Deep Learning

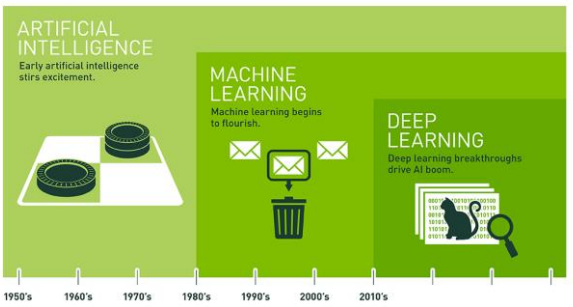
2020

Ando Ki, Ph.D.
adki@future-ds.com

Contents

- AI, ML, and DL
- Types of learning of machine learning
- Data sets
- Deep learning design flow
- Popular deep neural networks
- Data sets
  - MNIST
  - CIFAR-10
  - ImageNet
- Popular frameworks and popularity

AI, ML, and DL



Deep Learning is a domain of Machine Learning and they are state-of-the-art approaches of AI (source: NVIDIA Blog)

딥러닝은 머신러닝의 한 방법이며, 이들은 인공지능의 최신기술이다.

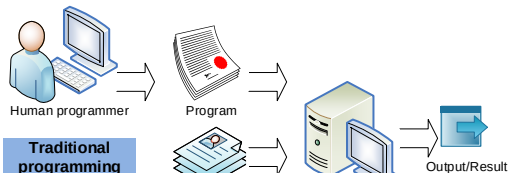
- Artificial Intelligence — Human intelligence exhibited by machines
- Machine Learning — An approach to achieve artificial intelligence
  - algorithmic approaches: decision tree learning, inductive logic programming, clustering, reinforcement learning, and Bayesian networks, Artificial Neural Networks
- Deep Learning — A technique for implementing machine learning based on artificial deep neural network

https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/

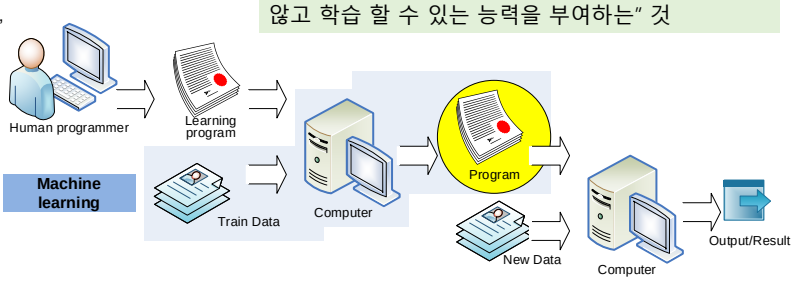
3

ML

- Machine learning (기계학습)
  - ML gives computers the abilities to learn without being explicitly programmed to complete a task. – Arthur Samuel, 1959.
    - (기계학습은 컴퓨터가 '특정한 일을 처리하도록' '명시적인 프로그래밍 없이' '배울 수 있게하는 능력')
    - 기계학습이란 "컴퓨터에 명시적으로 프로그래밍하지 않고 학습 할 수 있는 능력을 부여하는" 것
  - ML is designed to take a large set of data, analyze it, and learn from it.
  - ML systems recognize patterns, understand speech, and make predictions.

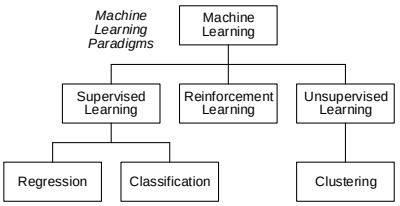


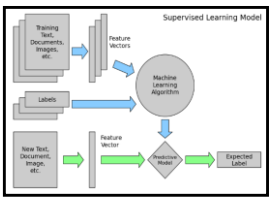
기계학습이란 "컴퓨터에 명시적으로 프로그래밍하지 않고 학습 할 수 있는 능력을 부여하는" 것

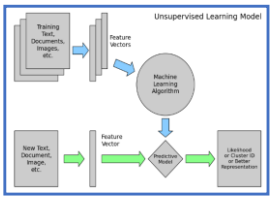


4

Types of learning of machine learning







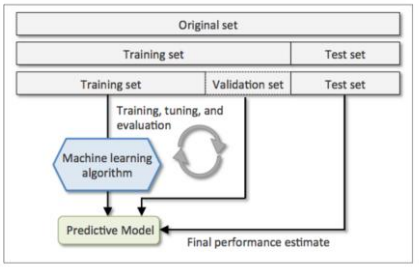
- Supervised learning (지도학습)
  - Labelled data (metrics) is already given to the computer.
  - Solving two types of problems
    - Regression problem: target variable is continuous
    - Classification problem: target variable is categorical
- Unsupervised learning (자율학습/비지도학습)
  - Finding hidden structures in datasets without any labels
  - Data is clustered using several clustering algorithms
  - Ex) Google News, Social Network Analysis, translation
- Reinforcement learning (강화학습)
  - No data given. Agent interacts with the environment calculating cost of actions.
  - Network is only provided with a grade, or score, which indicates network performance.
  - Gives reward instead of label
  - Action selection, policy learning, gaming
  - Ex) Google AlphGo

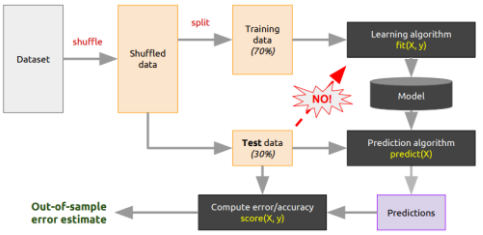
http://www.astroml.org/sklearn_tutorial/general_concepts.html

5

Data sets

- Training set
  - data for model building by training
  - a set of examples used for learning, where the target value is known.
- Validation set
  - data for estimating error while training
  - It should not be the same as training set and used as training.
  - a set of examples used to tune the architecture of a classifier and estimate the error.
- Test set
  - data for estimating error
  - used only to assess the performances of a classifier. It is never used during the training process so that the error on the test set provides an unbiased estimate of the generalization error.





