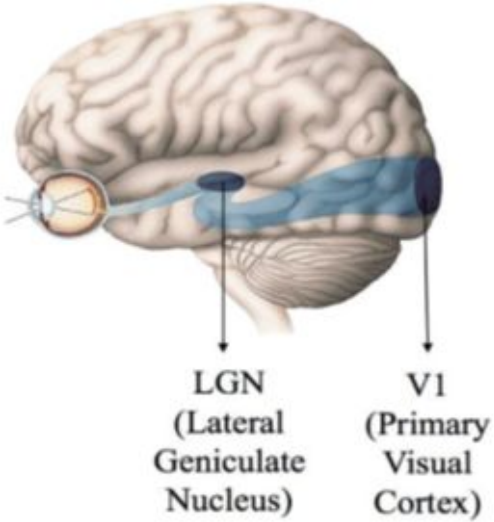
Lecture note 7 : Convolutional Neural Network

프로젝트 기반 딥러닝 이미지처리
한국인공지능아카데미 x Hub Academy

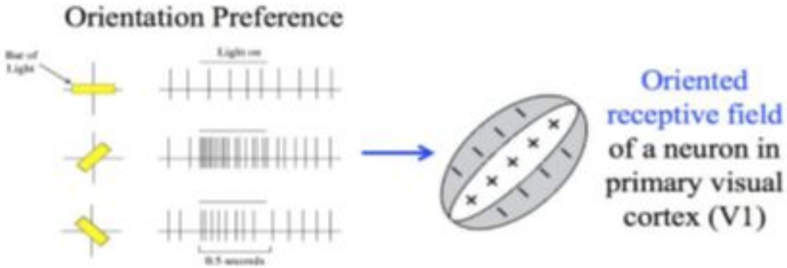
강사 : 김형욱 (hyounguk1112@gmail.com)

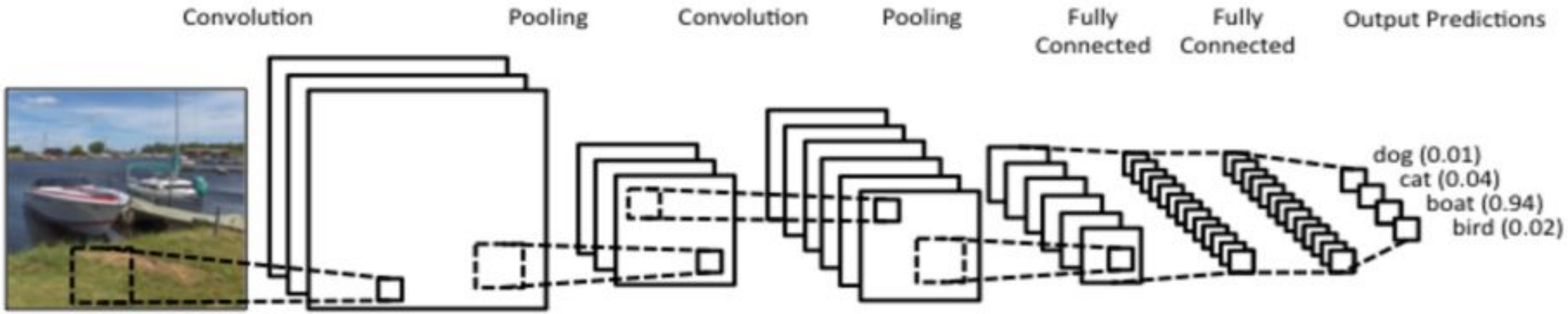
Convolutional Neural Network



Work by Hubel and Wiesel in the 1950s and 1960s showed that cat and monkey visual cortexes contain neurons that individually respond to small regions of the visual field.

the region of visual space within which visual stimuli affect the firing of a single neuron is known as its **receptive field**

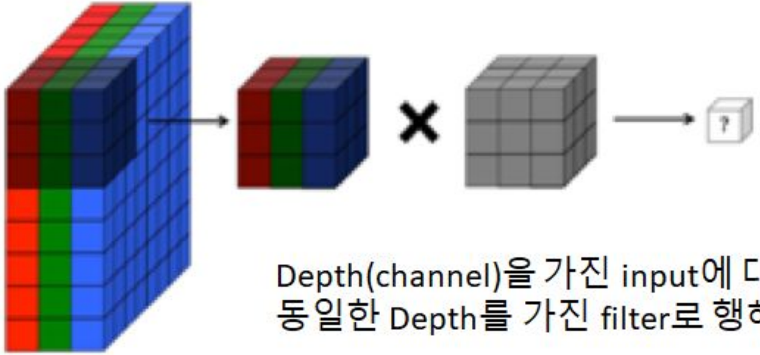




1. CNN(Convolutional Neural Network)는 Convolutional Layer와 Pooling Layer 그리고 FCN(Fully Connected Network)으로 구성되어있음.
2. 각 Convolutional Layer는 다수의 Convolutional Kernel로 구성되어 있고, 마지막에는 Activation function이 있다.
3. Pooling Layer는 feature map을 다운사이징하면서 일종의 정보 요약의 역할을 한다.
4. Fully Connected Network는 영상 특징 정보(Image feature information)를 토대로 비선형적 분류를 가능하게 한다.

CNN의 구조

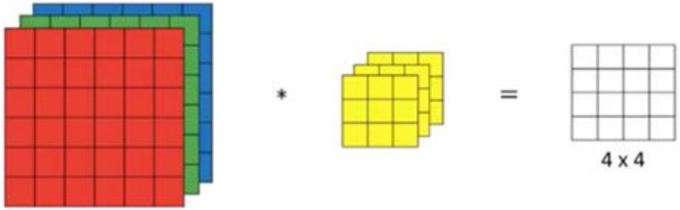
1. Convolutional kernel : RGB 영상 연산 방법



3차원 텐서에 대한 컨볼루션 필터는 마찬가지로 3차원 공간필터가 된다(Region이 3차원이 되므로)

Depth(channel)을 가진 input에 대한 Convolutional filtering은 동일한 Depth를 가진 filter로 행해진다.

Convolutions on RGB image

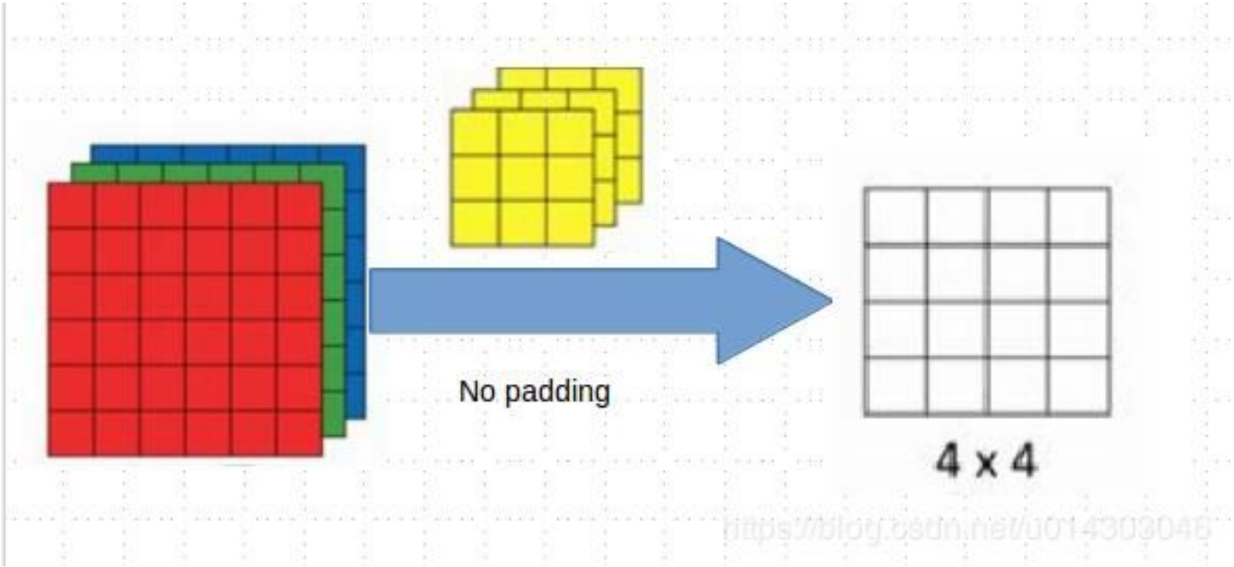


Input volume이 H x W x N차원이라면?
마찬가지로 K x K x N의 Kernel을 사용해준다.

