Inception Architecture

A 30 Minutes Introduction from Scratch

Shubhra Aich
s.aich.72@gmail.com

March 09, 2017

Outline

Architectural Overview

- RGB image basics
- ANN vs CNN
- Operations in CNN
  - Convolution
  - Pooling
  - Rectification
  - Dropout

Breaking Down Convolution

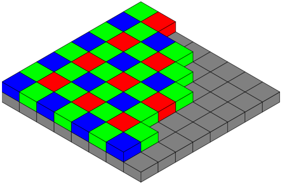
- From 5x5 to 3x3
- From 3x3 to 3x1 and 1x3

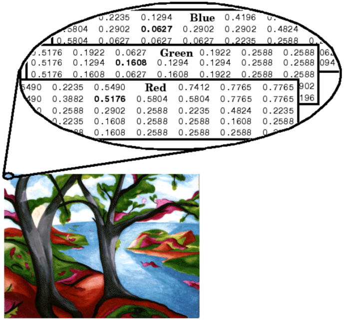
Inception Architecture

- Inception modules
- Final architecture

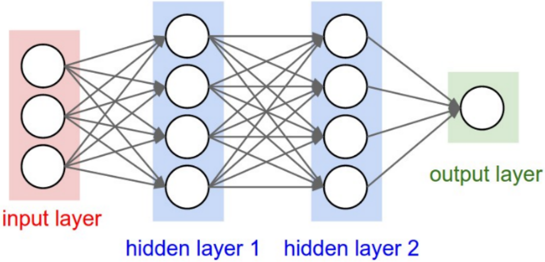
RGB image basics

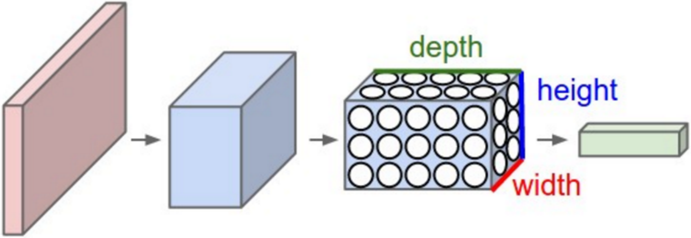
RGB image consists of 3 channels or planes, namely Red, Green and Blue.
We see the combined effect of these channels.





Artifical *vs* Convolutional NN





Operations in CNN

- Convolution
- Rectification
- Pooling
- Dropout

Convolution ...(1)

Input Volume (+pad 1) (7x7x3)

x[:,:,0]

0	0	0	0	0	0	0
0	0	2	2	0	0	0
0	1	0	2	0	1	0
0	1	2	1	1	2	0
0	0	1	1	1	1	0
0	2	2	1	0	2	0
0	0	0	0	0	0	0

x[:,:,1]

0	0	0	0	0	0	0
0	2	2	0	1	1	0
0	0	1	2	1	0	0
0	0	0	0	0	2	0
0	0	2	0	1	2	0
0	0	1	2	0	0	0
0	0	0	0	0	0	0

x[:,:,2]

0	0	0	0	0	0	0
0	2	2	0	2	2	0
0	2	1	1	1	2	0
0	2	0	1	0	0	0
0	0	1	2	0	2	0
0	1	2	2	2	1	0
0	0	0	0	0	0	0

Filter W0 (3x3x3)

w0[:,:,0]

-1	-1	-1
-1	1	1
1	1	-1

w0[:,:,1]

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

w0[:,:,2]

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

Bias b0 (1x1x1)

b0[:,:,0]

1

Filter W1 (3x3x3)

w1[:,:,0]

0	1	0
0	1	-1
-1	1	0

w1[:,:,1]

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

w1[:,:,2]

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

Bias b1 (1x1x1)

b1[:,:,0]

0

Output Volume (3x3x2)