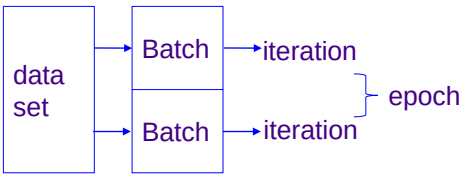
- Training error: error by training data set
- Generalization error (test error, out-of-sample error): error by test data set in order to evaluate the training model.

6

Data set, batch, epoch and iterations

- Data set
  - training data and validation data
  - it will be large amount of data.
- Batch
  - Training data set is divided into a number of parts.
  - The part of data set is 'batch'.
- Epoch
  - One Epoch is when an ENTIRE dataset is passed forward and backward through the neural network only ONCE.
  - Since, one epoch is too big to feed to the computer at once we divide it in several smaller batches.
- Iteration
  - The number of batches needed to complete one epoch

- We can divide the dataset of 2000 examples into batches of 500 then it will take 4 iterations to complete 1 epoch.
- If you have 1000 training examples, and your batch size is 500, then it will take 2 iterations to complete 1 epoch.



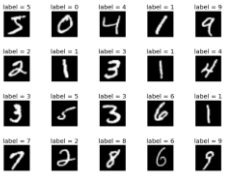
Popular Deep Neural Networks

Image classification case

	LeNet-5	AlexNet	GoogLeNet (V1)	ResNet-50	ResNet-152
Data set	MNIST	ImageNet	ImageNet	ImageNet	ImageNet
Purpose	Handwritten digit classification	Image classification	Image classificati on	Image classifica tion	Image classifica tion
Error (%)[Human]	0.95 [0.2~0.3]	16.4 [5]	6.7	5.3	3.57
Year	1998	2012	2014	2015	2015
Image size	28x28	227x227	224x224	224x224	
Layers	4	8	22	50	152
Weights	431k	61M	7M	25.5M	??
MACs	2.3M	724M	1.43G	3.9G	??
Training time		a week			
Inference time					
etc		2 GPU			

Data set: MNIST

- The MNIST database (Modified National Institute of Standards and Technology database)
  - a large database of handwritten digits that is commonly used for training various image processing systems.
- Digit classification
  - 28x28 pixels (B&W)
  - 10 classes: 0, 1, …, 9
  - training set: 60,000 training image
  - test set: 10,000 testing imag



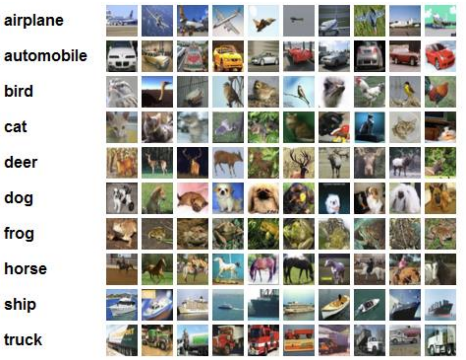
Type	Classifier	Error rate (%)
Linear classifier	Pairwise linear classifier	7.6
K-NN	K-Nearest Neighbors	0.52
Neural network	2-layer	1.6, 0.7
Deep NN (LeNet-5)	5-layer	0.95
Deep NN	6-layer	0.35
Convolutional NN	6-layer	0.31~0.21

1998: LeNet, 0.95% error
2013: ICML, 0.21% error

http://rodrigob.github.io/are_we_there_yet/build/classification_datasets_results.html

Data set: CIFAR-10/CIFAR-100

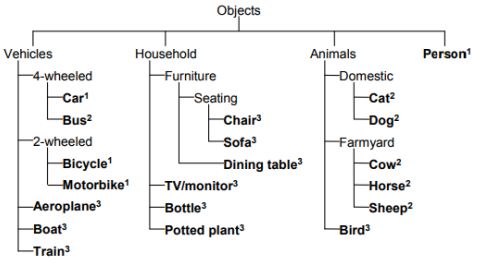
- CIFA: Canadian Institute For Advanced Research
- CIFA-10
  - Object classification
  - image dataset consists of 60,000 (32x32-pixesl/image) color images in 10 classes, with 6,000 images per class.
  - 32x32 pixels (color)
  - 10 classes containing 6,000 images each
  - 50,000 training
  - 10,000 testing
- CIFA-100
  - 100 classes containing 600 images each



https://www.cs.toronto.edu/~kriz/cifar.html

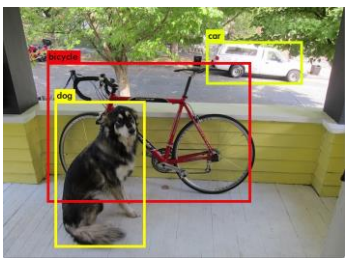
Data set: VOC2012

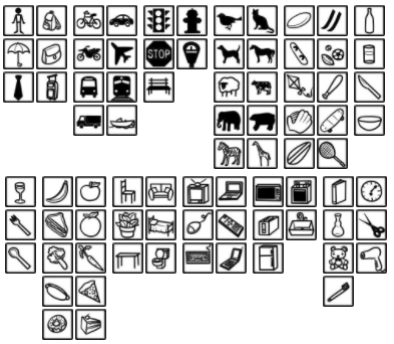
- VOC: PASCAL Visual Object Classes
  - Pattern Analysis, Statistical Modeling and Computational Learning
  - http://host.robots.ox.ac.uk/pascal/VOC/
  - VOCO: 2005~2012
  - VOC2012
    - 20 classes
    - 11k images
    - The train/val data has 11,530 images containing 27,450 ROI (region of interest) annotated objects and 6,929 segmentations.



