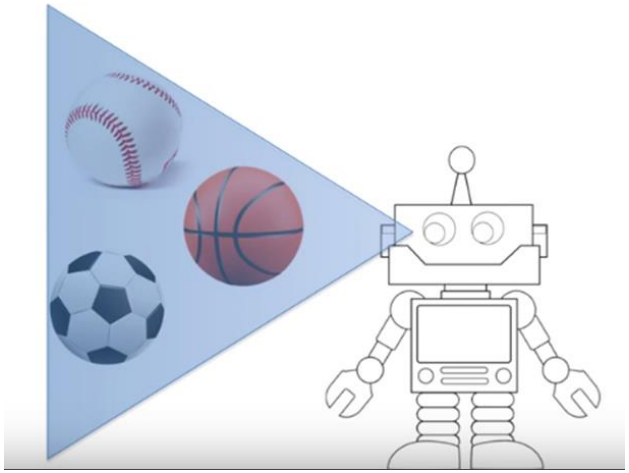
PNU Industrial Data Science

Convolutional Neural Network

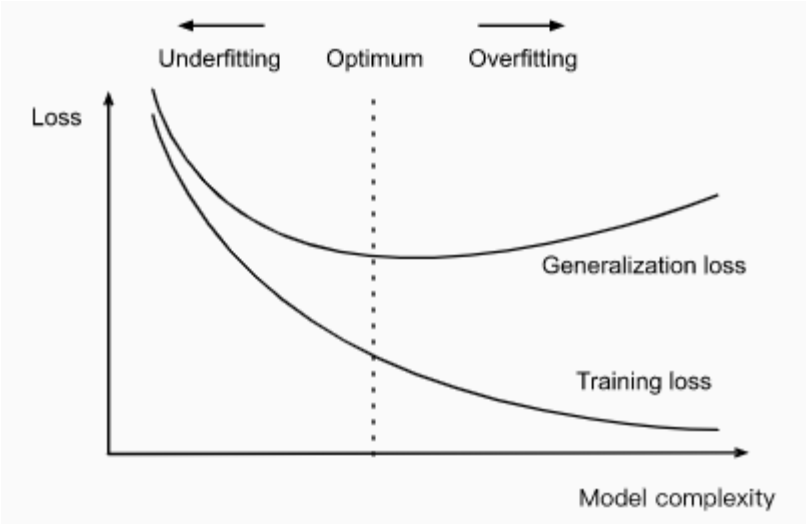
합성곱 신경망

Prof. Hyerim Bae

공을 구분하는 로봇



출처: https://nittaku.tistory.com/289



Contents

산업데이터과학은 산업현장에서 수집된 데이터를 분석하는데 필요한 기초 소양을 강의합니다.

Application of NN

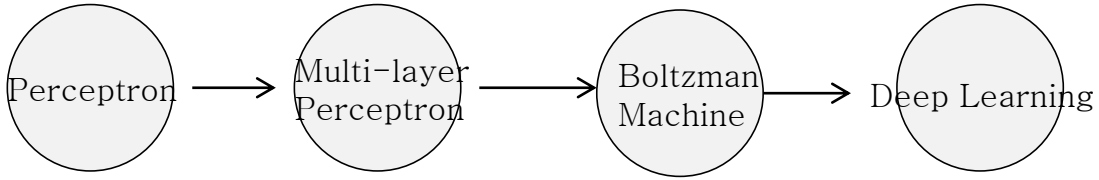
- Image recognition
  - Object recognition
  - Face recognition
- Voice recognition
- Spam-mail detection
- …

Backgrounds of Deep Learning?

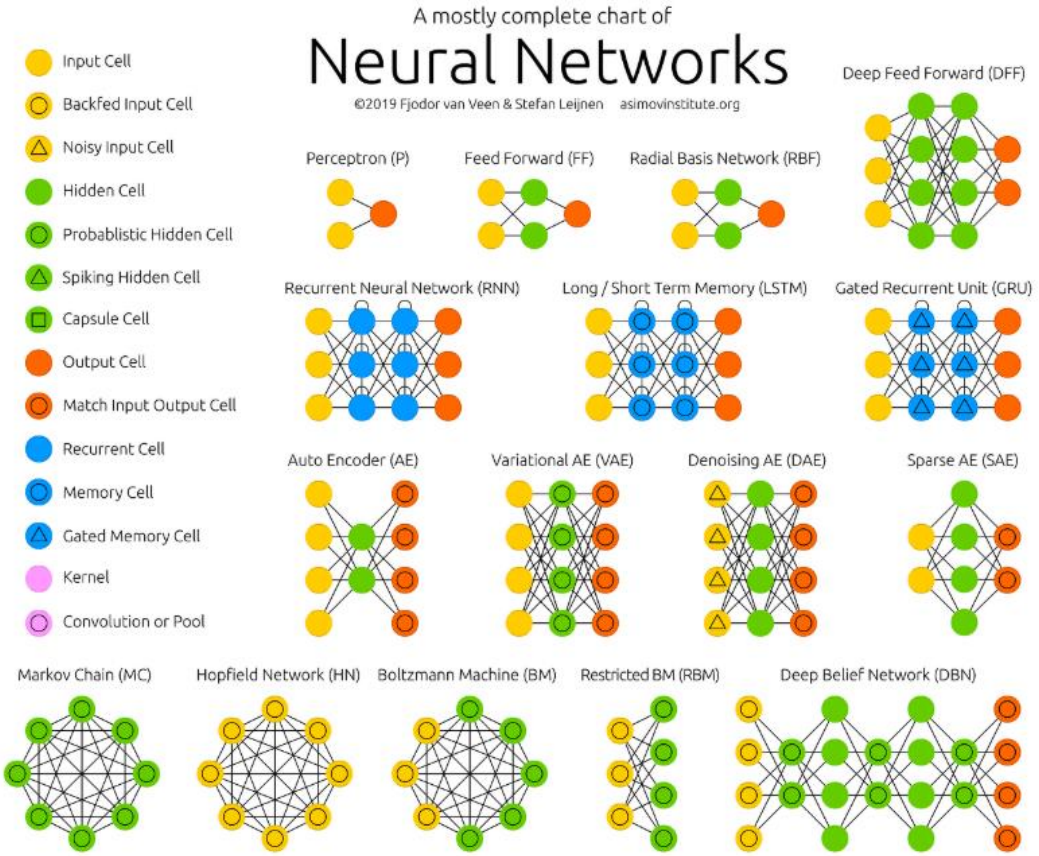
- Hard to program (?)
  - Automation (Decision system) vs. DSS
  - Problem is too complex
- Computing power
  - Big data
  - Mobile and SNS
- DL for business value
  - Ux
  - Advertisement
  - Recommendation

