Convolutional Neural Networks

IE643 - Lectures 13 & 14

September 23 & 27, 2022.

1. Recap - MLP
2. Digital Images
3. Point Processing
4. Filtering
5. Convolutional Neural Networks

Multi Layer Perceptron for Prediction Tasks

Multi Layer Perceptron for Prediction Tasks: Approximation property

Multi Layer Perceptron for Prediction Tasks

Multi Layer Perceptron: Encoder-Decoder Architecture

Digital Representation of Images

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

https://openframeworks.cc/ofBook/chapters/image_processing_computer_vision.html

Digital Representation of Images

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

https://openframeworks.cc/ofBook/chapters/image_processing_computer_vision.html

- $\Omega \subset \mathbb{N} \times \mathbb{N}$.
- Function form of an image given by $I : \Omega \to \mathbb{R}$.

Digital Representation of Images

RGB Channels

http://www.programmiamo.altervista.org/File/images/img9.html

Digital Representation of Images

http://www.programmiamo.altervista.org/File/images/img9.html

- $\Omega \subset \mathbb{N} \times \mathbb{N}$.
- Function form of a RGB image given by $I : \Omega \to \mathbb{R}^3$.

Basic Image Processing - Examples

Contrast Adjustment

https://in.mathworks.com/help/images/contrast-adjustment.html

Basic Image Processing - Examples

Noise Removal

http://what-when-how.com/embedded-image-processing-on-the-tms320c6000-dsp/non-linear-filtering-of-images-image-processing-part-1/

Basic Image Processing - Examples

Edge Detection

http://lipas.uwasa.fi/

Advanced Image Processing - Examples

Image Restoration

A critical survey of state-of-the-art image inpainting quality assessment metrics: Qureshi et al., JVCIR, 2017.

Advanced Image Processing - Examples

Image Segmentation

http://cvlab.postech.ac.kr/~tgx.lim/

Image Processing - Basic Form

- Most image processing applications boil down to the following form:

$$I' = f(I)$$

- I: Original image before processing
- I': Image obtained after processing

Point Processing

- f applied to each pixel of original image to obtain pixel of processed image

$$I'(x, y) = f(I(x, y))$$

http://what-when-how.com/introduction-to-video-and-image-processing/

Point Processing

- $f(I) = I + a$
- $f(I) = I - b$
- $f(I) = c \cdot I$

I ' (x,y) = I(x,y) + 1

Original Image

I ' (x,y) = 2 I(x,y)

http://www.di.univr.it/documenti/OccorrenzaIns/matdid/

Point Processing

- $I'(x, y) = P - I(x, y)$ (image inversion)

https://www.deviantart.com/selenapdx/art/Inverted-sketch-515936239

Point Processing

- $I'(x, y) = c \cdot I(x, y)^{\gamma}$ (gamma correction)

Gamma value: 0.45 | No gamma correction | Gamma value: 2.22

http://what-when-how.com/introduction-to-video-and-image-processing/

Point Processing

- $I'(x, y) = c \cdot \log(1 + I(x, y))$ (Log transformation)

https://www.tutorialspoint.com/dip/gray_level_transformations.htm

Point Processing

Clamping

- $I'(x,y) = \begin{cases} a \text{ if } I(x,y) \leq a \\ I(x,y) \text{ if } a \leq I(x,y) \leq b \\ b \text{ if } I(x,y) \geq b \end{cases}$

Binary thresholding

- $I'(x,y) = \begin{cases} 0 \text{ if } I(x,y) \leq a \\ 1 \text{ if } I(x,y) \geq a \end{cases}$

http://ict.udlap.mx/

Point Processing

- Other point processing techniques:
  - Intensity level slicing
  - Intensity stretching