Data set: COCO

- COCO: Common Objects in Context
  - 100k images
  - 80 classes
  - detection labels
  - http://cocodataset.org/#home

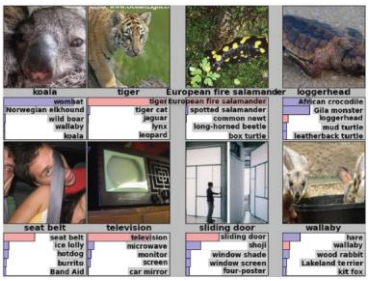




Data set: ImageNet and ILSVRC

- The **ImageNet** project is a large visual database designed for use in visual object recognition software research since 2009.
  - http://www.image-net.org/
  - Over 15M labeled high resolution images
    - Annotated
  - 256x256 pixels (color)
  - Roughly 22K categories
  - Collected from web and labeled by Amazon Mechanical Turk (Mturk)

- ILSVRC: ImageNet Large-Scale Visual Recognition Challenge
  - An annual software contest run by ImageNet project since 2010
  - http://image-net.org/challenges/LSVRC/
  - 150K images, 1K object classes
  - Error by human: ~5% (classification)

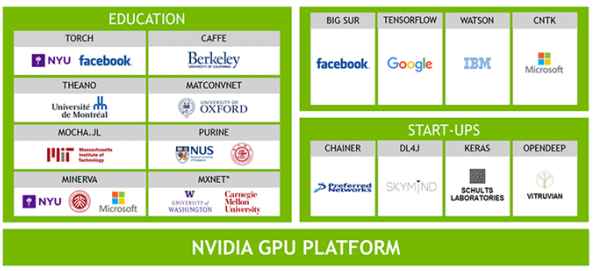


13

Popular Frameworks

- Popular Frameworks with supported interfaces
  - Caffe
    - Berkeley / BVLC (Berkeley Artificial Intelligence Research)
    - C, C++, Python, Matlab
  - TensorFlow
    - Google Brain
    - C++, Python
  - PyTorch
  - theano
    - U. Montreal
    - Python
  - torch
    - Facebook / NUU
    - C, C++, Lua
  - CNTK
    - Microsoft
  - MXNet
    - Carnegie Mellon University / DMLC (Distributed Machine Learning Community)

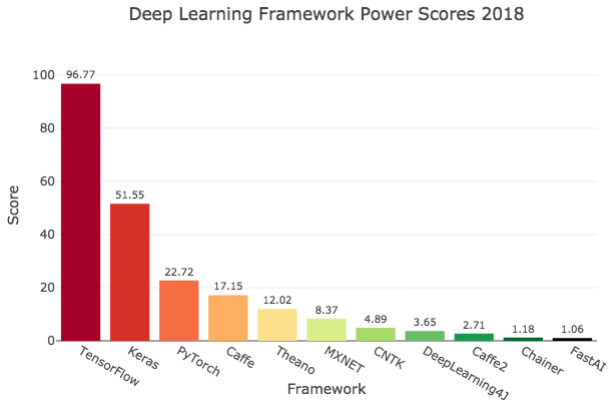
https://developer.nvidia.com/deep-learning-frameworks



https://blogs.nvidia.com/blog/2016/01/12/accelerating-ai-artificial-intelligence-gpus/

14

Popularity



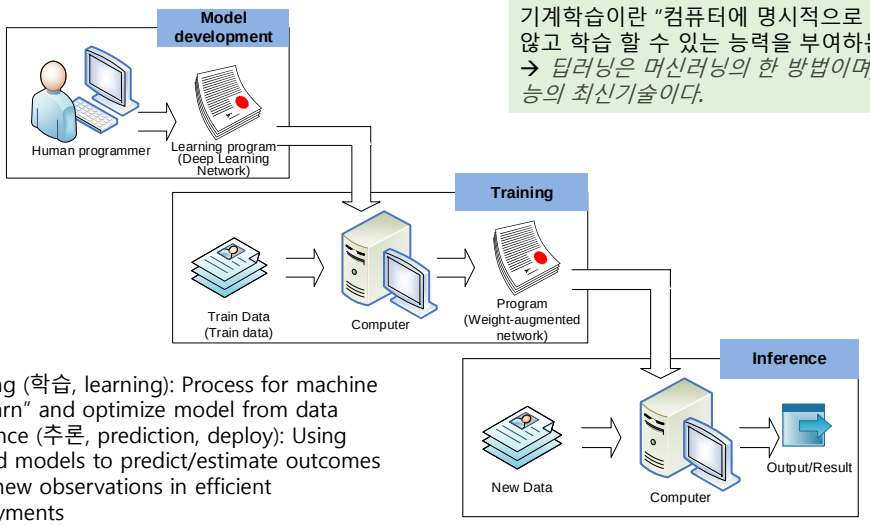
Deep Learning Framework Deep Learning Framework Power Scores (by Jeff Hale) http://bit.ly/2GBa3tU

https://towardsdatascience.com/deep-learning-framework-power-scores-2018-23607ddf297a

1. TensorFlow
2. Keras
3. PyTorch
4. Caffe
5. theano
6. mxnet
7. CNTK
8. DL4J
9. Caffe2
10. Chainer
11. fast.ai

15

Deep Learning Design Flow



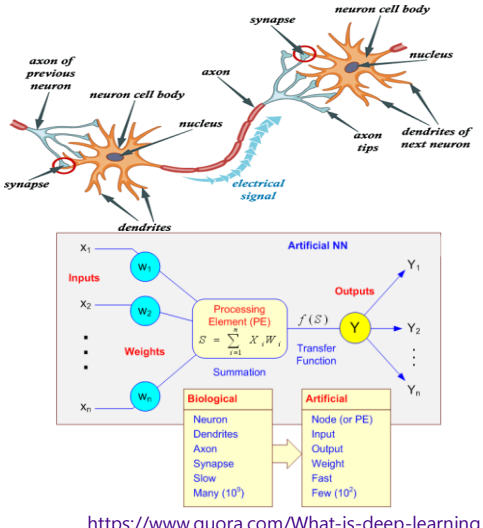
기계학습이란 "컴퓨터에 명시적으로 프로그래밍하지 않고 학습 할 수 있는 능력을 부여하는" 것
→ *딥러닝은 머신러닝의 한 방법이며, 이들은 인공지능의 최신기술이다.*