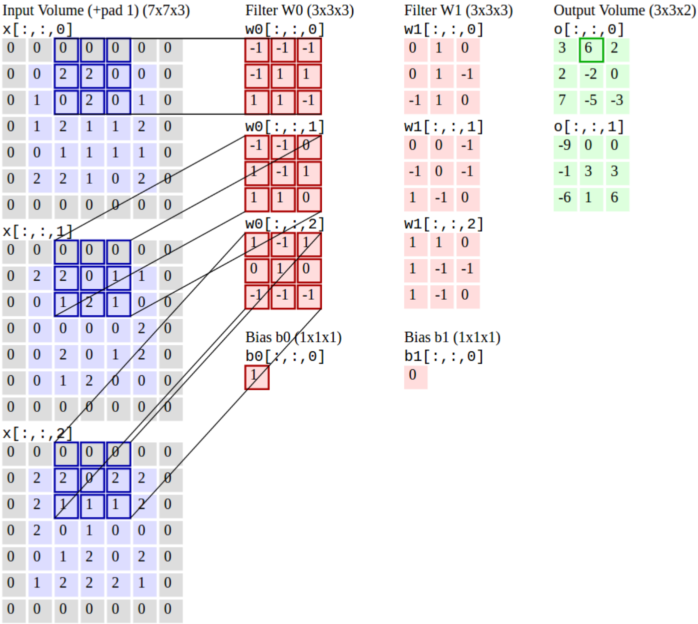
o[:,:,0]

3	6	2
2	-2	0
7	-5	-3

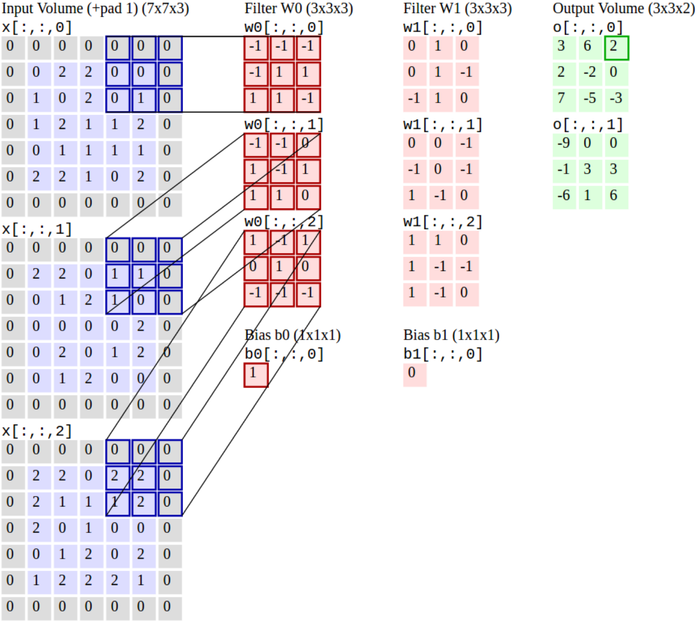
o[:,:,1]

-9	0	0
-1	3	3
-6	1	6

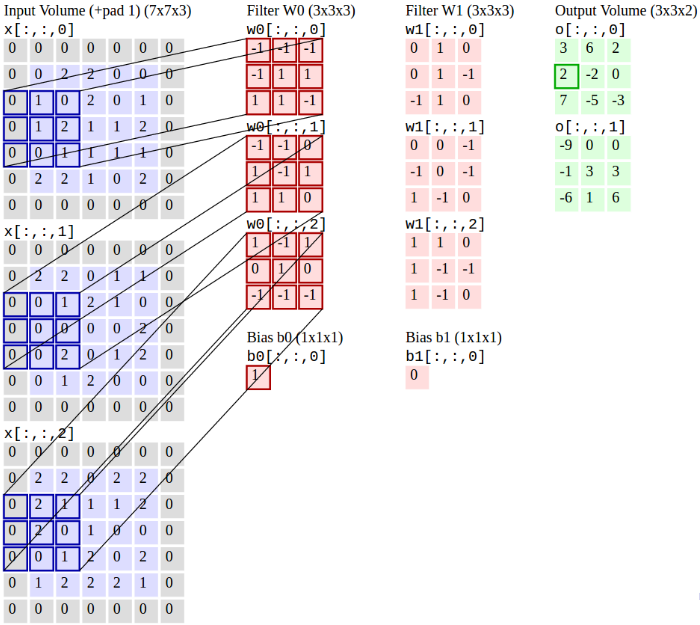
Convolution ...(2)