(H, W : 가로, 세로. K : kernel의 너비 및 높이)

CNN의 구조

1. Convolutional kernel : RGB 영상 연산 방법

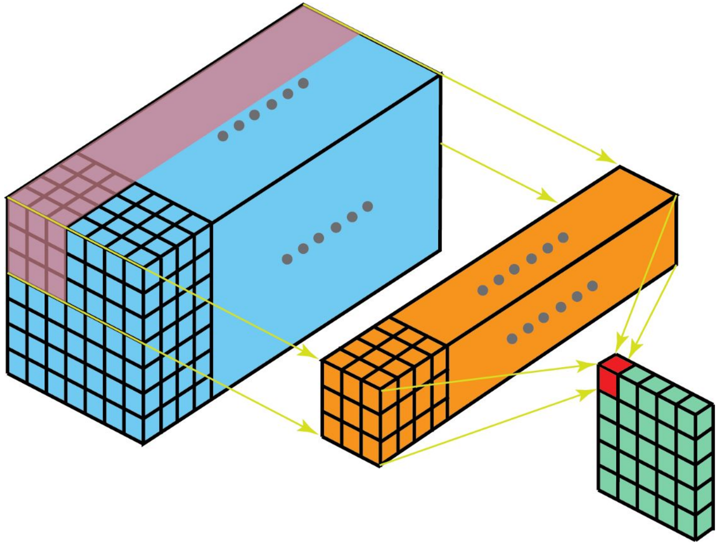


Padding이 있다면 feature map은 입력 텐서와 같은 너비와 깊이를 갖는다(깊이는 동일하게 1)

Ref : towardsdatascience.com

CNN의 구조

1. Convolutional Kernel : Depth가 N인 텐서에 대한 연산 방법

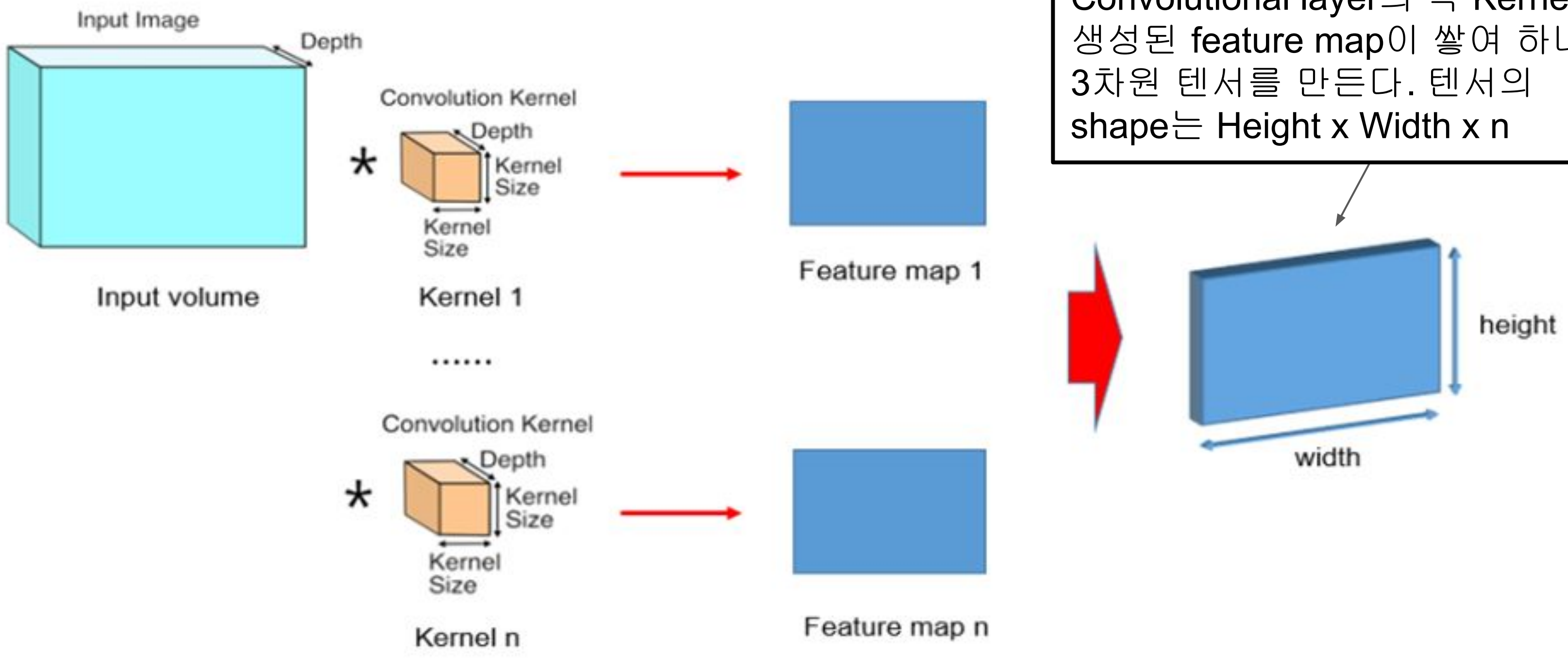


첫 번째 레이어 이후의 N(이전 레이어의 필터 갯수)개 만큼의 Depth를 가진 특징 정보는 동일한 Depth의 필터로 컨볼루션 연산을 수행한다.

Ref : towardsdatascience.com

CNN의 구조

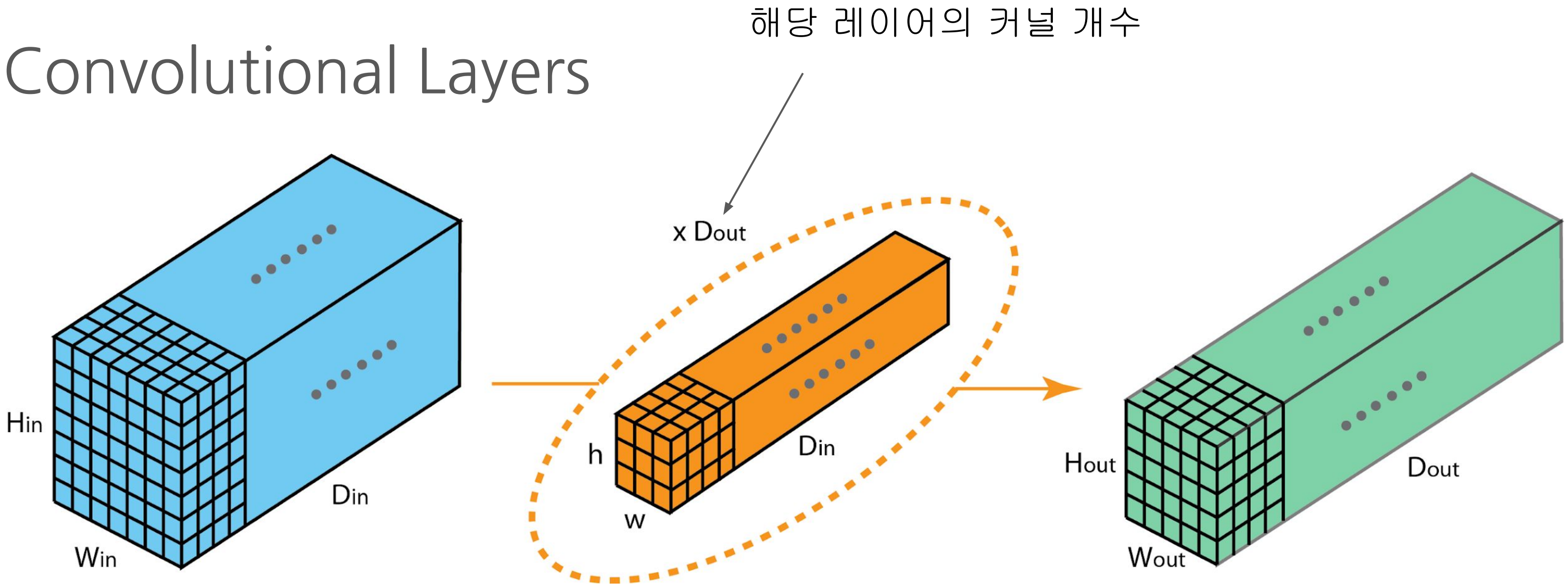
2. Convolutional Layers : 다중 필터의 feature map



Convolutional layer의 각 Kernel 별로 생성된 feature map이 쌓여 하나의 3차원 텐서를 만든다. 텐서의 shape는 Height x Width x n

