



빅데이터 사이언스 실무자 양성과정

CNN(Convolution Neural Networks) 특강

SBA Innovation Growth Intensive Training
Kim Jin Soo



◆ CNN(Convolution Neural Networks) Overview

◆ Step 1 – Convolution Operation

◆ Step 1 – ReLU Layer

◆ Step 2 – Pooling

◆ Step 3 – Flattening

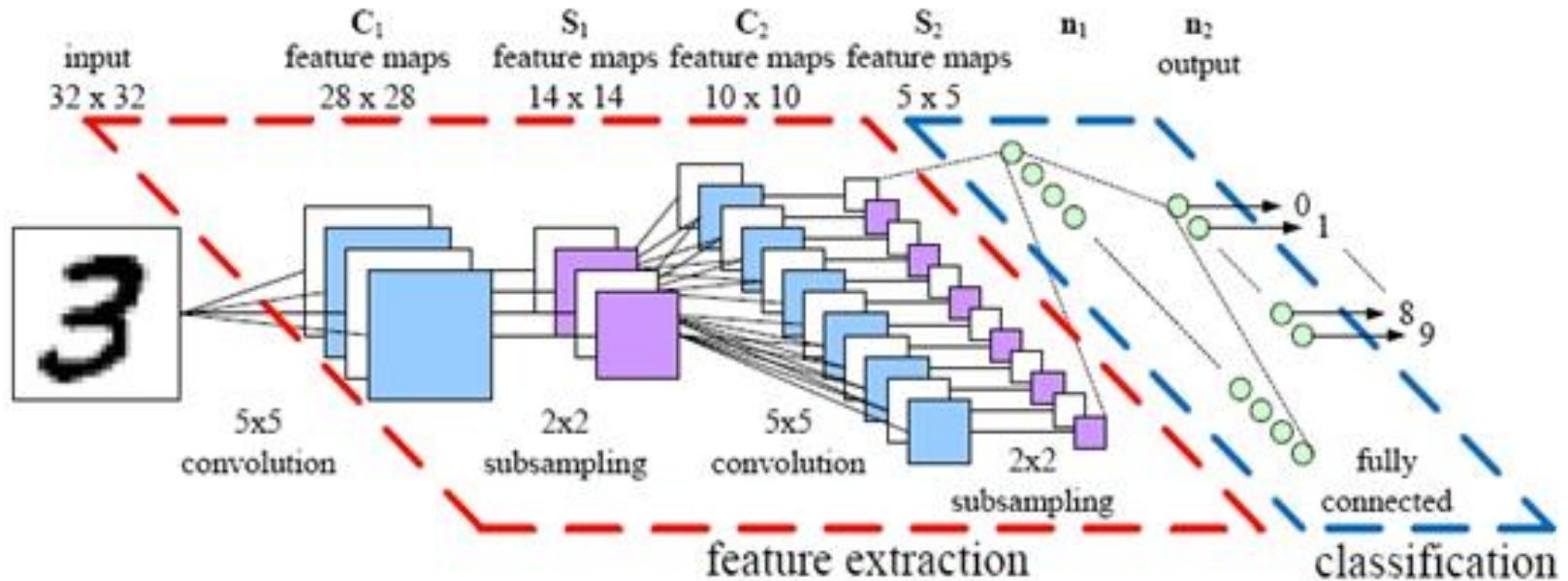
◆ Step 4 – Full Connection

◆ Summary

CNN (Convolutional Neural Network)



- ◆ 최근 신경망 연구에서 가장 두드러진 변화를 일으키고 있는 딥러닝의 핵심분야이다.
- ◆ 특히, 컴퓨터비전, 컴퓨터오디션, 강화학습 등에서 최첨단의 성과를 도출
- ◆ 컨볼루션넷은 새로운 서비스와 기능을 통해 기술기반의 회사에 널리 적용되고 있다.



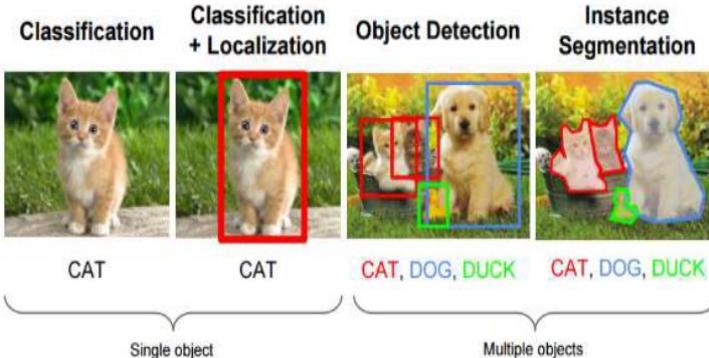
CNN 적용분야



CNN application in Handwritten Digit Recognizer



Object recognition and classification using Convolutional Networks



CNN application in Self Driving Cars



- 이미지 캡션, 이미지 인식, 다양한 시각 시스템 처리
- 이미지 기반의 물체 혹은 사람, 위치, 탐지 및 구분
- 음성의 문자변환, 오디오 합성
- 자연어로 이미지 및 동영상 묘사
- 무인차량에서 장애물 탐지 및 자율주행
- 생성모델을 통해 이미지, 사운드, 텍스트에 대한 홀루시네이팅(hallucinating)

CNN (Convolutional Neural Network)



Yann Lecun

CNN의 역사



Hubel & Wiesel,

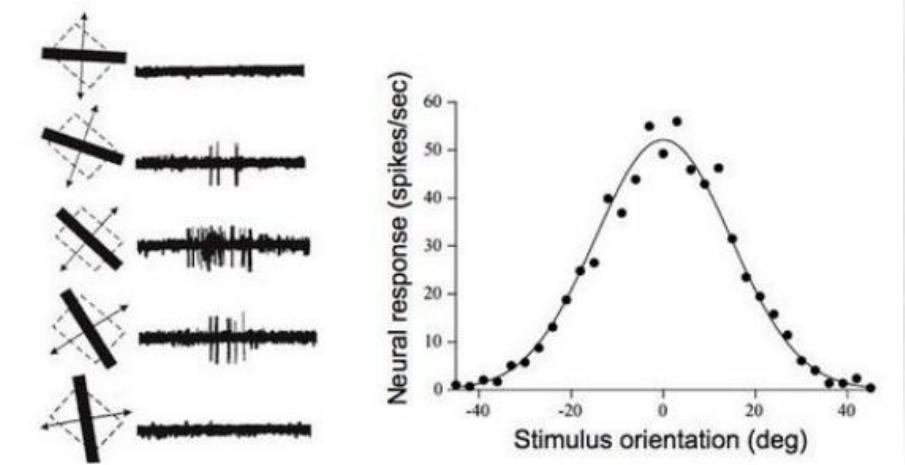
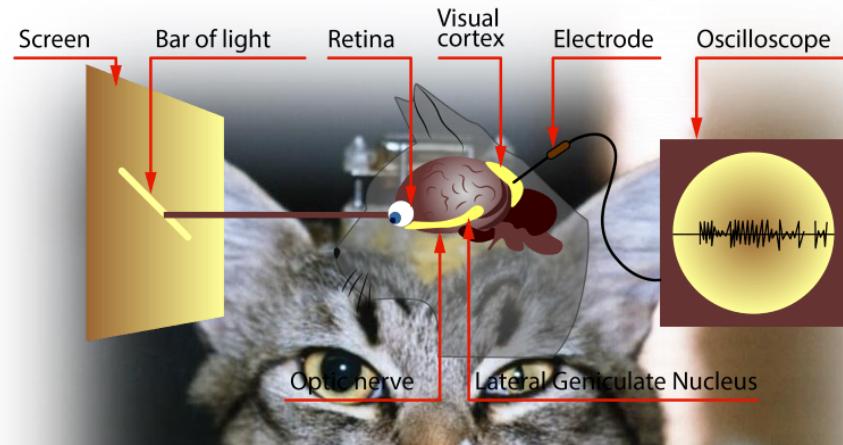
1959

RECEPTIVE FIELDS OF SINGLE
NEURONES IN
THE CAT'S STRIATE CORTEX

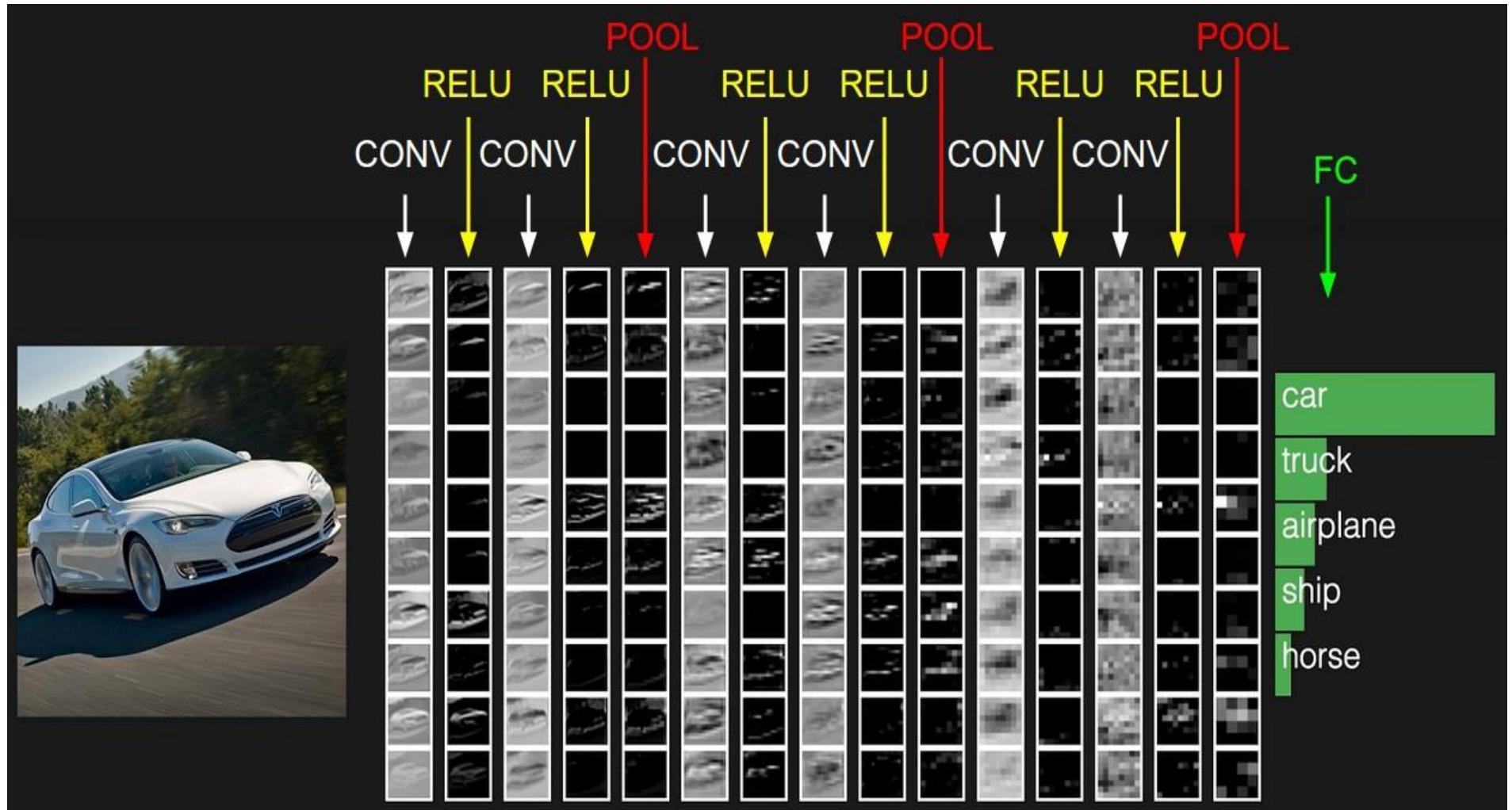
1962

RECEPTIVE FIELDS, BINOCULAR
INTERACTION
AND FUNCTIONAL ARCHITECTURE IN
THE CAT'S VISUAL CORTEX

1968



CNN DEMO



[ITP-NYU, CNN] <https://ml4a.github.io/classes/itp-S16/03/>

CNN (Convolutional Neural Network)

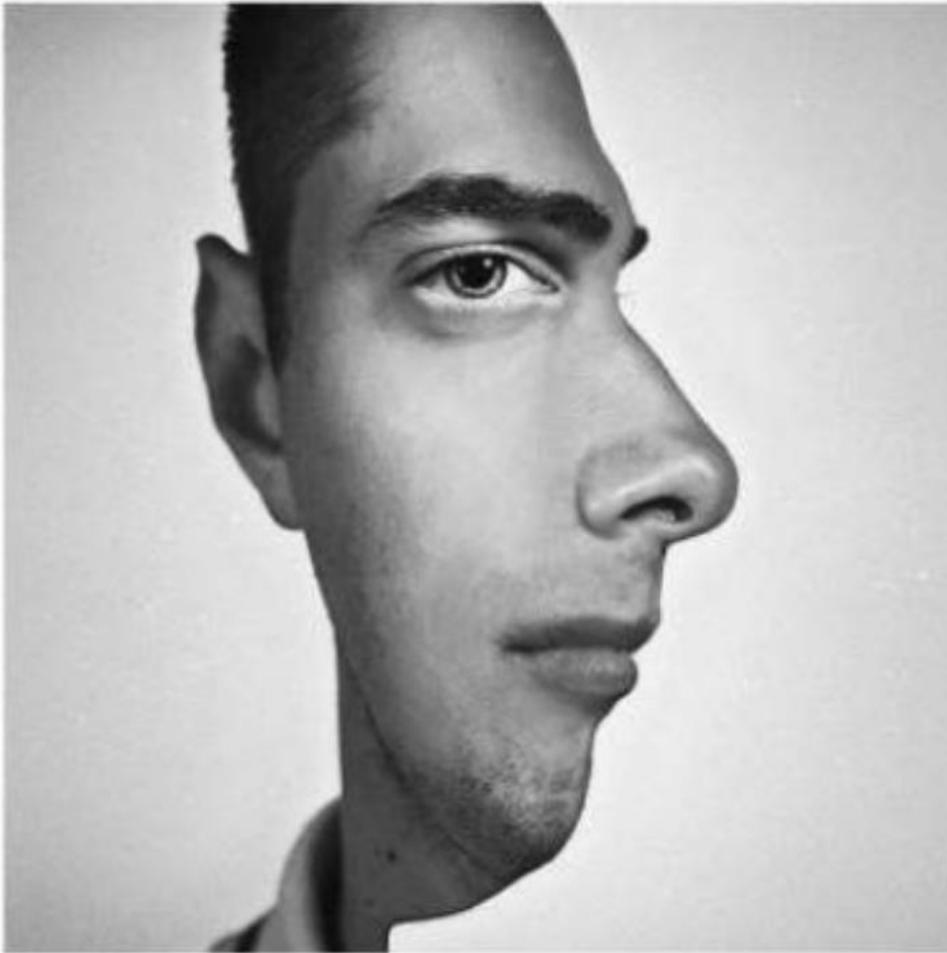


What we will learn in this section:

- What are Convolutional Neural Networks?
- Step 1 - Convolution Operation
- Step 1(b) - ReLU Layer
- Step 2 - Pooling
- Step 3 - Flattening
- Step 4 - Full Connection
- Summary

- EXTRA: Softmax & Cross-Entropy

CNN (Convolutional Neural Network)