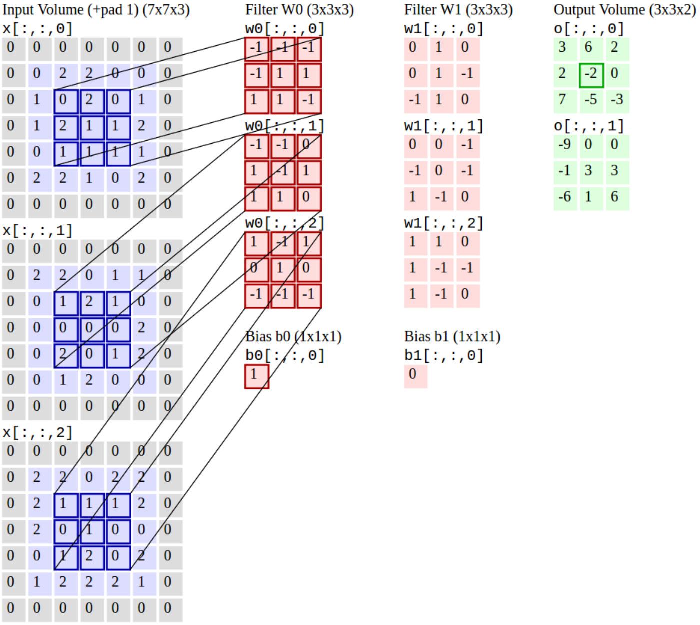
Convolution ...(3)



Convolution ...(4)



Convolution ...(5)



Convolution ...(6)

Input Volume (+pad 1) (7x7x3)

x[:,:,0]

0	0	0	0	0	0	0
0	0	2	2	0	0	0
0	1	0	2	0	1	0
0	1	2	1	1	2	0
0	0	1	1	1	1	0
0	2	2	1	0	2	0
0	0	0	0	0	0	0

x[:,:,1]

0	0	0	0	0	0	0
0	2	2	0	1	1	0
0	0	1	2	1	0	0
0	0	0	0	0	2	0
0	0	2	0	1	2	0
0	0	1	2	0	0	0
0	0	0	0	0	0	0

x[:,:,2]

0	0	0	0	0	0	0
0	2	2	0	2	2	0
0	2	1	1	1	2	0
0	2	0	1	0	0	0
0	0	1	2	0	2	0
0	1	2	2	2	1	0
0	0	0	0	0	0	0

Filter W0 (3x3x3)

w0[:,:,0]

-1	-1	-1
-1	1	1
1	1	-1

w0[:,:,1]

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

w0[:,:,2]

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

Bias b0 (1x1x1)

b0[:,:,0]

1

Filter W1 (3x3x3)

w1[:,:,0]

0	1	0
0	1	-1
-1	1	0

w1[:,:,1]

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

w1[:,:,2]

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

Bias b1 (1x1x1)

b1[:,:,0]

0

Output Volume (3x3x2)

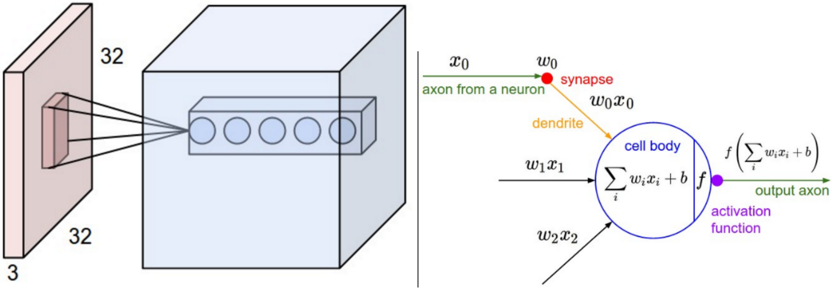
o[:,:,0]

3	6	2
2	-2	0
7	-5	-3

o[:,:,1]

