IoT·인공지능·빅데이터 개론 및 실습

Neural Networks

서울대학교 전기·정보공학부
윤성로

Contents

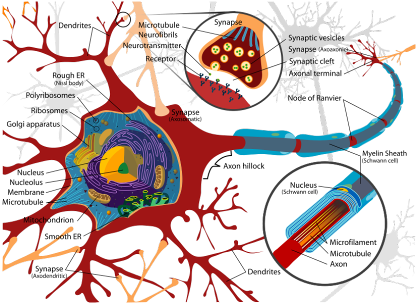
Contents

① Artificial Neural Networks

(1) Neuron

▶ 신경세포

- electrically excitable cell that processes and transmits information through electrical and chemical signals



[출처] Wikipedia

1 Artificial Neural Networks

(2) Artificial neural networks (ANN)

- **Computational models inspired by brain**
  - capable of machine learning and pattern recognition
  - popular until early 90's; popularity then diminished
  - renaissance: deep learning, AlphaGo, ...
- **Traditionally most studied: feedforward neural net**
  - comprises multiple layers of logistic regression models
  - also known as *multilayer perceptron* (MLP)

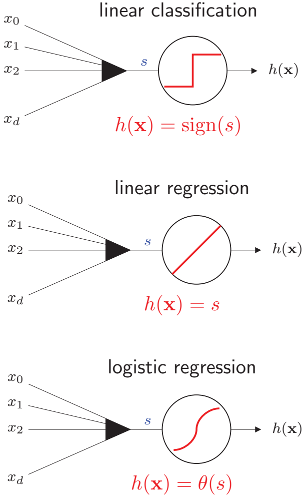
1 Artificial Neural Networks

(3) Recall

- **"signal"**

$$s = \sum_{i=0}^{d} w_i x_i$$

- **"activation"**
  - different



(4) Multilayer perceptron

- **Misnomer**
  - network of multiple *logistic* models (continuous nonlinearity)
  - rather than multiple *perceptrons* (discontinuous nonlinearity)
- **Central idea**
  - extract linear combinations of inputs as derived features
  - model the target as nonlinear function of these features

Contents

❷ Elements of a Neural Network

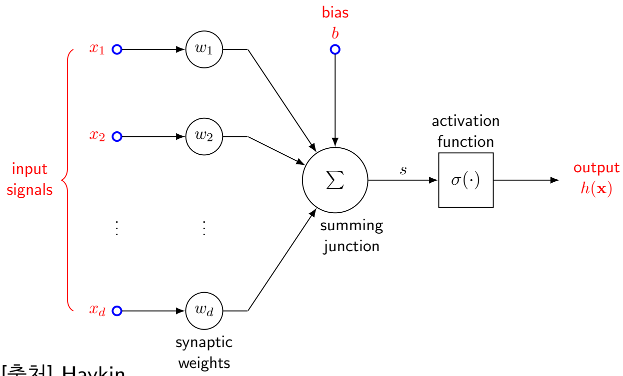
(1) Models of a neuron

- **Three basic elements**

    1. synapses (with weights)
    2. adder (input vector $\rightarrow$ scalar)
    3. activation function (possibly nonlinear)

(1) Models of a neuron

▶ Representation

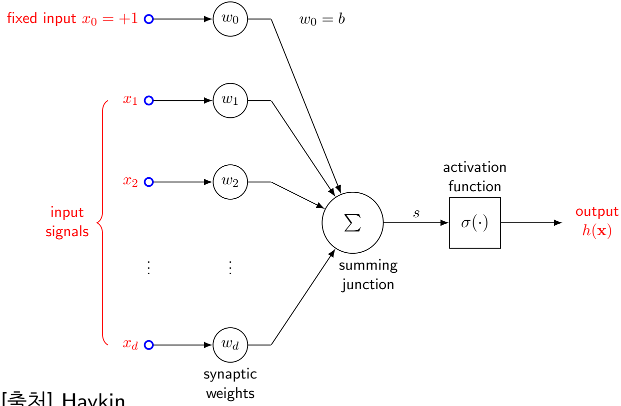


[출처] Haykin

❷ Elements of a Neural Network

(1) Models of a neuron

- **Alternative representation (w_0 for bias b)**

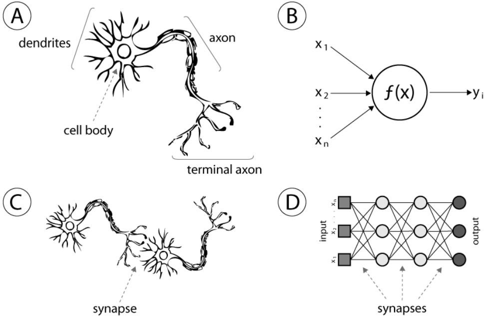


[출처] Haykin

❷ Elements of a Neural Network

(1) Models of a neuron

▶ Human neuron vs ANN neuron

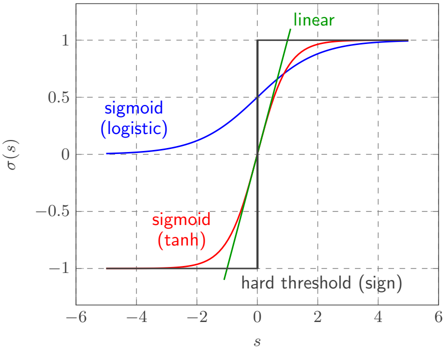


[출처] Maltarollo (2013)

