Teaching machines to recognize objects and actions

Dr. Alexandros Iosifidis
Tampere University of Technology

Finnish Academy of Science and Letters
January 18, 2016
Topics

- Image Classification
- Object Detection and Classification
- Action Recognition
Application examples

- Video surveillance
- Security
- Robotic systems:
  - Navigation, automated industrial operations, etc.
- Daily use:
  - Smart phones, smart homes, etc.
- Future technologies:
  - Assistive living.
Classification vs. Detection

😊 Dog

Dog

Dog
Image classification pipeline

• Two processing steps:
  – Image representation
  – Classification

• Image representation:
  – Image to vector mapping

• Image classification:
  – Reduced to vector classification
Traditional pipeline

- Dense SIFT
- Engineered SIFT descriptor
  - Learned (K-means) “Codebook”
  - VQ
- Dense code words
- Engineered Spatial pyramid
- Engineered Histogram pooling
- Learned (K-means) "Codebook"

Learned (K-means) "Codebook"

- [Luong & Malik, 1999]
- [Varma & Zisserman, 2003]
- [Csurka et al, 2004]
- [Vogel & Schiele, 2004]
- [Jurie & Triggs, 2005]
- [Lazebnik et al, 2006]
- [Bosch et al, 2006]
Current SoA approach

• Use of Neural Networks for combined:
  – Image representation learning,
  – Classification.

• Representation learning based on CNN layers:
  – Image convolution using 2D filters,
  – Local region pooling.

• Classification based on fully connected layers.
Practical ConvNets – mid 1990s

Gradient-Based Learning Applied to Document Recognition
Lecun et al., 1998
Weights as filters

Convolution

Image

Slide credit: Larry Zitnick
Pooling

Slide credit: Larry Zitnick
Practical ConvNets – mid 1990s

Gradient-Based Learning Applied to Document Recognition
Lecun et al., 1998
Fully-connected layers

Slide credit: Larry Zitnick
CNN training

Dog?
Horse?
Car?
Person?
Chair?
...

CNN
Training images representations class
1.4 2.7 1.9 0
3.8 3.4 3.2 0
6.4 2.8 1.7 1
4.1 0.1 0.2 0
etc …

Initialise with random weights

Network weights adaptation
Training data

<table>
<thead>
<tr>
<th>features</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4</td>
<td>0</td>
</tr>
<tr>
<td>3.8</td>
<td>0</td>
</tr>
<tr>
<td>6.4</td>
<td>1</td>
</tr>
<tr>
<td>4.1</td>
<td>0</td>
</tr>
<tr>
<td>etc …</td>
<td></td>
</tr>
</tbody>
</table>

Present a training pattern

Network weights adaptation
Training data

<table>
<thead>
<tr>
<th>features</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4 2.7 1.9</td>
<td>0</td>
</tr>
<tr>
<td>3.8 3.4 3.2</td>
<td>0</td>
</tr>
<tr>
<td>6.4 2.8 1.7</td>
<td>1</td>
</tr>
<tr>
<td>4.1 0.1 0.2</td>
<td>0</td>
</tr>
</tbody>
</table>

etc …

Feed it through to get output

Network weights adaptation
### Training data

<table>
<thead>
<tr>
<th>features</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4</td>
<td>0</td>
</tr>
<tr>
<td>3.8</td>
<td>0</td>
</tr>
<tr>
<td>6.4</td>
<td>1</td>
</tr>
<tr>
<td>4.1</td>
<td>0</td>
</tr>
<tr>
<td>etc …</td>
<td></td>
</tr>
</tbody>
</table>

### Compare with target output

![Diagram of neural network]

Network weights adaptation
### Training data

<table>
<thead>
<tr>
<th>features</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4</td>
<td>0</td>
</tr>
<tr>
<td>3.8</td>
<td>0</td>
</tr>
<tr>
<td>6.4</td>
<td>1</td>
</tr>
<tr>
<td>4.1</td>
<td>0</td>
</tr>
<tr>
<td>etc …</td>
<td></td>
</tr>
</tbody>
</table>