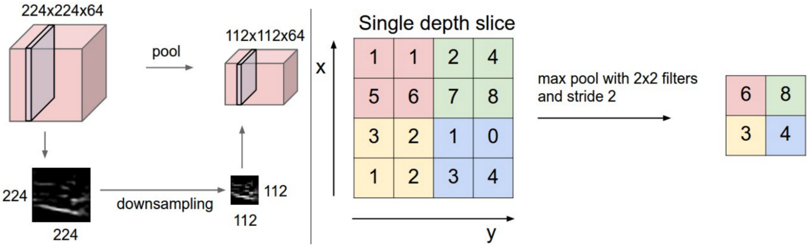
-9	0	0
-1	3	3
-6	1	6

Equations of Convolution



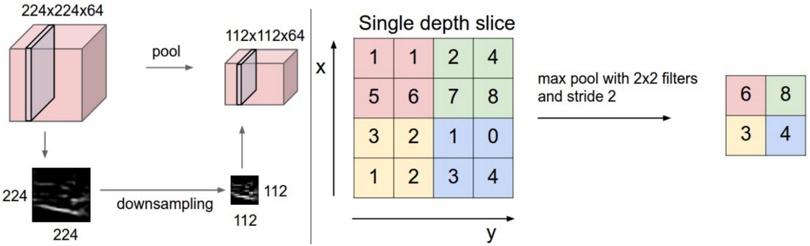
w_i = weights of the filter
x_i = outputs of the previous layer
$z = \sum_i w_i x_i + b = w.x + b \equiv w * x + b$
a = f(z)
f = activation/ rectification / squashing function

Pooling



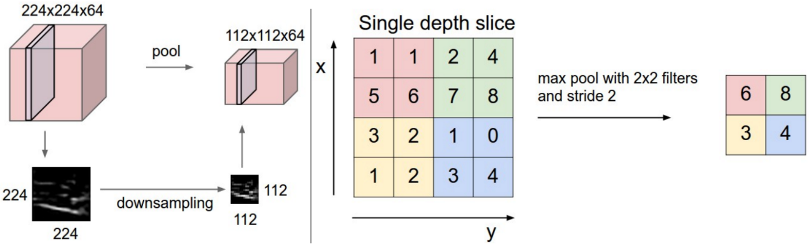
- Max-pooling.
- Average pooling.

Pooling



- Max-pooling.
- Average pooling.
- Min-pooling

Pooling



- Max-pooling.
- Average pooling.
- Min-pooling ☠

Why do we use Convolution and Pooling? ...(1)

CNN combines 3 architectural ideas to deal with shift, scale and distortion.

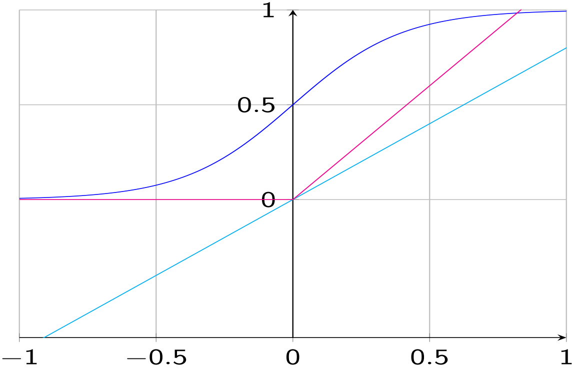
- Local receptive fields/filters
  - The idea of local filters dates back to Hubel-Wiesel's discovery of locally sensitive, orientation selective neurons in the cat's visual cortex.
  - It helps to extract elementary visual features, such as edges, corners, end-points etc.
- Shared weights
  - Distortions or shifts of the input causes the position of salient features to vary.
  - Elementary feature detectors that are useful on one region of the image are likely to be useful across the entire image.
  - This knowledge can be applied by forcing a set of units, whose receptive fields are located at different places on the to be identical.
  - These features are combined by subsequent layers to detect more meaningful features.

Why do we use Convolution and Pooling? ...(2)

- Sub-sampling/Pooling
  - Once a feature is detected, its exact location becomes less important, only approximate position relative to other features is relevant.
  - This precise position is even harmful they vary from image to image.
  - The simplest way to reduce this precision is sub-sampling or pooling.