CNN의 구조

2. Convolutional Layers



입력 텐서 형태

컨볼루션 필터 형태

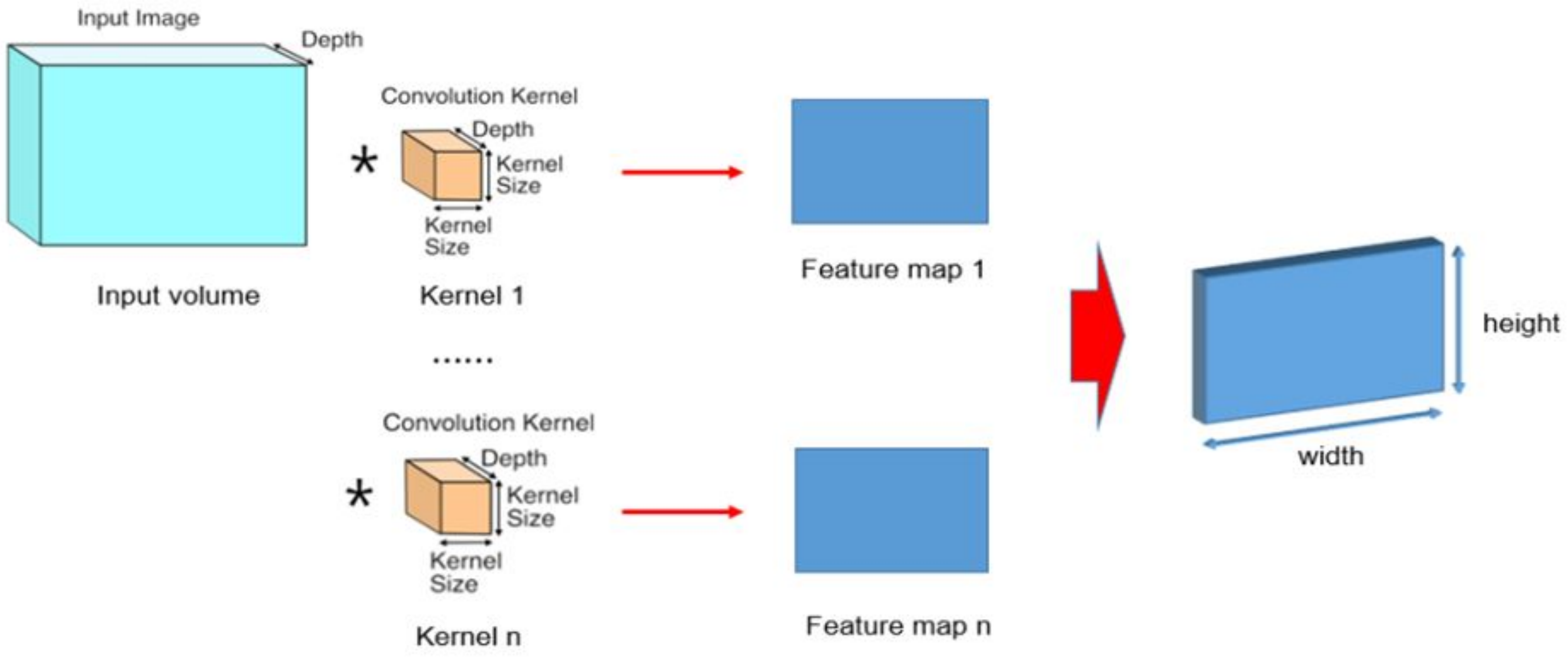
Dout개의 컨볼루션 필터를 갖는 컨볼루션 레이어의 아웃풋

첫 번째 레이어 이후의 N(이전 레이어의 필터 갯수)개 만큼의 Depth를 가진 특징 정보는 동일한 Depth의 필터로 컨볼루션 연산을 수행한다.

Ref : towardsdatascience.com

CNN의 구조

2. Convolutional Layers



CNN의 구조

2. Activation functions in Convolutional Layer

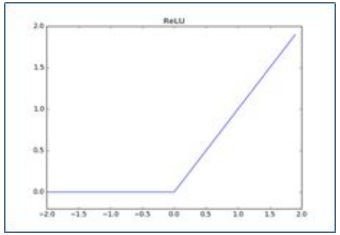
ReLU Layer

Filter 1 Feature Map

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

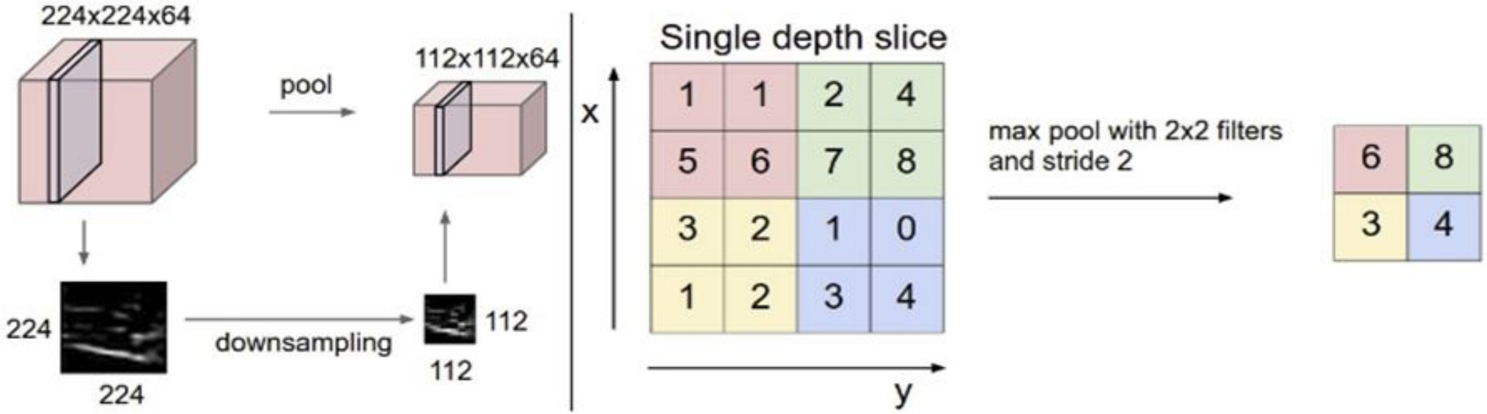
→

9	3	5	0
0	2	0	1
1	3	4	1
3	0	5	1



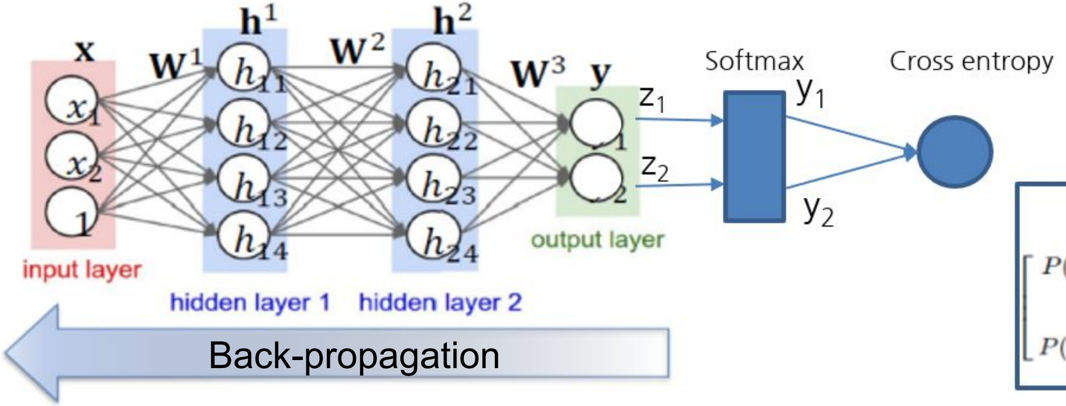
CNN의 구조

3. Pooling Layer



CNN의 구조

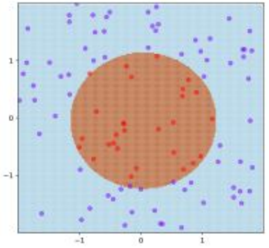
4. FCN

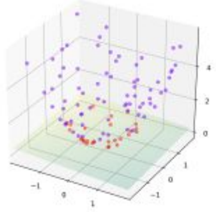


Softmax

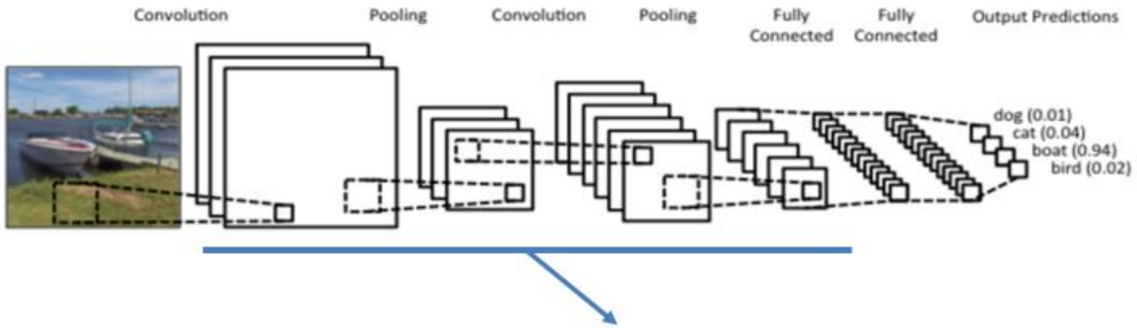
$$\begin{bmatrix} P(t=1|\mathbf{z}) \\ \vdots \\ P(t=C|\mathbf{z}) \end{bmatrix} = \begin{bmatrix} \varsigma(\mathbf{z})_1 \\ \vdots \\ \varsigma(\mathbf{z})_C \end{bmatrix} = \frac{1}{\sum_{d=1}^{C} e^{z_d}} \begin{bmatrix} e^{z_1} \\ \vdots \\ e^{z_C} \end{bmatrix}$$