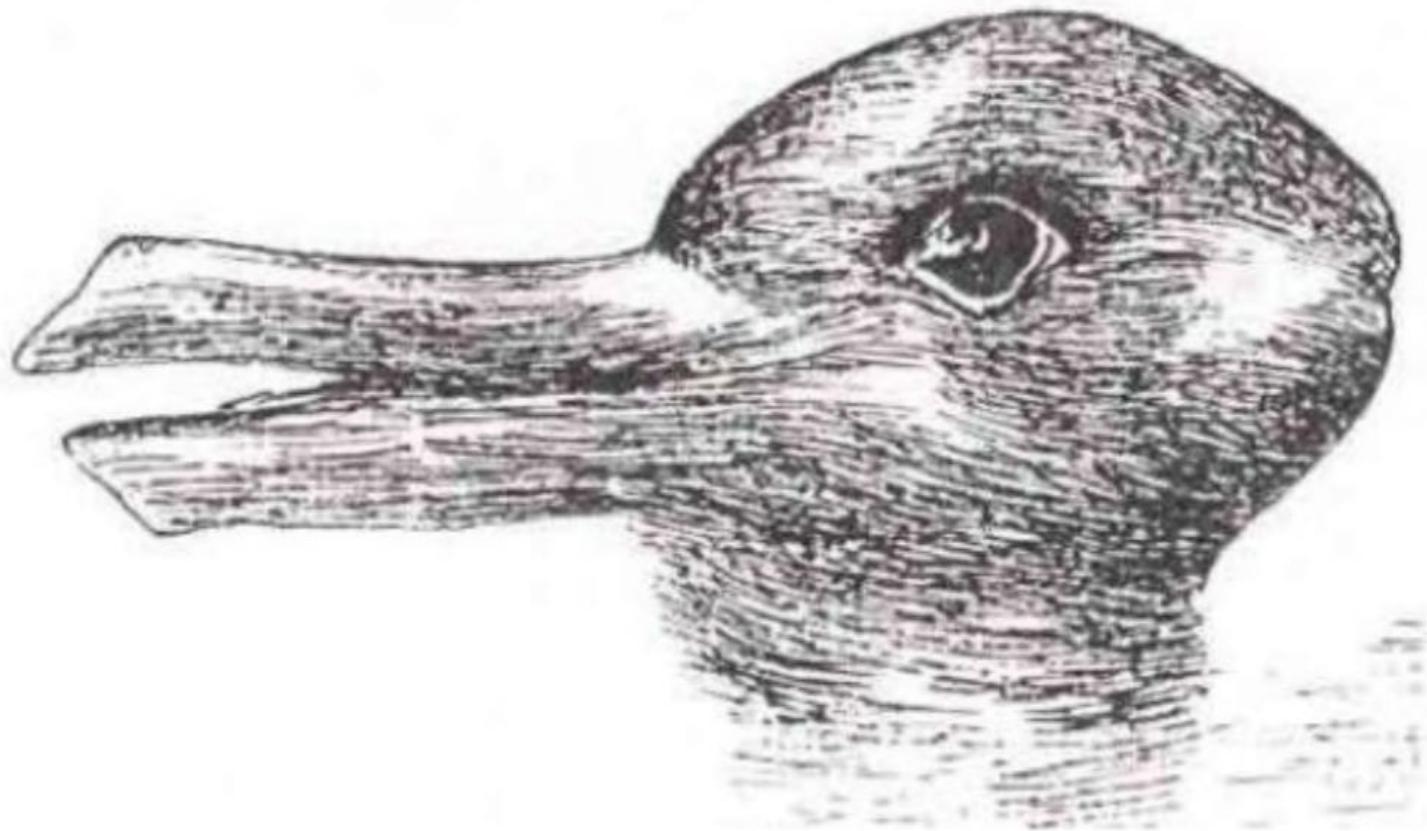


CNN (Convolutional Neural Network)

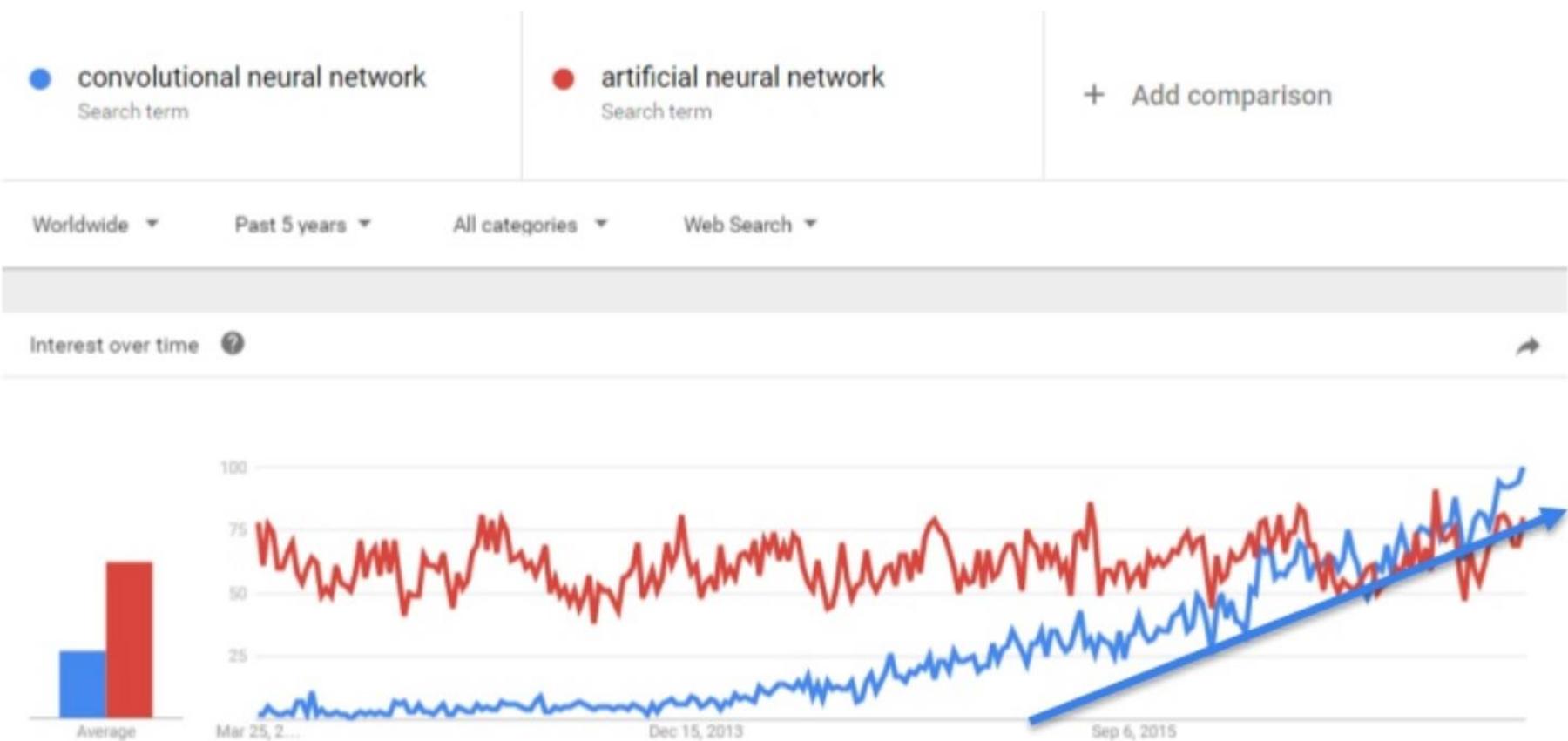




CNN (Convolutional Neural Network)



CNN (Convolutional Neural Network)

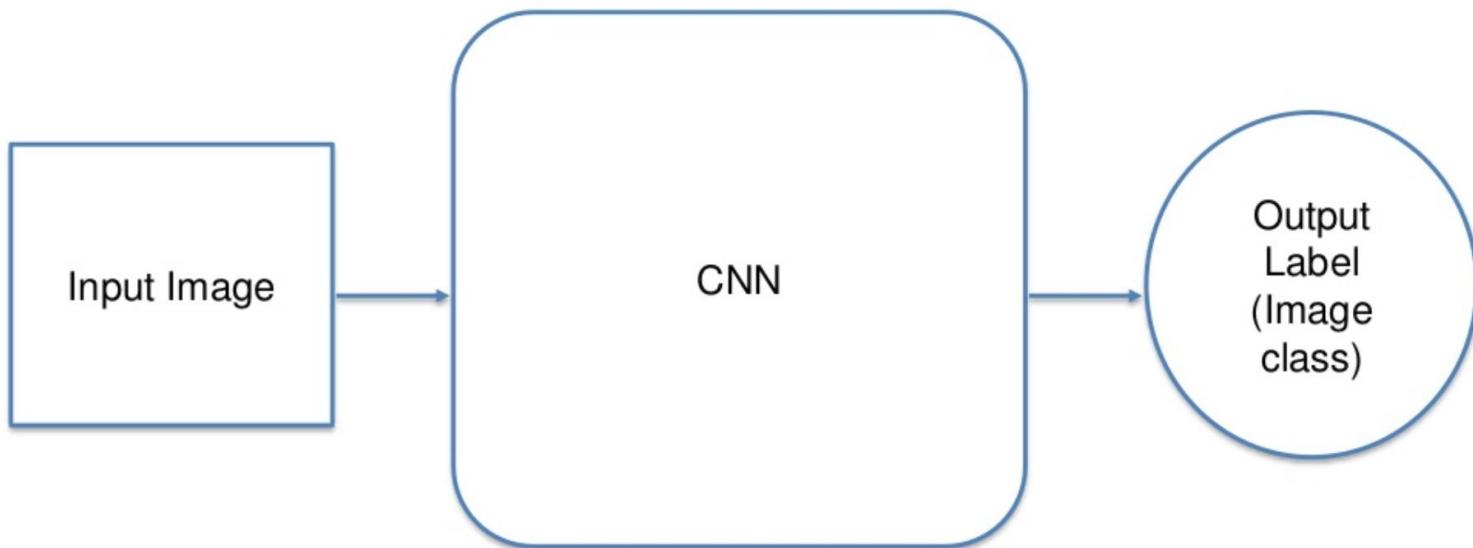


CNN (Convolutional Neural Network)

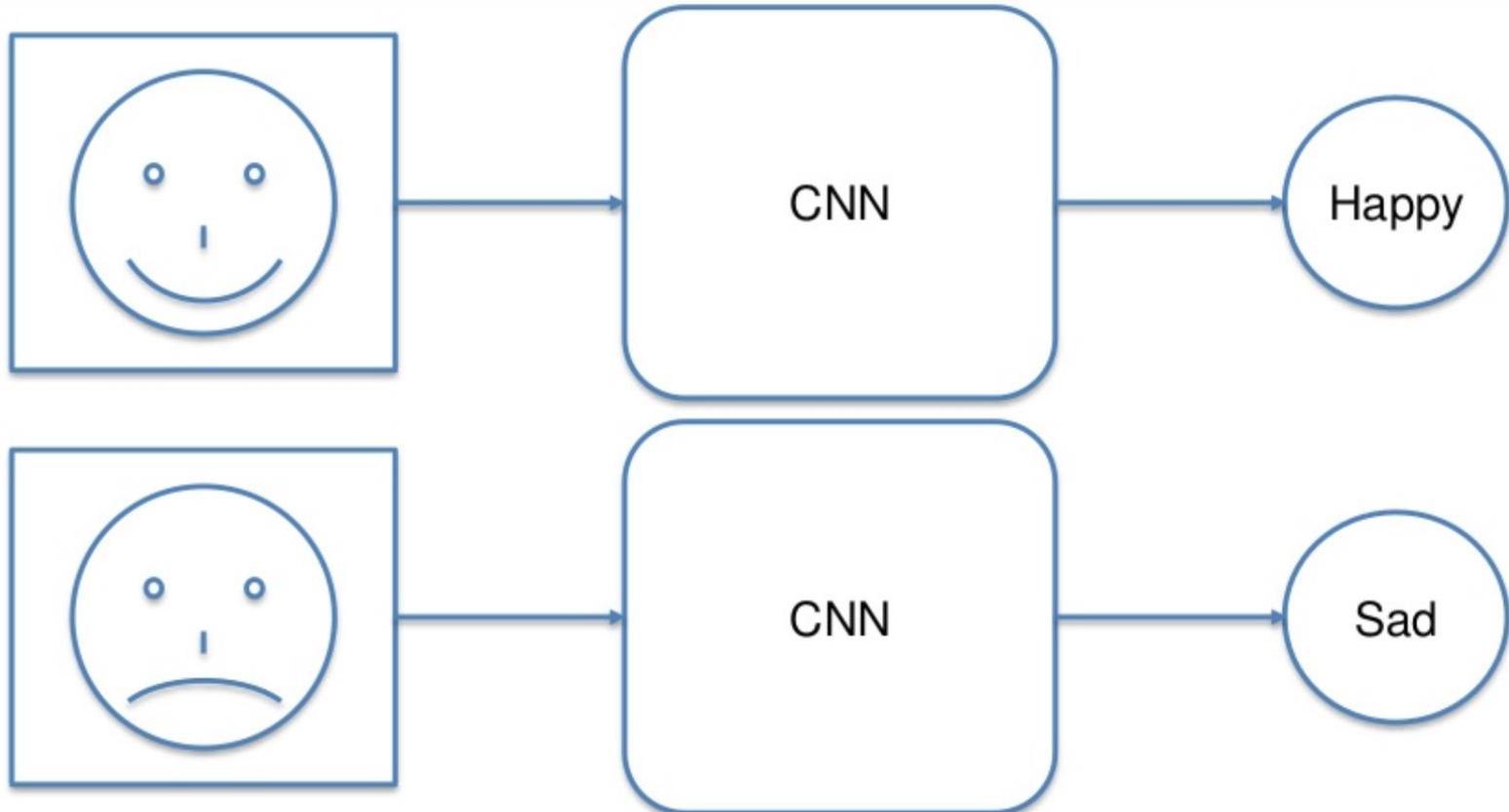


Yann Lecun

CNN (Convolutional Neural Network)



CNN (Convolutional Neural Network)



CNN (Convolutional Neural Network)



B / W Image 2x2px

Pixel 1	Pixel 2
Pixel 3	Pixel 4

2d array

Pixel 1 <small>0 ≤ pixel value ≤ 255</small>	Pixel 2 <small>0 ≤ pixel value ≤ 255</small>
Pixel 3 <small>0 ≤ pixel value ≤ 255</small>	Pixel 4 <small>0 ≤ pixel value ≤ 255</small>

Colored Image 2x2px

Pixel 1	Pixel 2
Pixel 3	Pixel 4

3d array

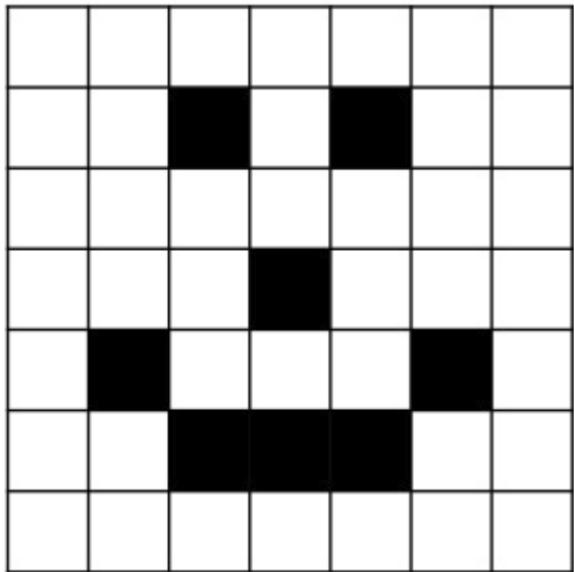
Red channel

Green channel

Blue channel

Pixel 1 <small>0 ≤ pixel value ≤ 255</small>	Pixel 2 <small>0 ≤ pixel value ≤ 255</small>
Pixel 3 <small>0 ≤ pixel value ≤ 255</small>	Pixel 4 <small>0 ≤ pixel value ≤ 255</small>

CNN (Convolutional Neural Network)



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

CNN (Convolutional Neural Network)