Adjust weights based on error

Network weights adaptation
**Training data**

<table>
<thead>
<tr>
<th>features</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4</td>
<td>2.7</td>
</tr>
<tr>
<td>3.8</td>
<td>3.4</td>
</tr>
<tr>
<td><strong>6.4</strong></td>
<td>2.8</td>
</tr>
<tr>
<td>4.1</td>
<td>0.1</td>
</tr>
</tbody>
</table>

eetc …

**Present a training pattern**

**Network weights adaptation**
### Training data

<table>
<thead>
<tr>
<th>features</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4 2.7 1.9</td>
<td>0</td>
</tr>
<tr>
<td>3.8 3.4 3.2</td>
<td>0</td>
</tr>
<tr>
<td><strong>6.4 2.8 1.7</strong></td>
<td><strong>1</strong></td>
</tr>
<tr>
<td>4.1 0.1 0.2</td>
<td>0</td>
</tr>
<tr>
<td>etc ...</td>
<td></td>
</tr>
</tbody>
</table>

Feed it through to get output

![Network weights adaptation](image)

Network weights adaptation
### Training data

<table>
<thead>
<tr>
<th>features</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4 2.7 1.9 0</td>
<td></td>
</tr>
<tr>
<td>3.8 3.4 3.2 0</td>
<td></td>
</tr>
<tr>
<td>6.4 2.8 1.7 1</td>
<td></td>
</tr>
<tr>
<td>4.1 0.1 0.2 0</td>
<td></td>
</tr>
<tr>
<td>etc …</td>
<td></td>
</tr>
</tbody>
</table>

### Compare with target output

```
6.4 2.8 1.7 0.9
```

### Network weights adaptation
**Training data**

<table>
<thead>
<tr>
<th>features</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4</td>
<td>2.7</td>
</tr>
<tr>
<td>3.8</td>
<td>3.4</td>
</tr>
<tr>
<td><strong>6.4</strong></td>
<td>2.8</td>
</tr>
<tr>
<td>4.1</td>
<td>0.1</td>
</tr>
</tbody>
</table>

etc …

Adjust weights based on error

Network weights adaptation
### Training data

<table>
<thead>
<tr>
<th>features</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4 2.7 1.9</td>
<td>0</td>
</tr>
<tr>
<td>3.8 3.4 3.2</td>
<td>0</td>
</tr>
<tr>
<td>6.4 2.8 1.7</td>
<td>1</td>
</tr>
<tr>
<td>4.1 0.1 0.2</td>
<td>0</td>
</tr>
</tbody>
</table>

etc …

And so on ….

Repeat this thousands, maybe millions of times – each time taking a random training image, and making slight weight adjustments.

*Algorithms for weight adjustment are designed to make changes that will reduce the training error*
Algorithms

1999 - 2012
- Classifier
- Encoding, pooling
- Filters, HOG, SIFT
- Image

2012
- Convolution
- Convolution
- Convolution
- Convolution
- Convolution
- Convolution
- Convolution
- Convolution
- Convolution
- Convolution
- Convolution

2014
- “Very deep CNNs”
  Simonyan & Zisserman
- Diminishing returns after ~16 layers

Very deep
Object detection – key ideas

Dominant paradigm: Reduce detection to image classification where each “image” is a window into the original image.

Is this a dog? bounding box or window
Key ideas – object detection

Dominant paradigm: Reduce detection to image classification where each “image” is a window into the original image.

Is this a dog?
Dominant paradigm:
Reduce detection to image classification where each “image” is a window into the original image.

Key ideas – object detection

Is this a dog?
Key ideas – object detection

Dominant paradigm: Reduce detection to image classification where each “image” is a window into the original image.

Is this a dog?
Key ideas – object detection

Which windows should be examined?