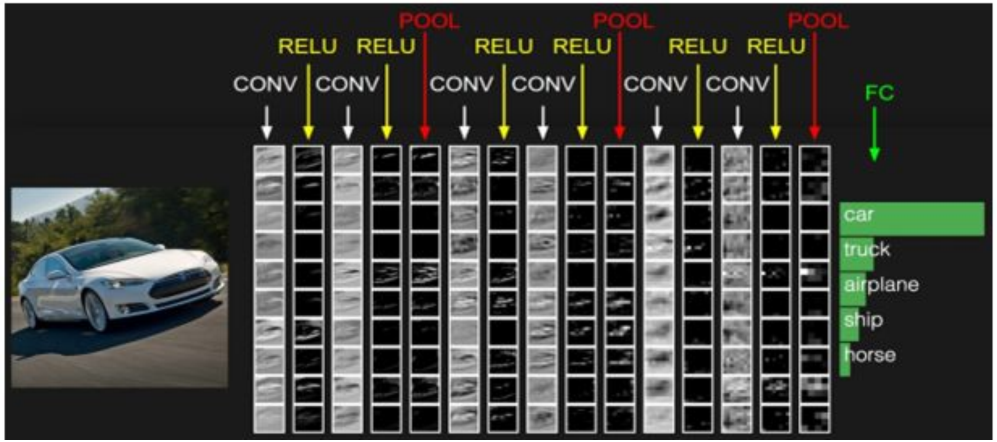
Cross entropy : $-\sum t_i * \log y_i$



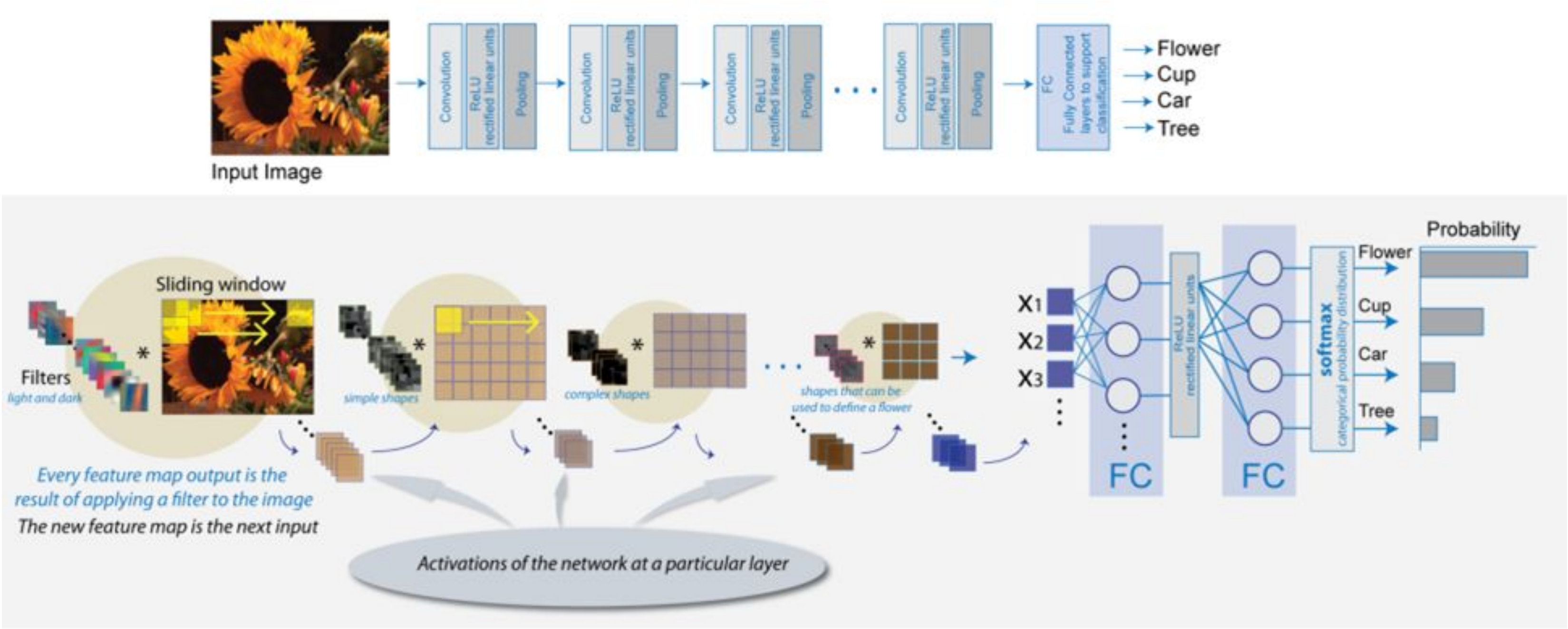


Overview of CNN(1)



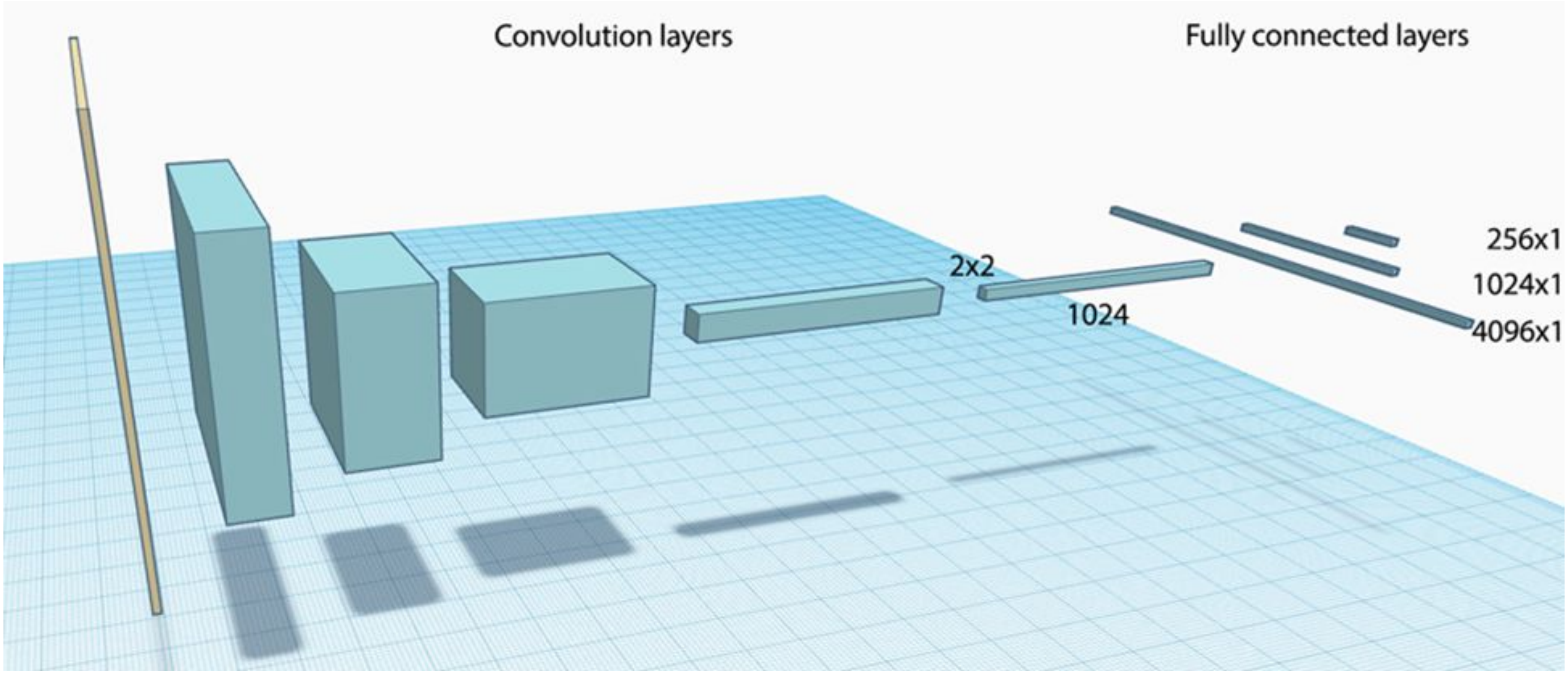


Overview of CNN(1)