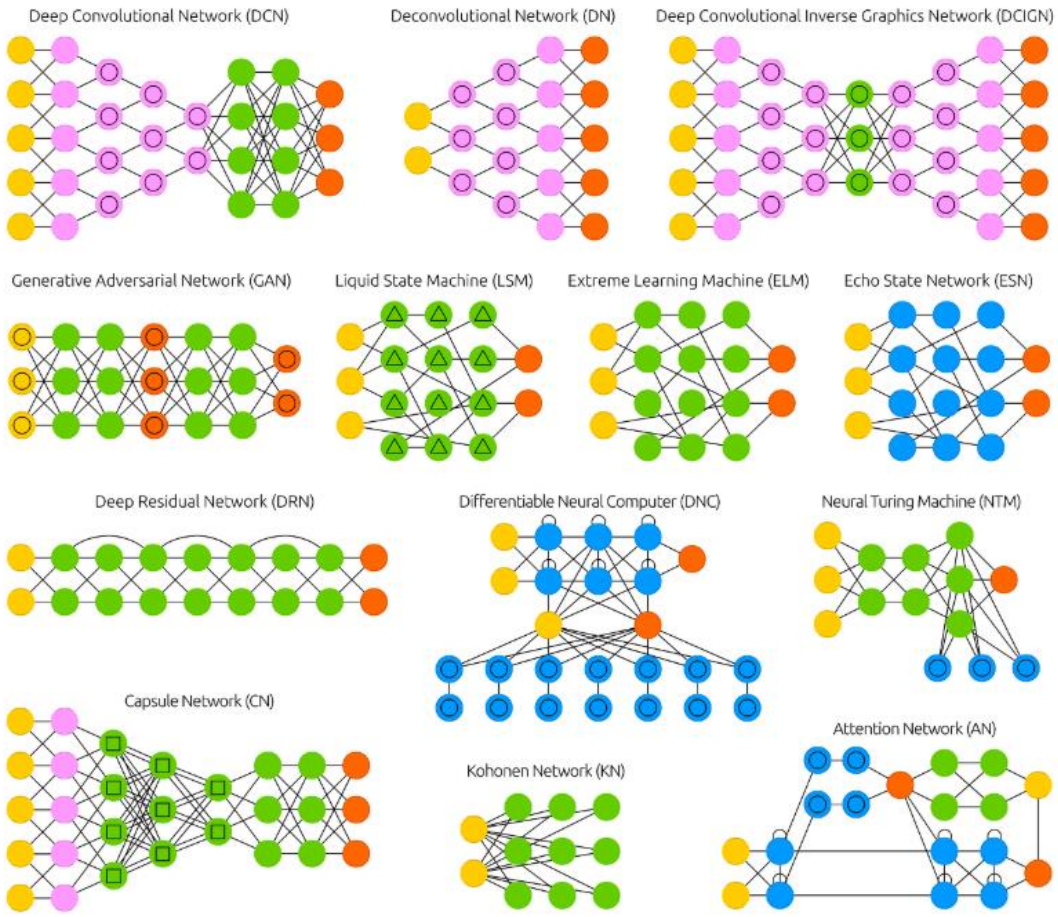
Brief history

- 1986 Back propagation algorithm
- 2004 DARPA Grand Challenge: Driverless car competition
- 2006 Deep Neural Networks by Hinton
- 2010 Turing award in learning theory (Valiant)
  - Probably approximately correct: Nature's Algorithms for learning and prospering in a Complex World
- 2011 Turing award in Bayesian networks (Pearl)
- 2012 DNNresearch (Hinton and Google)
- 2013 Quantum AI Lab for machine learning (Google)
- 2013 AI Labs for commercial ML research (Facebook)



Types of NN

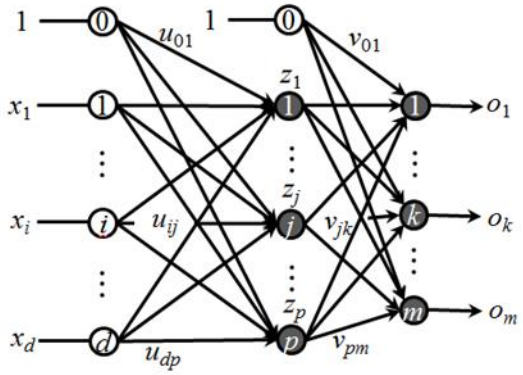

A mostly complete chart of
Neural Networks
©2019 Fjodor van Veen & Stefan Leijnen asimovinstitute.org
Input Cell
Backfed Input Cell
Noisy Input Cell
Hidden Cell
Probablistic Hidden Cell
Spiking Hidden Cell
Capsule Cell
Output Cell
Match Input Output Cell
Recurrent Cell
Memory Cell
Gated Memory Cell
Kernel
Convolution or Pool
Perceptron (P)
Feed Forward (FF)
Radial Basis Network (RBF)
Deep Feed Forward (DFF)
Recurrent Neural Network (RNN)
Long / Short Term Memory (LSTM)
Gated Recurrent Unit (GRU)
Auto Encoder (AE)
Variational AE (VAE)
Denoising AE (DAE)
Sparse AE (SAE)
Markov Chain (MC)
Hopfield Network (HN)
Boltzmann Machine (BM)
Restricted BM (RBM)
Deep Belief Network (DBN)



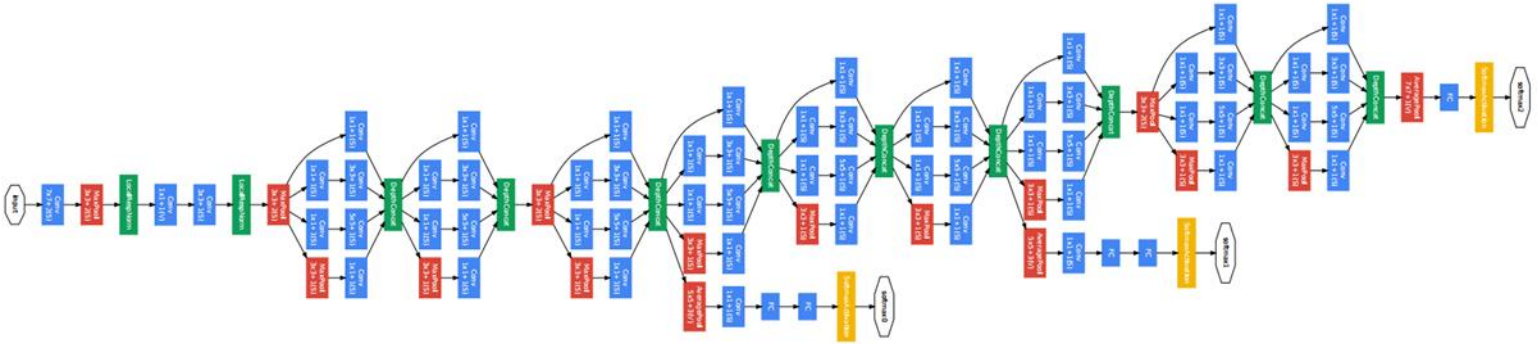
Source: https://www.asimovinstitute.org/neural-network-zoo

What is deep learning

- ANN



- DNN

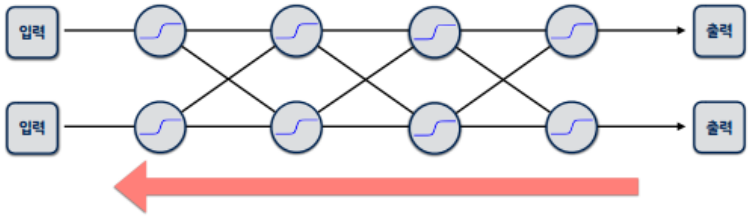


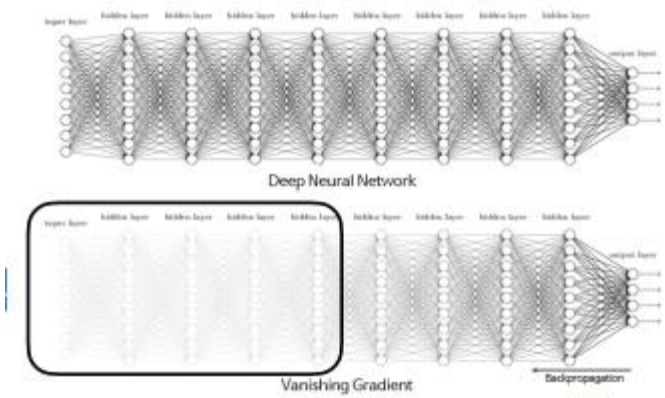
Three problems to be handled

- Underfitting
- Overfitting
- Slow learning

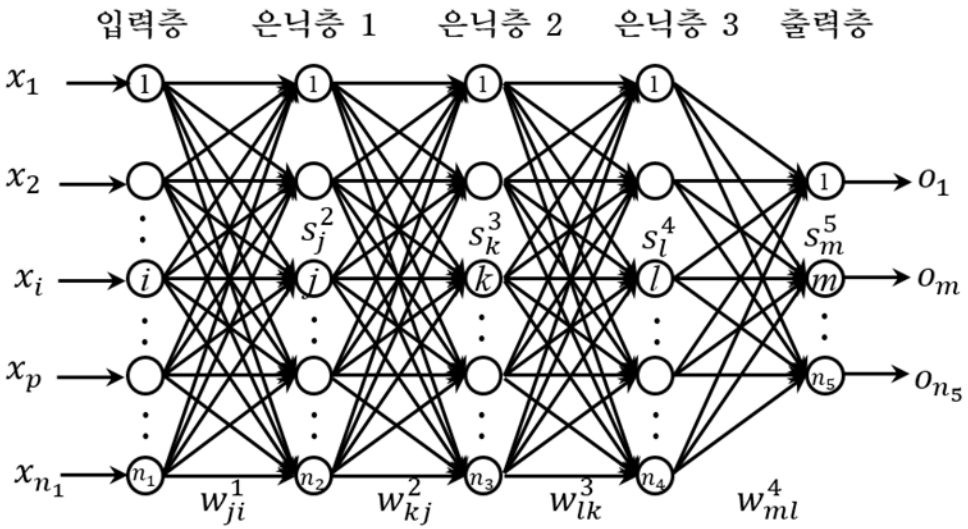
1. Underfitting

- Learning method: Back propagation
  - Classifier produces too extreme result (if f == s (=$\sum_i w_i x_i$))
    - Use Sigmoid function instead of step function