Image Filter

- Opposed to point processing, filter operations depend on a neighborhood of a pixel:

$$I'(x, y) = f(\mathscr{N}(I(x, y)))$$

- $\mathscr{N}(I(x, y))$: neighborhood of pixel $I(x, y)$

Mean Filter

- Mean filter:

$$I'(x,y) = \frac{1}{9} \sum_{i \in \{-1,0,1\}} \sum_{j \in \{-1,0,1\}} I(x+i, y+j)$$

Mean Filter - Equivalent Representation

- Consider H to be of size 3×3 with $H(i,j) = \frac{1}{9}, i = 1,2,3,\ j = 1,2,3.$
- Mean filter can be equivalently characterized as:

$$I'(x,y) = \sum_{i \in \{1,2,3\}} \sum_{j \in \{1,2,3\}} I(x+i-2, y+j-2) H(i,j)$$

Mean Filter - Equivalent Representation

- Mean filter

$$I'(x,y) = \sum_{i \in \{1,2,3\}} \sum_{j \in \{1,2,3\}} I(x+i-2, y+j-2) H(i,j)$$

- The matrix H sometimes referred to as Filter or Kernel.

Mean Filter - Equivalent Representation

- Mean filter - A General Form

$$I'(x,y) = \sum_{i=1}^{2N+1} \sum_{j=1}^{2N+1} I(x+i-(N+1), y+j-(N+1))H(i,j)$$

- Appropriate size and indexing to be used for the matrix H and I.
- General form often referred to as **correlation**

Mean Filter - Example

https://www.harrisgeospatial.com/docs/MEAN_FILTER.html

Mean Filter - Dealing with image boundaries

- Ignore the boundaries, perform filtering only for the interior pixels.

Mean Filter - Dealing with image boundaries

- Zero padding: Pad the boundaries with zeros and then apply filter.

1	1	2	5	6	3	6	7	3
2	3	4	6	7	5	1	8	4
8	7	6	5	7	6	3	3	4
2	3	5	6	7	8	2	7	3
4	5	3	2	1	6	8	7	2
1	4	5	3	2	6	7	8	1
2	3	4	5	6	8	9	2	1

Original Image

0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	2	5	6	3	6	7	3	0	0
0	0	2	3	4	6	7	5	1	8	4	0	0
0	0	8	7	6	5	7	6	3	3	4	0	0
0	0	2	3	5	6	7	8	2	7	3	0	0
0	0	4	5	3	2	1	6	8	7	2	0	0
0	0	1	4	5	3	2	6	7	8	1	0	0
0	0	2	3	4	5	6	8	9	2	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0

Zero padding across boundary

Mean Filter - Dealing with image boundaries

- Pixel Repetition: Pad the boundaries with replicas of boundary pixel values.

1	1	2	5	6	3	6	7	3
2	3	4	6	7	5	1	8	4
8	7	6	5	7	6	3	3	4
2	3	5	6	7	8	2	7	3
4	5	3	2	1	6	8	7	2
1	4	5	3	2	6	7	8	1
2	3	4	5	6	8	9	2	1

Original Image

2	2	2	3	4	6	7	5	1	8	4	4	4
1	1	1	1	2	5	6	3	6	7	3	3	3
1	1	1	1	2	5	6	3	6	7	3	3	7
3	2	2	3	4	6	7	5	1	8	4	4	8
7	8	8	7	6	5	7	6	3	3	4	4	3
3	2	2	3	5	6	7	8	2	7	3	3	7
5	4	4	5	3	2	1	6	8	7	2	2	7
4	1	1	4	5	3	2	6	7	8	1	1	8
3	2	2	3	4	5	6	8	9	2	1	1	2
3	2	2	3	4	5	6	8	9	2	1	1	2
3	1	1	4	5	3	2	6	7	8	1	1	2

Repetition of pixel values

Median Filter

- Median filter:

$$I'(x,y) = \text{Median}\{I(x+i, y+j)\}, i \in \{-1,0,1\}, j \in \{-1,0,1\}$$

.