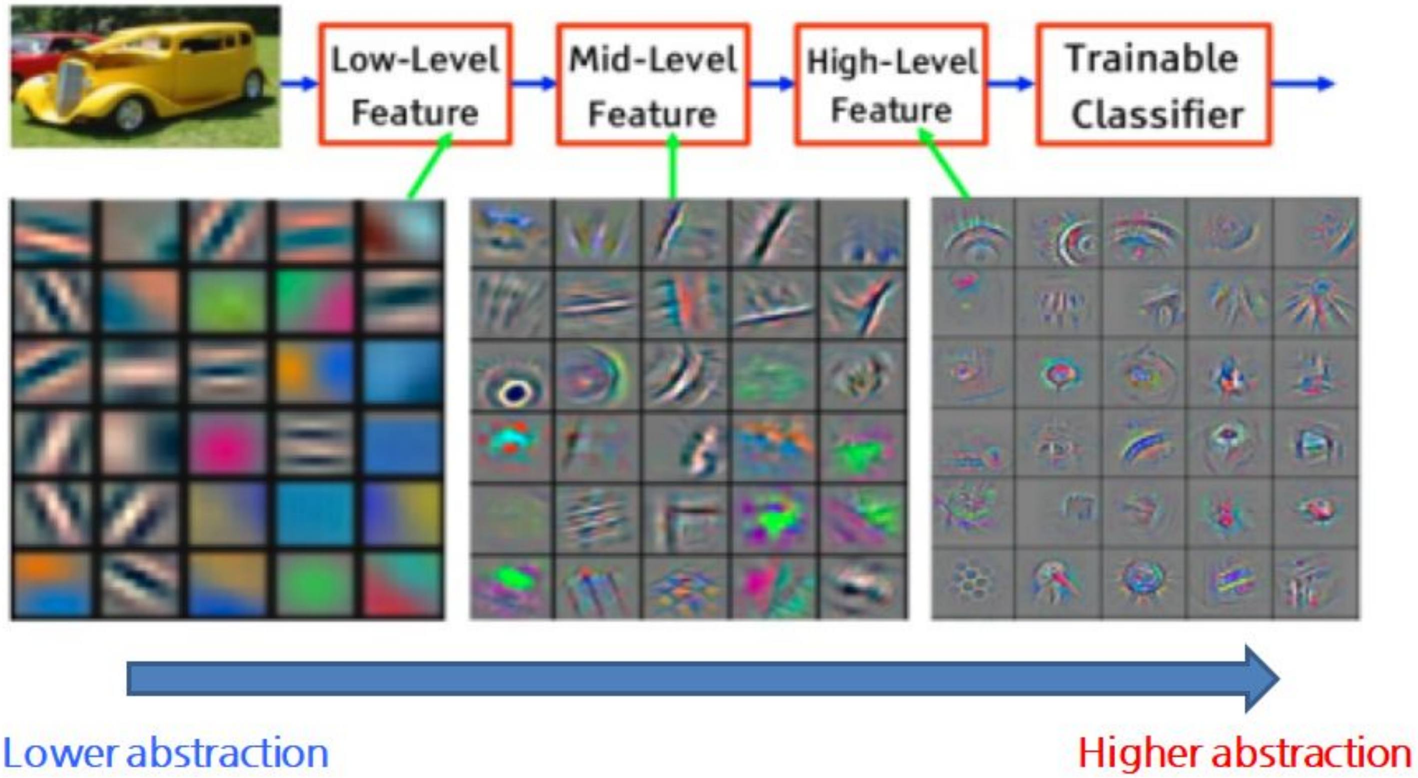
Ref : https://www.mathworks.com/help/nnet/deep_learning_architecture600pixels.png

Overview of CNN(3)

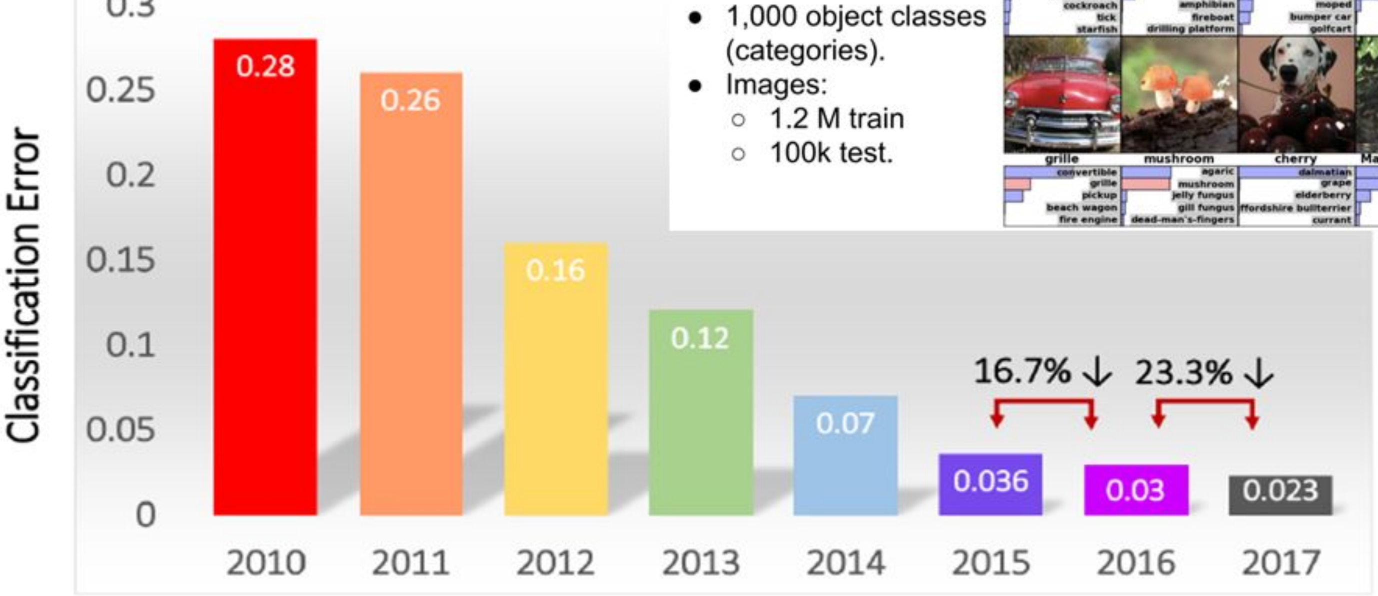


Ref : https://jhui.github.io/2017/03/16/CNN-Convolutional-neural-network/

Hierarchical Features of CNN



Imagenet Classification Competition

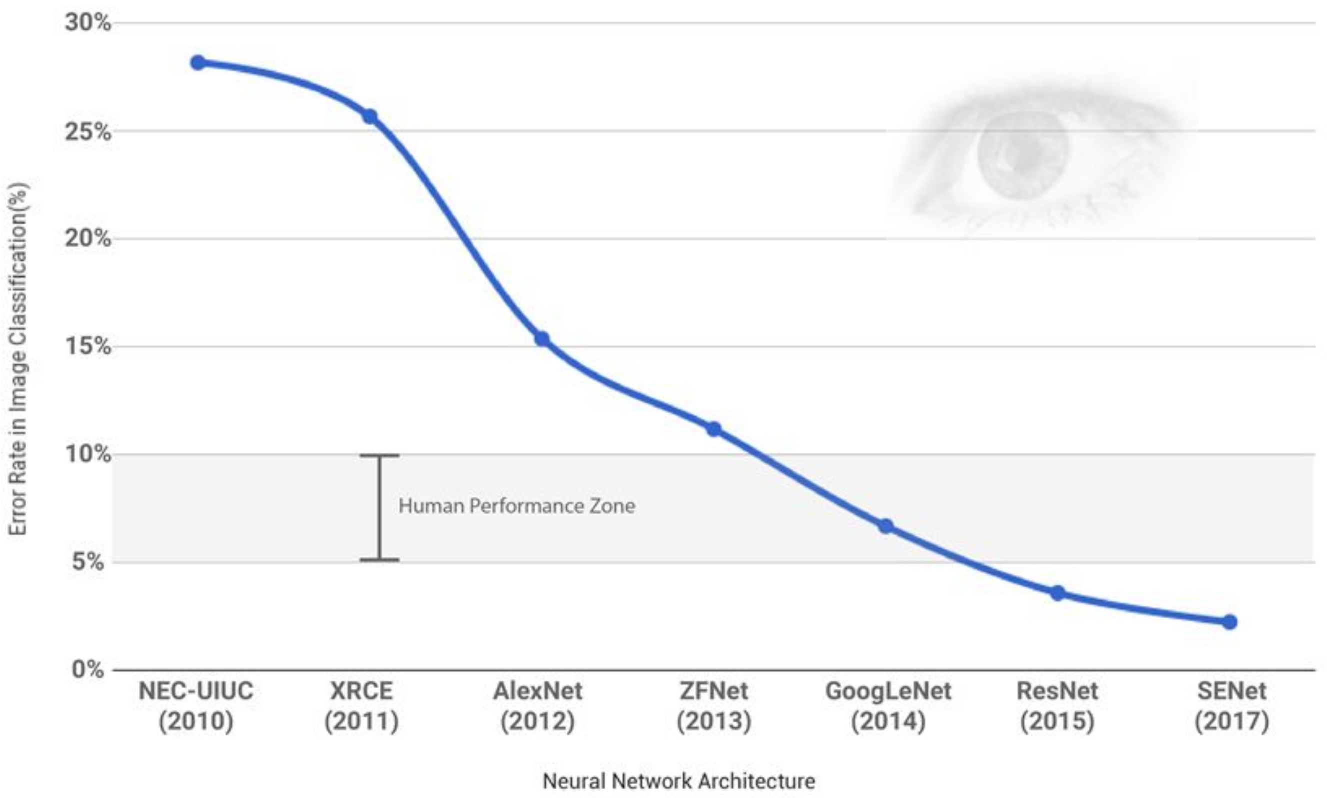


IMAGENET

- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.

Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

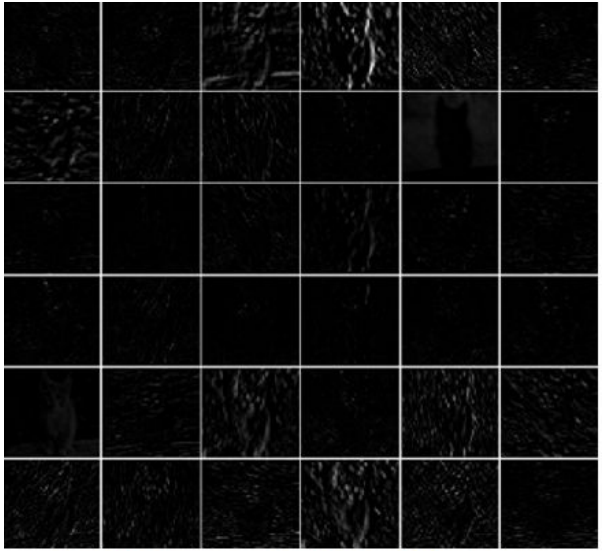
Imagenet Classification Competition



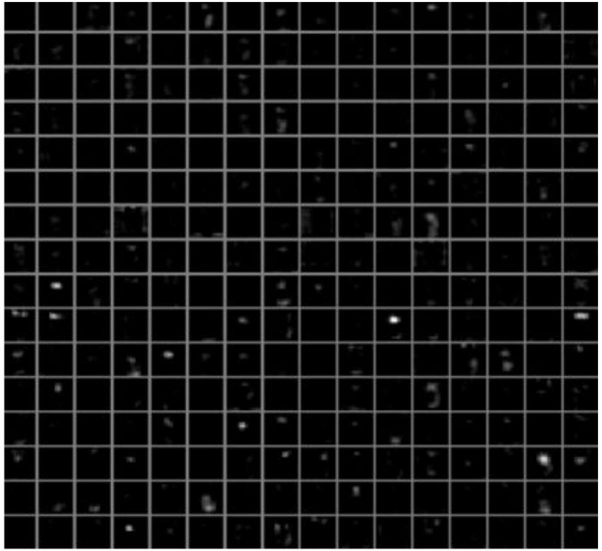
CNN insights

Visualization : Layer Activations

- More sparse and localized as the training processes
- Dead filters appear(symptom of high learning rates)



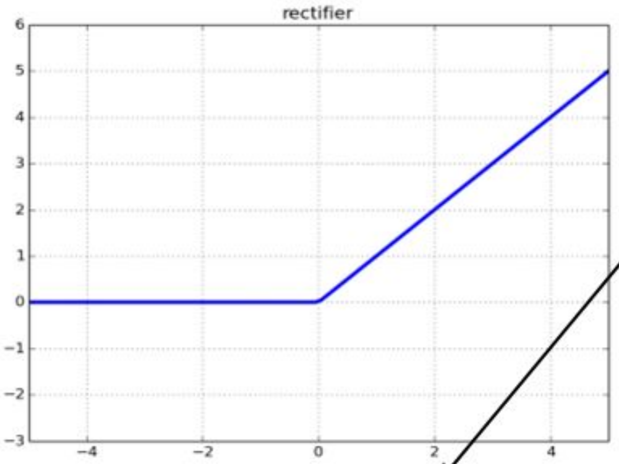
Activations on the first CONV layer



Activations on the second CONV layer

Ref : cs231n course of Stanford university

Visualization : Layer Activations



$$z_n = \sum_{i=0}^{k} w_i a_i^n$$

'a' : activations from the previous layer
'w' : weights

ReLU=max(0,Zn)

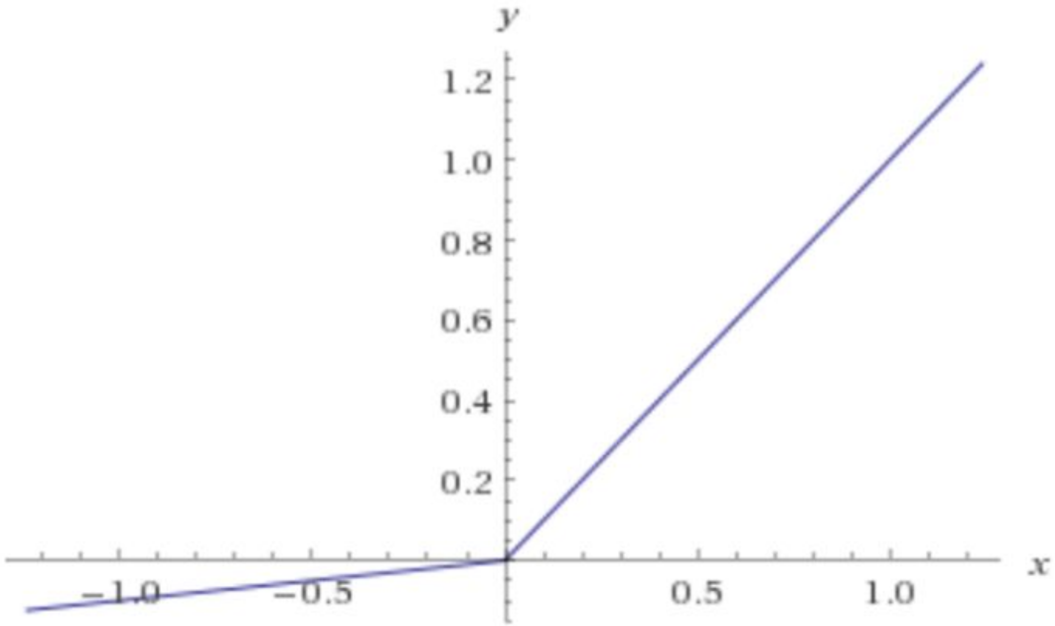
Simple error measure(like error = ReLu -y)

$$\frac{\partial error}{\partial z_n} = \delta_n = \begin{cases} 1 & z_n \geq 0 \\ 0 & z_n < 0 \end{cases}$$

$$\nabla error = \frac{\partial error}{\partial w_j} = \frac{\partial error}{\partial z_n} \times \frac{\partial z_n}{\partial w_j} = \delta_n \times a_j^n = \begin{cases} a_j^n & z_n \geq 0 \\ 0 & z_n < 0 \end{cases}$$

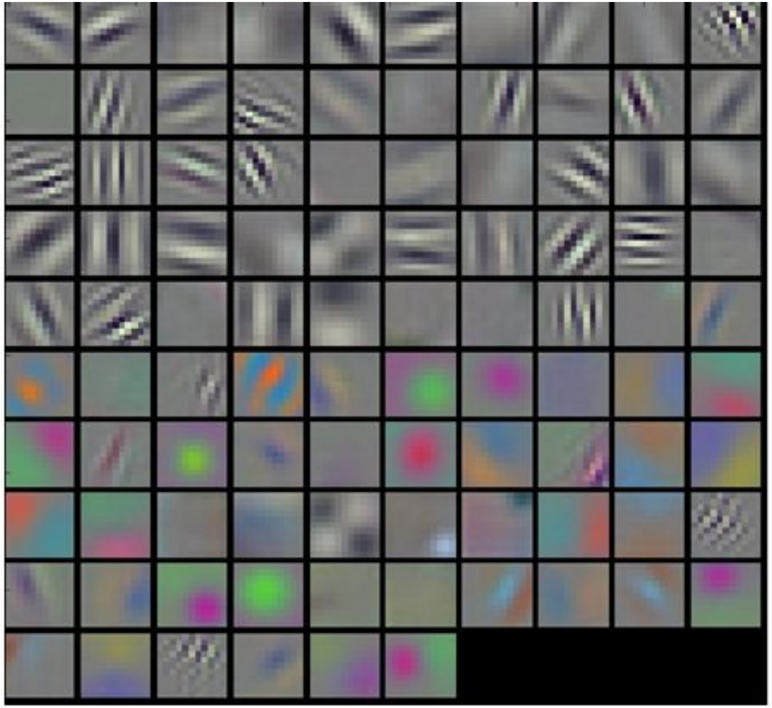
What if, weights put the ReLu on the flat side for all input of a batch? (large learning rate)