- Training (학습, learning): Process for machine to "learn" and optimize model from data
- Inference (추론, prediction, deploy): Using trained models to predict/estimate outcomes from new observations in efficient deployments

16

Table of contents

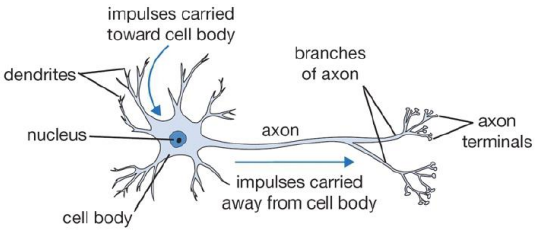
Modeling a neuron

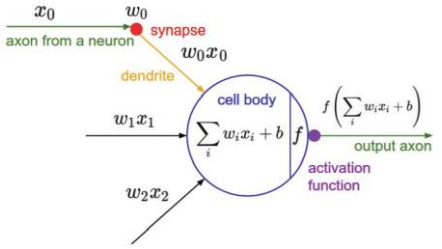


https://www.quora.com/What-is-deep-learning

- Neuron: 신경세포(神經細胞)
  - Dendrite: 수상돌기(樹狀突起)
    - input
  - Axon: 축삭돌기(軸索突起)
    - output
    - Branches of axon
    - Terminals of axon (axon tip)
      - synaptic knob
  - Synapse: 연접
    - junction between two nerve cells
- Human
  - whole brain
    - ~86 billion neurons (Giga, 10^9)
    - ~100 trillion synapses (Tera, 10^{12})
  - cerebral cortex: 대뇌피질
    - 19~23 billion neurons

Modeling a neuron





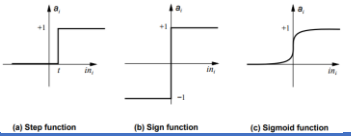
- Activation functions

https://en.wikipedia.org/wiki/Activation_function

sigmoid

Identity		$f(x) = x$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Logistic (a.k.a. Soft step)		$f(x) = \frac{1}{1+e^{-x}}$
TanH		$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$
ArcTan		$f(x) = \tan^{-1}(x)$
Softsign [7][8]		$f(x) = \frac{x}{1+\lvert x\rvert}$
Rectified linear unit (ReLU)[9]		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$

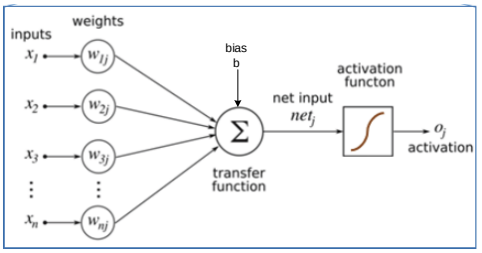
sigmoid

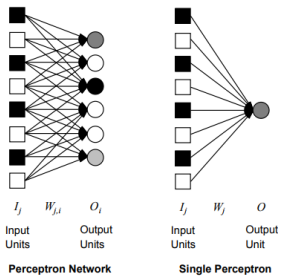


(a) Step function (b) Sign function (c) Sigmoid function

Perceptron: single layer neural network

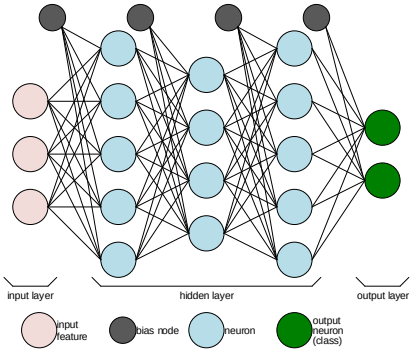
- Perceptron is a single artificial neuron that computes its weighted input and uses a threshold activation function.
  - It is also called a TLU (threshold logic unit).
  - It effectively separates the input space into two categories by the hyperplane: W*X+b = 0
  - Perceptron is a linear classifier.
    - Cannot deal with non-linear cases
  - Perceptron refers to a particular supervised learning model with backpropagation learning algorithm.
  - Perceptron is an algorithm for supervised learning of binary classifiers.





Layer-wise organization: MLP (multi-layer perceptron)

- 3 types of layers
  - Input layer
  - hidden layer
  - output layer



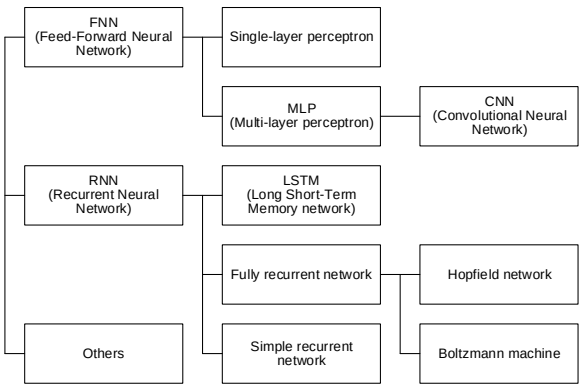
fully-connected multi-layered neural network

- input layer: not counted for the number of layers
- hidden layer
- output layer

- For the picture on the left
  - assume fully connected
  - 4-layered including 3-hidden layers
  - 16 neurons: 5+4+5+2
  - 65 weights: 3x5+5x4+4x5+5x2
    - ➲ not including bias
  - 16 biases: 5+4+5+2
  - 82 learnable parameters: 65+16

- Modern neural network
  - 10~20 layers, ~100 million parameters
  - How about 125 layers?