https://in.mathworks.com

Median Filter

- Median filter: Can you come up with an appropriate H?
- Median filter an example of non-linear filtering (and order statistic filtering)

Median Filter - Example

http://tracer.lcc.uma.es/problems/mfp/mfp.html

Median Filter vs Mean Filter

http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT5/node3.html

Gaussian Filter

$$\frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

Gaussian Filter

http://devdeeds.com/blur-view-using-renderscript/

Motion Blur Filter

```
1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1,
```

Motion Blur Filter

https://lodev.org/cgtutor/filtering.html

Gradient Filters

- Discrete gradients:

$$\frac{\partial f(x,y)}{\partial x} = \frac{f(x+1,y) - f(x-1,y)}{2}$$

$$\frac{\partial f(x,y)}{\partial y} = \frac{f(x,y+1) - f(x,y-1)}{2}$$

1	0	-1
2	0	-2
1	0	-1

1	2	1
0	0	0
-1	-2	-1

Sobel Filter for finding gradients

Gradient Filters

Gradient Filters

Sobel

1	0	-1
2	0	-2
1	0	-1

1	2	1
0	0	0
-1	-2	-1

Scharr

3	0	-3
10	0	-10
3	0	-3

3	10	3
0	0	0
-3	-10	-3

Prewitt

1	0	-1
1	0	-1
1	0	-1

1	1	1
0	0	0
-1	-1	-1

Roberts

0	1
-1	0

1	0
0	-1

Other Filters

- Laplacian filter (used for second order differencing)
- Edge filters - e.g. Canny
- Sharpening filters

Convolutional Neural Networks (CNNs)

At New York Postal Exchange

The Birth of Convolutional Neural Nets

Backpropagation Applied to Handwritten Zip Code Recognition. Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel. Neural Computation, vol.1(4), 1989.

CNN

What We See

08	02	22	97	38	15	00	40	00	75	04	05	07	78	52	12	50	77	91	08
49	49	99	40	17	81	18	57	60	87	17	40	98	43	69	48	04	56	62	00
81	49	31	73	55	79	14	29	93	71	40	67	53	88	30	03	49	13	36	65
52	70	95	23	04	60	11	42	69	24	68	56	01	32	56	71	37	02	36	91
22	31	16	71	51	67	63	89	41	92	36	54	22	40	40	28	66	33	13	80
24	47	32	60	99	03	45	02	44	75	33	53	78	36	84	20	35	17	12	50
32	98	81	28	64	23	67	10	26	38	40	67	59	54	70	66	18	38	64	70
67	26	20	68	02	62	12	20	95	63	94	39	63	08	40	91	66	49	94	21
24	55	58	05	66	73	99	26	97	17	78	78	96	83	14	88	34	89	63	72
21	36	23	09	75	00	76	44	20	45	35	14	00	61	33	97	34	31	33	95
78	17	53	28	22	75	31	67	15	94	03	80	04	62	16	14	09	53	56	92
16	39	05	42	96	35	31	47	55	58	88	24	00	17	54	24	36	29	85	57
86	56	00	48	35	71	89	07	05	44	44	37	44	60	21	58	51	54	17	58
19	80	81	68	05	94	47	69	28	73	92	13	86	52	17	77	04	89	55	40
04	52	08	83	97	35	99	16	07	97	57	32	16	26	26	79	33	27	98	66
88	36	68	87	57	62	20	72	03	46	33	67	46	55	12	32	63	93	53	69
04	42	16	73	38	25	39	11	24	94	72	18	08	46	29	32	40	62	76	36
20	69	36	41	72	30	23	88	34	62	99	69	82	67	59	85	74	04	36	16
20	73	35	29	78	31	90	01	74	31	49	71	48	86	81	16	23	57	05	54
01	70	54	71	83	51	54	69	16	92	33	48	61	43	52	01	89	19	67	48