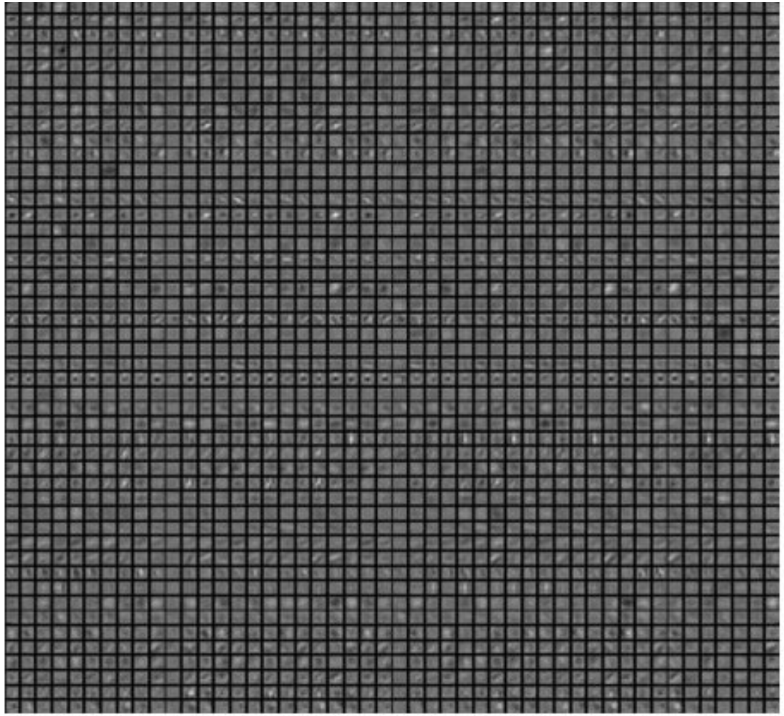
Visualization : Layer Activations



Leaky ReLu

Visualization : Convolutional Filters

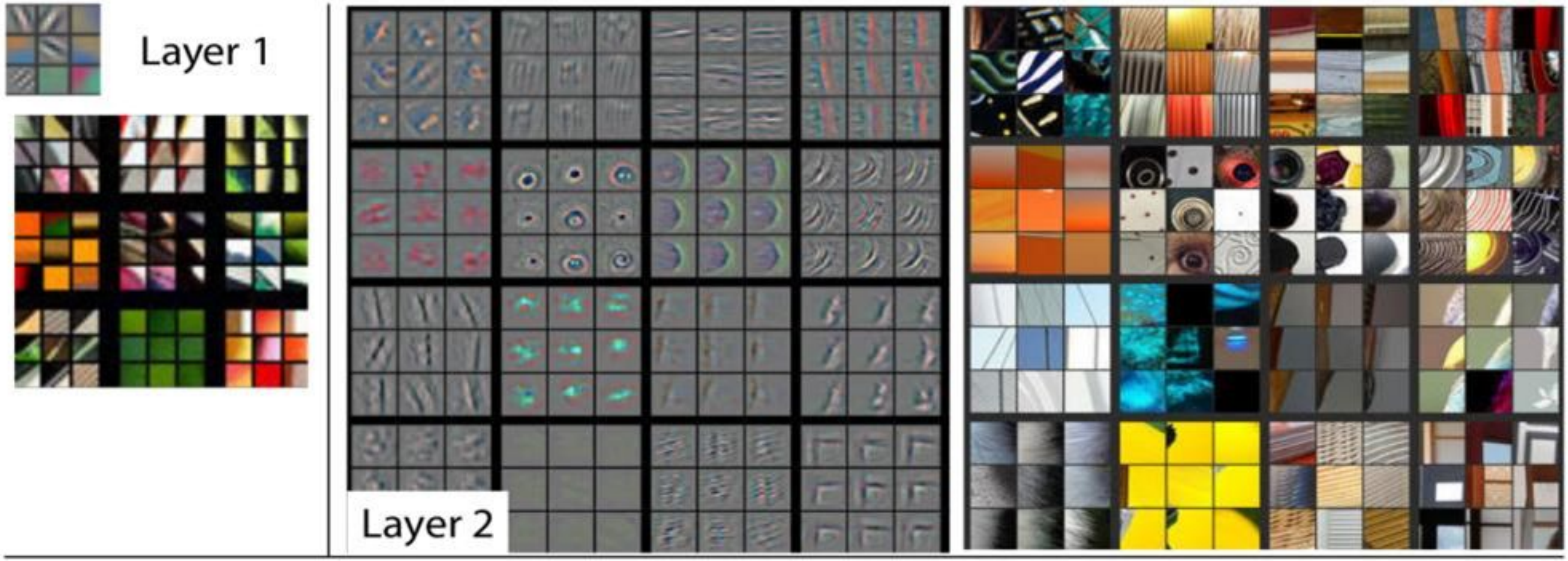




Well trained networks usually show nice and smooth filters without noisy patterns
(not trained enough, low regularization)

Ref : cs231n course of Stanford university

Visualization : Convolutional Filters



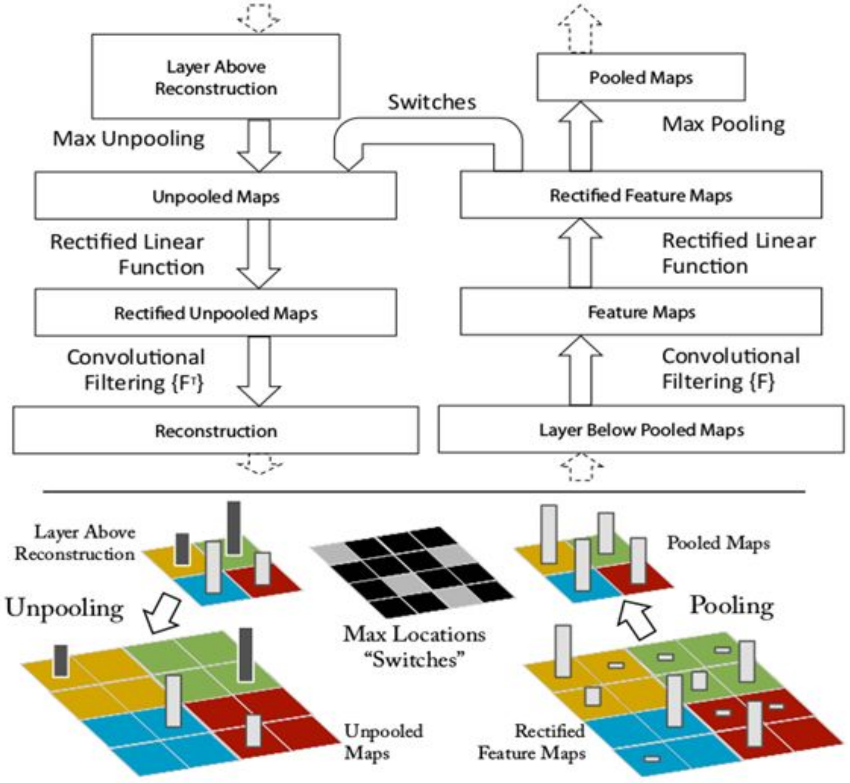
Ref：cs231n course of Stanford university

Visualization : Receptive field

It shows the characteristic of local connectivity of Convolutional networks

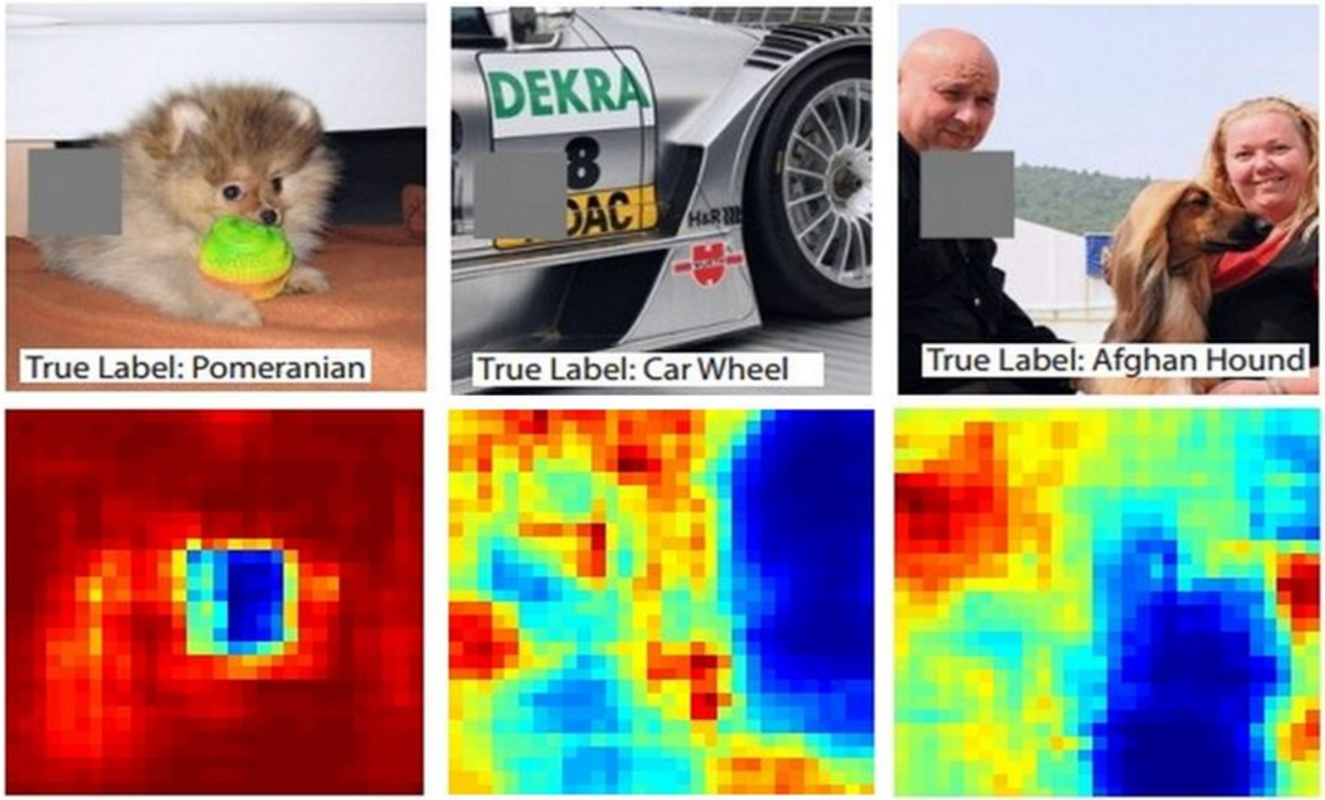
Ref : cs231n course of Stanford university

