>
<td>conv3-64</td>
<td>conv3-64</td>
<td>conv3-64</td>
</tr>
<tr>
<td>maxpool</td>
<td>conv3-128</td>
<td>conv3-128</td>
<td>conv3-128</td>
<td>conv3-128</td>
<td>conv3-128</td>
<td>conv3-128</td>
</tr>
<tr>
<td>maxpool</td>
<td>conv3-256</td>
<td>conv3-256</td>
<td>conv3-256</td>
<td>conv3-256</td>
<td>conv3-256</td>
<td>conv3-256</td>
</tr>
<tr>
<td>maxpool</td>
<td>conv3-512</td>
<td>conv3-512</td>
<td>conv3-512</td>
<td>conv3-512</td>
<td>conv3-512</td>
<td>conv3-512</td>
</tr>
<tr>
<td>maxpool</td>
<td>FC-4096</td>
<td>FC-4096</td>
<td>FC-1000</td>
<td>soft-max</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Comparison: AlexNet vs. VGGNet

- Receptive fields in the first layer
  - AlexNet: $11 \times 11$, stride 4
  - Zeiler & Fergus: $7 \times 7$, stride 2
  - VGGNet: $3 \times 3$, stride 1

- Why that?
  - If you stack a $3 \times 3$ layer on top of another $3 \times 3$ layer, you effectively get a $5 \times 5$ receptive field.
  - With three $3 \times 3$ layers, the receptive field is already $7 \times 7$.
  - But much fewer parameters: $3 \cdot 3^2 = 27$ instead of $7^2 = 49$.
  - In addition, non-linearities in-between $3 \times 3$ layers for additional discriminativity.
CNN Architectures: GoogLeNet (2014)

- **Main ideas**
  - “Inception” module as modular component
  - Learns filters at several scales within each module
  - 1x1 convolutions (“bottleneck layers”) for dimensionality reduction

GoogLeNet Visualization

• 22-layer network
  - No FC layers
  - Only 5M parameters
  - ILSVRC’14 winner with 6.7% top-5 error

Inception module + copies

Auxiliary classification outputs for training the lower layers (deprecated)

B. Leibe
### Results on ILSVRC

<table>
<thead>
<tr>
<th>Method</th>
<th>top-1 val. error (%)</th>
<th>top-5 val. error (%)</th>
<th>top-5 test error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG (2 nets, multi-crop &amp; dense eval.)</td>
<td>23.7</td>
<td>6.8</td>
<td>6.8</td>
</tr>
<tr>
<td>VGG (1 net, multi-crop &amp; dense eval.)</td>
<td>24.4</td>
<td>7.1</td>
<td>7.0</td>
</tr>
<tr>
<td>VGG (ILSVRC submission, 7 nets, dense eval.)</td>
<td>24.7</td>
<td>7.5</td>
<td>7.3</td>
</tr>
<tr>
<td>GoogLeNet (Szegedy et al., 2014) (1 net)</td>
<td>-</td>
<td>-</td>
<td>7.9</td>
</tr>
<tr>
<td>GoogLeNet (Szegedy et al., 2014) (7 nets)</td>
<td>-</td>
<td>-</td>
<td>6.7</td>
</tr>
<tr>
<td>MSRA (He et al., 2014) (11 nets)</td>
<td>-</td>
<td>-</td>
<td>8.1</td>
</tr>
<tr>
<td>MSRA (He et al., 2014) (1 net)</td>
<td>27.9</td>
<td>9.1</td>
<td>9.1</td>
</tr>
<tr>
<td>Clarifai (Russakovsky et al., 2014) (multiple nets)</td>
<td>-</td>
<td>-</td>
<td>11.7</td>
</tr>
<tr>
<td>Clarifai (Russakovsky et al., 2014) (1 net)</td>
<td>-</td>
<td>-</td>
<td>12.5</td>
</tr>
<tr>
<td>Zeiler &amp; Fergus (Zeiler &amp; Fergus, 2013) (6 nets)</td>
<td>36.0</td>
<td>14.7</td>
<td>14.8</td>
</tr>
<tr>
<td>Zeiler &amp; Fergus (Zeiler &amp; Fergus, 2013) (1 net)</td>
<td>37.5</td>
<td>16.0</td>
<td>16.1</td>
</tr>
<tr>
<td>OverFeat (Sermanet et al., 2014) (7 nets)</td>
<td>34.0</td>
<td>13.2</td>
<td>13.6</td>
</tr>
<tr>
<td>OverFeat (Sermanet et al., 2014) (1 net)</td>
<td>35.7</td>
<td>14.2</td>
<td>-</td>
</tr>
<tr>
<td>Krizhevsky et al. (Krizhevsky et al., 2012) (5 nets)</td>
<td>38.1</td>
<td>16.4</td>
<td>16.4</td>
</tr>
<tr>
<td>Krizhevsky et al. (Krizhevsky et al., 2012) (1 net)</td>
<td>40.7</td>
<td>18.2</td>
<td>-</td>
</tr>
</tbody>
</table>