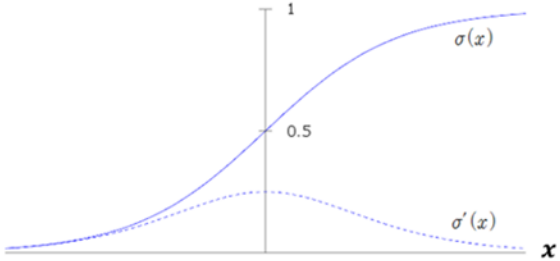


Problem of sigmoid BP

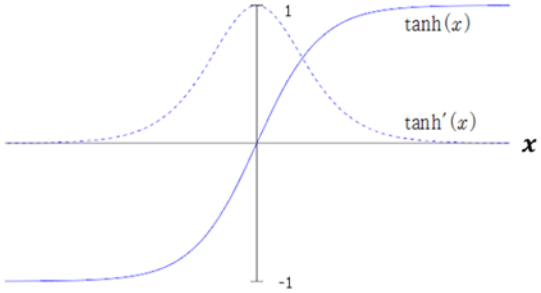


$$E = \frac{1}{2}\sum_{m=1}^{n_5}(y_m - o_m)^2$$

$$\frac{\partial E}{\partial w_{ji}^1} = \sum_{m=1}^{n_5}\sum_{l=1}^{n_4}\sum_{k=1}^{n_3}\sum_{j=1}^{n_2}(y_m - o_m) f'(s_m^5)\, w_{ml}^4 f'(s_l^4) w_{lk}^3 f'(s_k^3)\, w_{kj}^2 f'(s_j^2)\, x_i$$



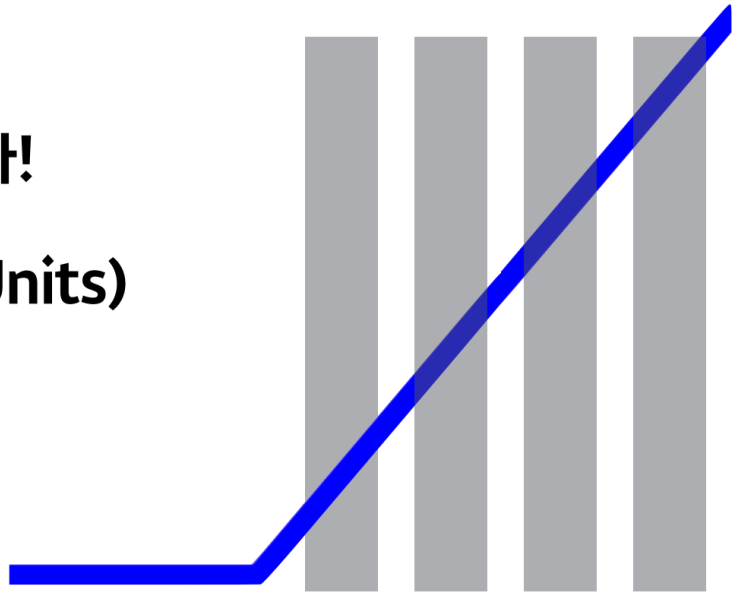
시그모이드(sigmoid)



쌍곡 탄젠트(hypertangent)

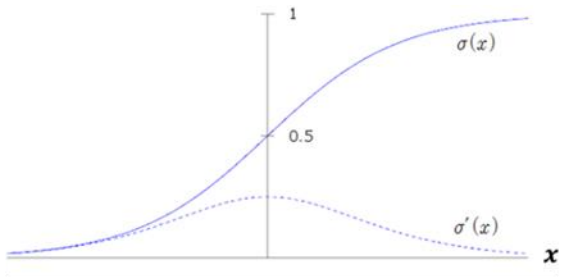
사그라드는 sigmoid대신
죽지않는 activation func을 쓰자!

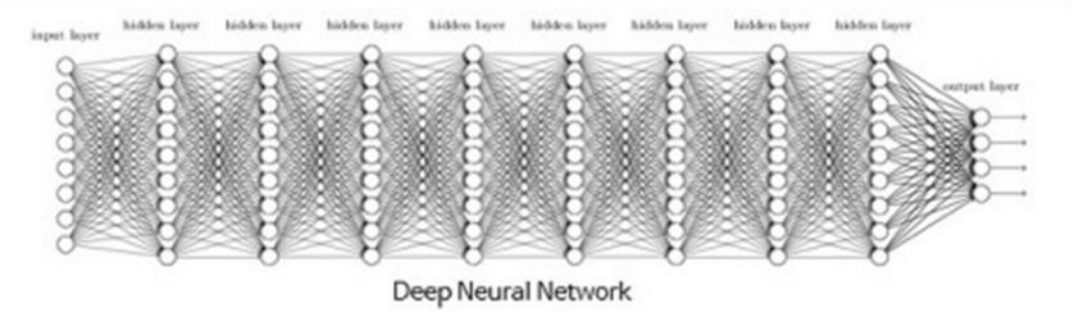
→ **ReLU** (Rectified Linear Units)



이녀석은 양의 구간에서 전부 미분 값(1)이 있다!

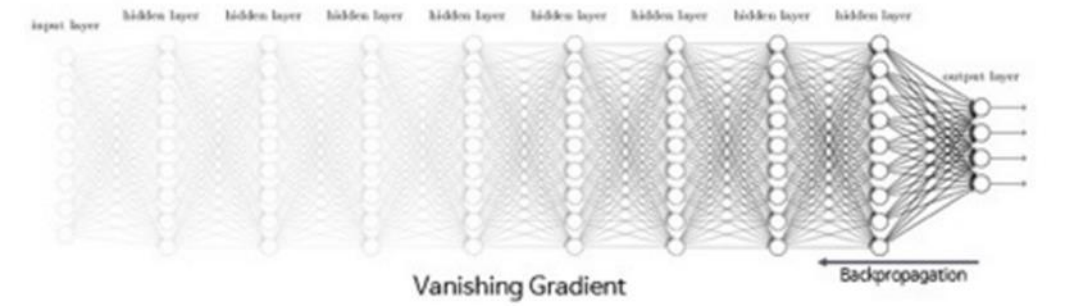
The effect of ReLU





Deep Neural Network

ReLU 사용

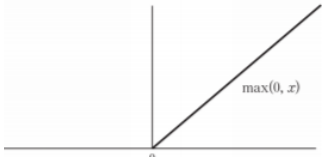


Vanishing Gradient

시그모이드 사용

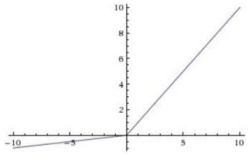
Variations of ReLU

- **ReLU**



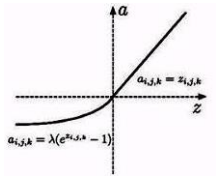
- **Leaky ReLU**

$$f(x) = \max(\alpha x, x)$$



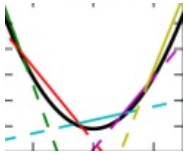
- **ELU (exponential Linear Unit)**

$$f(x)\begin{cases} x & if\ x > 0 \\ \alpha(\exp(x) - 1) & otherwise \end{cases}$$



- **Maxout**

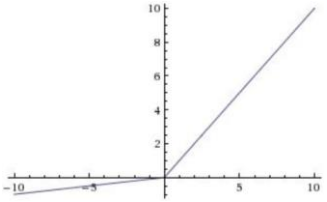
$$f(x) = max_{i\in\{1,\ldots,k\}}\{w_i^T x + b_i\}$$



- **PReLU** (parameteric ReLU)

$$f(x) = \max(\alpha x, x)$$

α: Parameter learned



2. Slow learning

- Gradient Decent

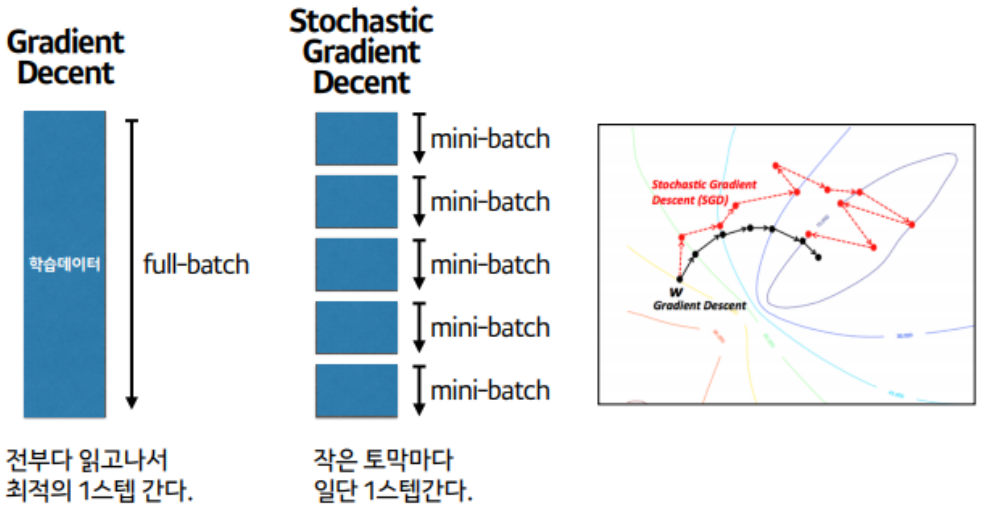
$$\mathbf{W} \leftarrow \mathbf{W} + c(d - f)\mathbf{X}$$