What Computers See

Image Source: `https://adeshpande3.github.io/assets/Corgi3.png`

CNN- Convolution

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Image

1	0	1
0	1	0
1	0	1

Convolution Filter

CNN- Convolution

Image Source: https://ujwlkarn.files.wordpress.com/2016/07/convolution_schematic.gif

CNN- Convolution

CNN - Convolution

CNN - Convolution

Image

Convolved Feature

CNN - Convolution

Image

Convolved Feature

CNN - Convolution

Image

Convolved Feature

CNN - Convolution

Image

Convolved Feature

CNN - Convolution

Image

4	3	4
2	4	3
2	3	

Convolved Feature

CNN - Convolution

Image

4	3	4
2	4	3
2	3	4

Convolved Feature

CNN - Convolution with stride=1

p_{11}	p_{12}	p_{13}	p_{14}	p_{15}
p_{21}	p_{22}	p_{23}	p_{24}	p_{25}
p_{31}	p_{32}	p_{33}	p_{34}	p_{35}
p_{41}	p_{42}	p_{43}	p_{44}	p_{45}
p_{51}	p_{52}	p_{53}	p_{54}	p_{55}

$*$

w_{11}	w_{12}	w_{13}
w_{21}	w_{22}	w_{23}
w_{31}	w_{32}	w_{33}

$\Longrightarrow$

q_{11}	q_{21}	q_{13}
q_{12}	q_{22}	q_{23}
q_{13}	q_{32}	q_{33}

Note: $kernelwidth = 3$, $stride = 1$

$$q_{ij} = \sum_{r=i}^{(i+kernelwidth-1)} \sum_{s=j}^{(j+kernelwidth-1)} p_{rs} w_{(r-i+1)(s-j+1)}$$

Exercise: Find the formula for $stride > 1$

CNN - Significance of Convolution

- Convolution operation is robust to ??
- Helps to extract local features which might be useful for the task.
- Significance of striding?

CNN - Significance of Convolution

- Changing the filter size (or kernel width) $\Longrightarrow$??
- Can we use a number of filters? If so, what sizes?
- Recall: In LeCun et al's work, **24** filters were used !
- What features do these convolutions capture?
  Answer:

CNN - Zero Padding

0	0	0	0	0	0
0	35	19	25	6	0
0	13	22	16	53	0
0	4	3	7	10	0
0	9	8	1	3	0
0	0	0	0	0	0

Image Source: https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=146227

CNN - Zero Padding

0	0	0	0	0	0
0	35	19	25	6	0
0	13	22	16	53	0
0	4	3	7	10	0
0	9	8	1	3	0
0	0	0	0	0	0

- Generally used to take care of the corner and border pixels
- Used to make the convolution operations unbiased towards these pixels

CNN - Convolution on single channel image

$$n_{out} = \left\lfloor \frac{n_{in} + 2 * \text{padsize} - kernelwidth}{stride} \right\rfloor + 1$$

- n_{out} = number of features in output image
- n_{in} = number of features in input image

CNN - RGB Convolution

Image Source: http://cs231n.github.io/convolutional-networks/

CNN - Pooling

Image Source:
https://computersciencewiki.org/index.php/Max-pooling_/_Pooling

CNN - Pooling

- Max-pooling sometimes called sub-sampling. Why?
- Max-pooling operation preserves what?
- Effect of successive max-pooling operations with zero-padding ?

Popular CNN Architectures

CNN - LeNet Architecture

Gradient-based learning applied to document recognition.
Y. Lecun, L. Bottou, Y. Bengio, P. Haffner. Proc. of the IEEE, 1998.

CNN - LeNet Architecture

Gradient-based learning applied to document recognition.
Y. Lecun, L. Bottou, Y. Bengio, P. Haffner. Proc. of the IEEE, 1998.