STEP 1: Convolution



STEP 2: Max Pooling



STEP 3: Flattening



STEP 4: Full Connection

CNN (Convolutional Neural Network)



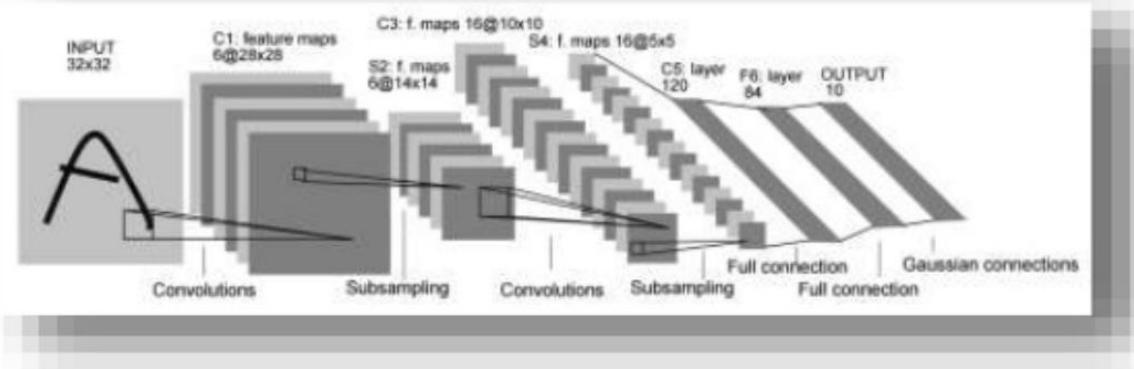
Additional Reading:

*Gradient-Based Learning
Applied to Document
Recognition*

By Yann LeCun et al. (1998)

Link:

<http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf>





CNN : Step 1 - Convolution

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau) g(t - \tau) d\tau$$

CNN : Step 1 - Convolution



Additional Reading:

Introduction to Convolutional Neural Networks

By Jianxin Wu (2017)

$$\begin{aligned}\frac{\partial z}{\partial (\text{vec}(\mathbf{y})^T)}(F^T \otimes I) &= \left((F \otimes I) \frac{\partial z}{\partial \text{vec}(\mathbf{y})} \right)^T \\ &= \left((F \otimes I) \text{vec} \left(\frac{\partial z}{\partial Y} \right) \right)^T \\ &= \text{vec} \left(I \frac{\partial z}{\partial Y} F^T \right)^T \\ &= \text{vec} \left(\frac{\partial z}{\partial Y} F^T \right)^T,\end{aligned}$$

Link:

<http://cs.nju.edu.cn/wujx/paper/CNN.pdf>

CNN : Step 1 - Convolution



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

Input Image

0	0	1
1	0	0
0	1	1

Feature Detector

CNN : Step 1 - Convolution



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



0	0	1
1	0	0
0	1	1



0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	

Input Image

Feature Detector

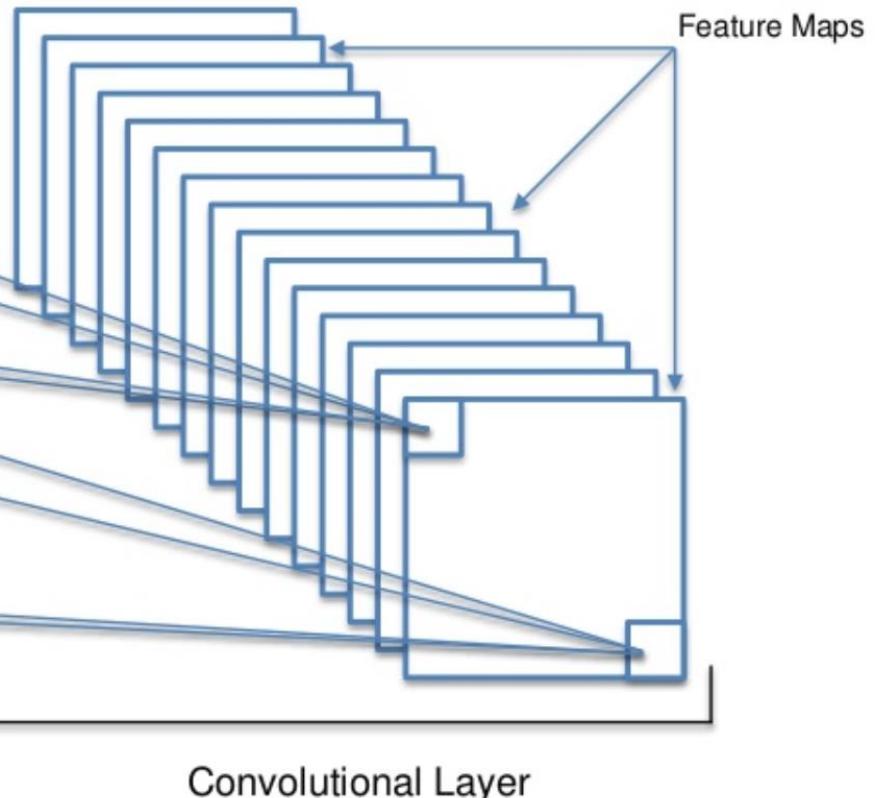
Feature Map

CNN : Step 1 - Convolution



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

We create many feature maps to obtain our first convolution layer



CNN : Step 1 - Convolution



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



0	0	1
1	0	0
0	1	1



0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	1

Input Image

Feature Detector

Feature Map

CNN : Step 1 - Convolution

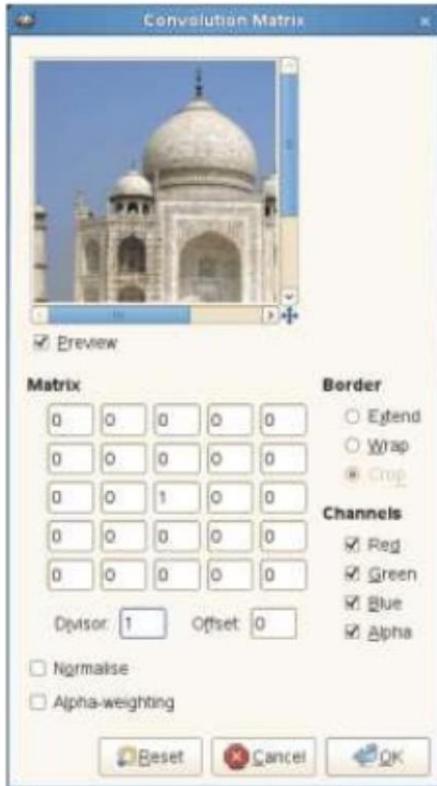


Image Source: docs.gimp.org/en/plug-in-convmatrix.html

CNN : Step 1 - Convolution



Sharpen:

0	0	0	0	0
0	0	-1	0	0
0	-1	5	-1	0
0	0	-1	0	0
0	0	0	0	0



Image Source: docs.gimp.org/en/plug-in-convmatrix.html

CNN : Step 1 - Convolution



Blur:

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



Image Source: docs.gimp.org/en/plug-in-convmatrix.html

CNN : Step 1 - Convolution



Edge Enhance:

$$\begin{matrix} 0 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}$$



Image Source: docs.gimp.org/en/plug-in-convmatrix.html

CNN : Step 1 - Convolution



Emboss:

$$\begin{array}{|c|c|c|} \hline & -2 & -1 & 0 \\ \hline -1 & 1 & 1 & \\ \hline 0 & 1 & 2 & \\ \hline \end{array}$$



Image Source: docs.gimp.org/en/plug-in-convmatrix.html

CNN : Step 1 - Convolution



*

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



Image Source: eonardoaraujosantos.gitbooks.io

CNN : Step 1 – ReLU Layer



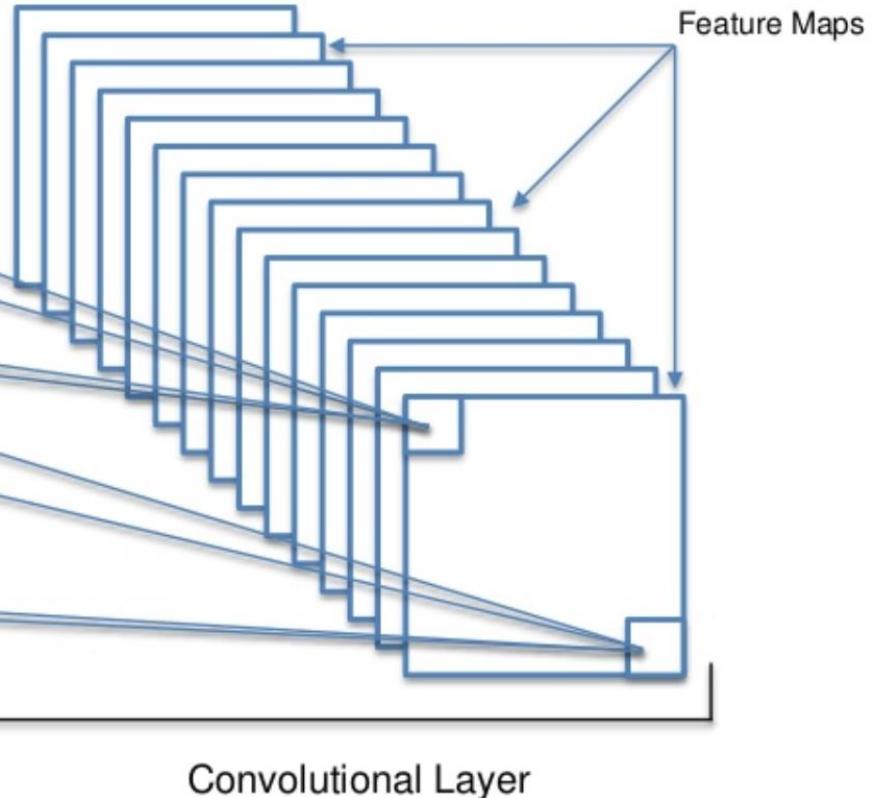
Step 1(B) – ReLU Layer

CNN : Step 1 – ReLU Layer

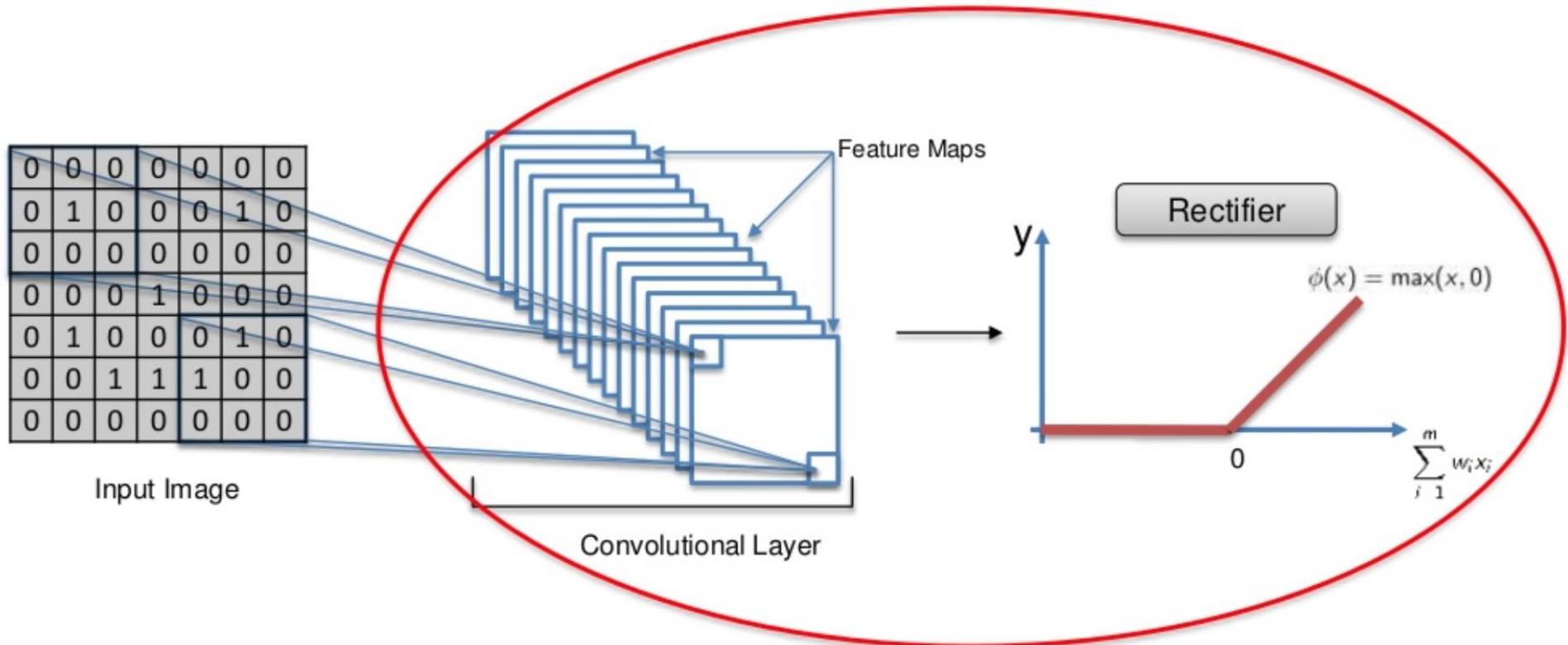


0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	1	0	0	0	1	0	0
0	0	1	1	1	0	0	0
0	0	0	0	0	0	0	0

We create many feature maps to obtain our first convolution layer



CNN : Step 1 – ReLU Layer



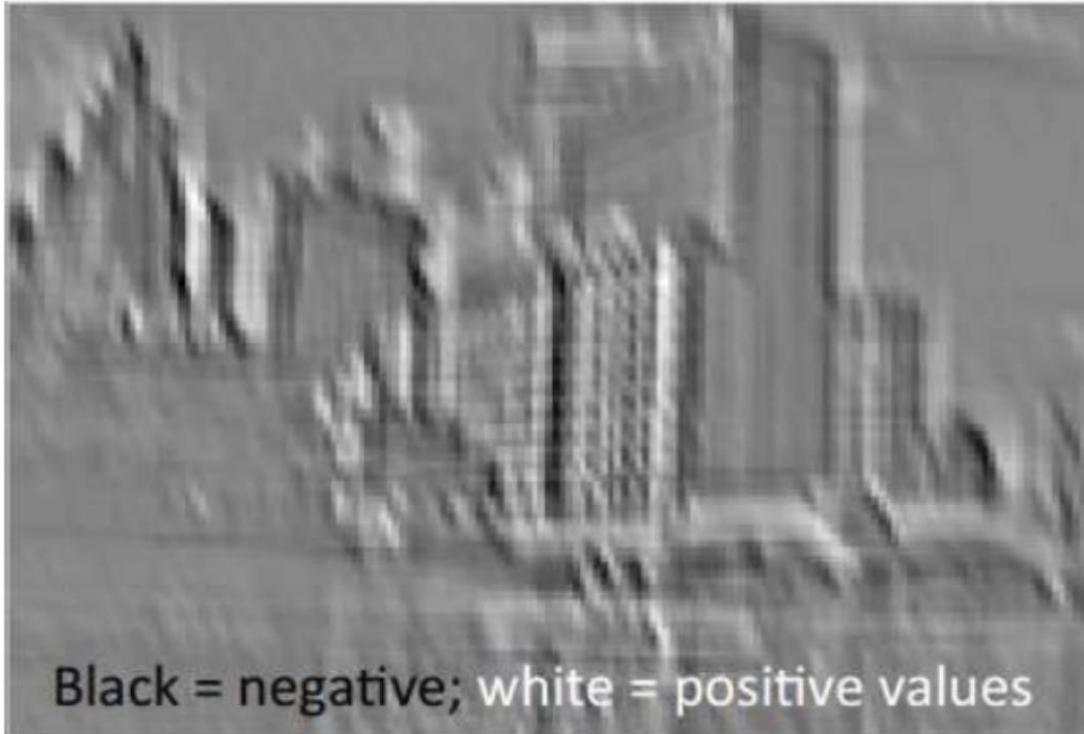
CNN : Step 1 – ReLU Layer



Image Source: http://mlss.tuebingen.mpg.de/2015/slides/fergus/Fergus_1.pdf



CNN : Step 1 – ReLU Layer



Black = negative; white = positive values

Image Source: http://mlss.tuebingen.mpg.de/2015/slides/fergus/Fergus_1.pdf



CNN : Step 1 – ReLU Layer



Image Source: http://mlss.tuebingen.mpg.de/2015/slides/fergus/Fergus_1.pdf



CNN : Step 1 – ReLU Layer



Image Source: http://mlss.tuebingen.mpg.de/2015/slides/fergus/Fergus_1.pdf

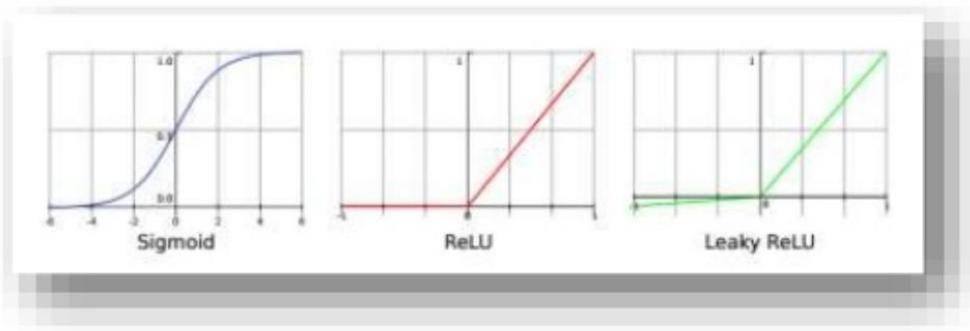
CNN : Step 1 – ReLU Layer



Additional Reading:

Understanding Convolutional Neural Networks with A Mathematical Model

By C.-C. Jay Kuo (2016)



Link:

<https://arxiv.org/pdf/1609.04112.pdf>

CNN : Step 1 – ReLU Layer



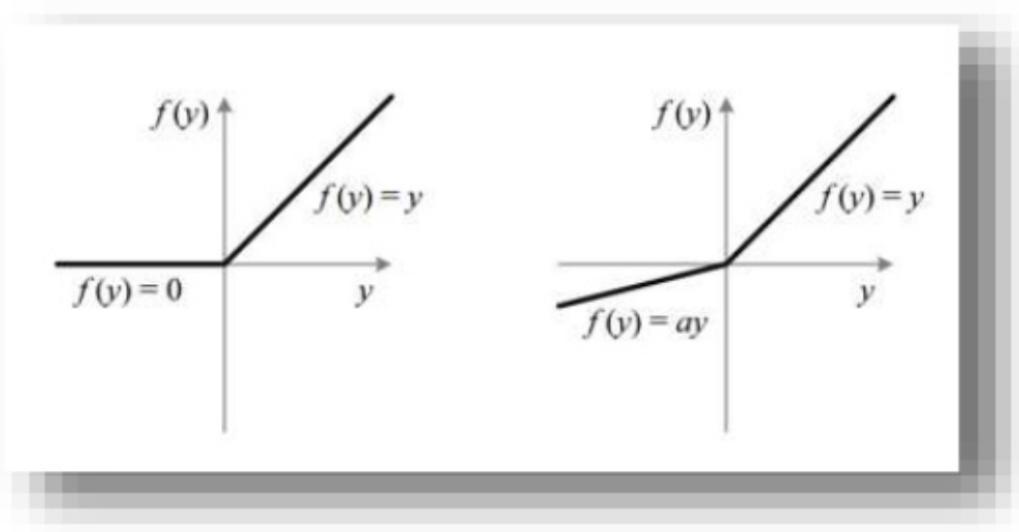
Additional Reading:

*Delving Deep into Rectifiers:
Surpassing Human-Level
Performance on ImageNet
Classification*

By Kaiming He et al. (2015)

Link:

<https://arxiv.org/pdf/1502.01852.pdf>



CNN : Step 2 – Max Pooling



Step 2 – Max Pooling



CNN : Step 2 – Max Pooling



Image Source: Wikipedia

CNN : Step 2 – Max Pooling

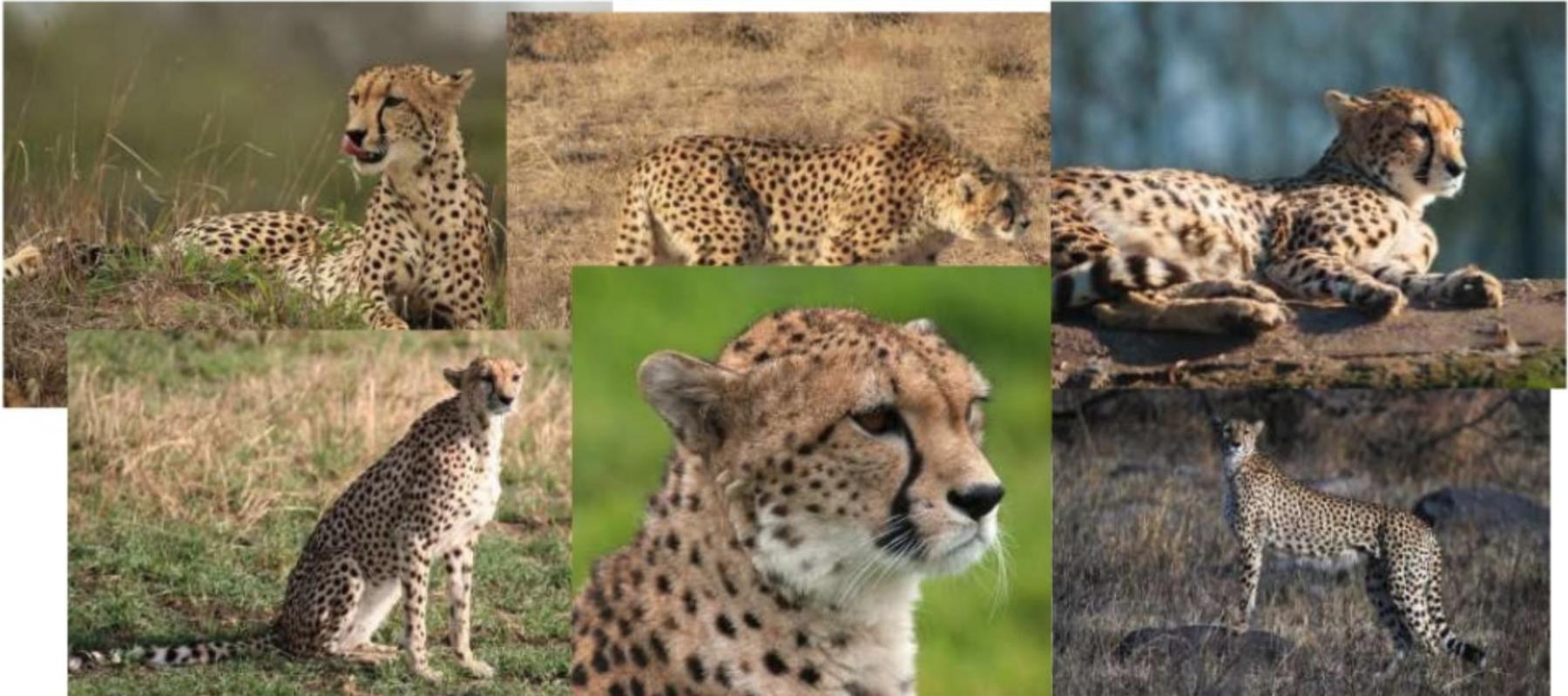


Image Source: Wikipedia

CNN : Step 2 – Max Pooling



0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	1

Feature Map

CNN : Step 2 – Max Pooling



0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	1

Feature Map

Max Pooling



Pooled Feature Map

CNN : Step 2 – Max Pooling



0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	1

Feature Map

Max Pooling



1		

Pooled Feature Map

CNN : Step 2 – Max Pooling



0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	1

Feature Map

Max Pooling



1	1	

Pooled Feature Map

CNN : Step 2 – Max Pooling



0	1	0	0	0	
0	1	1	1	0	
1	0	1	2	1	
1	4	2	1	0	
0	0	1	2	1	

Feature Map

Max Pooling

1	1	0

Pooled Feature Map

CNN : Step 2 – Max Pooling



0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	1

Feature Map

Max Pooling



1	1	0
4		

Pooled Feature Map

CNN : Step 2 – Max Pooling



0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	1

Feature Map

Max Pooling



1	1	0
4	2	

Pooled Feature Map

CNN : Step 2 – Max Pooling



0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	1

Feature Map

Max Pooling



1	1	0
4	2	1

Pooled Feature Map

CNN : Step 2 – Max Pooling



0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	1

Feature Map

Max Pooling



1	1	0
4	2	1
0		

Pooled Feature Map

CNN : Step 2 – Max Pooling



0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	1

Feature Map

Max Pooling



1	1	0
4	2	1
0	2	

Pooled Feature Map

CNN : Step 2 – Max Pooling



0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	1

Feature Map

Max Pooling



1	1	0
4	2	1
0	2	1

Pooled Feature Map

CNN : Step 2 – Max Pooling



Additional Reading:

Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition

By Dominik Scherer et al. (2010)

Link:

http://ais.uni-bonn.de/papers/icann2010_maxpool.pdf

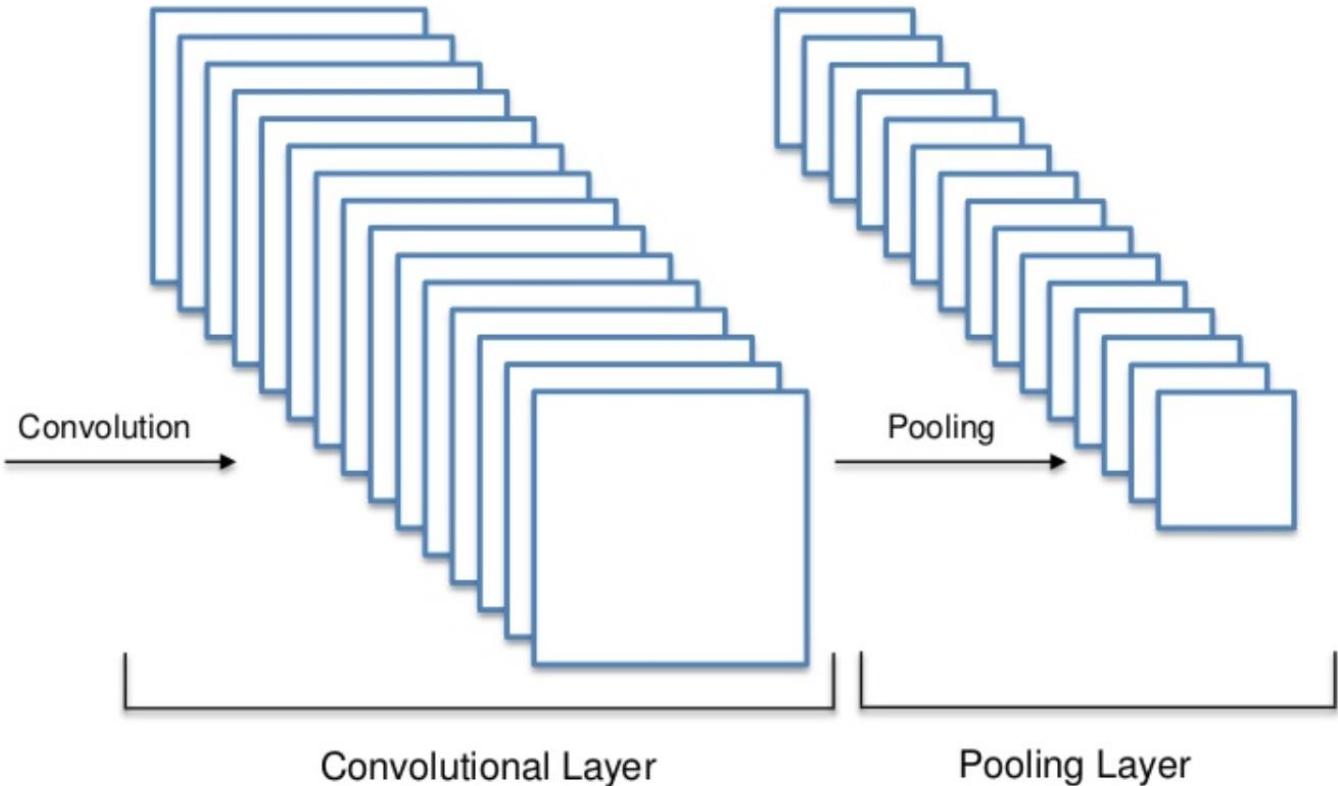


CNN : Step 2 – Max Pooling



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

Input Image



CNN : Step 2 – Max Pooling

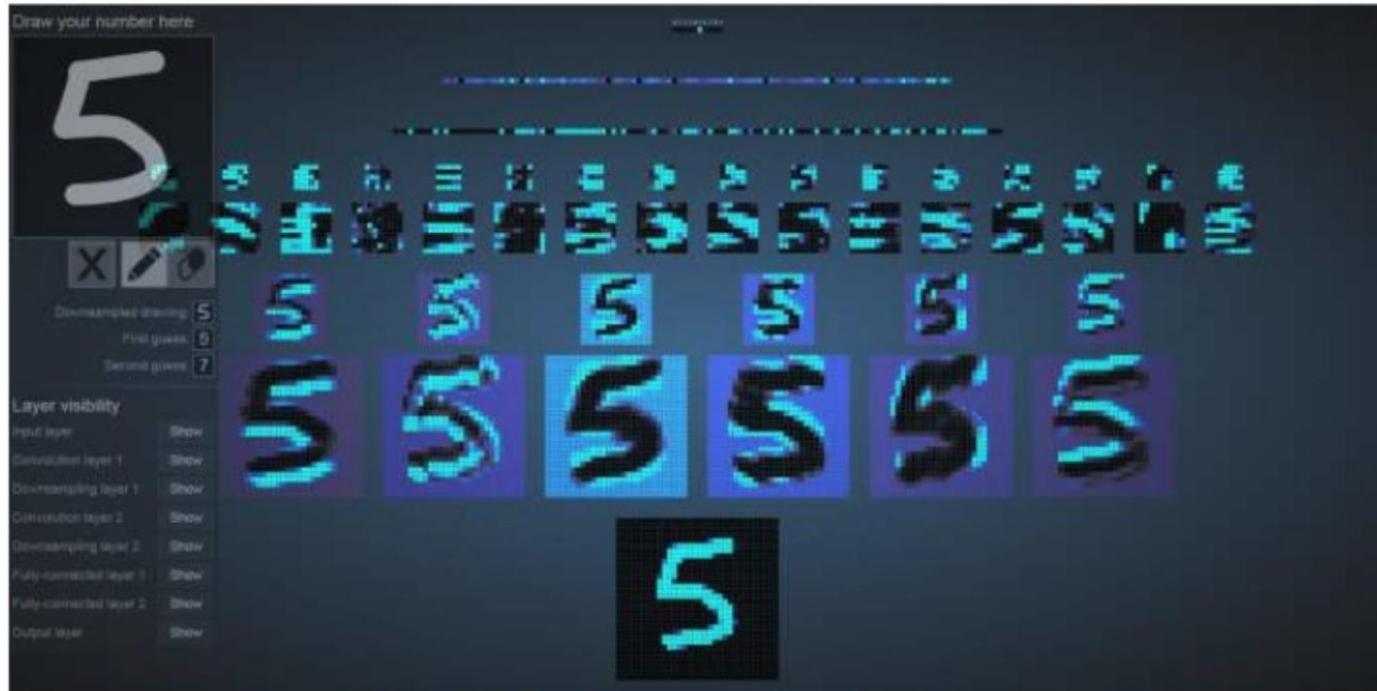


Image Source: scs.ryerson.ca/~aharley/vis/conv/flat.html

CNN : Step 3 – Flattening



Step 3 – Flattening

CNN : Step 3 – Flattening



1	1	0
4	2	1
0	2	1

Pooled Feature Map

CNN : Step 3 – Flattening



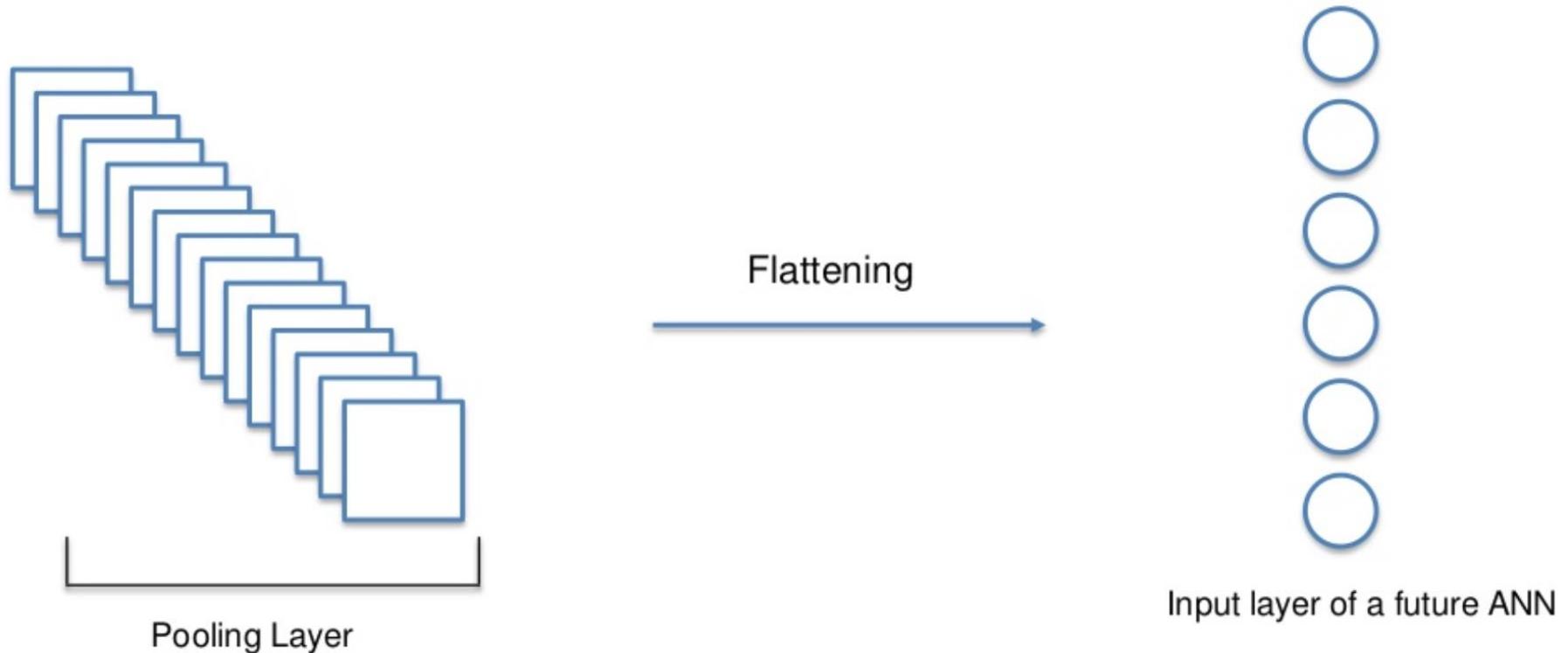
1	1	0
4	2	1
0	2	1

Pooled Feature Map

Flattening

1
1
0
4
2
1
0
2
1

CNN : Step 3 – Flattening

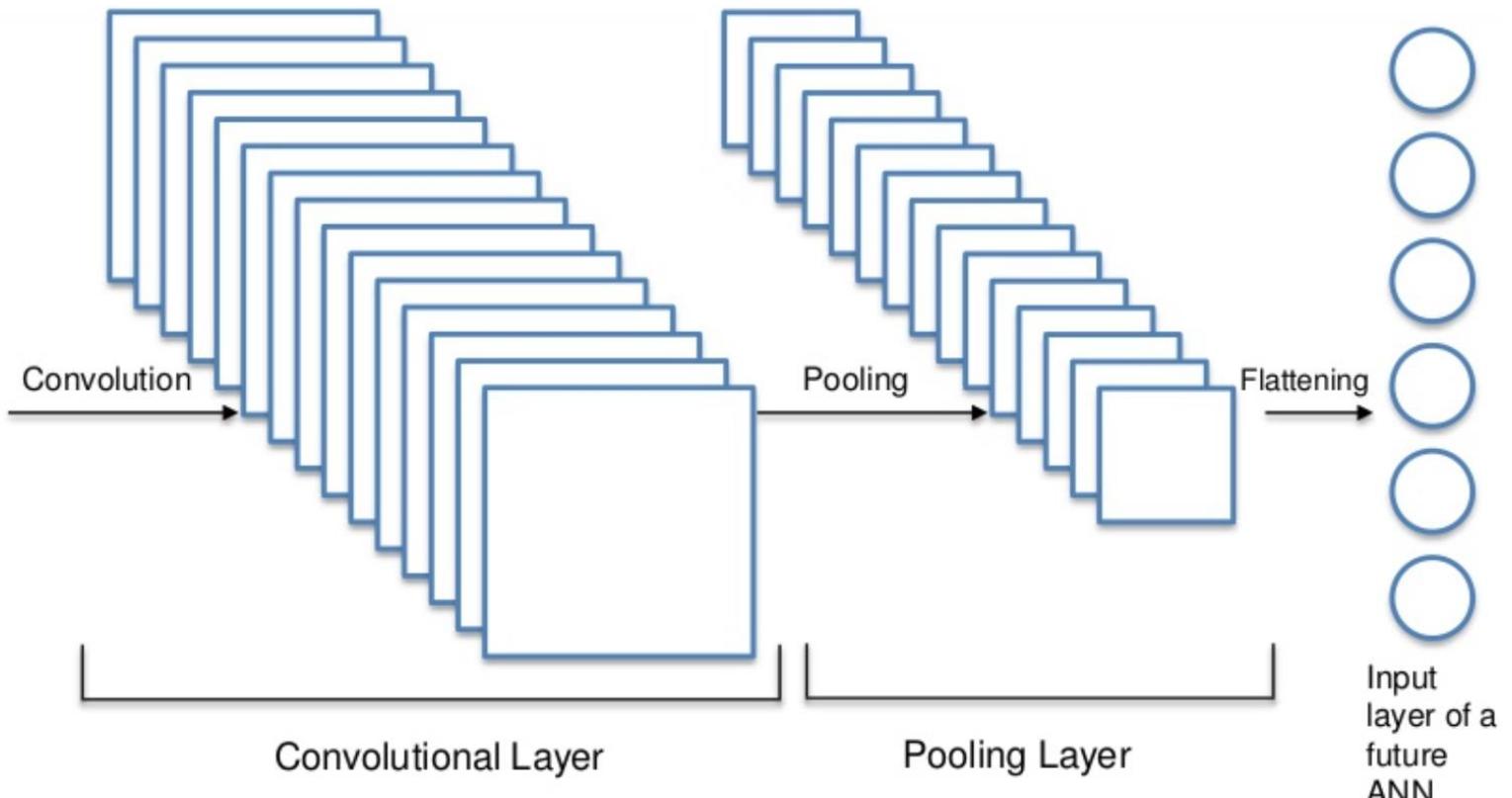