Weigh update = Direction reducing err. **(decent)** X Size of one step **(learning rate)** X slope **(gradient)**

$-\eta \nabla_\theta J(\theta)$ = $-$ X η X $\nabla_\theta J(\theta)$

- In big data
  - Weight update is too slow

- SGD
  - Use mini-batch for updating the weight

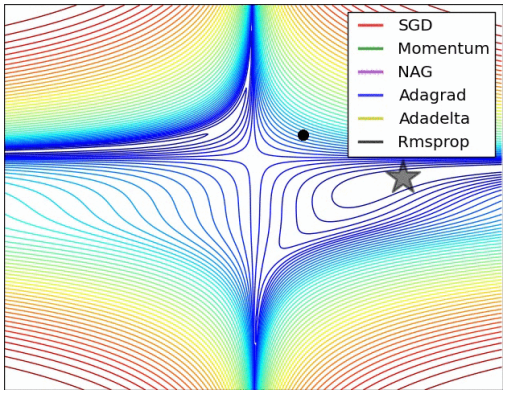


- Momentum vs. Adaptive
  - Momentum

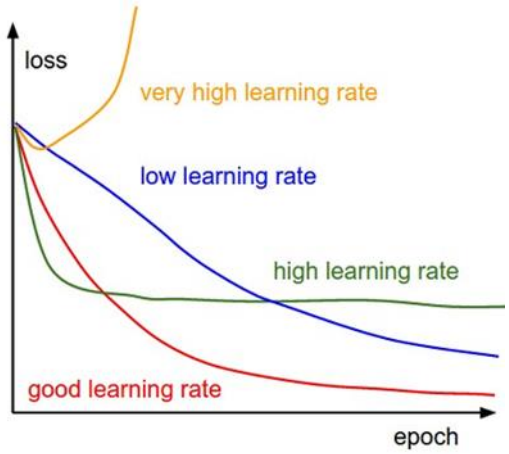
$$v_t = \gamma v_{t-1} + \eta \nabla_\theta J(\theta)$$

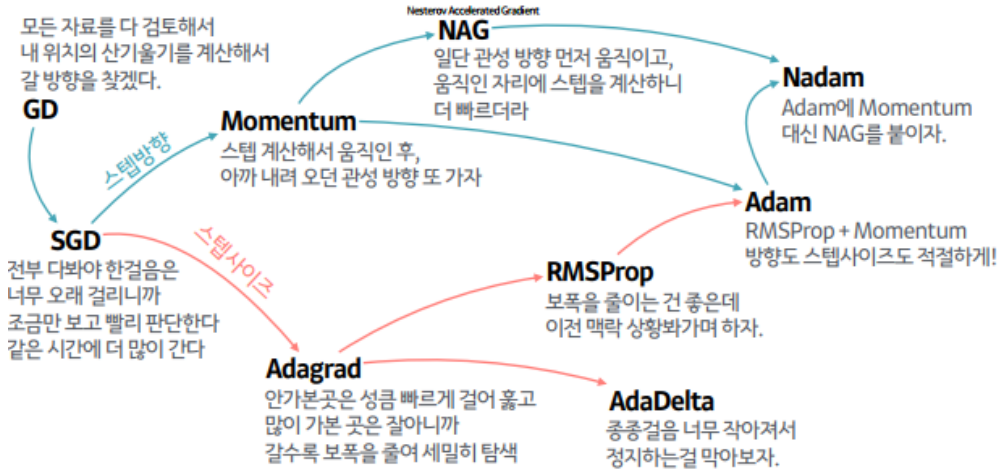
$$\theta = \theta - v_t$$

$$v_t = \eta \nabla_\theta J(\theta)_t + \gamma \eta \nabla_\theta J(\theta)_{t-1} + \gamma^2 \eta \nabla_\theta J(\theta)_{t-2} + \dots.$$