Categories of ANN (Artificial Neural network)



- Fully-Connected NN
  - feed forward
  - Multi-Layer Perceptron (MLP)
- Convolutional NN (CNN)
  - feed forward, sparsely-connected
  - Image recognition
  - AlphaGo
- Recurrent NN (RNN)
  - feedback
- Long Short-Term Memory (LSTM)
  - feedback + storage
  - Microsoft speech recognition
  - Google neural machine translation (GNMT)

See neural network topology: http://www.asimovinstitute.org/neural-network-zoo/

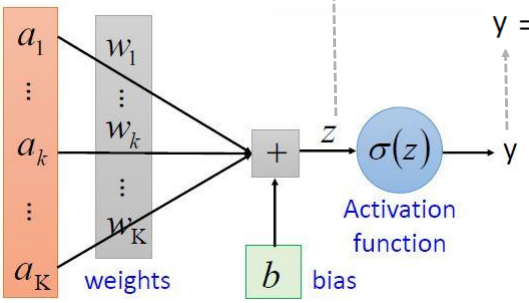
Artificial neuron: Perceptron

- Artificial Neuron: Perceptron
  - inputs
  - output
  - weights
  - bias
  - activation function

$$(a_1, a_2, \cdots, a_K) \times \begin{pmatrix} W_1 \\ W_2 \\ \vdots \\ W_K \end{pmatrix} + b = z$$

$$z = a_1 w_1 + \cdots + a_k w_k + \cdots + a_K w_K + b$$

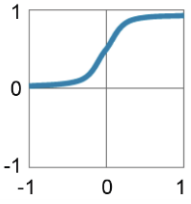
$$y = \sigma(z)$$



23

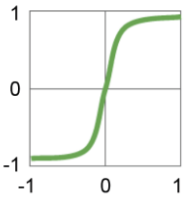
Artificial neuron: activation functions

Logistic, soft step
Sigmoid



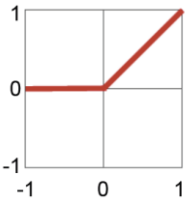
$$y = \frac{1}{1 + e^{-z}}$$

Hyperbolic Tangent



$$y = \frac{2}{1 + e^{-2z}} - 1$$

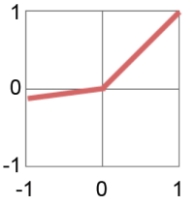
Rectified Linear Unit (ReLU)



$y = 0 \; for \; z < 0$
$y = z \; for \; z \geq 0$

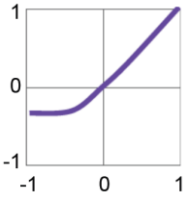
y=max(x,0)

Leaky ReLU



$y = 0.01z \; for \; z < 0$
$y = z \; for \; z \geq 0$

Exponential LU



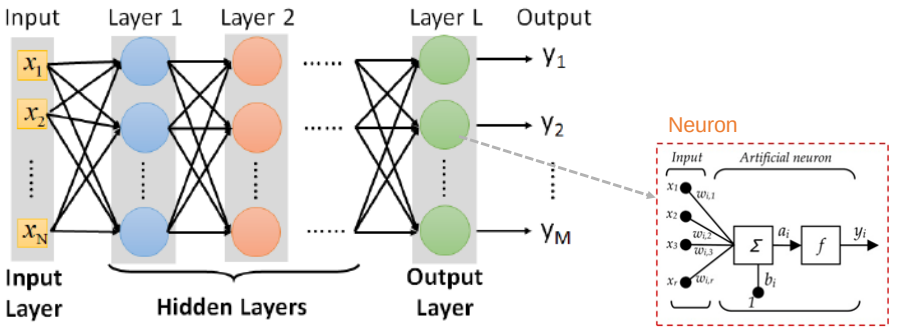
$y = \alpha(e^z - 1) \; for \; z < 0$
$y = z \; for \; z \geq 0$

$$\frac{dy(z)}{dz} = \frac{d}{dz}\left[\frac{1}{1 + e^{-z}}\right] = \frac{d}{dz}(1 + e^{-z})^{-1} = -(1 + e^{-z})^{-2}(-e^{-z}) = y(z) \cdot (1 - y(z))$$

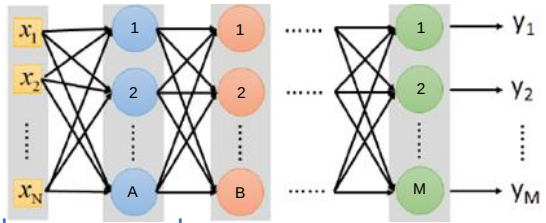
24

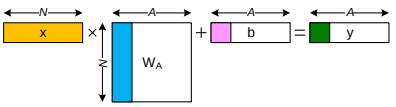
Artificial Neural Network: ANN

- Artificial Neural Network: ANN
  - Network structure by different connections
  - Each neuron can has different values of weights and bias
  - Weights and biases are network parameter θ .