❷ Elements of a Neural Network

(2) Activation function σ

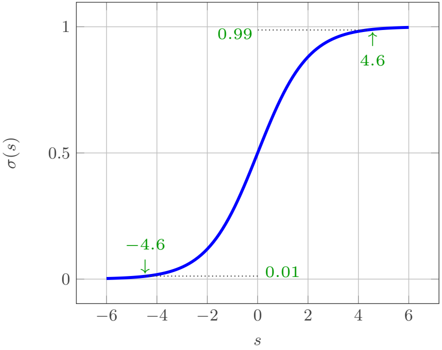
- Defines output of neuron in terms of signal s



(2) Activation function σ

- **Simplifying assumption: all neurons use identical σ**
  - *e.g.* logistic sigmoid

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

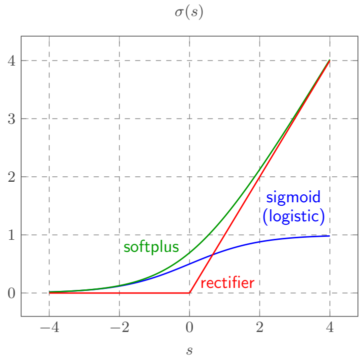


(2) Activation function σ

▶ ReLU (rectifier linear unit)

- very popular in deep neural nets

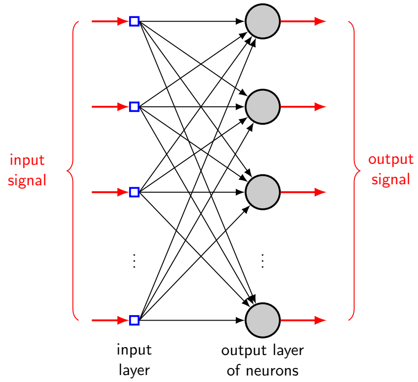
$$\sigma(s) = \max(0, s)$$



Contents

(1) Single-layer feedforward network

- Layered, feedforward

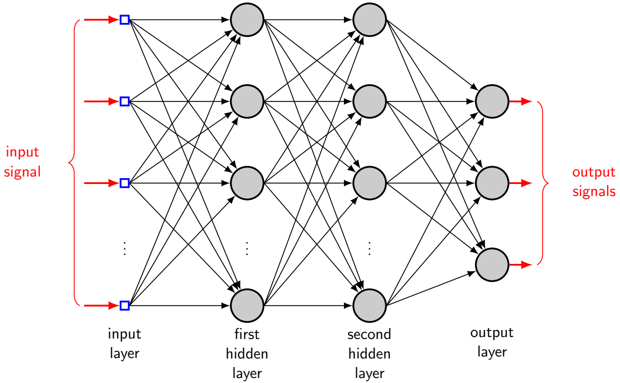


[출처] Haykin

❸ Network Architecture

(2) Multi-layer feedforward network

- One or more hidden layers of neurons



[출처] Haykin

(3) Softmax function

- **Generalization of the logistic function**
  - final layer of a network for multi-class classification
- **Given a K-dimensional vector $\mathbf{h} = (h_1, h_2, \ldots, h_K)$**
  - softmax function $\sigma : \mathbb{R}^K \mapsto \mathbb{R}^K$ s.t.

$$\sigma(\mathbf{h})_j = \frac{e^{h_j}}{\sum_{k=1}^{K} e^{h_k}} \qquad \text{for } j = 1, ..., K$$

- **Components of vector $\sigma(\mathbf{h})$**
  - sum to one and are all strictly between zero and one

  $\Rightarrow$ represent a categorical probability distribution

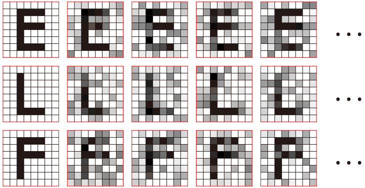
(4) Role of hidden neurons

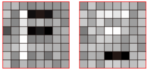
- **Play critical role in operation of MLP**
  - each layer corresponds to "distributed representation"
- **Hidden neurons act as feature detectors**
  - as learning goes on, they gradually "discover" salient features characterizing training data
  - they do so by performing nonlinear transformation on input data into new space called *feature space*
- **Increasing the number of hidden layers**
  - results in a more power model
  - but makes training more difficult

(4) Role of hidden neurons

- **Example**

(a) sample training patterns





(b) learned input-to-hidden weights

[출처] Duda, Hart, Stork (2001)

(5) Deep learning

▶ Based on deep neural nets

- a neural network with many hidden layers
- conventionally considered very difficult to train
  (major reason: vanishing/exploding gradient problem)
- recent success: effective training of deep neural nets