- **VGGNet and GoogLeNet perform at similar level**
  - Comparison: human performance ~5% [Karpathy]

http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet/
Residual Networks

AlexNet, 8 layers (ILSVRC 2012)
- 11x11 conv, 96, /4, pool/2
- 5x5 conv, 256, pool/2
- 3x3 conv, 384
- 3x3 conv, 384
- 3x3 conv, 256, pool/2
- fc, 4096
- fc, 4096
- fc, 1000

VGG, 19 layers (ILSVRC 2014)
- 3x3 conv, 64
- 3x3 conv, 64, pool/2
- 3x3 conv, 128
- 3x3 conv, 128, pool/2
- 3x3 conv, 256
- 3x3 conv, 256
- 3x3 conv, 256
- 3x3 conv, 256, pool/2
- 3x3 conv, 512
- 3x3 conv, 512
- 3x3 conv, 512
- 3x3 conv, 512, pool/2
- 3x3 conv, 512
- 3x3 conv, 512
- 3x3 conv, 512
- 3x3 conv, 512, pool/2
- fc, 4096
- fc, 4096
- fc, 1000

GoogleNet, 22 layers (ILSVRC 2014)
Residual Networks

- Core component
  - Skip connections bypassing each layer
  - Better propagation of gradients to the deeper layers

\[ H(x) = F(x) + x \]
ILSRVC Winners

- 2010: Lin et al
- 2011: Sanchez & Perronnin
- 2012: Krizhevsky et al (AlexNet)
- 2013: Zeiler & Fergus
- 2014: Simonyan & Zisserman (VGG)
- 2014: Szegedy et al (GoogLeNet)
- 2015: He et al (ResNet)
- 2016: Shao et al
- 2017: Hu et al (SENet)
- Human

Layers:
- 2010: Shallow
- 2011: 8 layers
- 2012: 8 layers
- 2013: 19 layers
- 2014: 22 layers
- 2015: 152 layers
- 2016: 152 layers
- 2017: 152 layers

Slide credit: FeiFei Li
PASCAL VOC Object Detection Performance

Engines of visual recognition

HOG, DPM
- Shallow 34 layers

AlexNet (RCNN)
- 8 layers 58 mAP

VGG (RCNN)
- 16 layers 66 mAP

ResNet (Faster RCNN)*
- 101 layers 86 mAP

PASCAL VOC 2007 Object Detection mAP (%)
Comparing Complexity


Figure credit: Alfredo Canziano, Adam Paszke, Eugenio Culurcello
The Learned Features are Generic

- **Experiment: feature transfer**
  - Train AlexNet-like network on ImageNet
  - Chop off last layer and train classification layer on CalTech256
  - State of the art accuracy already with only 6 training images!
Transfer Learning with CNNs

1. Train on ImageNet
2. If small dataset: fix all weights (treat CNN as fixed feature extractor), retrain only the classifier

I.e., swap the Softmax layer at the end
Transfer Learning with CNNs

1. Train on ImageNet

3. If you have medium sized dataset, “finetune” instead: use the old weights as initialization, train the full network or only some of the higher layers.

Retrain bigger portion of the network

Slide credit: Andrej Karpathy
Topics of This Lecture

• CNN Architectures
  ➢ LeNet
  ➢ AlexNet
  ➢ VGGNet
  ➢ GoogLeNet
  ➢ ResNet

• CNNs for Object Detection
  ➢ R-CNN
  ➢ Fast R-CNN
  ➢ Faster R-CNN
  ➢ Mask R-CNN
  ➢ YOLO / SSD
Object Detection: R-CNN

**R-CNN: Regions with CNN features**

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

- Results on PASCAL VOC Detection benchmark
  - Pre-CNN state of the art: 35.1% mAP [Uijlings et al., 2013]
  - 33.4% mAP DPM
  - R-CNN: 53.7% mAP

R-CNN Pipeline

Input image

Slide credit: Ross Girshick
R-CNN Pipeline

Regions of Interest (RoI) from a proposal method (~2k)

Input image

Slide credit: Ross Girshick
R-CNN Pipeline

Slide credit: Ross Girshick
R-CNN Pipeline

Slide credit: Ross Girshick
R-CNN Pipeline

- SVMs
- ConvNet
- SVMs
- SVMs
- ConvNet
- SVMs
- ConvNet
- SVMs
- SVMs
- ConvNet