CNN : Step 3 – Flattening



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

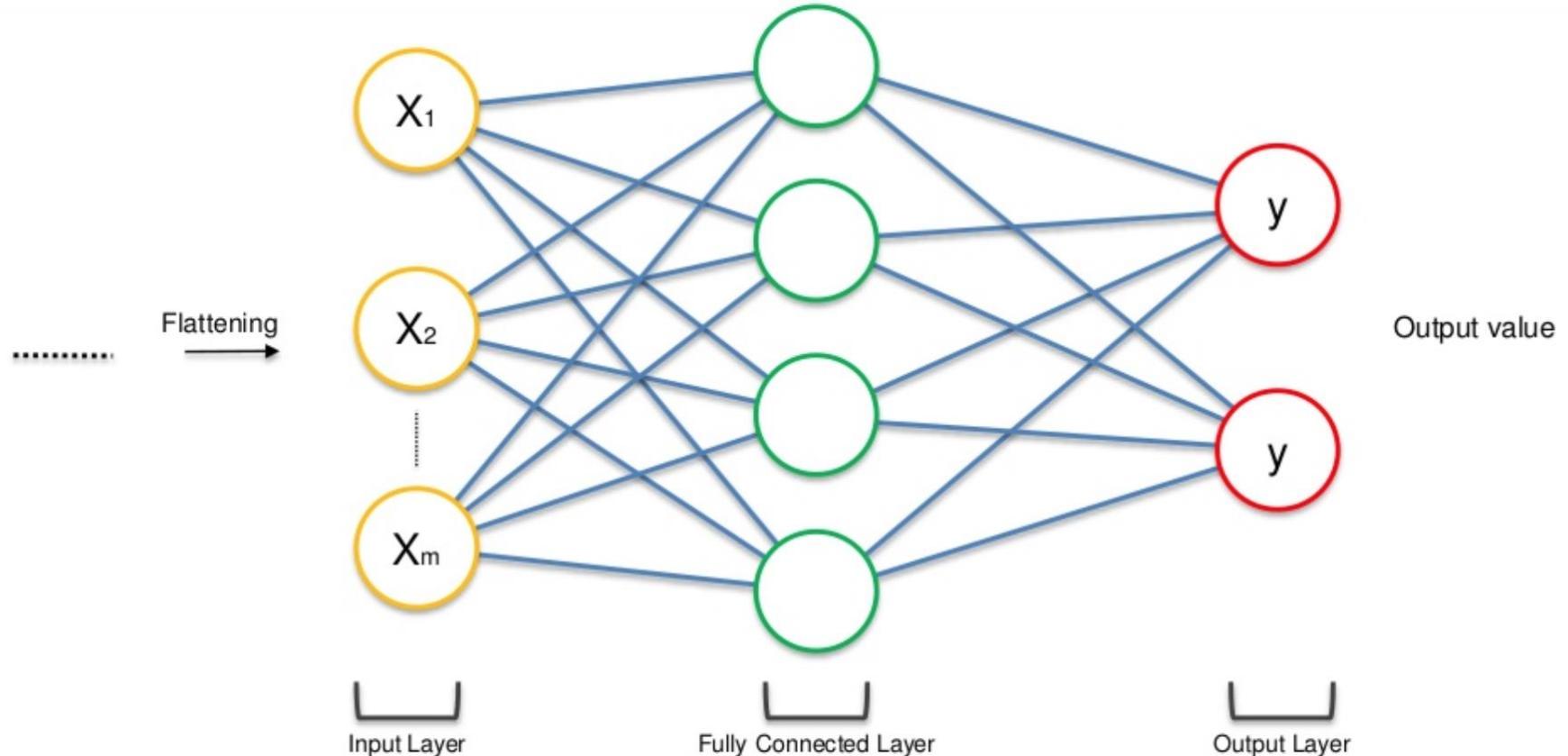
Input Image



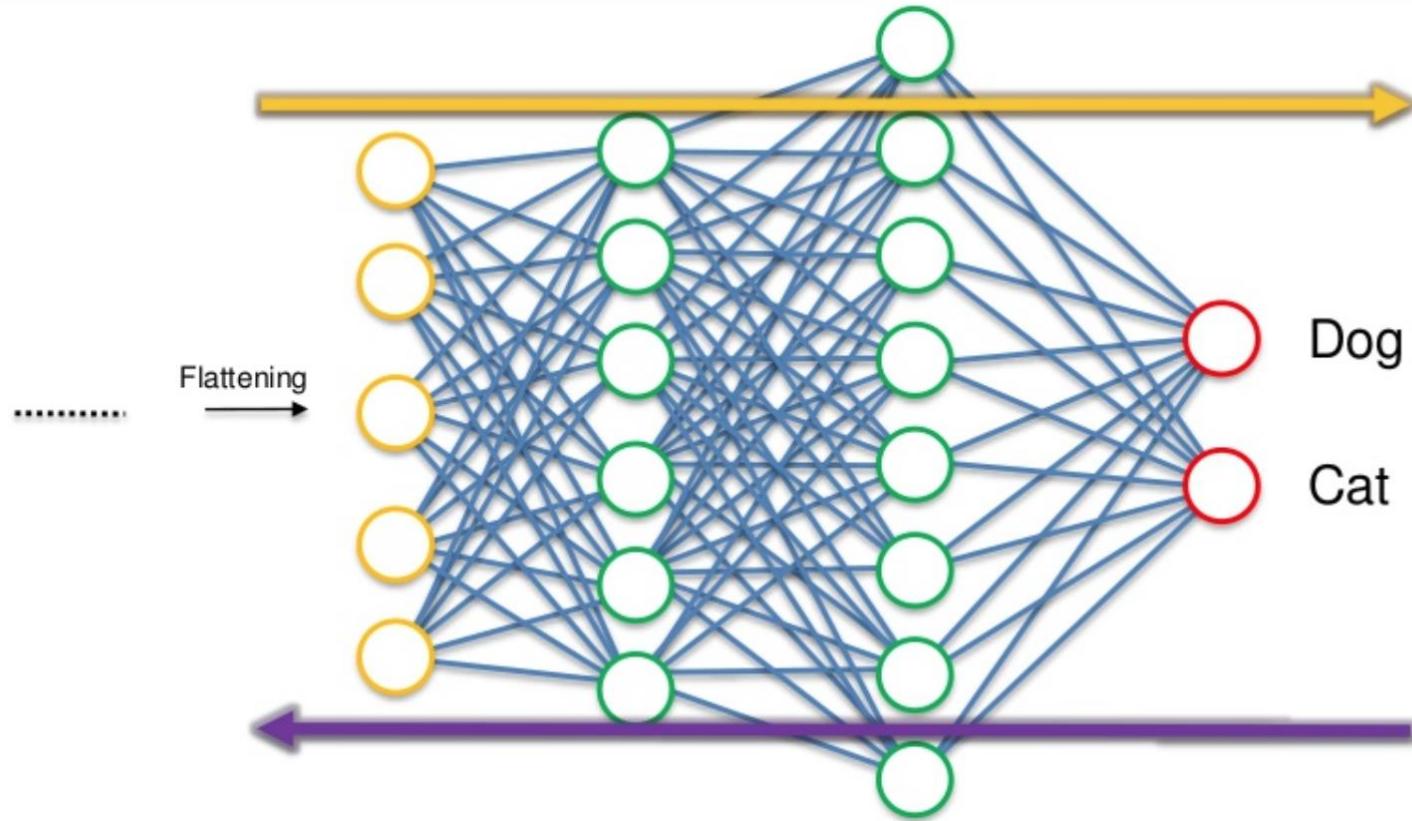


Step 4 – Full Connection

CNN : Step 4 – Full Connection

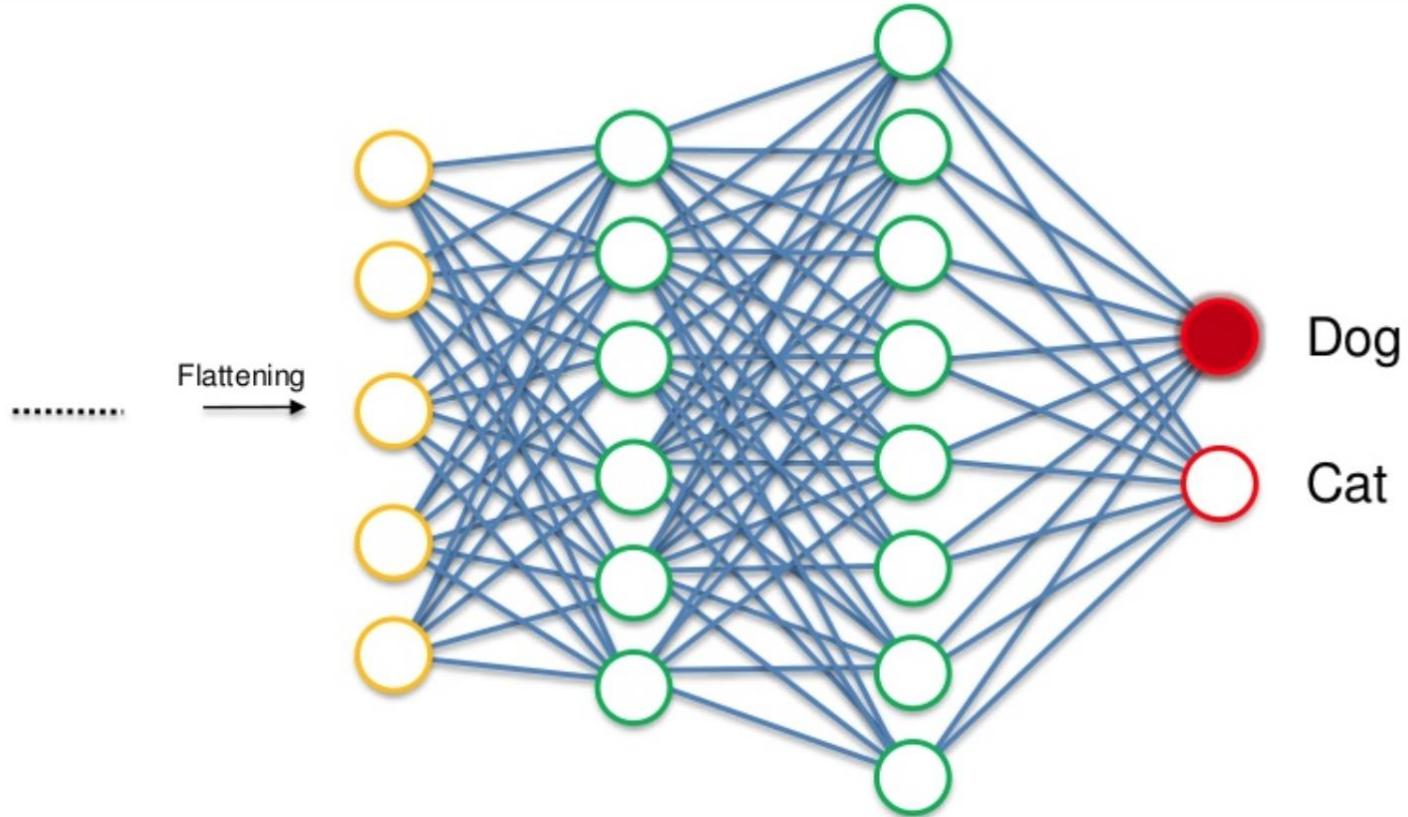


CNN : Step 4 – Full Connection



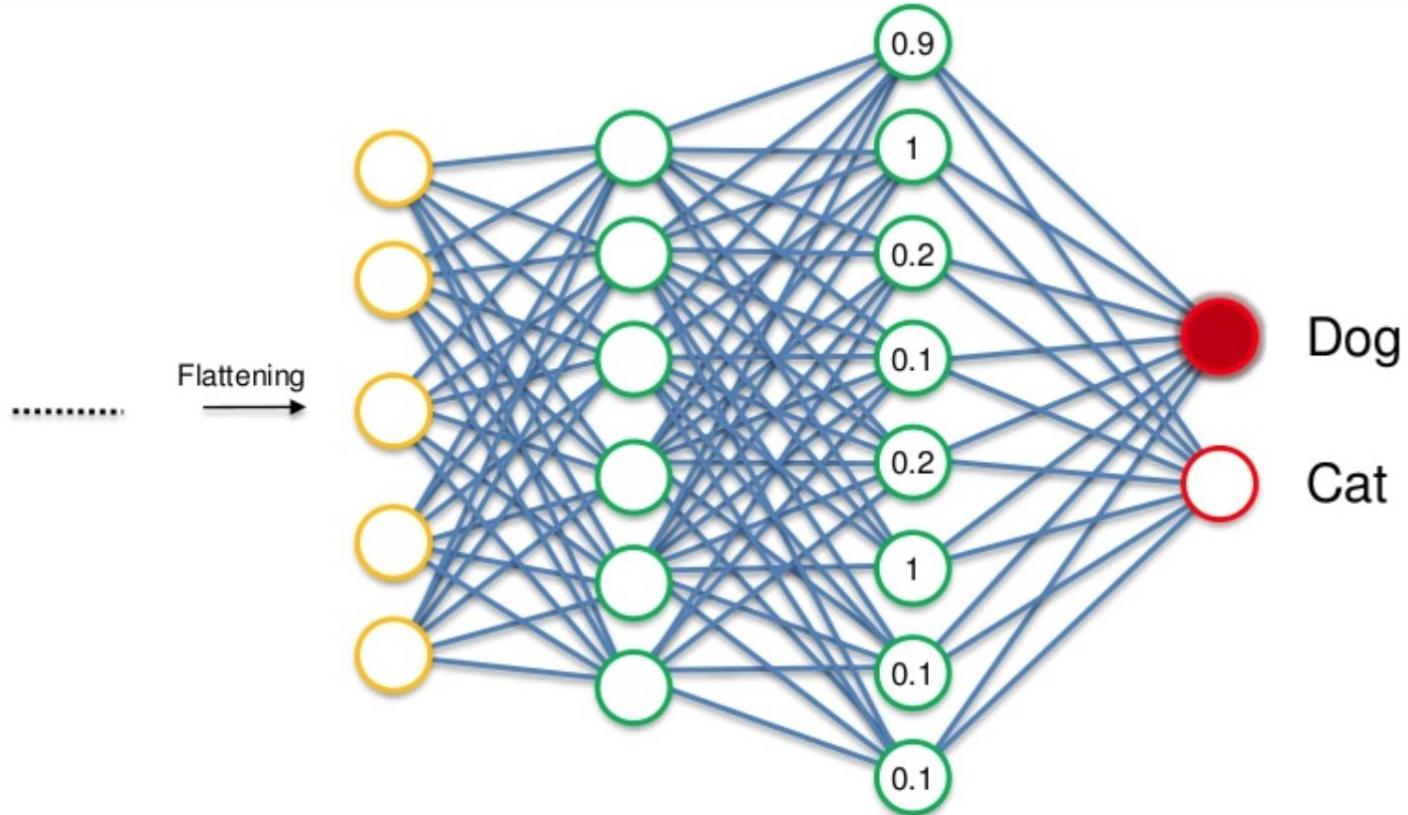


CNN : Step 4 – Full Connection



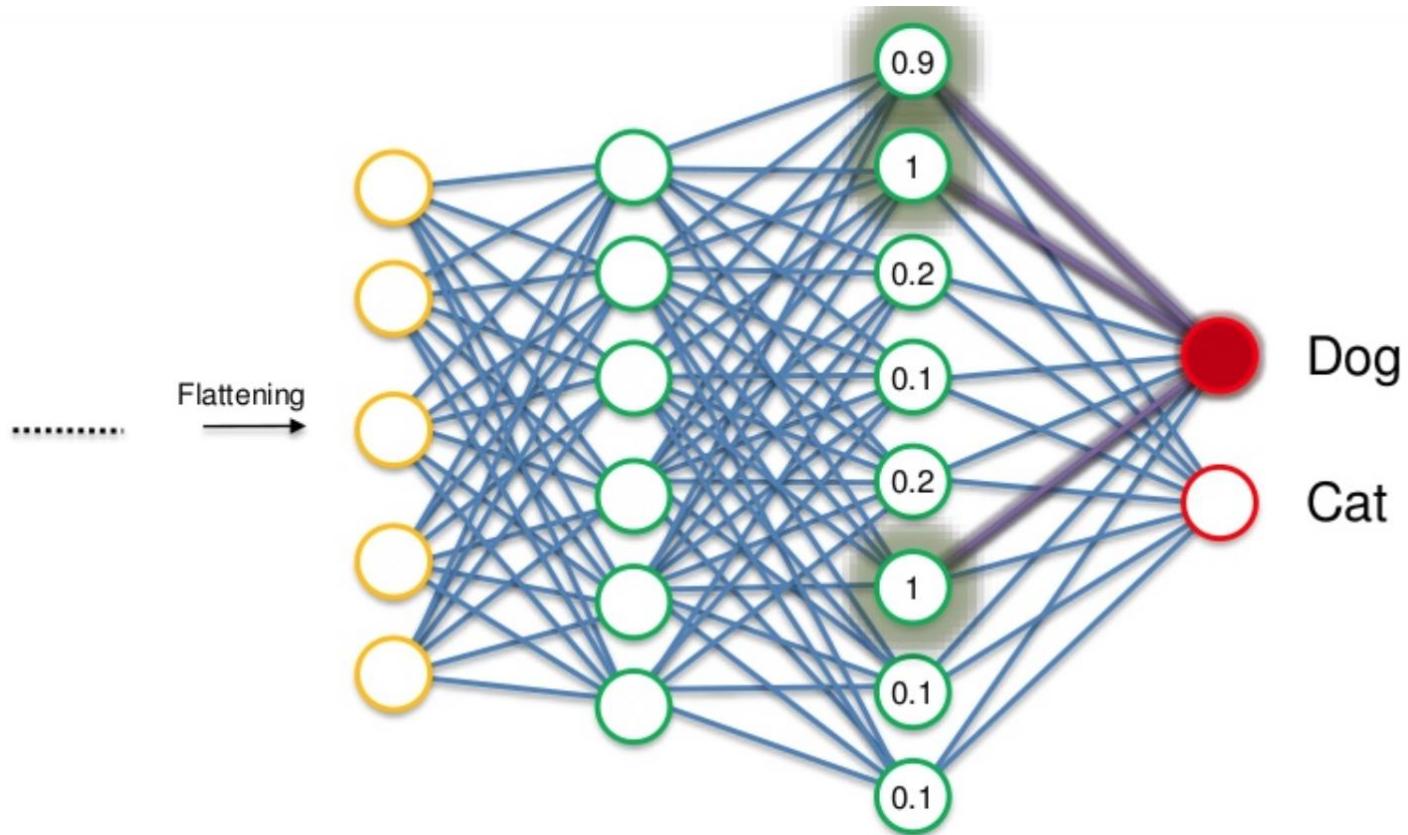


CNN : Step 4 – Full Connection



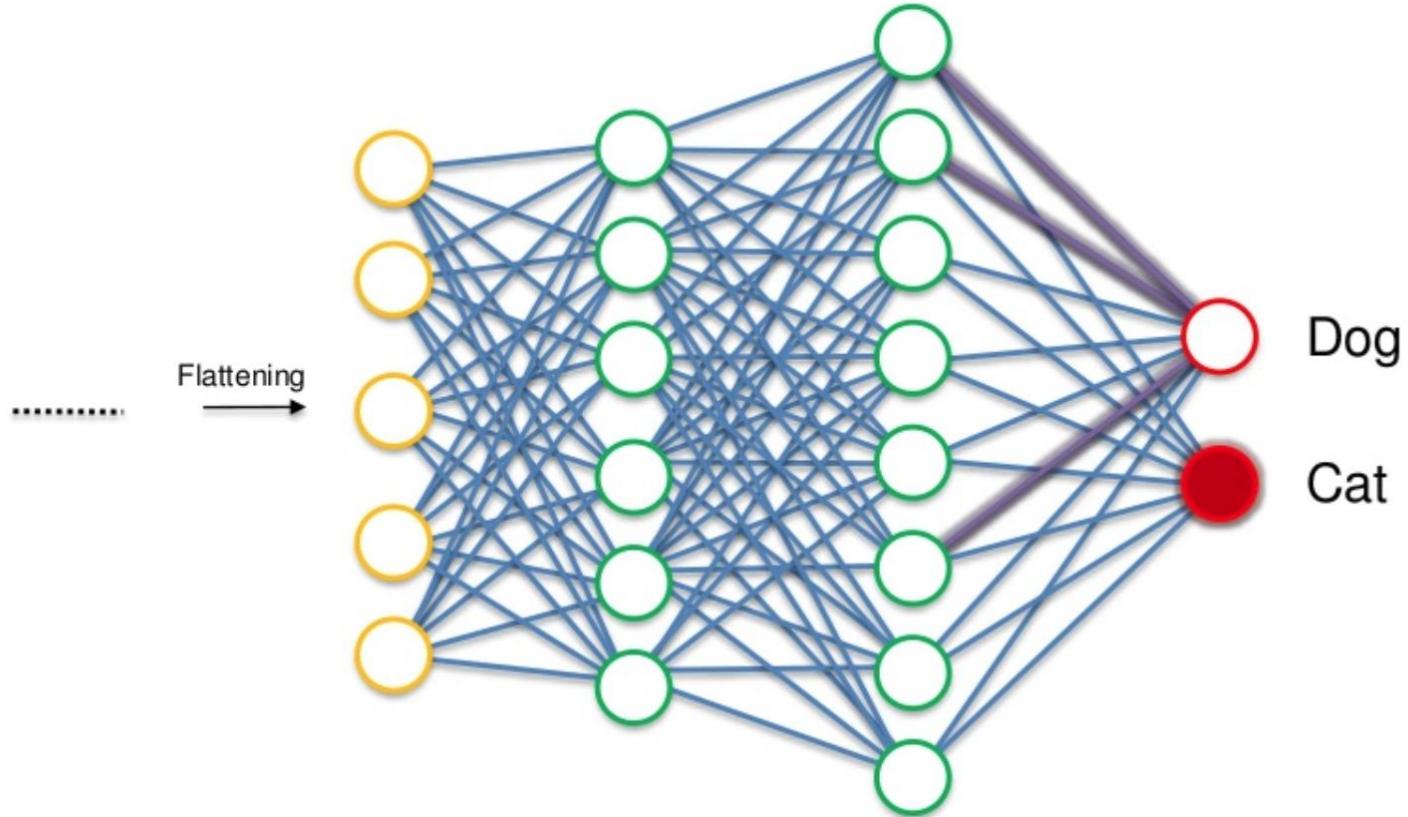


CNN : Step 4 – Full Connection



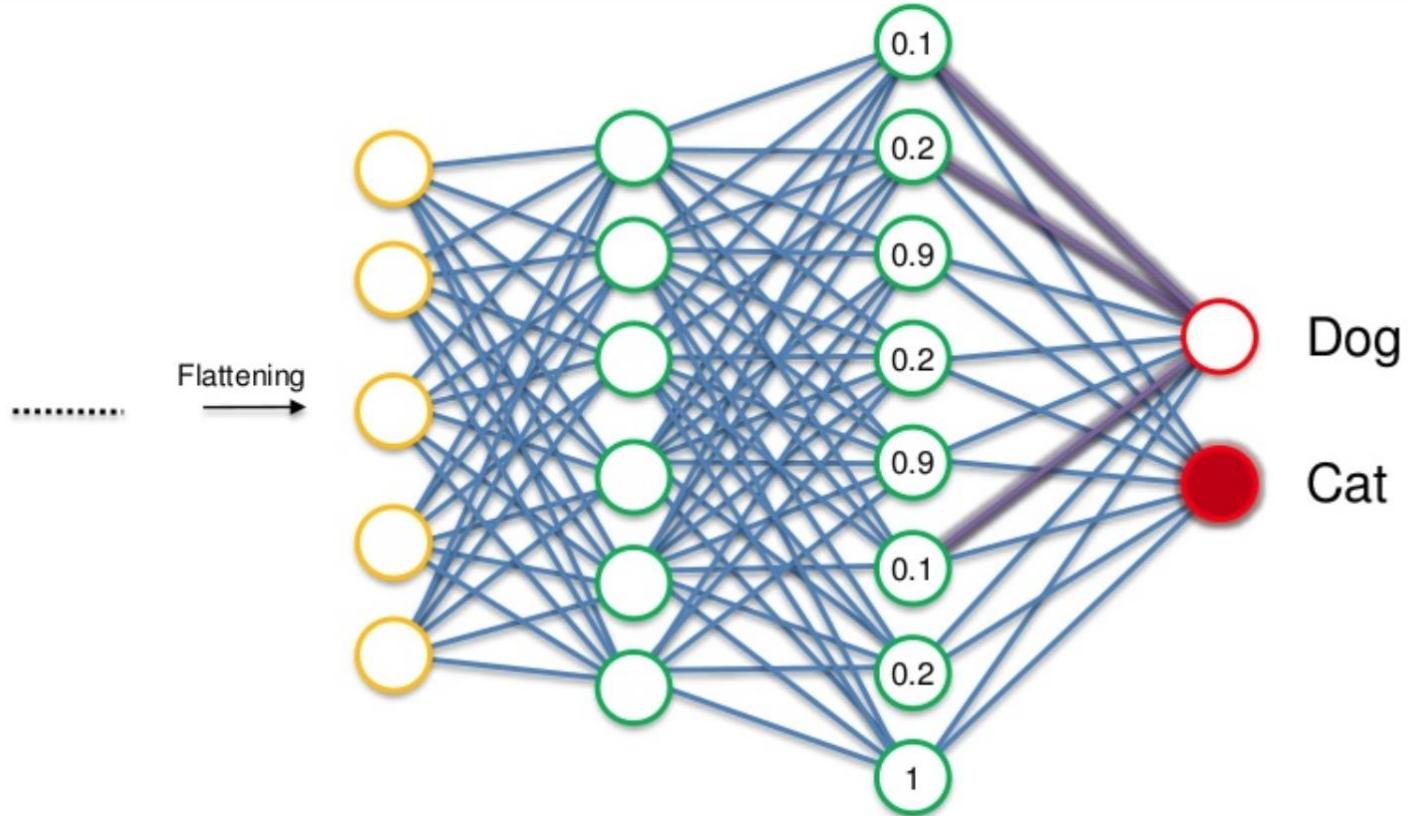


CNN : Step 4 – Full Connection

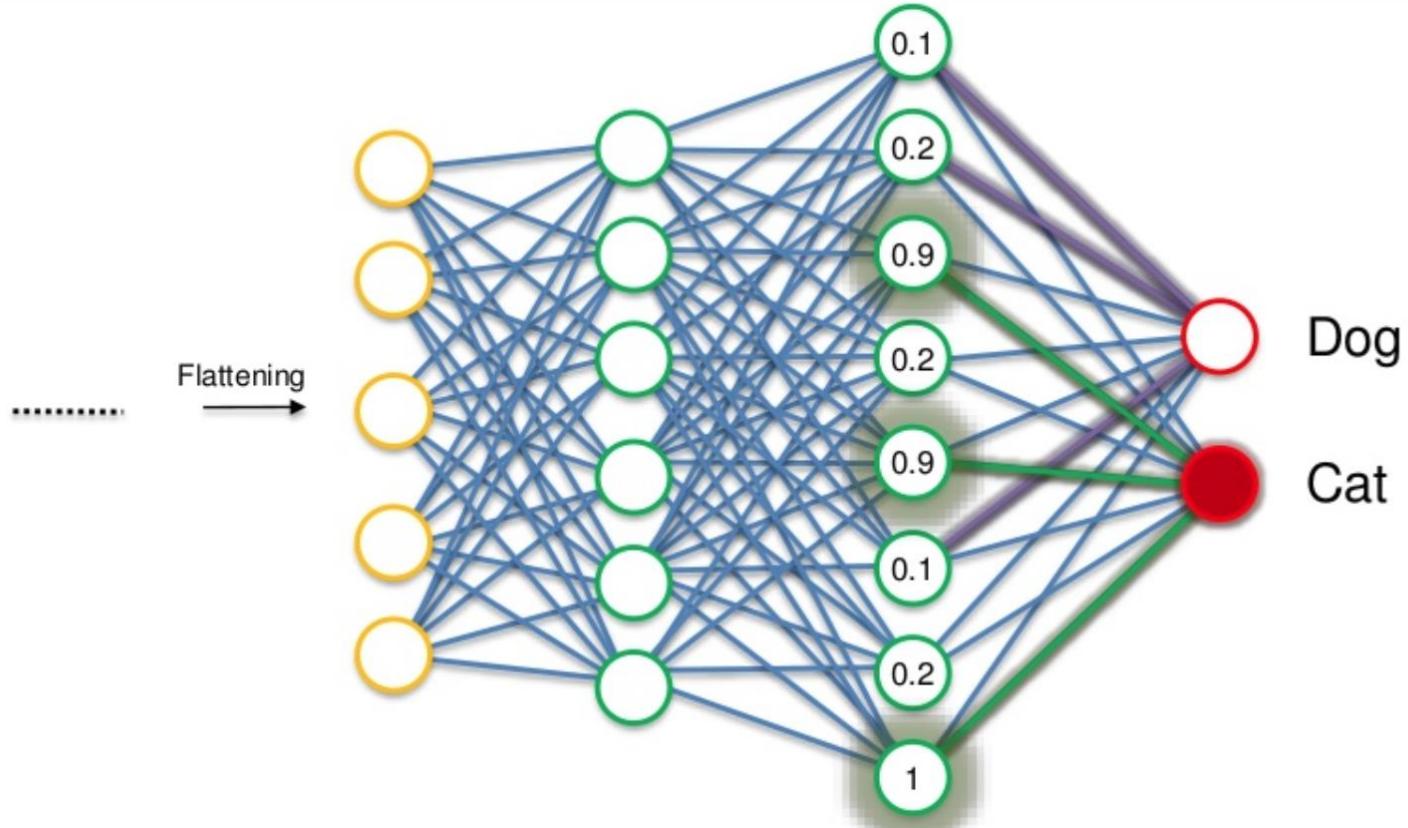