Artificial Neural Network: ANN





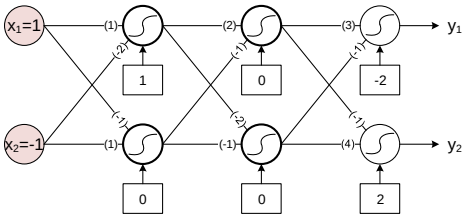
N: number of inputs
A: number of hidden layers

$$\begin{bmatrix} X_1 & X_2 & \cdots & X_N \end{bmatrix} \times \begin{bmatrix} w_{1,1} & w_{2,1} & \cdots & w_{A,1} \\ w_{1,2} & w_{2,2} & \cdots & w_{A,N} \\ \vdots & & & \vdots \\ w_{1,N} & w_{2,N} & \cdots & w_{A,N} \end{bmatrix} + \begin{bmatrix} b_1 & b_2 & \cdots & b_A \end{bmatrix} = \begin{bmatrix} y_1 & y_2 & \cdots & y_A \end{bmatrix}$$

$$\begin{bmatrix} w_{1,1} & w_{2,1} & \cdots & w_{A,1} \\ w_{1,2} & w_{2,2} & \cdots & w_{A,N} \\ \vdots & & & \vdots \\ w_{1,N} & w_{2,N} & \cdots & w_{A,N} \end{bmatrix}^T \times \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_N \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_A \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_A \end{bmatrix}$$

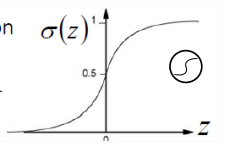
Fully connected feed-forward network: FC-FFN

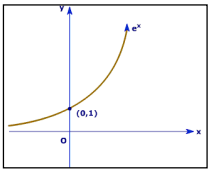
- Activation function: E.g., Sigmoid – S-shaped function



Sigmoid Function

$$\sigma(z) = \frac{1}{1+e^{-z}}$$





$$[1, -1] \times \begin{bmatrix} 1, -1 \\ -2, 1 \end{bmatrix} + [1, 0] = [4, -2] \xrightarrow{\text{sigmoid}} [0.98, 0.12]$$

$$[0.98, 0.12] \times \begin{bmatrix} 2, -2 \\ -1, -1 \end{bmatrix} + [0, 0] = [1.84, -2.08] \xrightarrow{\text{sigmoid}} [0.86, 0.11]$$

$$[0.86, 0.11] \times \begin{bmatrix} 3, -1 \\ -1, 4 \end{bmatrix} + [-2, 2] = [\ ??,\ ??] \xrightarrow{\text{sigmoid}} [\ ??,\ ??]$$

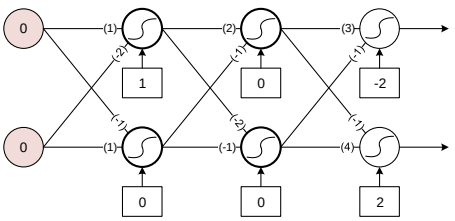
x_1 x_2 y_1 y_2

f ([1, -1]) = [0.62, -0.83]

f ([0, 0]) = [051, 0.85]

Do it yourself

- Calculate the output



Optional output layer: Softmax

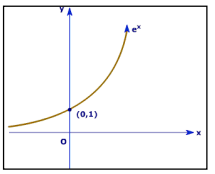
- Outputs of artificial neural network will be any values from very small to very large including negative.

f ([1, -1]) = [0.62, -0.83]
f ([0, 0]) = [051, 0.85]

The output can be any value.
→ Hard to interpret.

- Softmax for output layer
  - Softmax is a function to transform a number of values to a range of value to between 0 ~ 1.
    - Score (-inf, inf) ==> probabilities [0,1]
  - Multinomial logistic or normalized exponential function

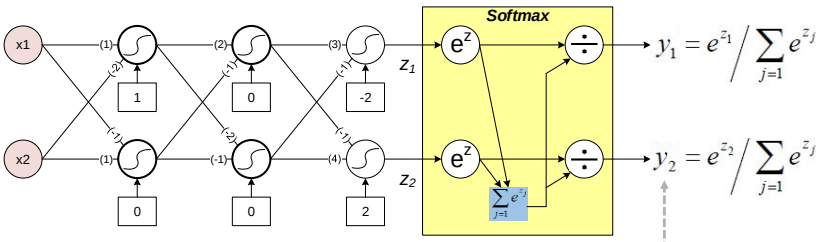
$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad \text{for } j = 1, \ldots, K.$$

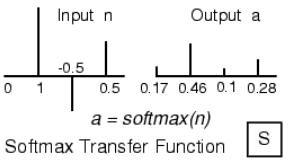


29

Optional output layer: Softmax

- Softmax converts score to probability: Score (-inf, inf) ==> probabilities [0,1]
  - un-normalized probabilities (summation will not give 1): e^{z_j} for result j.
  - normalized probabilities (summation will give 1): -- see below --



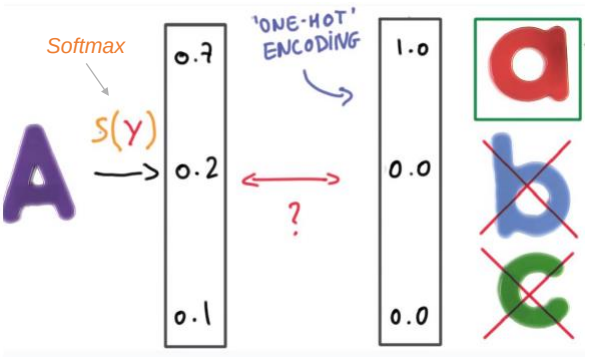


$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad \text{for } j = 1, \ldots, K.$$

30

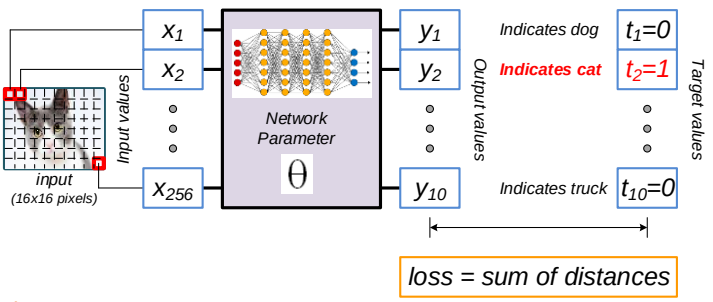
Optional output layer: one-hot encoding

- One-hot encoding by encoding class labels
- Select one only among many.