Visualization : Receptive field



DeConvnet (Zeiler and Fergus, 2013)

Visualization : Receptive field



Visualization : Embedding the codes with t-SNE

- Convolutional networks can be interpreted as gradually transforming the images into a representation in which the classes are separable by a linear classifier.

Ref : cs231n course of Stanford university

Visualization : Embedding the codes with t-SNE

Ref : cs231n course of Stanford university

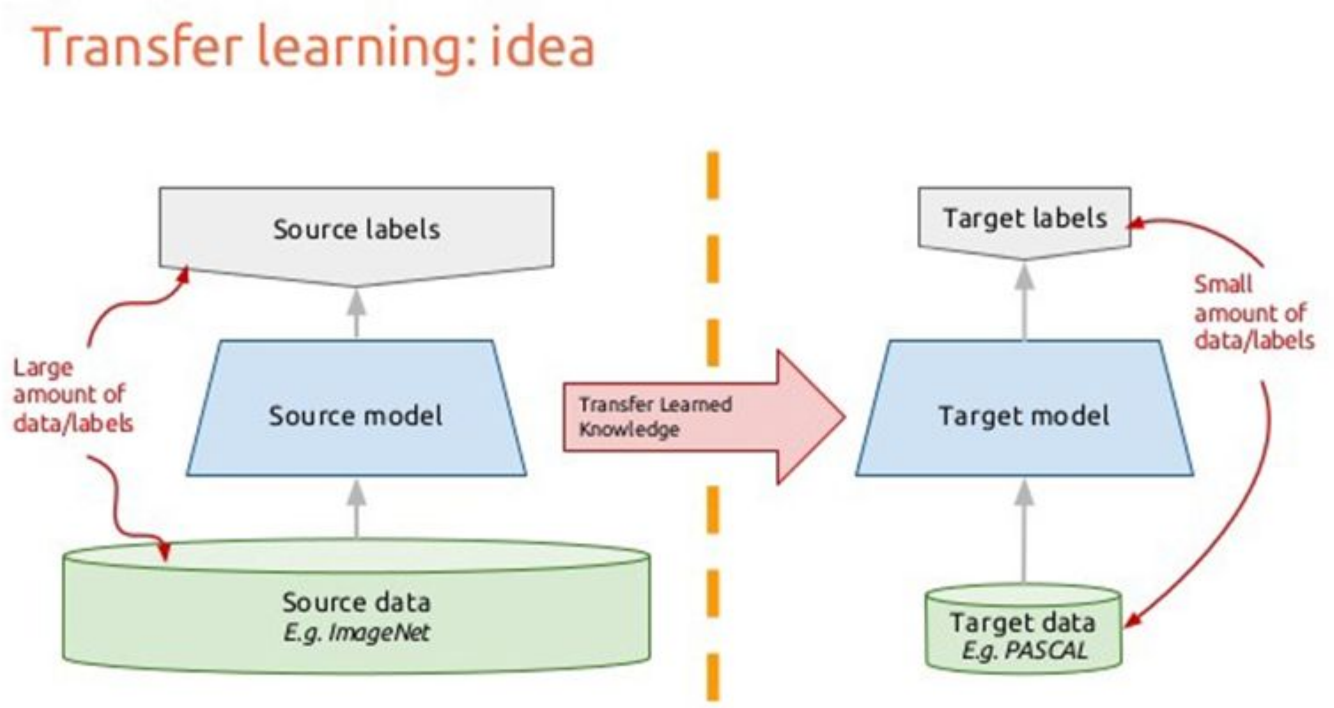
Transfer learning

In practice, We do not train an entire Convolutional Network from scratch(with random initialization) everytime.

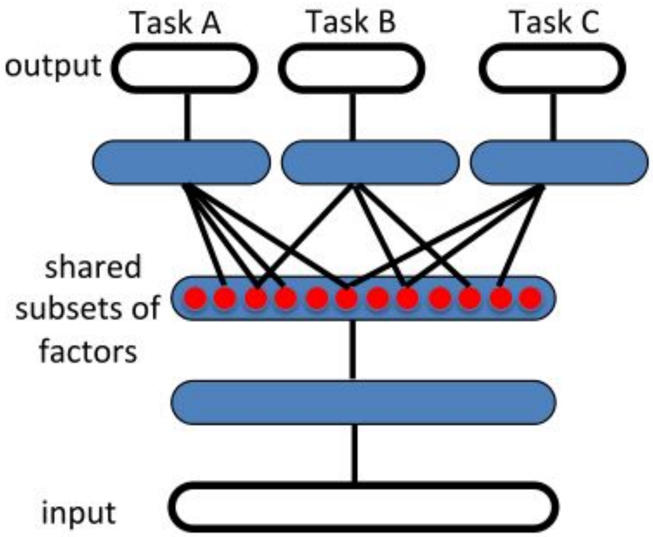
Instead, It is common to pretrain a ConvNet on a very large dataset such as the ImageNet dataset(1.2 million images with 1000 categories)

You can use the ConvNet either as an initialization or a fixed feature extractor for the task of interest.

Transfer learning



Transfer learning

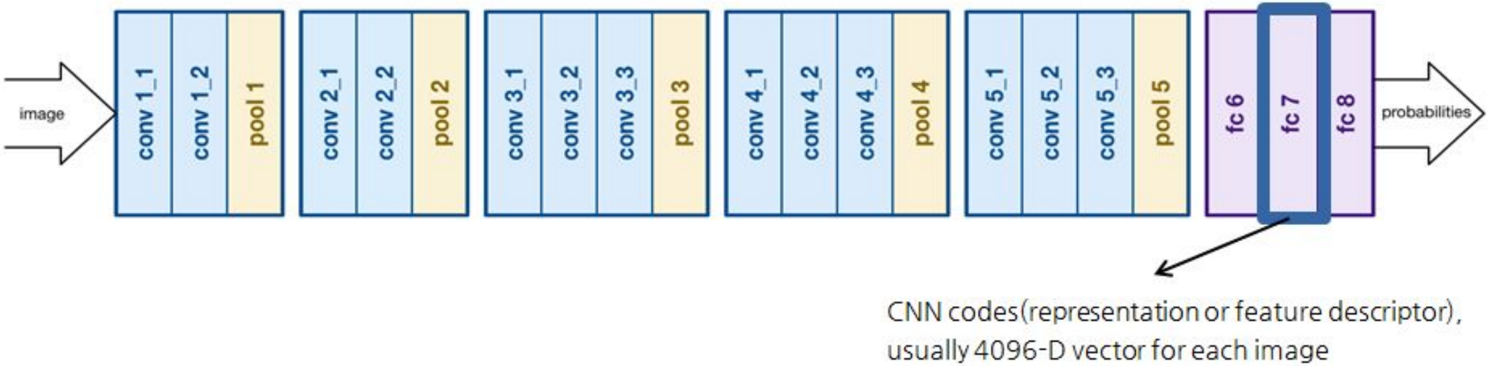


Transfer Learning is the ability of a learning algorithm to exploit commonalities between different learning tasks in order to share statistical strength and transfer knowledge across tasks.

Transfer learning

ConvNet as fixed feature extractor

- Take a ConvNet pretrained, remove the last fully-connected layer.
- Train a new classifier during treating the rest of the ConvNet as a fixed feature extractor.



Fine-tuning the ConvNet with Pretrained models

- Train all layers for specific domain.
- Due to overffing concerns and the ideas of the earlier feature to contain more generic features, sometime we only fine-tune higher-level portion of the network.

Thank you!