Interpretation

- prediction of class
- distributed representations
- feature sharing
- compositionality

high-level parts

mid-level parts

low level parts

Input image

Lee et al. “Convolutional DBN's ...” ICML 2009
Object Detectors Emerge in Deep Scene CNNs

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, Antonio Torralba

Massachusetts Institute of Technology
How Objects are Represented in CNN?

CNN uses **distributed code** to represent objects.


Estimating the Receptive Fields

Estimated receptive fields

pool1

conv3

pool5

Actual size of RF is much smaller than the theoretic size

Segmentation using the RF of Units

More semantically meaningful
Annotating the Semantics of Units

Top ranked segmented images are cropped and sent to Amazon Turk for annotation.

Task 1
Word/Short description:
lower

Task 2
Mark (by clicking on them) the images which don’t correspond to the short description you just wrote.

Task 3
Which category does your short description mostly belong to?
- Scene (kitchen, corridor, street, beach, ...)
- Region or surface (road, grass, wall, floor, sky, ...)
- Object (bed, car, building, tree, ...)
- Object part (leg, head, wheel, roof, ...)
- Texture or material (striped, rugged, wooden, plastic, ...)
- Simple elements or colors (vertical line, curved line, color blue, ...)

![Image of lighthouses with annotations]
Annotating the Semantics of Units

Pool5, unit 76; Label: ocean; Type: scene; Precision: 93%
Annotating the Semantics of Units
Annotating the Semantics of Units

Pool5, unit 77; Label:legs; Type: object part; Precision: 96%
Annotating the Semantics of Units

Pool5, unit 112; Label: pool table; Type: object; Precision: 70%
Annotating the Semantics of Units

Pool5, unit 22; Label: dinner table; Type: scene; Precision: 60%
Distribution of Semantic Types at Each Layer

Simple elements & colors

Object part

Object

Scene

percent units (perf>75%)
Object detectors emerge within CNN trained to classify scenes, without any object supervision!
ConvNets perform classification

< 1 millisecond

1000-dim vector

“tabby cat”

end-to-end learning

[Slides from Long, Shelhamer, and Darrell]
CONV NETS: EXAMPLES

- Object detection

Szegedy et al. “DNN for object detection” NIPS 2013
At test time, run only the forward mode (FPROP).
ConvNets: Test

At test time, run only is forward mode (FPROP).

Naturally, convnet can process larger images.

Traditional methods use inefficient sliding windows.
At test time, run only is forward mode (FPROP).

Naturally, convnet can process larger images

Traditional methods use inefficient sliding windows.
ConvNets: Test

At test time, run only is forward mode (FPROP).

Naturally, convnet can process larger images

Traditional methods use inefficient sliding windows.
ConvNets: Test

At test time, run only is forward mode (FPROP).

Naturally, convnet can process larger images

Traditional methods use inefficient sliding windows.
R-CNN does detection
R-CNN: Region-based CNN

Figure: Girshick et al.
Fast R-CNN

RoI = Region of Interest

Figure: Girshick et al.
Fast R-CNN

- Convolve whole image into feature map (many layers; abstracted)
- For each candidate RoI:
  - Squash feature map weights into fixed-size ‘RoI pool’ – adaptive subsampling!
    - Divide RoI into H x W subwindows, e.g., 7 x 7, and max pool
  - Learn classification on RoI pool with own fully connected layers (FCs)
  - Output classification (softmax) + bounds (regressor)

Figure: Girshick et al.
What if we want pixels out?

monocular depth estimation Eigen & Fergus 2015

boundary prediction Xie & Tu 2015

optical flow Fischer et al. 2015

convolutional network

Semantic segmentation
~1/10 second

end-to-end learning

[Long et al.]
Fully Convolutional Networks for Semantic Segmentation

Jonathan Long*    Evan Shelhamer*    Trevor Darrell
UC Berkeley

CVPR 2015
A classification network...

convolution

Number of filters, e.g., 64

227 × 227  55 × 55  27 × 27  13 × 13

fully connected

Number of perceptrons in MLP layer, e.g., 1024

“tabby cat”
A classification network...

[Diagram showing convolution and fully connected layers with dimensions: 227 x 227, 55 x 55, 27 x 27, 13 x 13]

“tabby cat”

[Long et al.]
A classification network...

The response of every kernel across all positions are attached densely to the array of perceptrons in the fully-connected layer.

[Long et al.]
A classification network...

The response of every kernel across all positions are attached densely to the array of perceptrons in the fully-connected layer.

AlexNet: 256 filters over 6x6 response map
Each 2,359,296 response is attached to one of 4096 perceptrons, leading to 37 mil params.
Problem

• We want a label at every pixel
• Current network gives us a label for the whole image.
• We want a matrix of labels

• Approach:
  • Make CNN for sub-image size
  • ‘Convolutionalize’ all layers of network, so that we can treat it as one (complex) filter and slide around our full image.
Figure 2. Transforming fully connected layers into convolution layers enables a classification net to output a heatmap. Adding layers and a spatial loss (as in Figure 1) produces an efficient machine for end-to-end dense learning.
A classification network...

The response of every kernel across all positions are attached densely to the array of perceptrons in the fully-connected layer.