  - Adaptive
    - Large step size for the variables with small change so far



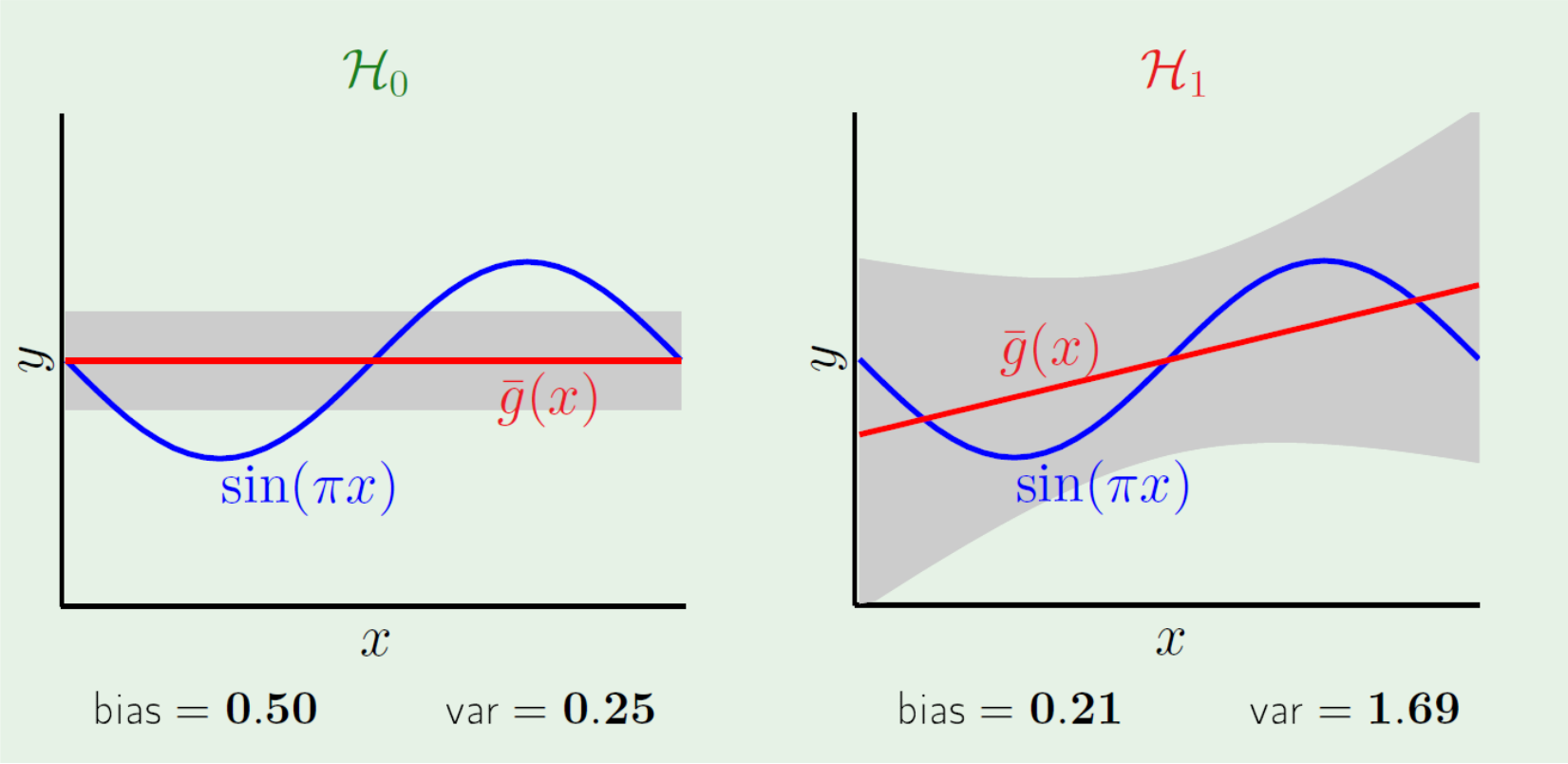


https://seamless.tistory.com/38

3. Overfitting

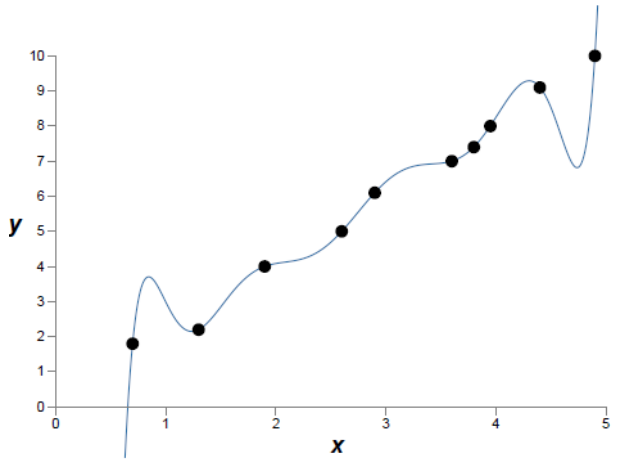
Bias-Variance Tradeoff

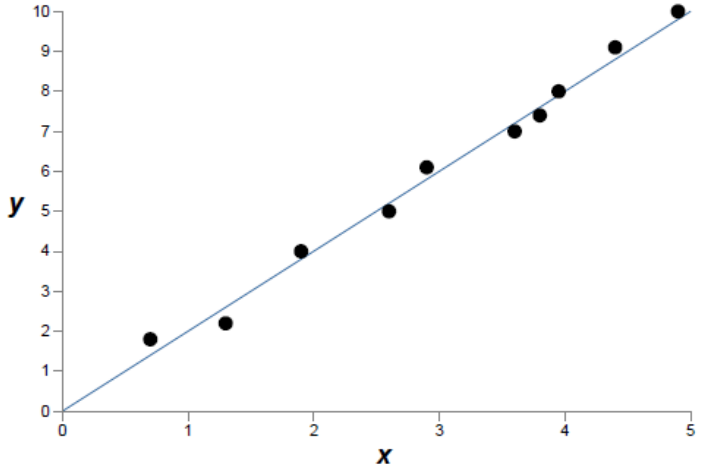
- Bias: Underfitting
- Variance: Overfitting



Overfitting

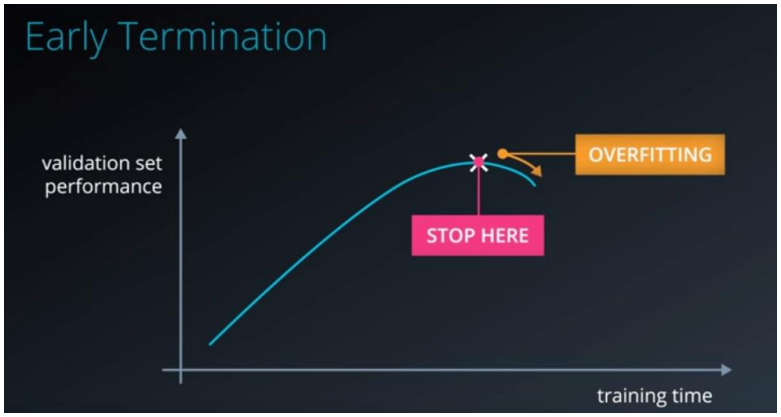
- Which one fits better?



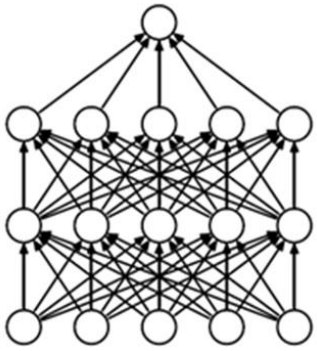


3. overfitting?

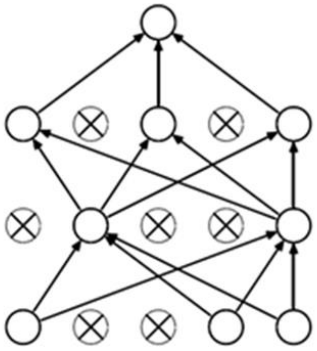
- Early Termination



- Regularization
  - L1, L2 normalization
  - Dropout
    - Remove some hidden nodes



(a) Standard Neural Net



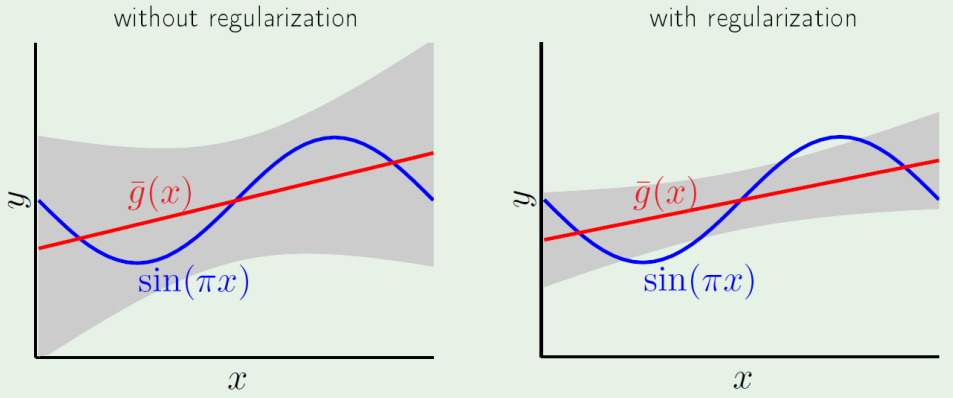
(b) After applying dropout.

Regularization

- L1 Regularization
  - L1 penalty uses the sum of the absolute weight.
  - $min\ (Y - BX)^T(Y - BX) + \lambda\|B\|_1$
- L2 Regularization
  - L2 penalty is based on the squared weights
  - $min\,(Y - BX)^T(Y - BX) + 0.5\lambda B^T B$

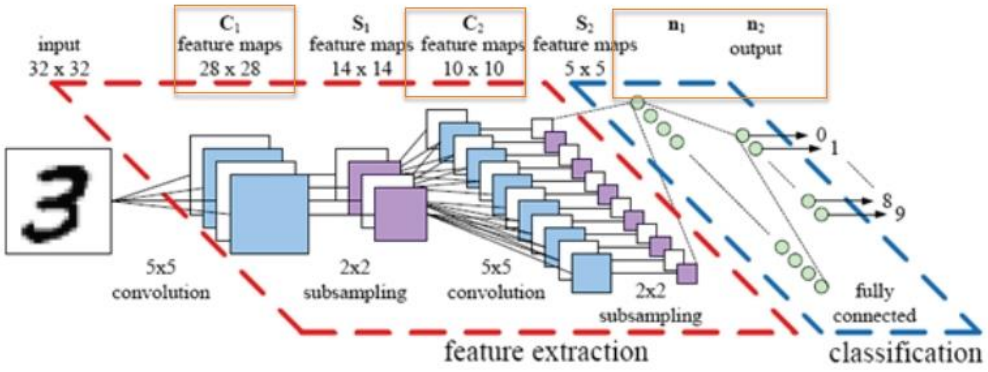
Minimizing $E_{\text{aug}}(\mathbf{w}) = E_{\text{in}}(\mathbf{w}) + \frac{\lambda}{N}\mathbf{w}^{\mathsf{T}}\mathbf{w}$

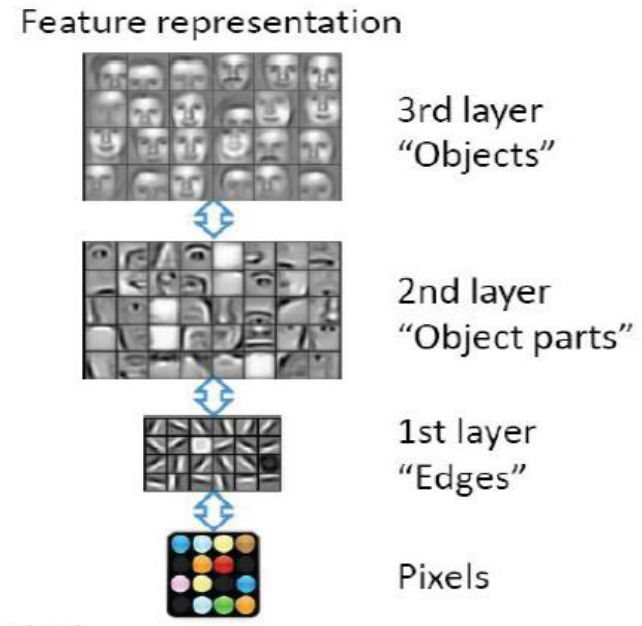
$= \frac{1}{N}(\mathrm{Z}\mathbf{w} - \mathbf{y})^{\mathsf{T}}(\mathrm{Z}\mathbf{w} - \mathbf{y}) + \frac{\lambda}{N}\mathbf{w}^{\mathsf{T}}\mathbf{w}$