CNN : Step 4 – Full Connection



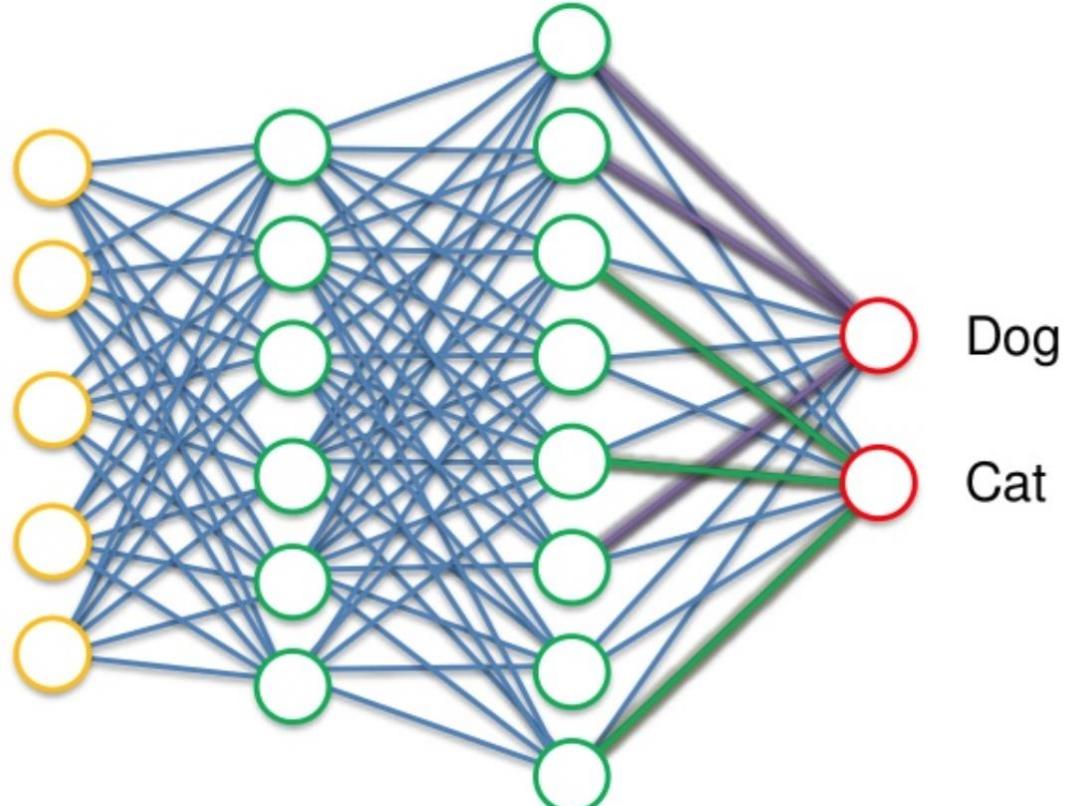
CNN : Step 4 – Full Connection



CNN : Step 4 – Full Connection



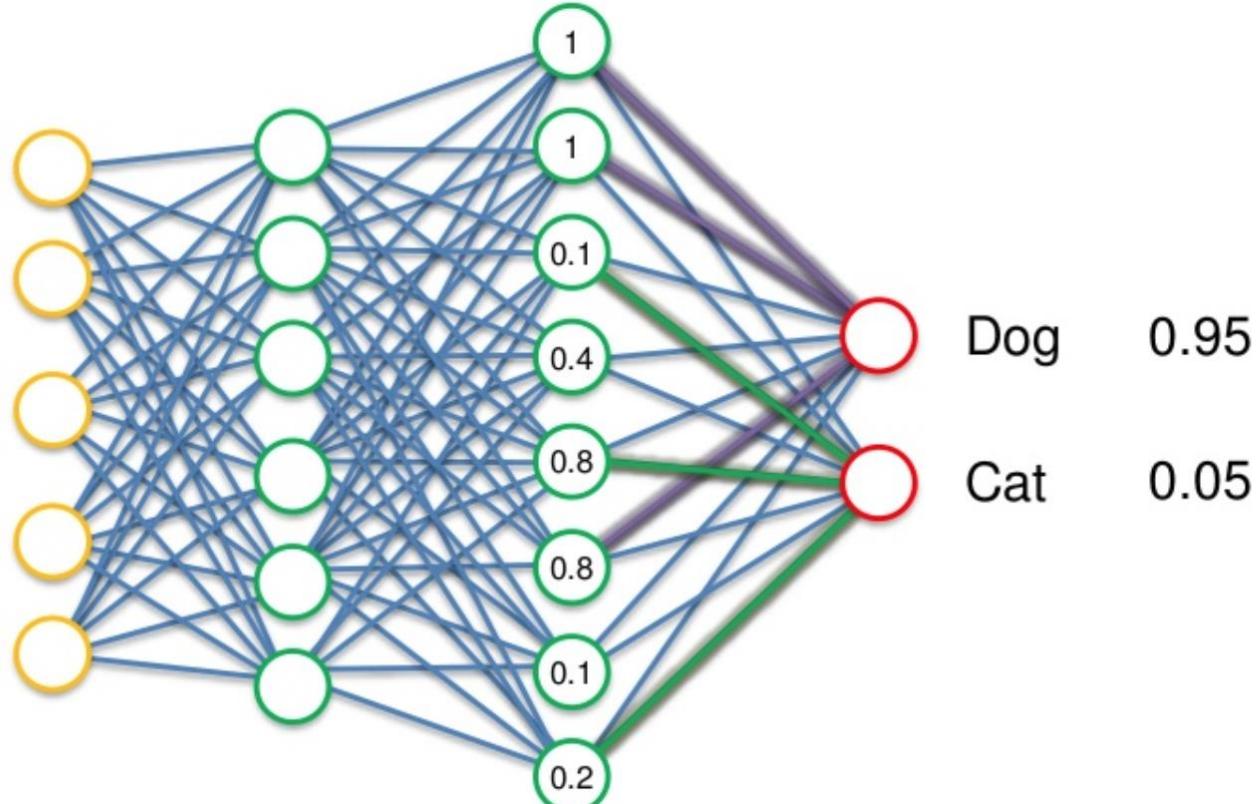
Flattening



CNN : Step 4 – Full Connection



Flattening

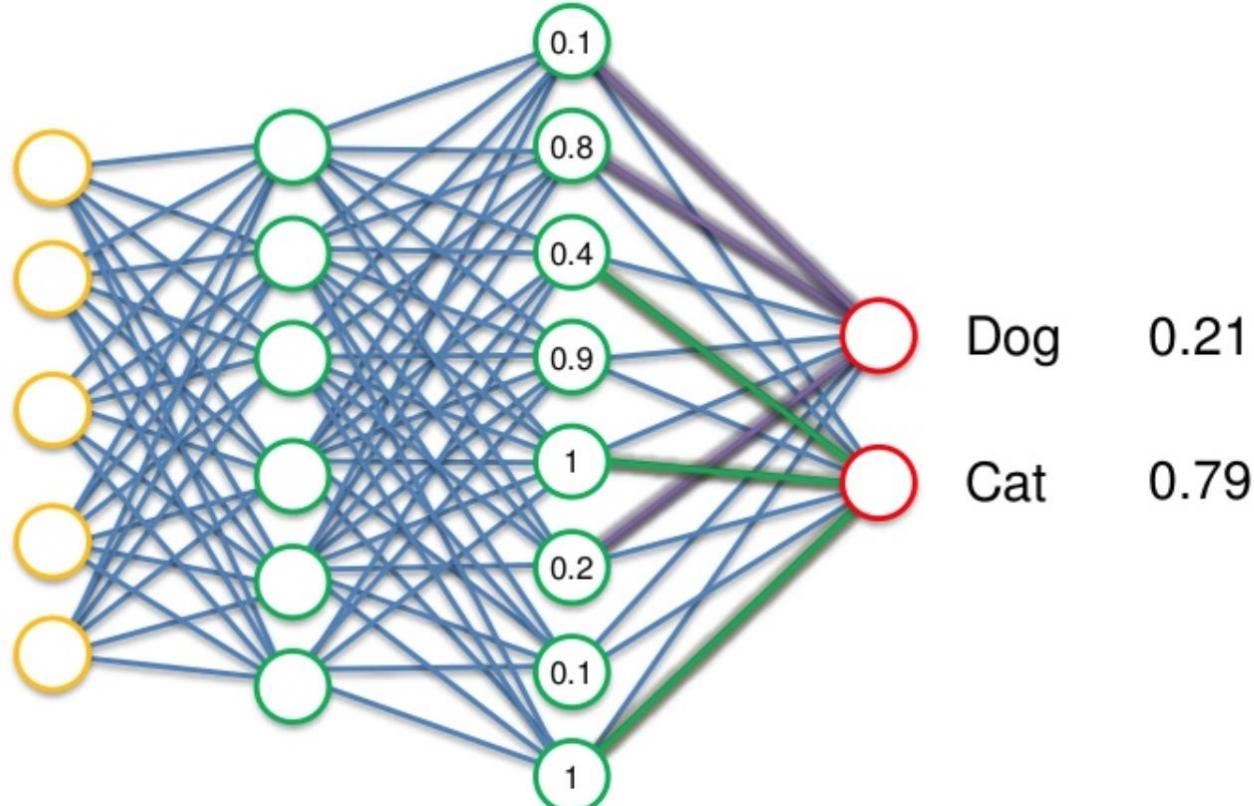




CNN : Step 4 – Full Connection



Flattening



CNN : Step 4 – Full Connection



Examples from the test set
(with the network's guesses)

The figure displays three examples of images with their corresponding predicted categories by a neural network. Each example consists of a small image followed by a list of five predicted categories.

