Convolutional Networks for Computer Vision

Roberto Paredes

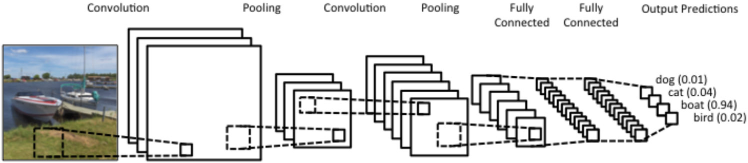
Centro de Investigación
Pattern Recognition and Human Language Technologies
Universidad Politécnica de Valencia

Index

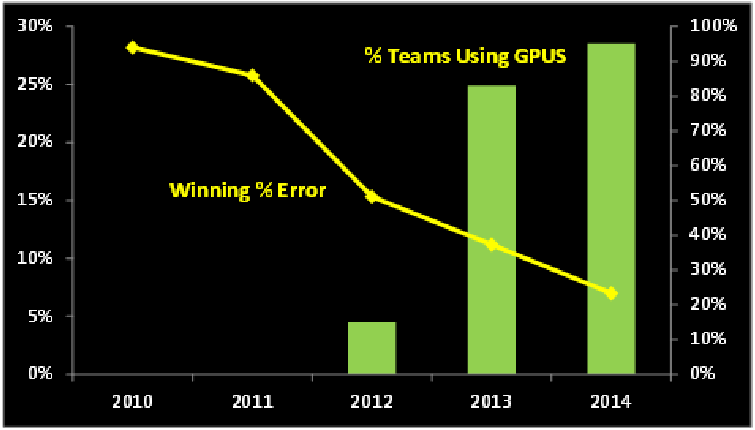
- Introduction
- Convolution Operator
- Pooling Operator
- Reshape Layers
- Special Layers
- Convolutional Networks

Introduction

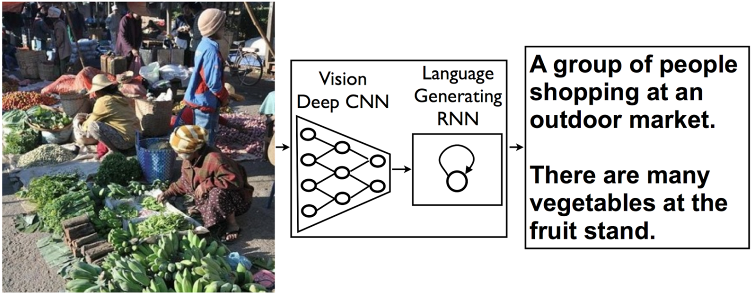
- Deep Learning → Bridge the gap between raw representation and categories



- ImageNet Challenge



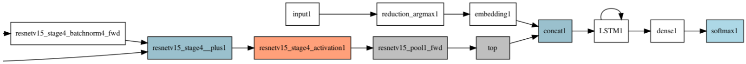
Introduction. Image to text



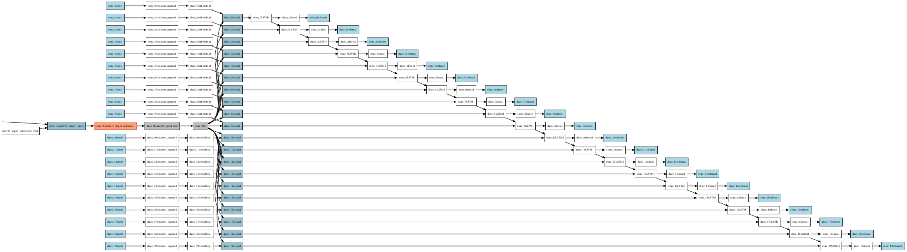
A group of people shopping at an outdoor market.

There are many vegetables at the fruit stand.

Introduction. Image to text example



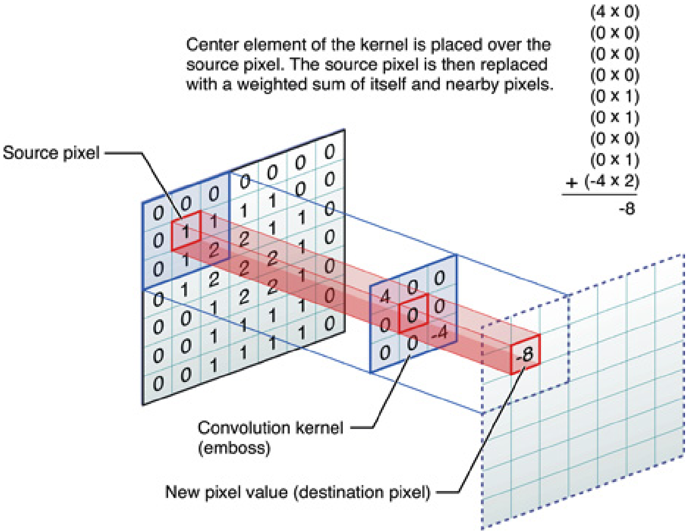
Introduction. Image to text example (unrolled)