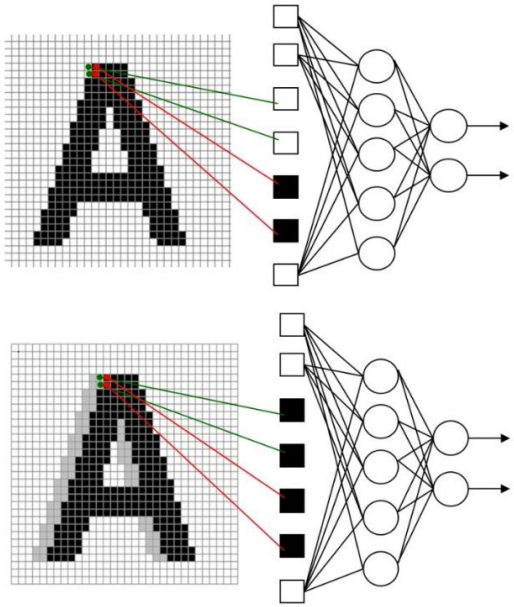


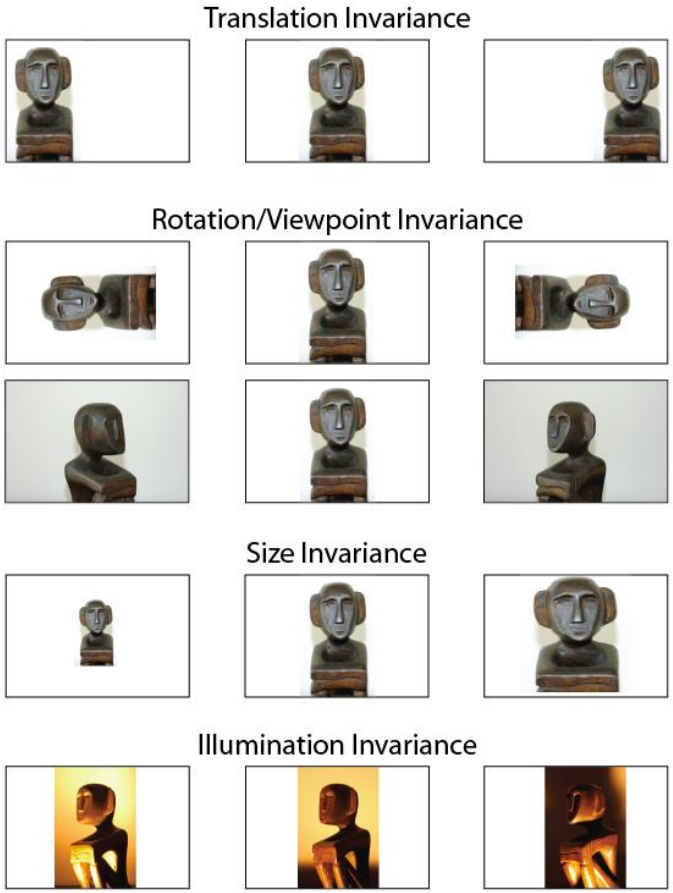
Deep learning architecture

- CNN (Convolutional NN)
  - Repeat convolution and pooling

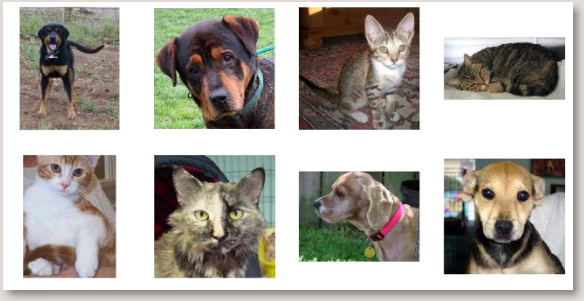






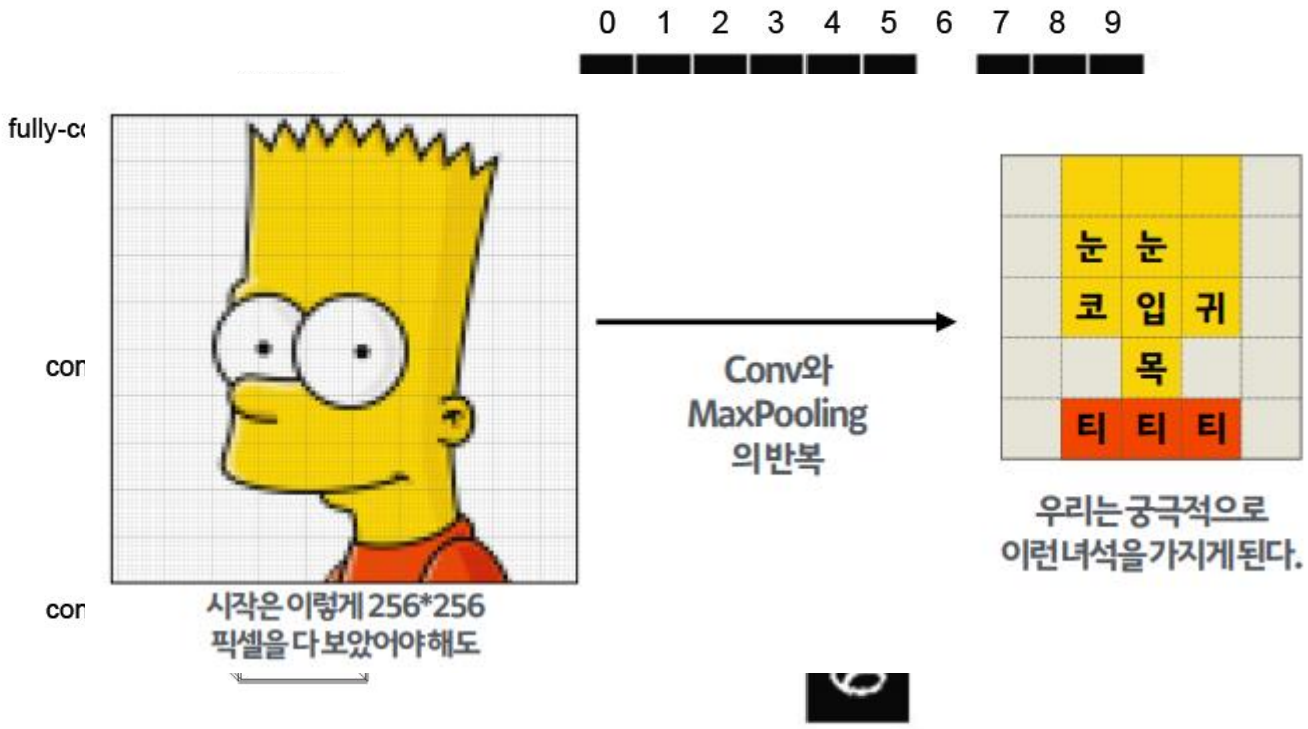

Translation Invariance
Rotation/Viewpoint Invariance
Size Invariance
Illumination Invariance
Matt Krause
mattkrau.se

- How do human know the classification?



Which one is cat and which is dog and how do you know ?

How computer can know ?

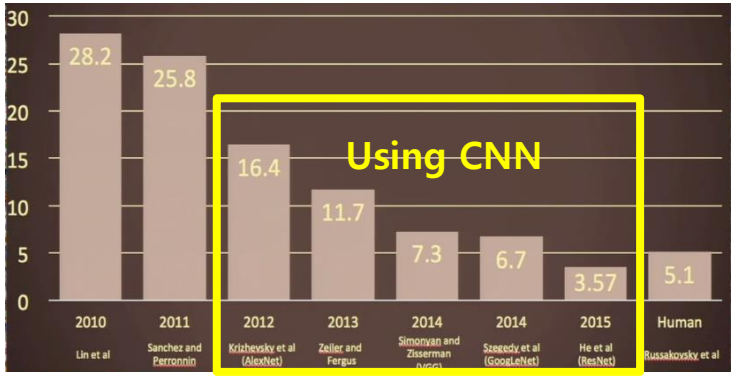

0 1 2 3 4 5 6 7 8 9
fully-c
cor
cor
시작은 이렇게 256*256
픽셀을 다 보았어야해도
Conv와
MaxPooling
의반복
눈 눈
코 입 귀
목
티 티 티
우리는 궁극적으로
이런녀석을 가지게 된다.