Classify regions with SVMs
Forward each region through ConvNet
Warped image regions
Regions of Interest (RoI) from a proposal method (~2k)
Input image

Slide credit: Ross Girshick
R-CNN Pipeline

B. Leibe

Slide credit: Ross Girshick
Classification

- Linear model with class-dependent weights
  - Linear SVM
    \[ f_c(x_{fc7}) = w_c^T x_{fc7} \]
  - where
    - \( x_{fc7} \) = features from the network (fully-connected layer 7)
    - \( c \) = object class
Bounding Box Regressors

- Prediction of the 2D box
  - Necessary, since the proposal region might not fully coincide with the (annotated) object bounding box
  - Perform regression for location \((x^*, y^*)\), width \(w^*\) and height \(h^*\)
    \[
    \begin{align*}
    \frac{x^* - x}{w} &= w^T_{c,x} x_{pool5} \\
    \frac{y^* - y}{h} &= w^T_{c,y} x_{pool5} \\
    \ln \frac{w^*}{w} &= w^T_{c,w} x_{pool5} \\
    \ln \frac{h^*}{h} &= w^T_{c,h} x_{pool5}
    \end{align*}
    \]
  - Where \(x_{pool5}\) are the features from the pool5 layer of the network.
Problems with R-CNN

- Ad hoc training objectives
  - Fine tune network with softmax classifier (log loss)
  - Train post-hoc linear SVMs (hinge loss)
  - Train post-hoc bounding-box regressors (squared loss)

- Training (3 days) and testing (47s per image) is slow.
  - Many separate applications of region CNNs

- Takes a lot of disk space
  - Need to store all precomputed CNN features for training the classifiers
  - Easily 200GB of data

Slide credit: Ross Girshick
Fast R-CNN

• Forward Pass
Fast R-CNN

• Forward Pass
Fast R-CNN

- Forward Pass

Slide credit: Ross Girshick
Fast R-CNN Training

• Backward Pass
Region Proposal Networks (RPN)

• Idea
  - Remove dependence on external region proposal algorithm.
  - Instead, infer region proposals from same CNN.
  ⇒ Feature sharing
  ⇒ Object detection in a single pass becomes possible.

• Faster R-CNN = Fast R-CNN + RPN

Slide credit: Ross Girshick
Faster R-CNN

- One network, four losses
  - Joint training
Faster R-CNN (based on ResNets)

Faster R-CNN (based on ResNets)

Object Detection Performance

- Before deep convnets
- Using deep convnets

PASCAL VOC

Year:
- 2006
- 2007
- 2008
- 2009
- 2010
- 2011
- 2012
- 2013
- 2014
- 2015
- 2016

Mean Average Precision (mAP):
- 0%
- 10%
- 20%
- 30%
- 40%
- 50%
- 60%
- 70%
- 80%

Slide credit: Ross Girshick
Runtime Comparison

![R-CNN Test-Time Speed](image)

- R-CNN: 49
- SPP-Net: 4.3
- Fast R-CNN: 2.3
- Faster R-CNN: 0.2
Most Recent Version: Mask R-CNN


Slide credit: FeiFei Li
Mask R-CNN Results

- Detection + Instance segmentation

- Detection + Pose estimation

Figure credit: K. He, G. Gkioxari, P. Dollar, R. Girshick
YOLO / SSD

• Idea: Directly go from image to detection scores
• Within each grid cell
  - Start from a set of anchor boxes
  - Regress from each of the B anchor boxes to a final box
  - Predict scores for each of C classes (including background)

Slide credit: FeiFei Li
YOLO-v3 Results

Summary

• Object Detection
  ➢ Find a variable number of objects by classifying image regions
  ➢ Before CNNs: dense multiscale sliding window (HoG, DPM)

• Region proposal based detectors
  ➢ Idea: Avoid dense sliding window with region proposals
  ➢ R-CNN: Selective Search + CNN classification / regression
  ➢ Fast R-CNN: Swap order of convolutions and region extraction
  ➢ Faster R-CNN: Compute region proposals within the network
  ➢ Mask R-CNN: Detection + instance segmentation + pose estimation

• Anchor box based detectors
  ➢ Idea: Perform detection in a single step using grid of anchor boxes
  ➢ YOLO, YOLO-v2, YOLO-v3
  ➢ SSD
References and Further Reading

• **LeNet**

• **AlexNet**

• **VGGNet**

• **GoogLeNet**
References and Further Reading

• ResNet
References: Computer Vision Tasks

• Object Detection