Index

- Introduction
- **Convolution Operator**
- Pooling Operator
- Special Layers
- Reshape Layers
- Convolutional Networks

Convolution Operator



Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.

(4 x 0)
(0 x 0)
(0 x 0)
(0 x 0)
(0 x 1)
(0 x 1)
(0 x 0)
(0 x 1)
+ (-4 x 2)
-8

Source pixel
Convolution kernel (emboss)
New pixel value (destination pixel)
4 0 0
0 0 0
0 0 -4
-8

Convolution Operator

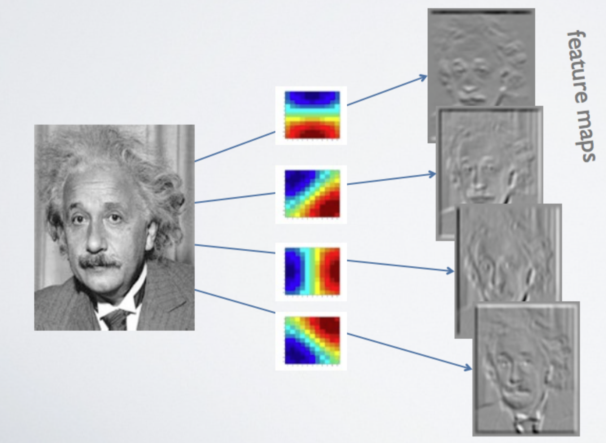
- The simplest case:
  - Size of input image: $I_R \times I_C$
  - Size of kernel: $k_r \times k_c$
  - Size of output image: $O_R \times O_C$
    - $O_R = (I_R - k_r) + 1$
    - $O_C = (I_C - k_c) + 1$
  - Convolution cost: $O_R \times O_C \times k_r \times k_c$

Convolution Operator

- Padding, same output size than input size
  - $O_R = I_R$
  - $O_C = I_C$
  - Add a frame of $k_r/2 \times k_c/2$ of 0's to the input image
- Stride, jump scanning the input image
- In general:
  - $O_R = \lfloor (I_R + 2 * pad - k_r)/stride_r + 1 \rfloor$
  - $O_C = \lfloor (I_C + 2 * pad - k_c)/stride_c + 1 \rfloor$

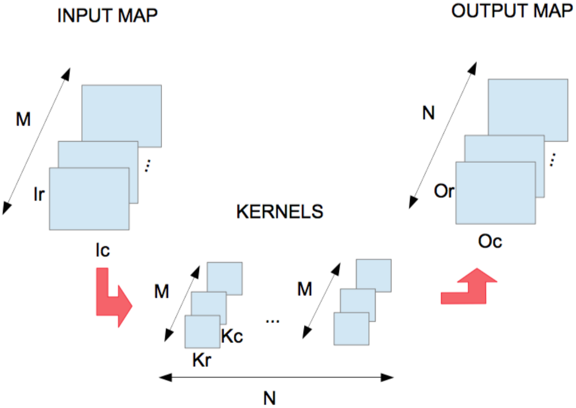
Convolution Operator

- The general case:
  - Input: Images (2D)
  - Apply more than one kernel (3D)
  - Output: Maps (3D)



Convolution Operator

- The general case:
  - Input: Maps (3D)
  - Apply more than one kernel (4D)
  - Output: Maps (3D)

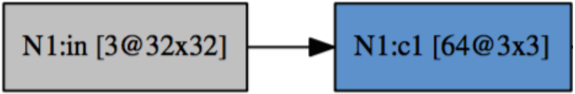


Convolution Operator

- Implementation tricks: **LOWERING**
  - A convolution becomes a standard multipication $I \times K$
- Implementation tricks: **Multi-threading**
  - For instance, split the batch into different threads
- Implementation tricks: **FFT**
  - A convolution is a multiplication in the frecuency domain
- Implementation tricks: **Winograd** algorithm:
  - A fast method to obtain the convolution with less multiplication and adition operations

Convolution Operator as a layer

- Input map $M = 3$, $O_R = 32$ and $O_C = 32$
- Kernels $N = 64$, $K_r = 3$ and $k_c = 3$
- By default a convolutional layer as N bias



- Output map sizes?