Rectification

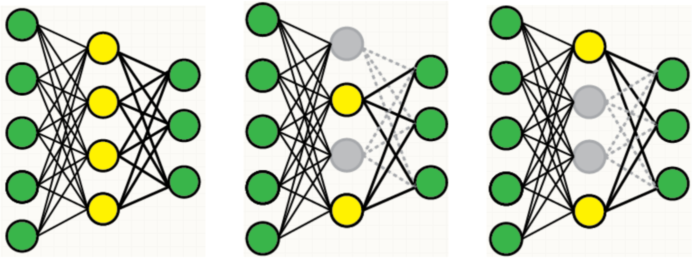
- Linear neuron, $y = wx + b$
- Rectified linear neuron (ReLU), $y = \begin{cases} wx + b, & \text{if} x > 0 \\ 0, & \text{otherwise} \end{cases}$
- Sigmoid/logistic neuron, $y = \frac{1}{1+exp(-\alpha(wx+b))}$



Choice of Recitification Functions

- Backpropagation is the most popular learning algorithm for a network comprising real-valued units.
- This learning algorithm adjusts the weights in each iteration in proportion to the derivative of the final error function w.r.t. the weights.
- To facilitate learning, rectification functions are expected to have more or less smooth derivatives.
- Linear neurons can map only linear input-output relationships.
- Linear neurons are unbounded both below and above.
- Sigmoid activation works well for simple ANN. However, in deeper models like CNN, tiny error-gradients of sigmoid units make learning horrendously slow.

Dropout



- In each iteration of learning, we randomly omit each hidden unit with a prespecified probability.
- Using dropout, for a network comprising single hidden layer with H hidden units, in each iteration, we randomly sample from 2^H different architectures.
- All architectures share weights.
- Dropout prevents overfitting.
- In test time, the full network is used with parameter values halved.

Breaking down Convolution ... (1)

- Straightforward 5×5 convolution
  M $\equiv$ Multiplication and A $\equiv$ Addition

$$
\begin{array}{|c|c|c|c|c|}
\hline
a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ \hline
a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ \hline
a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ \hline
a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ \hline
a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \\ \hline
\end{array}
\quad * \quad
\begin{array}{|c|c|c|c|c|}
\hline
h_{11} & h_{12} & h_{13} & h_{14} & h_{15} \\ \hline
h_{21} & h_{22} & h_{23} & h_{24} & h_{25} \\ \hline
h_{31} & h_{32} & h_{33} & h_{34} & h_{35} \\ \hline
h_{41} & h_{42} & h_{43} & h_{44} & h_{45} \\ \hline
h_{51} & h_{52} & h_{53} & h_{54} & h_{55} \\ \hline
\end{array}
$$

$output_{33} =$
$a_{11}h_{11} + a_{12}h_{12} + \cdots + a_{31}h_{31} + a_{32}h_{32} + \cdots + a_{54}h_{54} + a_{55}h_{55}$
$\equiv 25\text{M} + 24\text{A}$
The filter (h_{ij}) will slide over the image (a_{ij}) for each position once totalling 25 times.
So, total number of gross multiplication and addition $= 25(25\text{M} + 24\text{A}) = 625\text{M} + 600\text{A}$

Breaking down Convolution ... (2)

- Breaking down 5×5 convolution into 2 consecutive 3×3 convolutions
  - Step - 01: Convolving 5×5 matrix with $3 \times 3 filter$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{bmatrix} * \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} = \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} & b_{15} \\ b_{21} & b_{22} & b_{23} & b_{24} & b_{25} \\ b_{31} & b_{32} & b_{33} & b_{34} & b_{35} \\ b_{41} & b_{42} & b_{43} & b_{44} & b_{45} \\ b_{51} & b_{52} & b_{53} & b_{54} & b_{55} \end{bmatrix}$$

$b_{22} = a_{11}h_{11} + a_{12}h_{12} + \cdots + a_{32}h_{32} + a_{33}h_{33}$

$\equiv$ 9M + 8A

This set of operation is performed 25 times, once for each a_{ij}

So, total multiplication + addition in the first step

$= 25(9M + 8A) = 225M + 200A$

Breaking down Convolution ... (3)