- Achieved $\sim$ 92% accuracy on handwritten digit recognition

ILSVRC 2014 Challenge

www.image-net.org/challenges/LSVRC/2014/ 133% Search

IMAGENET Large Scale Visual Recognition Challenge 2014 (ILSVRC2014)

Introduction History Data Tasks FAQ Development kit Timetable Citation[new] Organizers Sponsors Contact

News

- June 2, 2015: Additional announcement regarding submission server policy is released.
- May 19, 2015: Announcement regarding submission server policy is released.
- December 17, 2014: ILSVRC 2015 is announced.
- September 2, 2014: A new paper which describes the collection of the ImageNet Large Scale Visual Recognition Challenge dataset, analyzes the results of the past five years of the challenge, and even compares current computer accuracy with human accuracy is now available. *Please cite it when reporting ILSVRC2014 results or using the dataset.*
- August 18, 2014: Check out the New York Times article about ILSVRC2014.
- August 18, 2014: Results are released.
- August 18, 2014: Test server is open.
- July 25, 2014: Submission server is now open.
- July 15, 2014: Computational resources available, courtesy of NVIDIA.
- July 3, 2014: **Please note that the August 15th deadline is firm this year and will not be extended.**
- June 25, 2014: You can now browse all annotated detection images.
- May 3, 2014: ILSVRC2014 development kit and data are available. Please register to obtain the download links.

Image Net Data

ImageNet: A Large-Scale Hierarchical Image Database

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li and Li Fei-Fei
Dept. of Computer Science, Princeton University, USA
{jiadeng, wdong, rsocher, jial, li, feifeili}@cs.princeton.edu

ILSVRC 2014 Challenge

Example ILSVRC2014 images:

Two Tasks:
1. Object Detection
2. Classification

Image Net Data

- **Detection Task:** 200 classes, $\sim$ 457K images in training set, $\sim$ 60K images in test and validation set.
- **Classification Task:** 1000 categories, validation and test data consist of 150,000 photographs collected from Flickr and other search engines

www.kaggle.com/c/imagenet-object-localization-challenge

Pascal VOC Data

- Goal of the challenge: To recognize objects from a number of visual object classes in realistic scenes (i.e. not pre-segmented objects)
- Person: person
- Animal: bird, cat, cow, dog, horse, sheep
- Vehicle: aeroplane, bicycle, boat, bus, car, motorbike, train
- Indoor: bottle, chair, dining table, potted plant, sofa, tv/monitor

Webpage: `http://host.robots.ox.ac.uk/pascal/VOC/voc2007/`
`http://host.robots.ox.ac.uk/pascal/VOC/voc2012/`

Pascal VOC Data

Two main objectives:

- Classification
- Detection

20 classes

CNN - Alex Net

ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

CNN - Alex Net Architecture

Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

CNN - Alex Net

Peculiarities:

- ReLU non-linearity used
- Training done on multiple GPUs
- Local response normalization applied after ReLU

$$b^i_{x,y} = \frac{a^i_{x,y}}{\left(k + \alpha \sum_{j=\max\{0,i-n/2\}}^{\min\{N-1,i+n/2\}} (a^j_{x,y})^2\right)^\beta}$$

- $k = 2$, $\alpha = 10^{-4}$, $n = 5$ and $\beta = 0.75$ chosen using cross-validation
- Local response normalization decreased the error rate
- Overlapping pooling was also employed (3×3 window with stride of length 2)

CNN - Alex Net

Peculiarities during Training:

- Data augmentation using
  - image translations
  - horizontal reflections
- Original image size: 256×256
- Patches of size 224×224 considered by random patching and their reflections
- Dropout considered

Peculiarities during Testing:

- M patches of size 224×224 considered for an input image
- Predictions were averaged over M patches
- All neurons' output was multiplied by 0.5 (to take dropout into account)

CNN - Alex Net Kernels After Training

Figure 3: 96 convolutional kernels of size $11 \times 11 \times 3$ learned by the first convolutional layer on the $224 \times 224 \times 3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

CNN - Alex Net Results

Figure 4: **(Left)** Eight ILSVRC-2010 test images and the five labels considered most probable by our model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5). **(Right)** Five ILSVRC-2010 test images in the first column. The remaining columns show the six training images that produce feature vectors in the last hidden layer with the smallest Euclidean distance from the feature vector for the test image.