Index

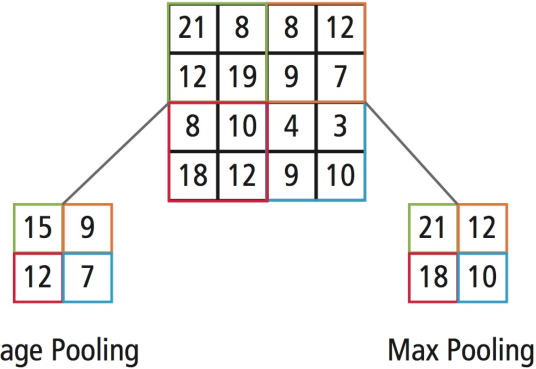
- Introduction
- Convolution Operator
- **Pooling Operator**
- Reshape Layers
- Special Layers
- Convolutional Networks

Pooling Operator

- The main goal is to reduce the size of the maps:
    - Reduce the computational cost
    - Deal with multiscale
    - Capture higher level features

Pooling Operator

- Maxpool and Average Pool



21	8	8	12
12	19	9	7
8	10	4	3
18	12	9	10

15	9
12	7

Average Pooling

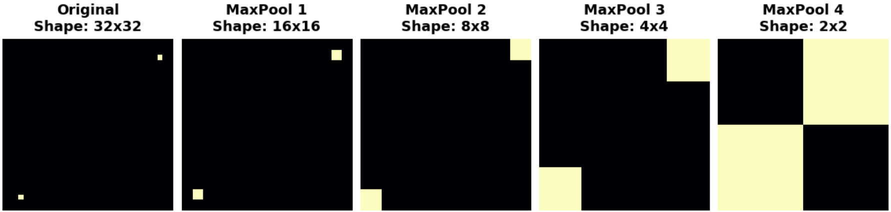
21	12
18	10

Max Pooling

- Normally *stride* $=$ *size*

Pooling Operator

- Results after applying several pooling operators:

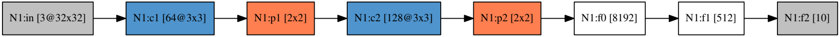


Index

- Introduction
- Convolution Operator
- Pooling Operator
- **Reshape Layers**
- Special Layers
- Convolutional Network

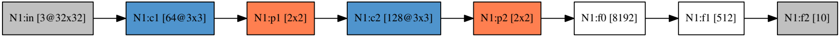
Reshape Layers

- The goal of the reshape layer is to present the maps to the next layers Fully Connected layers
- The maps become a raw vector and the spatial relationship is not considered
- After the reshape several hidden layers can be stacked to reach the output layer, conforming a **Convolutional Network**:



Reshape Layers - Exercise

- Given the following CNN, How many parameters (weights) are?
- Where is the big amount of parameters?

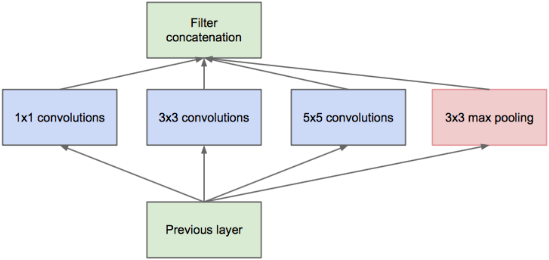


Index

- Introduction
- Convolution Operator
- Pooling Operator
- Reshape Layers
- **Special Layers**
- Convolutional Networks

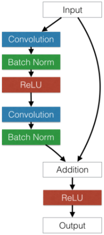
Special Layers - Cat Layers

- A layer that cat **in depth** the maps of the layers that are connected to
- The sizes of the maps must be equivalent
- Used in the Inception Model (GoogleNet)



Special Layers - Agregation Layers

- Similar to cat layers
- A layer that sum the maps of the layers that are connected to
- The size and number of maps must be the same
- Used on the Residual Nets (Microsoft Research)