History of CNN

- CNN (Convolutional Neural Networks) was invented by Yann LeCun in 1998
- The first paper title: Gradient-Based Learning Applied to Document Recognition
- The popularity of CNN (Deep Learning) was extremely increasing after it always won in ***ImageNet Challenge*** (A compu ter vision world-cup) in 2012

For text recognition

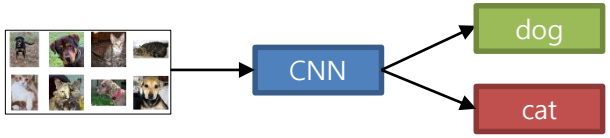
Error rate



Fe-fei Li, et. al, CNNn23 Stanford Univ. 2017

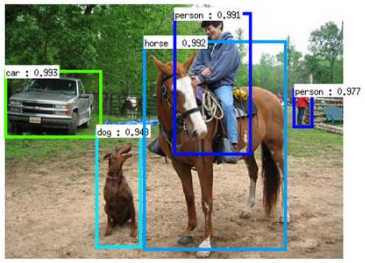
Application of CNN

- Image Classification



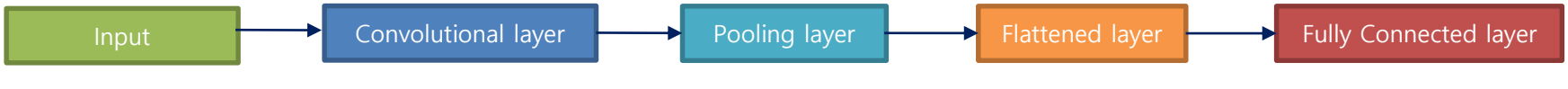
- OCR (Optical Character Recognition)

- Object Detection



- Image Generation

CNN Architecture



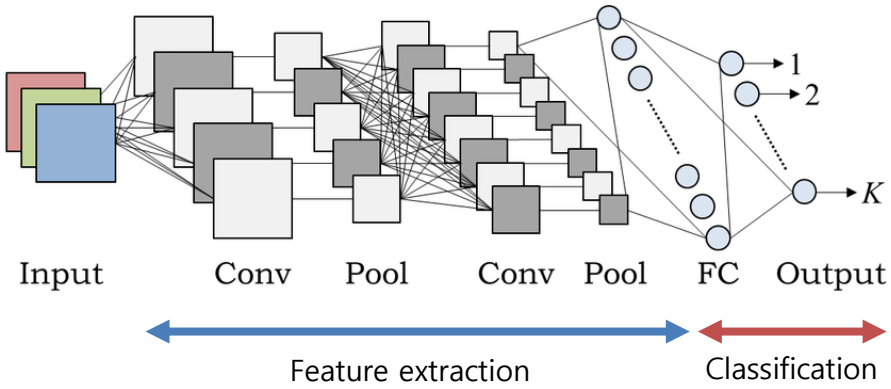
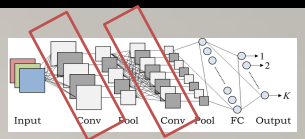
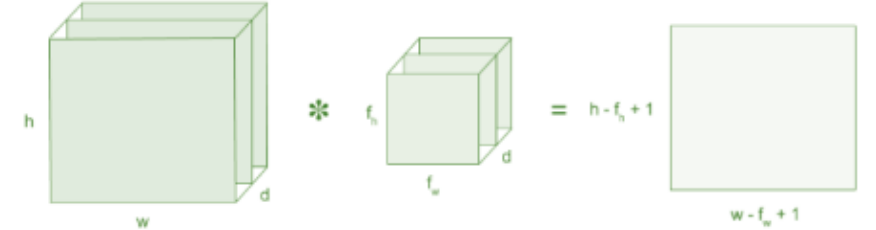


Image source: Akinori Hidaka, Takio Kurita, *Consecutive Dimensionality Reduction by Canonical Correlation Analysis for Visualization of Convolutional Neural Networks*

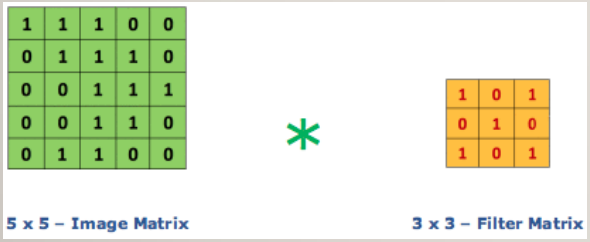
Convolution Layer (1)

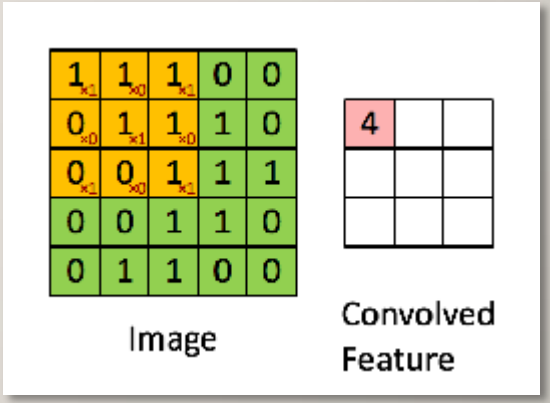




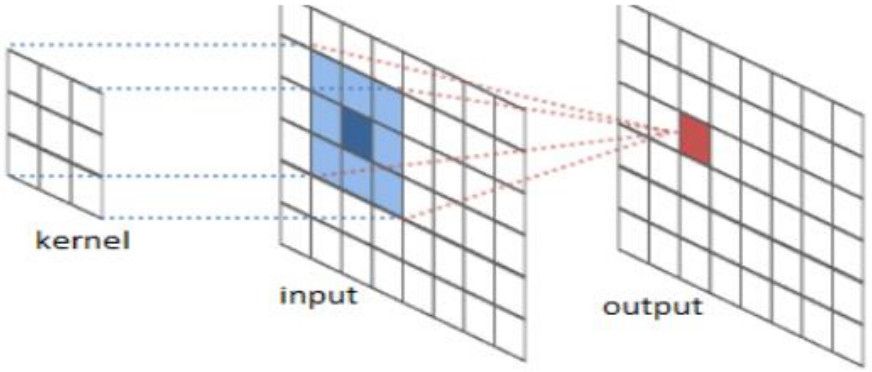
- An image matrix (volume) of dimension **(h x w x d)**
- A filter **(f_h x f_w x d)**
- Outputs a volume dimension **(h - f_h + 1) x (w - f_w + 1) x 1**

- filter/kernel
- Receptive field





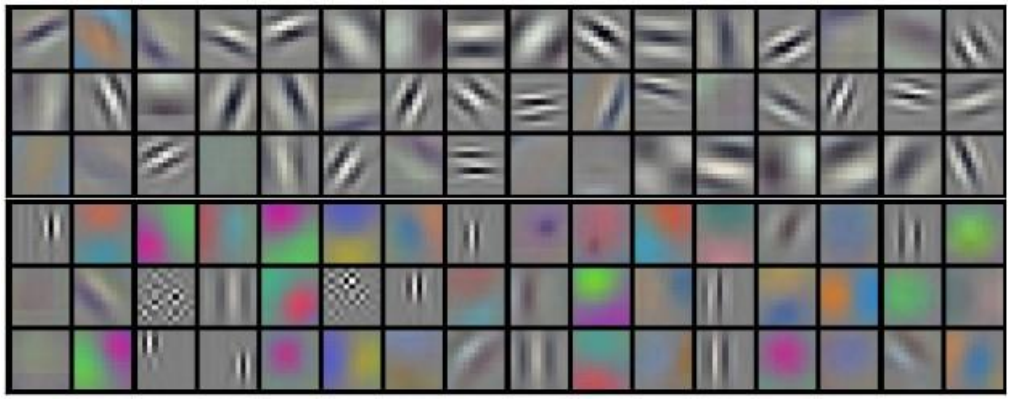
Convolution Layer (2)

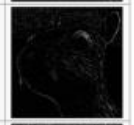


Convolutional Layer (3)

The output of convolutional layer is the important features like edge, curve, dominant colour etc.