AlexNet: 256 filters over 6x6 response map
Each 2,359,296 response is attached to one of 4096 perceptrons, leading to 37 mil params.
In Convolutional Nets, there is no such thing as "fully-connected layers". There are only convolution layers with 1x1 convolution kernels and a full connection table.
Convolutionalization

1x1 convolution operates across all filters in the previous layer, and is slid across all positions.

[Long et al.]
Back to the fully-connected perceptron...

\[ \text{output} = \begin{cases} 0 & \text{if } w \cdot x \leq 0 \\ 1 & \text{if } w \cdot x > 0 \end{cases} \]

\[ w \cdot x \equiv \sum_j w_j x_j \]

Perceptron is connected to every value in the previous layer (across all channels; 1 visible).

[Long et al.]
Convolutional Layer

Learn multiple filters.

E.g.: 200x200 image
    100 Filters
    Filter size: 10x10
    10K parameters
Convolutional Layer

E.g.: 200x200 image
100 Filters
Filter size: 1x1
100 parameters
Convolutionalization

1x1 convolution operates across all filters in the previous layer, and is slid across all positions.

e.g., 64x1x1 kernel, with shared weights over 13x13 output, x1024 filters = 11mil params.
Becoming fully convolutional

When we turn these operations into a convolution, the 13x13 just becomes another parameter and our output size adjust dynamically.

Now we have a *vector/matrix* output, and our network acts itself like a complex filter.

[Long et al.]
Figure 2. Transforming fully connected layers into convolution layers enables a classification net to output a heatmap. Adding layers and a spatial loss (as in Figure 1) produces an efficient machine for end-to-end dense learning.
Upsampling the output

Some upsampling algorithm to return us to $H \times W$
End-to-end, pixels-to-pixels network

[Long et al.]
End-to-end, pixels-to-pixels network

conv, pool, nonlinearity

upsampling

pixelwise output + loss

[Long et al.]
What is the upsampling layer?

Hint: it’s actually an upsampling _network._
Upsampling with convolution

Convolution

Transposed convolution =
weighted kernel ‘stamp’

Often called “deconvolution”,
but not actually the deconvolution
that we previously saw in deblurring ->
that is division in the Fourier domain.
Spectrum of deep features

Combine *where* (local, shallow) with *what* (global, deep)

Fuse features into **deep jet**

(cf. Hariharan et al. CVPR15 “hypercolumn”)
Learning upsampling kernels with skip layer refinement

End-to-end, joint learning of semantics and location

(interp + sum)

(interp + sum)

(dense output)
Skip layer refinement

input image  stride 32  stride 16  stride 8  ground truth

no skips  1 skip  2 skips

[Long et al.]
Results

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

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

*Simultaneous Detection and Segmentation
Hariharan et al. ECCV14
What can we do with an FCN?
How much can an image tell about its geographic location?

6 million geo-tagged Flickr images

http://graphics.cs.cmu.edu/projects/im2gps/

im2gps (Hays & Efros, CVPR 2008)
Nearest Neighbors according to gist + bag of SIFT + color histogram + a few others
PlaNet - Photo Geolocation with Convolutional Neural Networks

Tobias Weyand, Ilya Kostrikov, James Philbin

ECCV 2016
Discretization of Globe

Figure 2. Left: Adaptive partitioning of the world into 26,263 S2 cells. Right: Detail views of Great Britain and Ireland and the San...
Network and Training

• Network Architecture: Inception with 97M parameters
• 26,263 “categories” – places in the world

• 126 Million Web photos
• 2.5 months of training on 200 CPU cores
PlaNet vs im2gps (2008, 2009)

<table>
<thead>
<tr>
<th>Method</th>
<th>Street 1 km</th>
<th>City 25 km</th>
<th>Region 200 km</th>
<th>Country 750 km</th>
<th>Continent 2500 km</th>
</tr>
</thead>
<tbody>
<tr>
<td>Im2GPS (orig) [17]</td>
<td>12.0%</td>
<td>15.0%</td>
<td>23.0%</td>
<td>47.0%</td>
<td></td>
</tr>
<tr>
<td>Im2GPS (new) [18]</td>
<td>2.5%</td>
<td>21.9%</td>
<td>32.1%</td>
<td>35.4%</td>
<td>51.9%</td>
</tr>
<tr>
<td>PlaNet</td>
<td>8.4%</td>
<td>24.5%</td>
<td>37.6%</td>
<td>53.6%</td>
<td>71.3%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Manmade Landmark</th>
<th>Natural Landmark</th>
<th>City Scene</th>
<th>Natural Scene</th>
<th>Animal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Im2GPS (new)</td>
<td>61.1</td>
<td>37.4</td>
<td>3375.3</td>
<td>5701.3</td>
<td>6528.0</td>
</tr>
<tr>
<td>PlaNet</td>
<td>74.5</td>
<td>61.0</td>
<td>212.6</td>
<td>1803.3</td>
<td>1400.0</td>
</tr>
</tbody>
</table>
Spatial support for decision
PlaNet vs Humans
PlaNet vs. Humans

The graph shows the geolocation error in kilometers for different levels of GeoGuessr questions: Street, City, Region, Country, Continent, and Earth diameter. The blue line represents PlaNet, and the green line represents Humans. The y-axis is on a log scale, and the x-axis represents the percentage of GeoGuessr questions.
PlaNet summary

• Very fast geolocalization method by categorization.
• Uses far more training data than previous work (im2gps)
• Better than humans!