- Breaking down 5×5 convolution into 2 consecutive 3×3 convolutions
  - Step - 02: Convolving 3×3 intermediate matrix with another 3×3 filter

$$\begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} & b_{15} \\ b_{21} & b_{22} & b_{23} & b_{24} & b_{25} \\ b_{31} & b_{32} & b_{33} & b_{34} & b_{35} \\ b_{41} & b_{42} & b_{43} & b_{44} & b_{45} \\ b_{51} & b_{52} & b_{53} & b_{54} & b_{55} \end{bmatrix} * \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix} = \begin{bmatrix} o_{11} & o_{12} & o_{13} & o_{14} & o_{15} \\ o_{21} & o_{22} & o_{23} & o_{24} & o_{25} \\ o_{31} & o_{32} & o_{33} & o_{34} & o_{35} \\ o_{41} & o_{42} & o_{43} & o_{44} & o_{45} \\ o_{51} & o_{52} & o_{53} & o_{54} & o_{55} \end{bmatrix}$$

$o_{22} = b_{11}k_{11} + b_{12}k_{12} + \cdots + b_{32}k_{32} + b_{33}k_{33}$

$\equiv$ 9M + 8A

Like before, this set of operation is performed 25 times, once for each b_{ij}

So, total multiplication + addition in the first step

$= 25(9M + 8A) = 225M + 200A$

Breaking down Convolution ... (4)

$$\begin{array}{|c|c|c|c|c|}\hline a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ \hline a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ \hline a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ \hline a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ \hline a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \\ \hline \end{array} * \begin{array}{|c|c|c|c|c|}\hline h_{11} & h_{12} & h_{13} & h_{14} & h_{15} \\ \hline h_{21} & h_{22} & h_{23} & h_{24} & h_{25} \\ \hline h_{31} & h_{32} & h_{33} & h_{34} & h_{35} \\ \hline h_{41} & h_{42} & h_{43} & h_{44} & h_{45} \\ \hline h_{51} & h_{52} & h_{53} & h_{54} & h_{55} \\ \hline \end{array}$$

- Computational cost of straightforward 5×5 convolution $= 625M + 600A$.

$$\begin{array}{|c|c|c|c|c|}\hline a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ \hline a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ \hline a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ \hline a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ \hline a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \\ \hline \end{array} * \begin{array}{|c|c|c|}\hline h_{11} & h_{12} & h_{13} \\ \hline h_{21} & h_{22} & h_{23} \\ \hline h_{31} & h_{32} & h_{33} \\ \hline \end{array} * \begin{array}{|c|c|c|}\hline k_{11} & k_{12} & k_{13} \\ \hline k_{21} & k_{22} & k_{23} \\ \hline k_{31} & k_{32} & k_{33} \\ \hline \end{array}$$

- Computational cost of 5×5 convolution using consecutive 3×3 convolutions $= 450M + 400A$.

Breakding down Convolution ... (5)

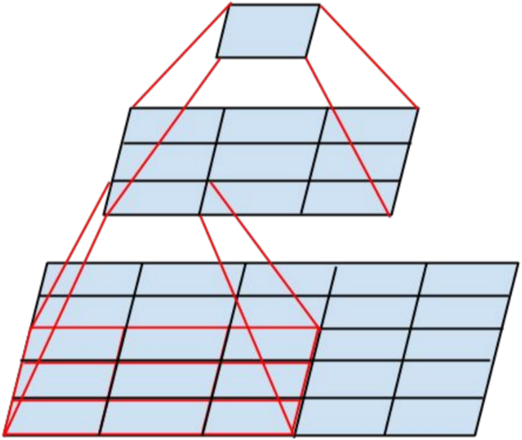


Figure 1. Mini-network replacing the 5 × 5 convolutions.

Breakding down Convolution ... (6)

Breakding down 3×3 convolution into 3×1 and 1×3 convolutions (assymetric breakdown).