CNN - VGGNet

VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

Karen Simonyan* **& Andrew Zisserman+**
Visual Geometry Group, Department of Engineering Science, University of Oxford
{karen,az}@robots.ox.ac.uk

CNN - A VGG Net Architecture

CNN - VGG Net

Peculiarities:

- Minimal pre-processing (*e.g.* Subtraction of mean value from RGB pixel values)
- Small receptive field of size 3×3 with stride 1 (as opposed to 7×7 with stride 2 and 11×11 with stride 4 used in other previous approaches)
- Small receptive field leads to reduction in number of weights to be learned (check this claim!)

CNN - VGG Net

Peculiarities:

- Multinomial logistic regression objective
- Mini-batch SGD with momentum
- Weight decay employed for regularization
- Initial learning rate set to 10^{-2} and then decreased when validation error stalls
- Pre-initialization helped

CNN - VGG Net

Peculiarities in Training:

- Multiple scales considered ($S = 256$, $S = 384$, $S \in [256, 512]$)
- Training for larger S value performed by initializing with weights obtained from smaller S values

Peculiarities in Testing:

- Multiple scales considered in testing too
- Scale values not exactly the same as training scales
- FC layers converted to convolution maps
- Dense vs multi-crop evaluation

CNN - VGG Net

Table 1: **ConvNet configurations** (shown in columns). The depth of the configurations increases from the left (A) to the right (E), as more layers are added (the added layers are shown in bold). The convolutional layer parameters are denoted as "conv⟨receptive field size⟩-⟨number of channels⟩". The ReLU activation function is not shown for brevity.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

CNN - VGG Net

Table 2: **Number of parameters** (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

CNN - VGG Net

Table 3: **ConvNet performance at a single test scale.**

ConvNet config. (Table 1)	smallest image side		top-1 val. error (%)	top-5 val. error (%)
	train (S)	test (Q)		
A	256	256	29.6	10.4
A-LRN	256	256	29.7	10.5
B	256	256	28.7	9.9
C	256	256	28.1	9.4
	384	384	28.1	9.3
	[256;512]	384	27.3	8.8
D	256	256	27.0	8.8
	384	384	26.8	8.7
	[256;512]	384	25.6	8.1
E	256	256	27.3	9.0
	384	384	26.9	8.7
	[256;512]	384	**25.5**	**8.0**

CNN - VGG Net

Table 4: **ConvNet performance at multiple test scales.**

ConvNet config. (Table 1)	smallest image side		top-1 val. error (%)	top-5 val. error (%)
	train (S)	test (Q)		
B	256	224,256,288	28.2	9.6
C	256	224,256,288	27.7	9.2
	384	352,384,416	27.8	9.2
	[256; 512]	256,384,512	26.3	8.2
D	256	224,256,288	26.6	8.6
	384	352,384,416	26.5	8.6
	[256; 512]	256,384,512	**24.8**	**7.5**
E	256	224,256,288	26.9	8.7
	384	352,384,416	26.7	8.6
	[256; 512]	256,384,512	**24.8**	**7.5**

CNN - VGG Net

Table 5: **ConvNet evaluation techniques comparison.** In all experiments the training scale S was sampled from $[256; 512]$, and three test scales Q were considered: $\{256, 384, 512\}$.

ConvNet config. (Table 1)	Evaluation method	top-1 val. error (%)	top-5 val. error (%)
D	dense	24.8	7.5
	multi-crop	24.6	7.5
	multi-crop & dense	**24.4**	**7.2**
E	dense	24.8	7.5
	multi-crop	24.6	7.4
	multi-crop & dense	**24.4**	**7.1**

CNN - VGG Net

Table 7: **Comparison with the state of the art in ILSVRC classification**. Our method is denoted as "VGG". Only the results obtained without outside training data are reported.