- Cheetah:** The image shows a cheetah. The predicted categories are: cheetah (highlighted in red), leopard, snow leopard, and Egyptian cat.
- Bullet train:** The image shows a bullet train at a station. The predicted categories are: bullet train (highlighted in red), passenger car, subway train, and electric locomotive.
- Hand glass:** The image shows a hand holding a magnifying glass. The predicted categories are: hand glass (highlighted in red), scissors, frying pan, and stethoscope.

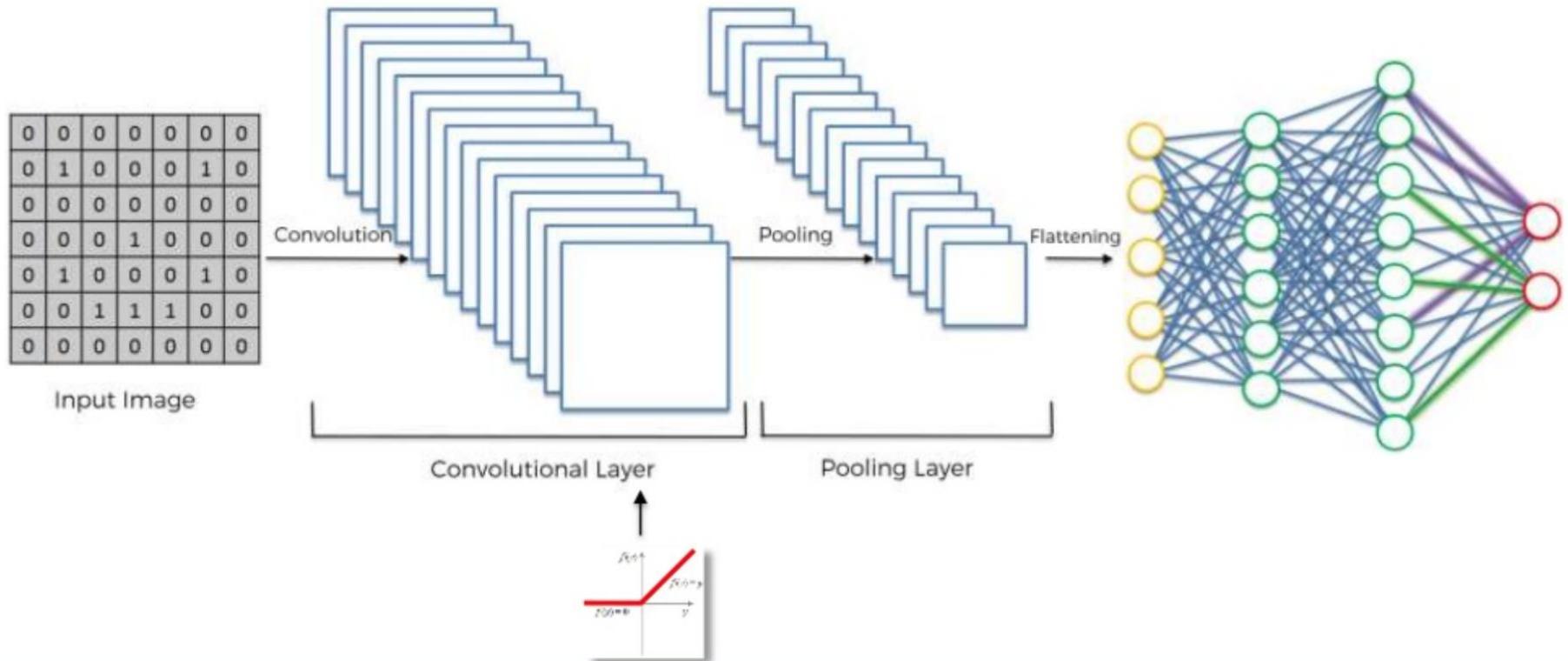
Image Source: a talk by Geoffrey Hinton

Summary



Summary

Summary



Summary



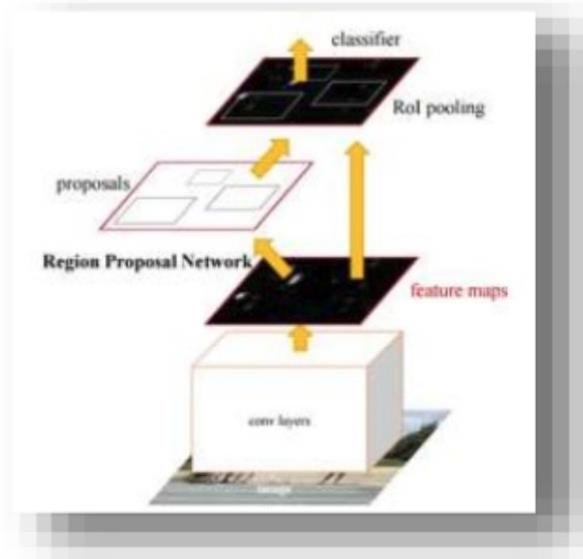
Additional Reading:

*The 9 Deep Learning Papers
You Need To Know About
(Understanding CNNs Part 3)*

Adit Deshpande (2016)

Link:

<https://adeshpande3.github.io/adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html>



Softmax & Cross-Entropy



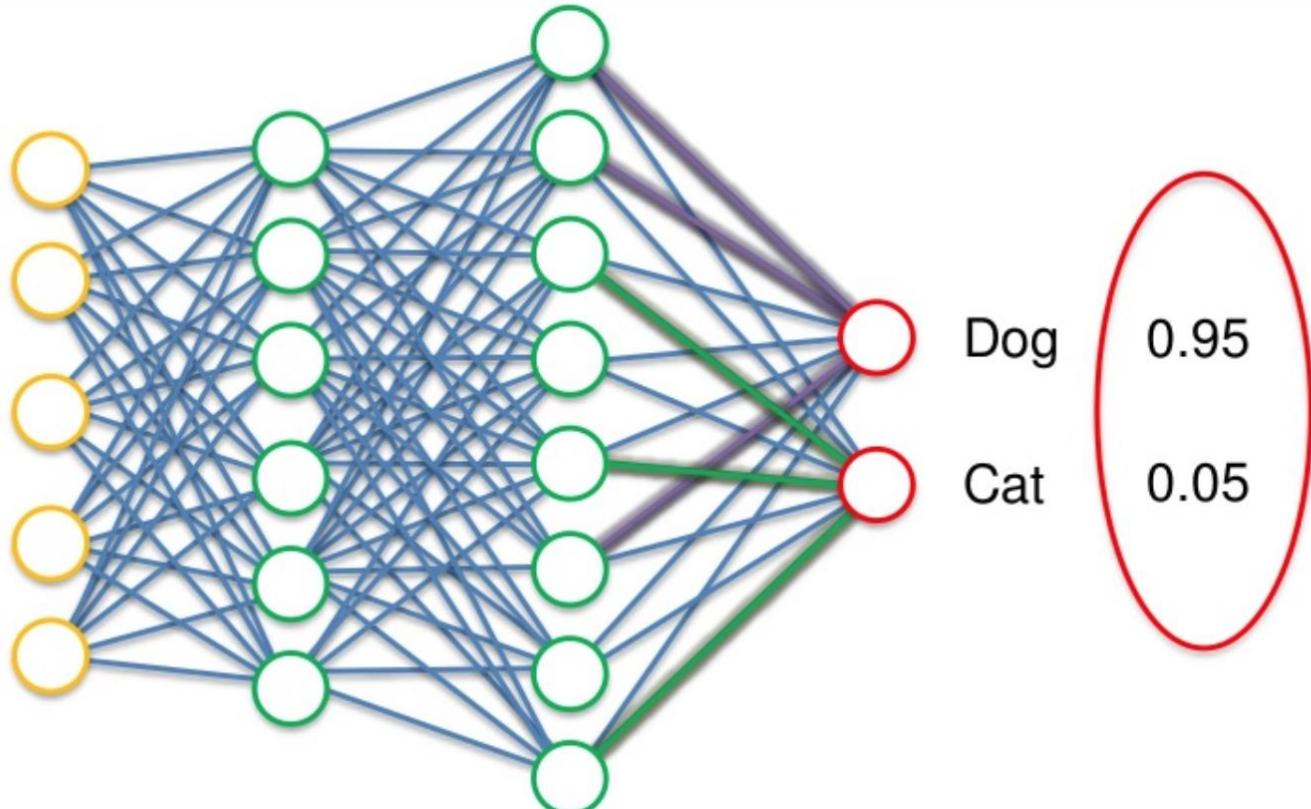
Softmax & Cross-Entropy



Softmax & Cross-Entropy



Flattening

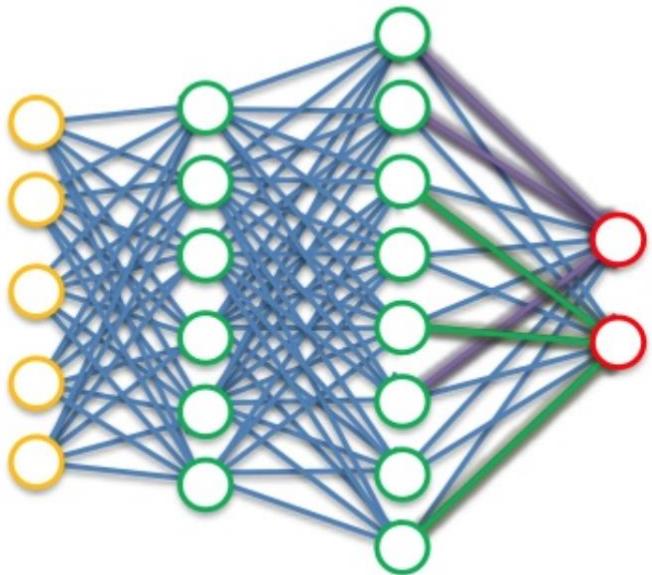




Softmax & Cross-Entropy



..... $\xrightarrow{\text{Flattening}}$



$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}}$$

Dog $\longrightarrow z_1 \longrightarrow 0.95$
Cat $\longrightarrow z_2 \longrightarrow 0.05$



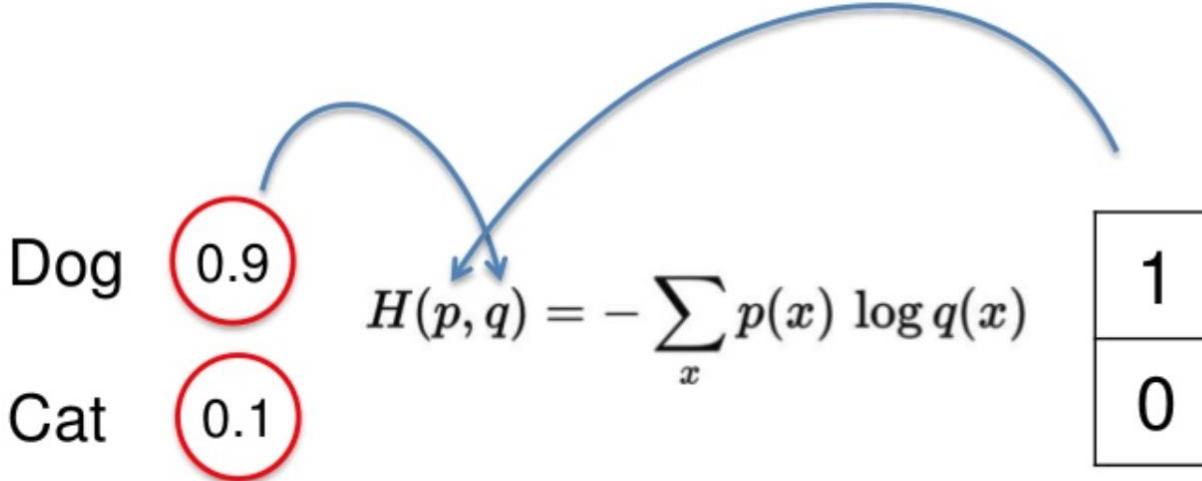
Softmax & Cross-Entropy

$$L_i = -\log \left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}} \right)$$

$$H(p, q) = - \sum_x p(x) \log q(x)$$



Softmax & Cross-Entropy





Softmax & Cross-Entropy

NN1 NN2



Dog	1
Cat	0

0.9
0.1

0.6
0.4



Dog	0
Cat	1

0.1
0.9

0.3
0.7



Dog	1
Cat	0

0.4
0.6

0.1
0.9



Softmax & Cross-Entropy

NN1

Row	Dog^	Cat^	Dog	Cat
#1	0.9	0.1	1	0
#2	0.1	0.9	0	1
#3	0.4	0.6	1	0

NN2

Row	Dog^	Cat^	Dog	Cat
#1	0.6	0.4	1	0
#2	0.3	0.7	0	1
#3	0.1	0.9	1	0

Classification Error

$$1/3 = 0.33$$

$$1/3 = 0.33$$

Mean Squared Error

$$0.25$$

$$0.71$$

Cross-Entropy

$$0.38$$

$$1.06$$

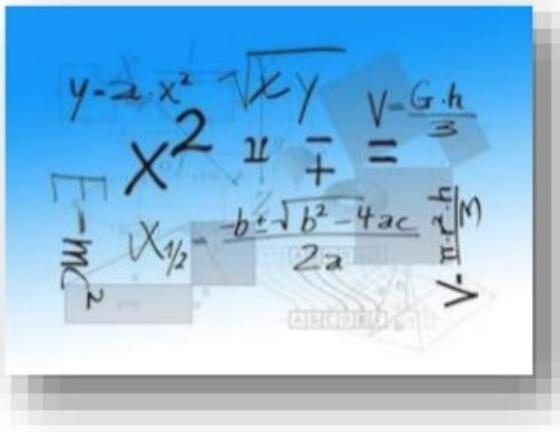


Softmax & Cross-Entropy

Additional Reading:

A Friendly Introduction to Cross-Entropy Loss

By Rob DiPietro (2016)



Link:

<https://rdipietro.github.io/friendly-intro-to-cross-entropy-loss/>

Summary



Additional Reading:

*How to implement a neural network
Intermezzo 2*

By Peter Roelants (2016)

$$\begin{aligned}\frac{\partial \xi}{\partial z_i} &= - \sum_{j=1}^C \frac{\partial t_j \log(y_j)}{\partial z_i} = - \sum_{j=1}^C t_j \frac{\partial \log(y_j)}{\partial z_i} = - \sum_{j=1}^C t_j \frac{1}{y_j} \frac{\partial y_j}{\partial z_i} \\ &= - \frac{t_i}{y_i} \frac{\partial y_i}{\partial z_i} - \sum_{j \neq i}^C \frac{t_j}{y_j} \frac{\partial y_j}{\partial z_i} = - \frac{t_i}{y_i} y_i (1 - y_i) - \sum_{j \neq i}^C \frac{t_j}{y_j} (-y_j y_i) \\ &= -t_i + t_i y_i + \sum_{j \neq i}^C t_j y_i = -t_i + \sum_{j=1}^C t_j y_i = -t_i + y_i \sum_{j=1}^C t_j \\ &= y_i - t_i\end{aligned}$$

Link:

http://peterroelants.github.io/posts/neural_network_implementation_intermezzo2/



김 진 수

CEO, Data Actionist

100-791 서울특별시 중구 청파로 463번지 3F BigData R&D Center

CP. 010-5670-3847 Tel. 02-360-4047 Fax. 02-360-4899

E-mail. bigpycraft@gmail.com

<http://www.biggycraft.com>

수고하셨습니다!