- Let us calculate the average of a 3×3 matrix.

$$\begin{array}{|c|c|c|}\hline 9 & 8 & 7 \\ \hline 6 & 5 & 4 \\ \hline 3 & 2 & 1 \\ \hline \end{array} * \begin{array}{|c|c|c|}\hline \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \hline \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \hline \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \hline \end{array} = \frac{9}{9} + \frac{8}{9} + \cdots + \frac{2}{9} + \frac{1}{9} \equiv 9(9M + 8A)$$

$$\begin{array}{|c|c|c|}\hline \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \hline \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \hline \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \hline \end{array} = \begin{array}{|c|}\hline \frac{1}{3} \\ \hline \frac{1}{3} \\ \hline \frac{1}{3} \\ \hline \end{array} \quad \begin{array}{|c|c|c|}\hline \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \hline \end{array} \begin{array}{l} \equiv 9(3M + 2A) + 9(3M + 2A) \\ = 9(6M + 4A) \end{array}$$

- The average kernel is a separable filter (rank-1 matrix).

Inception Modules ... (1)

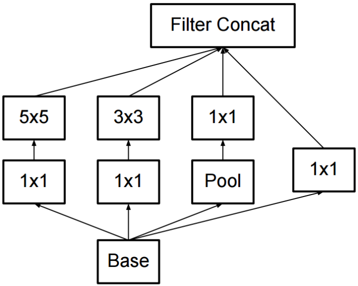
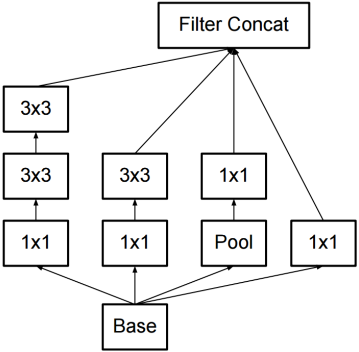
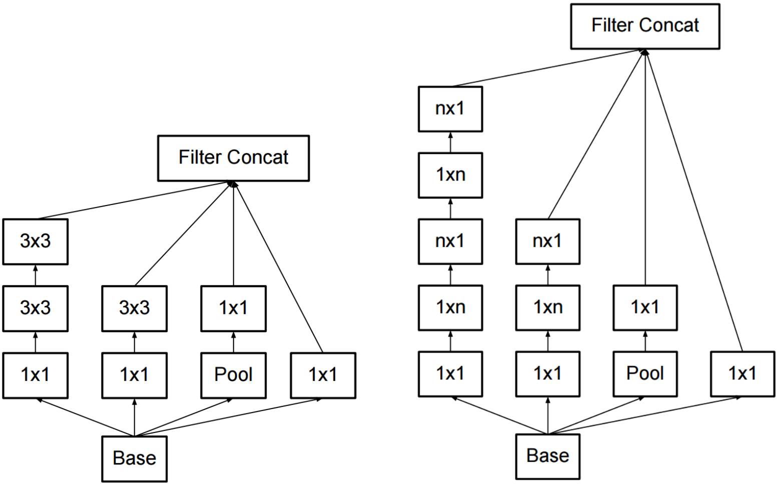


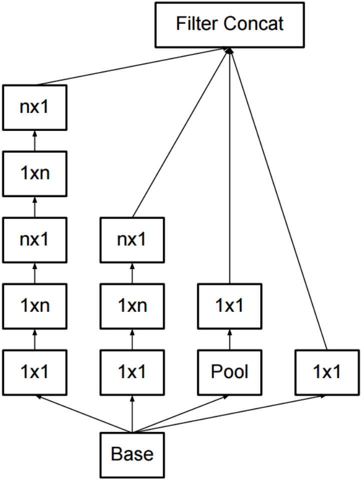
Figure 4. Original Inception module as described in [20].

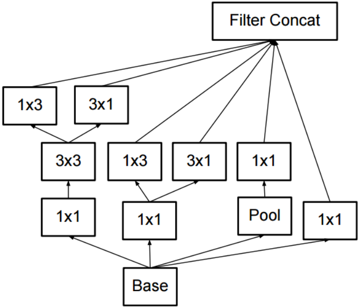


Inception Modules ... (2)



Inception Modules - Final





Complete Architecture