How to find a good or the best network: Loss/Cost

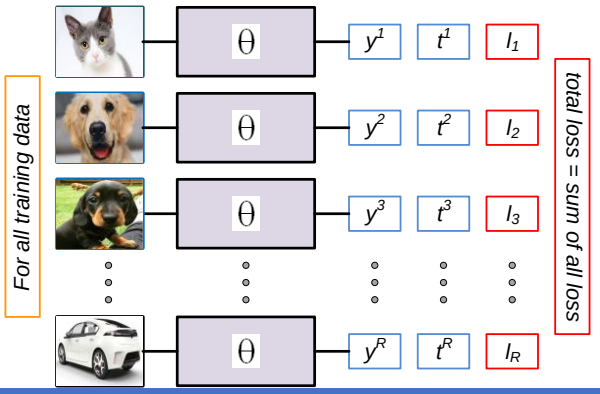
- **Loss function** is the distance between the network output and the target
  - cost function or error function
  - It indicates how good the result is.
  - There can be different loss functions.
    - The simplest one will be a summation of | t – y |.
      - Perfect match will give 0.



- Training error: error by training data set
- Generalization error (test error): error by test data set in order to evaluate the training model.

How to find a good or the best network: Total Loss

- Total loss (L) is a sum of losses (l_r)
  - Make it as small as possible
- Training means to find the network parameter θ that minimize total loss L.
  - This means we should modify the network parameter according to the total loss.



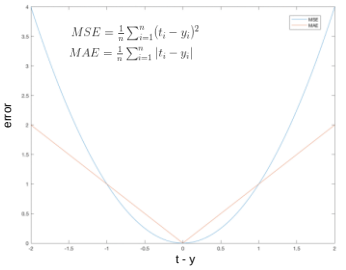
$$L = \sum_{r=1} l_r$$

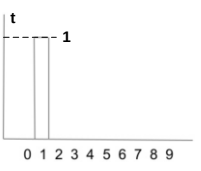
Sum of losses for R test images

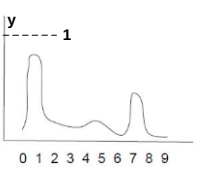
Cost functions (error function)

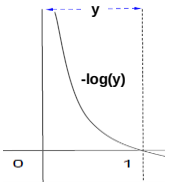
- Absolute error
  - Sum of absolute errors
    - sum(|t – y|)
  - Mean absolute errors (MAE)
    - sum(|t-y|)/n
- Squared error loss
  - Sum of squared errors
    - sum((t - y)**2)
  - Mean squared errors (MSE)
    - sum((t-y)**2)/n
  - Root mean square errors (RMSE)
    - (MSE)**(1/2)
- Cross-entropy loss
  - For classification after Softmax
  - Sum of cross-entropy loss
    - -sum(t*log(y))
      - all except t=1 does not contribute
    - or - sum[t*log(y) + (1-t)*log(1-y)]
      - add cost when t is not 1.

- y: inference value or calculated value
- t: target value





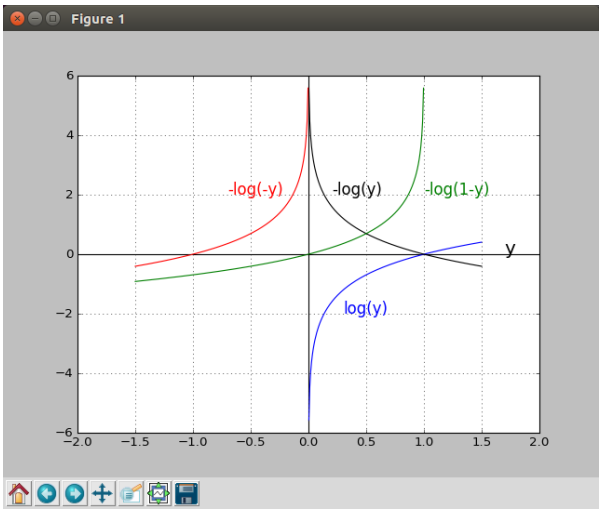




-log(y) emphasizes error (y) when softmax result (y) is small.
y<1 means error, y=1 means correct.

Log plots

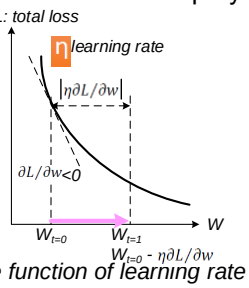
```
mport numpy as np
from matplotlib import pyplot as plt

y = np.linspace(-1.5, 1.5, 400)

plt.plot(y, np.log(y), color='blue')
plt.text(0.3, -2, 'log(y)', fontsize=15, color='blue')

plt.plot(y, -np.log(y), color='black')
plt.text(0.2, 2, '-log(y)', fontsize=15, color='black')

plt.plot(y, -np.log(-y), color='red')
plt.text(-0.7, 2, "-log(-y)", fontsize=15, color='red')

plt.plot(y, -np.log(1-y), color='green')
plt.text(1.0, 2, "-log(1-y)", fontsize=15, color='green')

plt.grid()
plt.show()
```

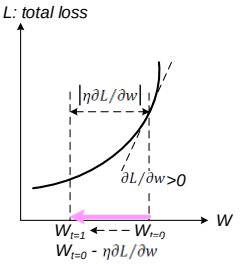


35

How to minimize total loss by changing [W] and [b]

- If we can find how the network parameters affect the total loss, it may be possible to figure out how to minimize the total loss.
- However, the number of parameters is too larger to figure out.
  - AlexNet: 650K neurons, 8 layers, 60 Million parameters
- So we apply gradual iterative progress method in step by step called '**Gradient descent**'. It is called *optimization algorithm.*