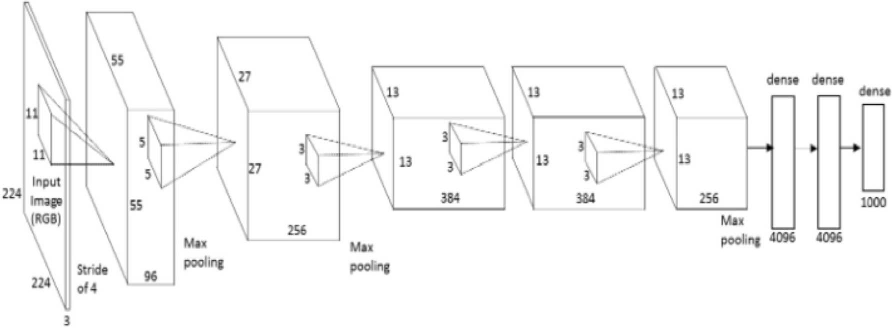


Index

- Introduction
- Convolution Operator
- Pooling Operator
- Reshape Layers
- Special Layers
- **Convolutional Networks**

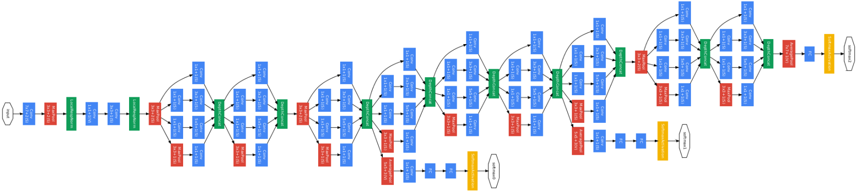
Convolutional Networks - AlexNet

- ILSVRC 2012 Winner



Convolutional Networks - GoogleNet

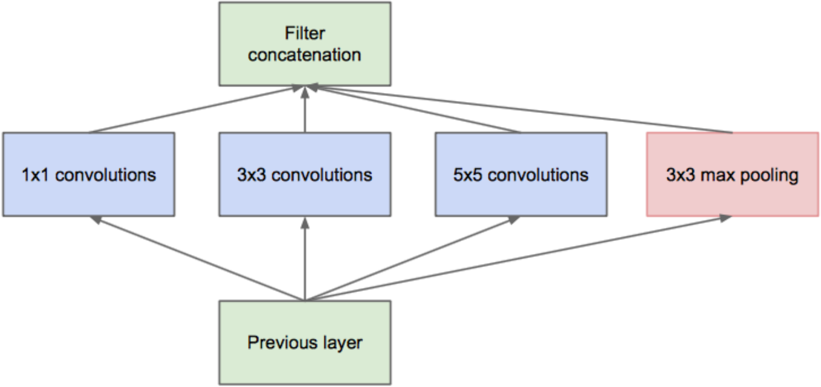
- ILSVRC 2014 Winner



- `http://www.cs.unc.edu/~wliu/papers/GoogLeNet.pdf`

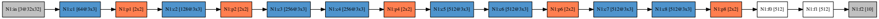
Convolutional Networks - GoogleNet

- Inception model



Convolutional Networks - OxfordNet (VGG)

- ILSVRC2014 2nd, but best single model

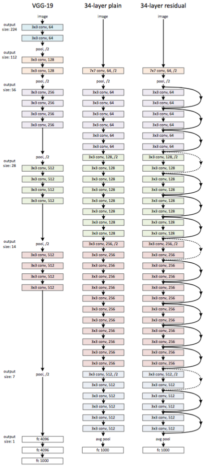


N1:in [3@32x32]
N1:c1 [64@3x3]
N1:p1 [2x2]
N1:c2 [128@3x3]
N1:p2 [2x2]
N1:c3 [256@3x3]
N1:c4 [256@3x3]
N1:p4 [2x2]
N1:c5 [512@3x3]
N1:c6 [512@3x3]
N1:p6 [2x2]
N1:c7 [512@3x3]
N1:c8 [512@3x3]
N1:p8 [2x2]
N1:f0 [512]
N1:f1 [512]
N1:f2 [10]

- `http://arxiv.org/pdf/1409.1556v6.pdf`

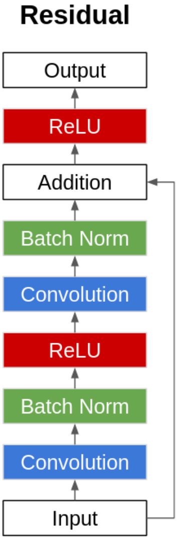
Convolutional Networks - ResidualNet (MSR)

- ILSVRC 2015 Winner



- `http://arxiv.org/pdf/1512.03385v1.pdf`

Convolutional Networks - ResidualNet (MSR)


Residual
Output
ReLU
Addition
Batch Norm
Convolution
ReLU
Batch Norm
Convolution
Input