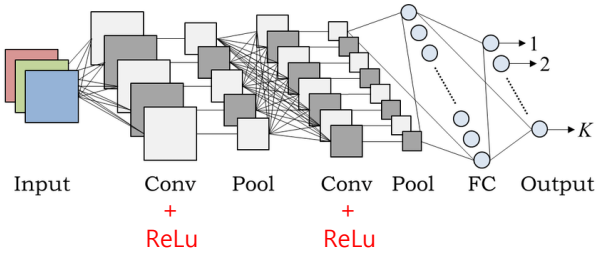
- Filter / kernels

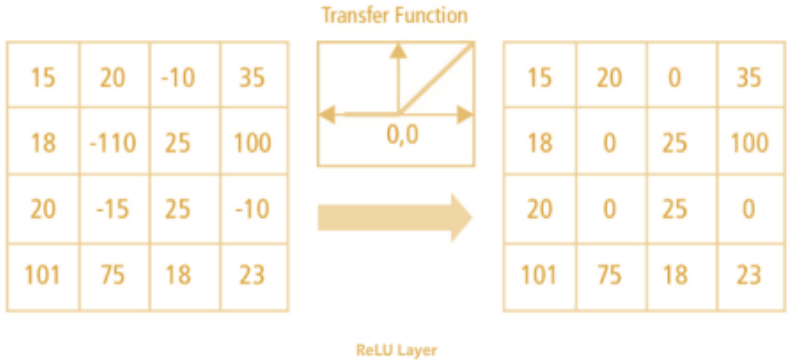


Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

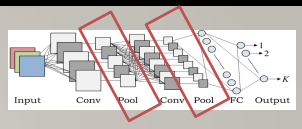
ReLu Activation Function

- ReLU stands for Rectified Linear Unit for a non-linear operation.
- $f(x) = max(0,x)$



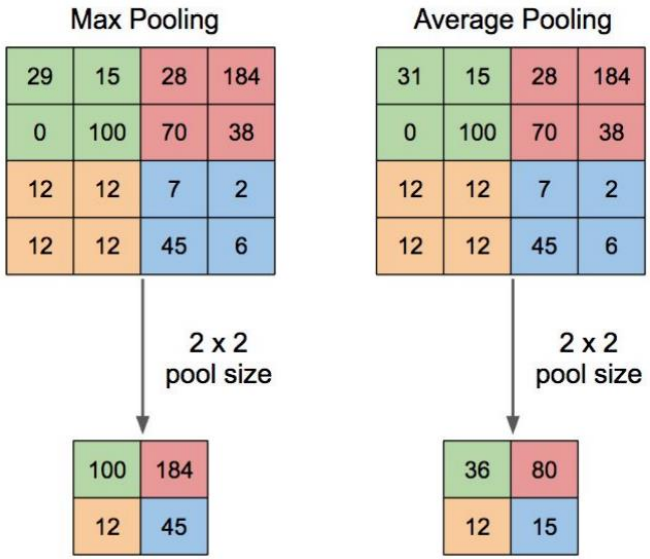


Pooling Layer

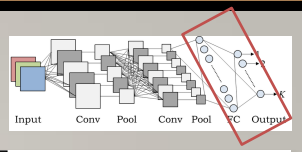


- Pooling layer applies non-linear downsampling on activation maps.
- Pooling is aggressive in discarding information

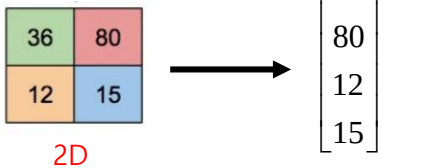
Types of pooling layer:
- Max Pooling
- Average Pooling
- Sum Pooling



Fully Connected Layer (1)



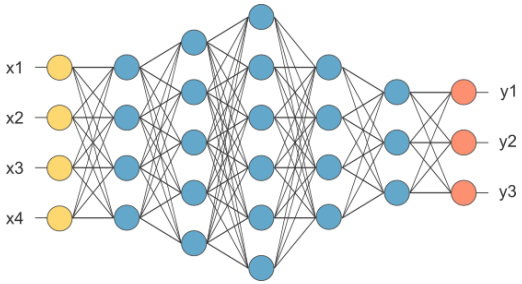
- Neural Networks !
- **Flattened** the output from convolutional layer



2D

Vector 1D

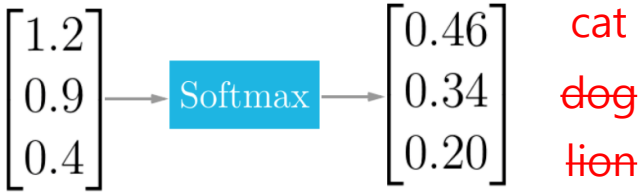
- The fully connected networks is used for feature classification
- The last layer uses **SoftMax** activation function (for classification)

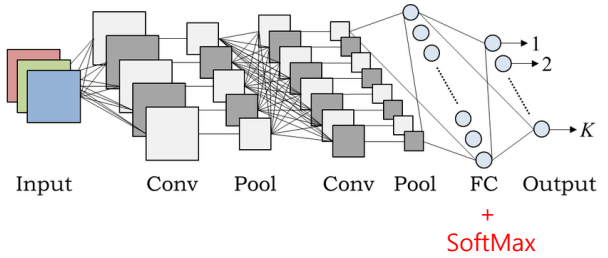


How many neuron in each layer?

Fully Connected Layer (2)

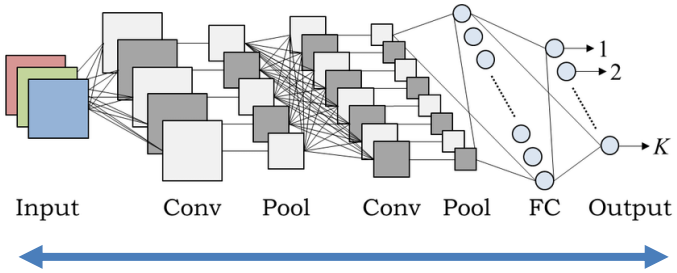
- The softmax function squashes the outputs of each unit to be between 0 and 1
- The output of the softmax function is equivalent to a categorical probability dist ribution
- Index having highest value represent the output





Learning

- What dimension and kind of filter/kernel that we use ?
- What is the weight value that we use ?

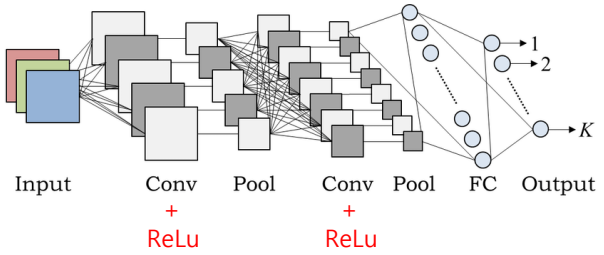


Back-propagation as supervised learning!

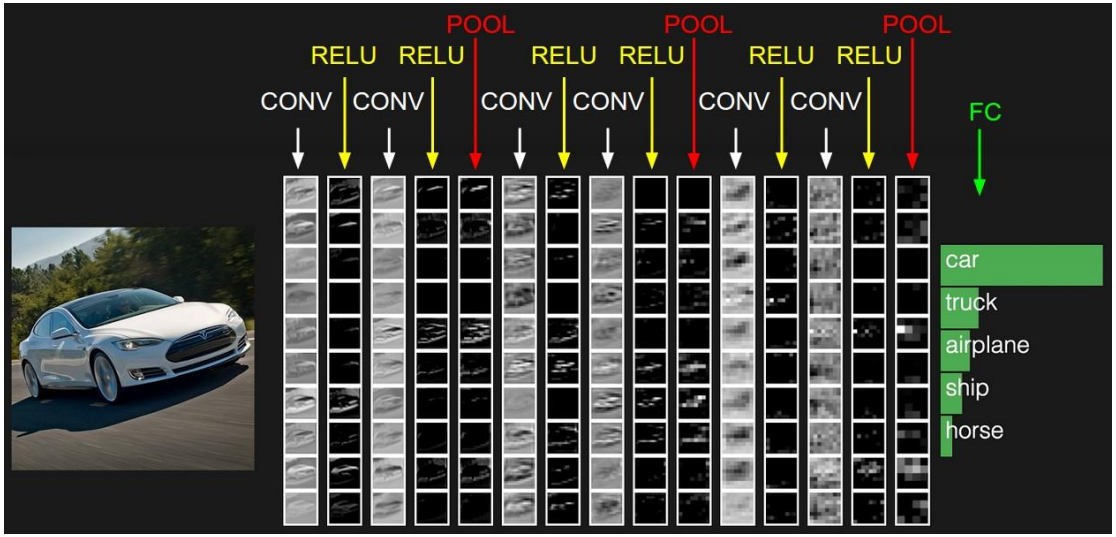
- We just need to initialize the value, and let the CNN through learning set the fi nal filter and weight value itself.
- However, the number of layer and type of activation function are chosen based on your trial experiments.

Summary of How CNN works (1)

1. Provide input image into convolution layer
2. Choose parameters, apply filters with strides, padding if requires. Perform convolution on the image a nd apply ReLU activation to the matrix.
3. Perform pooling to reduce dimensionality size
4. Add as many convolutional layers until satisfied
5. Flatten the output and feed into a fully connected layer (FC Layer)
6. Output the class using an activation function (SoftMax) and classifies images.



Summary of How CNN works (2)



Advance of CNN

- How to choose the best architecture ? it depends on our case
- There is no standard architecture, like how many conv. Layer, pooling layer, strides, and number of fully connected layer. Do a trial experiment !