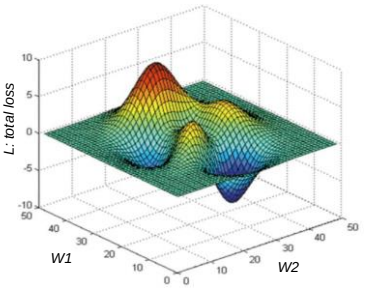


- *Negative slope ➔ increase W by some function of learning rate*
- *Positive slope ➔ decrease W*
- *Steep slope ➔ large change of W for the next time*
- *go on until the slope is small enough, i.e., inflection point*

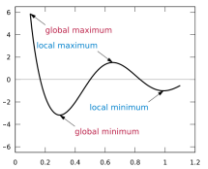
36

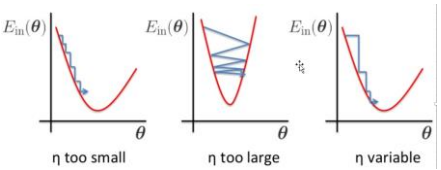
Optimization algorithm: gradient descent

- *Initial value problem*
  - *different initial point leads to different minima*
- *Local minimum problem (get stuck in local minima)*
  - *never guarantee global minima*
- *Learning rate problem*
  - *large learning rate could cause oscillation*
  - *small learning rate results in slow learning*



- *Vanishing gradient problem*
  - If a change in the parameter's value causes very small change in the network's output - the network just can't learn the parameter effectively, which is a problem.
- Gradient Exploding





37

How to compute gradient

- Backpropagation
  - 1. Feed-forward computation
  - 2. Back-propagation to the output layer
  - 3. Back-propagation to the hidden layers
  - 4. Weight updates

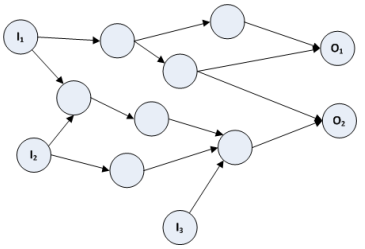
$\partial L / \partial w$

Do not panic. You do not need to worry about it because the program will do it.

38

Neural Network (NN)

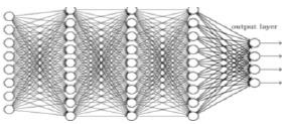
- NN has three elements
  - Architecture: the graph , weights/biases, activation functions
  - Activity Rule: weights/biases, activation functions
  - Learning Rule: a typical one is backpropagation algorithm
- The architecture basically determines the capability of a specific NN
  - Different architectures are suitable for different applications.
  - The most general architecture of an ANN is a DAG (directed acyclic graph).



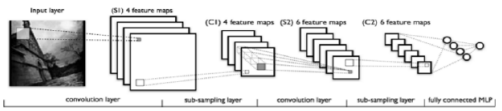
39

Popular types of Neural Network (NN)

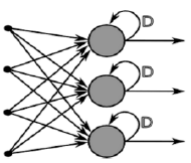
- DNN: Deep NN
  - More general model
  - fully connected
  - feed-forward (i.e., MLP: multilayer perceptron)
  - speech, image processing, natural language processing (NLP)



- CNN: Convolutional NN
  - Common image optimization
  - connected locally (i.e., sparsely-connected)
  - feed-forward
  - object/facial recognition



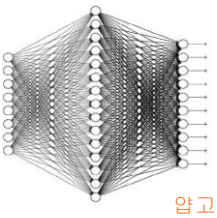
- RNN: Recurrent NN
  - context driven, time-series optimization
  - variable connectivity
  - feed-back in addition to feed-forward
  - NLP and speech recognition
  - Long Short-Term Memory (LSTM)
    - feed-back + storage



40

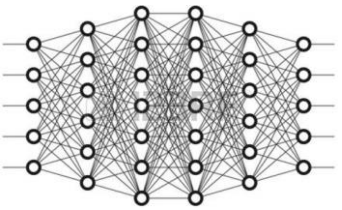
Deep neural net

- Any continuous function can be realized by a network with one hidden layer with sufficient neurons. (Universality theorem, universal approximation theorem)
  - A hidden layer network can represent any continuous function
  - A shallow fat neural net.



얇고 굵다(두껍다)

- Deep thin neural net (deep NN) is better than shallow fat net.
  - Using multiple layers of neurons to represent some functions are much simpler.
    - Less parameters ➔ less computation



깊고 가늘다(얇다)

41

NN categories by applications

Category	Theme	Technology (NN)	Application
Pattern classification	Image classification	CNN	Number, character, image
	Text classification	RNN, NLP	Text
Pattern recognition	Image segmentation	CNN	CT cancer
	Face recognition	CNN	Door lock
	Voice recognition	RNN	AI secretary
Synthesis	Voice synthesis	RNN, NLP	Voice style transform
	Image synthesis	CAN, GAN	Photo transform
Forecasting	Time-serial	RNN, DNN	Traffic estimation
	Weather forecasting	RNN, DNN	Weather
	Anomaly detection	DNN	Network security; Credit card
Control	Robot control	Reinforcement learning	Intelligent IoT control, DRONE
	Game	Reinforcement learning	Intelligent game agent
Optimization	Decision making	DNN, Tree search	Financial decision making
	Mathematical analysis	DNN	Product planning

42

Popular DNNs and Frameworks

- Popular DNNs
  - AlexNet
    - First CNN Winner of ILSVRC
    - Uses LRN (deprecated after this)
  - VGG-16
    - Goes Deeper (16+ layers)
    - Uses only 3x3 filters (stack for larger filters)
  - GoogLeNet (v1)
    - Reduces weights with Inception and only one FC layer
    - Inception: 1x1 and DAG (parallel connections)
    - Batch Normalization
  - ResNet
    - Goes Deeper (24+ layers)
    - Shortcut connections
- Popular Frameworks
  - TensorFlow
  - Caffe