▶ Popular deep neural nets

- convolutional neural net (CNN)
- recurrent neural net (RNN)
- autoencoder (AE)
- generative adversarial net (GAN)

Contents

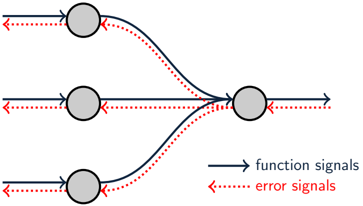
(1) Types of signals in neural nets

▶ Function signals: forward propagation

- input signal comes in at input
- propagates forward through network, and
- emerges at output

▶ Error signals: back(ward) propagation

- originates at an output neuron, and
- propagates backward



[출처] Haykin

(2) Computations at hidden/output layer

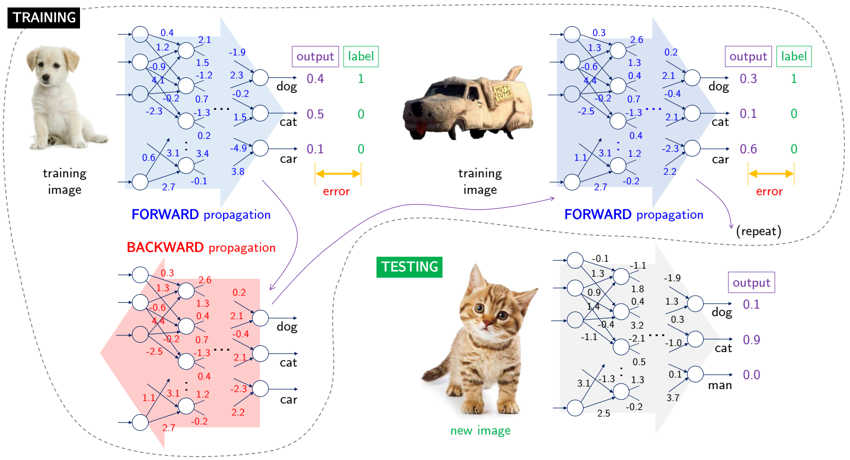
- **Each neuron performs two types of computations:**

  1. function signal appearing at neuron output
     - continuous nonlinear function of input and synaptic weights
  2. estimate of gradient vector needed for back pass
     - gradients of error surface with respect to weights: $\nabla\mathcal{E}(\mathbf{w})$

Contents

❷ Backpropagation

(2) Example: training by forward/backward prop



(3) Credit assignment problem

▶ For output layer

- explicit 'teacher' (correct output y) exists for h
- error signal can be evaluated directly: $e = h - y$
- it is straightforward to find how output (thus error) depends on hidden-to-output weights

$\Rightarrow$ hidden-to-output 'sensitivity' $\dfrac{\partial \mathcal{E}}{\partial w}$: easy to evaluate

(3) Credit assignment problem

- **For hidden layer**
  - no teacher to tell what hidden unit's output should be
  - this is called *credit assignment* problem

  $\Rightarrow$ input-to-hidden sensitivity $\dfrac{\partial \mathcal{E}}{\partial w}$: difficult to evaluate

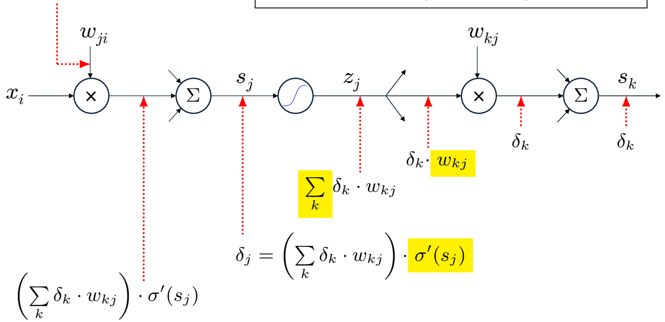
- **Power of back-propagation**
  - it allows us to calculate error for each hidden unit
  - we can derive learning rule for input-to-hidden weights

② Backpropagation

(4) More details of backprop

▶ 실습을 통해 학습

$$\left(\sum_k \delta_k \cdot w_{kj}\right) \cdot \sigma'(s_j) \cdot x_i \implies \boxed{\frac{\partial \mathcal{E}}{\partial w_{ji}} = \delta_j \cdot x_i = \left(\sum_k \delta_k \cdot w_{kj}\right) \cdot \sigma'(s_j) \cdot x_i}$$



Summary

- **Neural network: universal function approximator**
  - neuron: synapse (weights), adder, activation functions
  - conventional: feed-forward multilayer perceptrons
  - recent: deep neural networks ("deep learning")
- **Hidden layers act as feature detectors**
  - # hidden layers $\uparrow \Rightarrow$ intelligence $\uparrow$
    but training difficulty $\uparrow$
  - practical solutions exist for deep neural nets
- **Two types of signals for training MLP**
  1. function signals: forward propagation
  2. error signals: backward propagation
- **Backpropagation algorithm**
  - Popular method for training neural networks