1. All windows – $O(n^2m^2)$ windows in $n \times m$ image; 23B windows in a 300x500 image

2. Sliding window – 100k to 1M windows

3. Object proposals – 1k to 5k windows
Finding Objects with object proposals

Objectives:
1. Minimize number of proposed regions
2. Maintain high recall of all objects
Challenges

• Objects are extremely diverse
  – Variety of shapes, sizes, colors, etc.
  – Many different appearances of the same object

• Within object variation
  – Multiple materials and textures
  – Strong interior boundaries

• Multiple objects in an image
Selective Search

- Selective search


Merge of multiple segmentation to propose candidate box

1536 boxes = 96.7 recall
Edge Boxes


# of contours wholly within in a box indicates the objectness

Method

– Edge detection. \((m, \theta)\)
– Group edges using connectivity and orientation. Affinity between edge groups:

\[
a(s_i, s_j) = |\cos(\theta_i - \theta_{ij}) \cos(\theta_j - \theta_{ij})|^\gamma
\]

\[
w_b(s_i) = 1 - \max_{|T|-1} \prod_j a(t_j, t_{j+1}),
\]

\[
h_b = \frac{\sum_i w_b(s_i) m_i}{2(b_w + b_h)\kappa},
\]

Rank \(wb\) on sliding window.
Region-based Convolutional Network

Input image

Extract region proposals (~2k / image)

Compute CNN features

Classify regions (linear SVM)
**R-CNN at test time: Step 1**

Input image → Extract region proposals (~2k / image)

Segmentation As Selective Search for Object Recognition

van de Sande, Uijlings, Gevers, Smeulders, ICCV 2011
R-CNN at test time: Step 2

Input image

Extract region proposals (~2k / image)

Compute CNN features
R-CNN at test time: Step 2

Input image

Extract region proposals (~2k / image)

Compute CNN features

Dilate proposal
R-CNN at test time: Step 2

a. Crop

Input image

Extract region proposals (~2k / image)

Compute CNN features
R-CNN at test time: Step 2

Input image

Extract region proposals (~2k / image)

Compute CNN features

b. Scale (anisotropic)
R-CNN at test time: Step 2

- Input image
- Extract region proposals (~2k / image)
- Compute CNN features
  - Different filters
  - Image representation

**Step 2:**

**c. Forward propagate**

Output: “fc7” features
R-CNN at test time: Step 3

Input image

Extract region proposals (~2k / image)

Compute CNN features

Classify regions

Classification

linear classifiers (SVMs)
Image Classification

wall table chair curtain railing shelves sideboard

lighthouse tower building sky dome house ship
Person Car Bicycle & Bus Detection

road sidewalk cars wall buildings ground floor runway trees people platform grandstand fence finger truck
From Objects to Actions

• Three types of methods:
  – Silhouette/skeleton-based
  – Space-Time Interest Point-based
  – CNN-based
Human Action Recognition

- Silhouette/skeleton-based:
  - Action defined as a series of successive human body poses
Human Action Recognition

- Silhouette/skeleton-based:
  - Action defined as a series of successive human body poses
  - Vectorial action representation based on human body prototypes (dynemes)

Human Action Recognition

- Silhouette/skeleton-based:
  - Exploitation of multiple observation angles for achieving better performance

Human Action Recognition

- Silhouette/skeleton-based:

Human Action Recognition

- Silhouette/skeleton-based:

Human Action Recognition

- What can we do for this kind of videos?
Human Action Recognition

- Space-Time Interest Point-based:
  - Video description based on shape/motion on video frame interest points

Human Action Recognition

• Space-Time Interest Point-based:
  – Video description based on shape/motion on the trajectories of video frame interest points

Human Action Recognition

• Space-Time Interest Point-based:
  – Video description based on shape/motion on the trajectories of video frame interest points

Human Action Recognition

- Space-Time Interest Point-based:

  Training videos

  codebook

  video descriptors (e.g. HOG)

  BoF-based video representation
Human Action Recognition

- Use of stereoscopic videos:

3D CNNs

3D CNNs

Thank You!