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

CNN - VGG Net

Table 10: **Comparison with the state of the art in ILSVRC localisation**. Our method is denoted as "VGG".

Method	top-5 val. error (%)	top-5 test error (%)
VGG	**26.9**	**25.3**
GoogLeNet (Szegedy et al., 2014)	-	26.7
OverFeat (Sermanet et al., 2014)	30.0	29.9
Krizhevsky et al. (Krizhevsky et al., 2012)	-	34.2

CNN - VGG Net

Table 11: **Comparison with the state of the art in image classification on VOC-2007, VOC-2012, Caltech-101, and Caltech-256**. Our models are denoted as "VGG". Results marked with * were achieved using ConvNets pre-trained on the *extended* ILSVRC dataset (2000 classes).

Method	VOC-2007 (mean AP)	VOC-2012 (mean AP)	Caltech-101 (mean class recall)	Caltech-256 (mean class recall)
Zeiler & Fergus (Zeiler & Fergus, 2013)	-	79.0	86.5 ± 0.5	74.2 ± 0.3
Chatfield et al. (Chatfield et al., 2014)	82.4	83.2	88.4 ± 0.6	77.6 ± 0.1
He et al. (He et al., 2014)	82.4	-	$\mathbf{93.4 \pm 0.5}$	-
Wei et al. (Wei et al., 2014)	81.5 (85.2*)	81.7 (**90.3***)	-	-
VGG Net-D (16 layers)	89.3	89.0	91.8 ± 1.0	85.0 ± 0.2
VGG Net-E (19 layers)	89.3	89.0	92.3 ± 0.5	85.1 ± 0.3
VGG Net-D & Net-E	**89.7**	**89.3**	92.7 ± 0.5	$\mathbf{86.2 \pm 0.3}$

CNN - VGG Net

Table 12: **Comparison with the state of the art in single-image action classification on VOC-2012**. Our models are denoted as "VGG". Results marked with * were achieved using ConvNets pre-trained on the *extended* ILSVRC dataset (1512 classes).

Method	VOC-2012 (mean AP)
(Oquab et al., 2014)	70.2*
(Gkioxari et al., 2014)	73.6
(Hoai, 2014)	76.3
VGG Net-D & Net-E, image-only	**79.2**
VGG Net-D & Net-E, image and bounding box	**84.0**

CNN - GoogLeNet

Going deeper with convolutions

Christian Szegedy
Google Inc.

Wei Liu
University of North Carolina, Chapel Hill

Yangqing Jia
Google Inc.

Pierre Sermanet
Google Inc.

Scott Reed
University of Michigan

Dragomir Anguelov
Google Inc.

Dumitru Erhan
Google Inc.

Vincent Vanhoucke
Google Inc.

Andrew Rabinovich
Google Inc.

CNN - GoogLeNet

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

Table : GoogLeNet incarnation of the Inception architecture

CNN - GoogLeNet

GoogLeNet Network (From Left to Right)

- Testing is a bit complicated: involves 7 models, 144 crops per image, softmax probabilities averaged over all crops
- Achieved $\sim$ **93.3%** accuracy on ILSVRC 2014 Classification Challenge

CNN - GoogLeNet

(a) Inception module, naïve version (b) Inception module with dimension reductions

Figure 2: Inception module

CNN - GoogLeNet

CNN - GoogLeNet

CNN - GoogLeNet

ILSVRC 2015 Challenge

image-net.org/challenges/LSVRC/2015/ 133% Search

IMAGENET Large Scale Visual Recognition Challenge 2015 (ILSVRC2015)

News History Timetable Introduction Main challenges Taster challenges GPU resources Registration FAQ Citation Contact

News

- Feburary 25, 2016: Per-class results are available.
- January 28, 2016: Test server for 2015 is open.
- January 18, 2016: Videos of the workshop are now available.
- December 10, 2015: New York Times covers ILSVRC2015.
- December 10, 2015: Results announced.
- November 23, 2015: Submission server for VID is open.
- November 2, 2015: Submission server is open.
- October 19, 2015: ImageNet and MS COCO Visual Recognition Challenges Joint Workshop schedule is announced.
- October 2, 2015: Object detection from video initial release is ready for download.
- August 15, 2015: Registration is up. Register your team today.
- August 13, 2015: Computational resources for registered teams, provided by NVIDIA and IBM Cloud.
- June 12, 2015: Tentative timetable is announced. Please check firm time at the end of July.
- June 2, 2015: Additional announcement regarding submission server policy is released.
- May 19, 2015: Announcement regarding submission server policy is released.
- December 17, 2015: Stay tuned. Coming soon.

ILSVRC 2015 Challenge

1. Two main competitions:
   I. Object detection for 200 fully labeled categories.
   II. Object localization for 1000 categories.
2. Two taster competitions New:
   I. Object detection from video for 30 fully labeled categories
   II. Scene classification for 401 categories.

CNN - ResNet

Deep Residual Learning for Image Recognition

Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun
Microsoft Research
{kahe, v-xiangz, v-shren, jiansun}@microsoft.com

CNN - ResNet

- Testing involves 6 models, multiple crops per image, softmax probabilities averaged over all crops
- ResNet-152 Achieved $\sim$ **97%** accuracy on ILSVRC 2015 Classification Challenge

CNN - ResNet

- **Aim**: To model a map $\mathcal{H} : I \to S$
- I can denote the input space or some suitable intermediate representation space.
- S can denote the output space or some suitable intermediate representation space.

CNN - ResNet

- **Idea**: To model $\mathcal{H}$ using $\mathcal{H}(x) = \mathcal{F}(x) + Wx$ and learn $\mathcal{F}$ and W.
- If I and S are of same dimensions, W can be considered (or hypothesized) to be the identity, hence the aim reduces to learning $\mathcal{F}$.

CNN - ResNet

- **Motivation**: Learning the actual map $\mathcal{H}$ is difficult.
- **Hope**: Can we ease the task by focusing the learning task related to $\mathcal{F}$ and then use the original input to get back $\mathcal{H}$.
- **Caveat**: Learning $\mathcal{F}$ is also not that easy! (Exploit deep nets to learn $\mathcal{F}$)

CNN - ResNet

Figure 2. Residual learning: a building block.

CNN - ResNet

model	top-1 err.	top-5 err.
VGG-16 [41]	28.07	9.33
GoogLeNet [44]	-	9.15
PReLU-net [13]	24.27	7.38
plain-34	28.54	10.02
ResNet-34 A	25.03	7.76
ResNet-34 B	24.52	7.46
ResNet-34 C	24.19	7.40
ResNet-50	22.85	6.71
ResNet-101	21.75	6.05
ResNet-152	**21.43**	**5.71**

Table 3. Error rates (%, **10-crop** testing) on ImageNet validation. VGG-16 is based on our test. ResNet-50/101/152 are of option B that only uses projections for increasing dimensions.

CNN - ResNet

method	top-1 err.	top-5 err.
VGG [41] (ILSVRC'14)	-	8.43†
GoogLeNet [44] (ILSVRC'14)	-	7.89
VGG [41] (v5)	24.4	7.1
PReLU-net [13]	21.59	5.71
BN-inception [16]	21.99	5.81
ResNet-34 B	21.84	5.71
ResNet-34 C	21.53	5.60
ResNet-50	20.74	5.25
ResNet-101	19.87	4.60
ResNet-152	**19.38**	**4.49**

Table 4. Error rates (%) of **single-model** results on the ImageNet validation set (except † reported on the test set).