Convolutional Neural Networks 101

Jaehyun Ahn
Machine Learning and Data Mining
Department of Computer Science
Sogang University
Seoul, Korea
jaehyunahn@sogang.ac.kr

May 13, 2015

Convolutional Neural Networks (CNN)

- In machine learning, a convolutional neural network (or CNN) is a type of feed-forward artificial neural network where the individual neurons are tiled in such a way that they respond to overlapping regions in the visual field. Convolutional networks were inspired by biological processes and are variations of multilayer perceptrons which are designed to use minimal amounts of preprocessing. They are widely used models for image and video recognition.

http://en.wikipedia.org/wiki/Convolutional_neural_network

Convolutional Neural Networks (CNN)

- In machine learning, a convolutional neural network (or CNN) is a type of **feed-forward artificial neural network** where the individual neurons are tiled in such a way that they respond to overlapping regions in the visual field. Convolutional networks were inspired by biological processes and are variations of **multilayer perceptrons** which are designed to use minimal amounts of **preprocessing**. They are widely used models for image and video recognition.

http://en.wikipedia.org/wiki/Convolutional_neural_network

Neural Networks + Convolution layer

- Neural Networks using ‘**neurons**’, which send messages to each other. The connections have **numeric weights** that can be **tuned** based on experience, making neural nets adaptive to inputs and capable of learning

Neural Networks + Convolution layer

- Neural Networks using '**neurons**', which send messages to each other. The connections have **numeric weights** that can be **tuned** based on experience, making neural nets adaptive to inputs and capable of learning

Where this came from

(origin,

Neural Networks + Convolution layer

- Neural Networks using ‘**neurons**’, which send messages to each other. The connections have **numeric weights** that can be **tuned** based on experience, making neural nets adaptive to inputs and capable of learning

Where this came from / How to set

(origin, structure

Neural Networks + Convolution layer

- Neural Networks using ‘**neurons**’, which send messages to each other. The connections have **numeric weights** that can be **tuned** based on experience, making neural nets adaptive to inputs and capable of learning

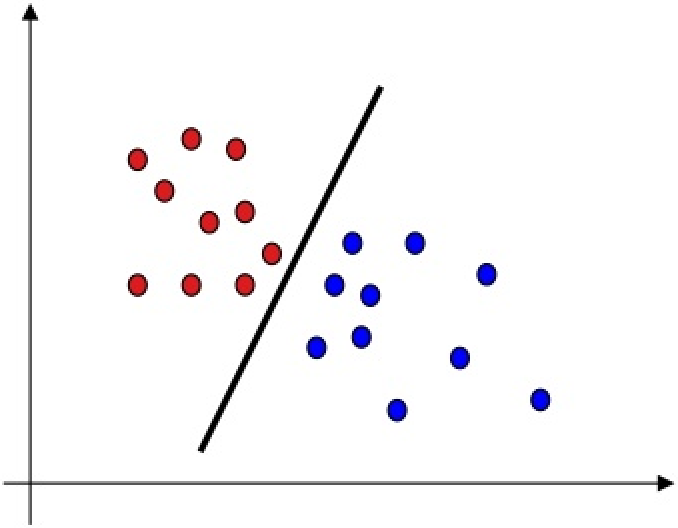
Where this came from / How to set / How to tune ?

(origin, structure and learning)

origin

XOR Problem

- Comes from AB Problem (similar with SVM)

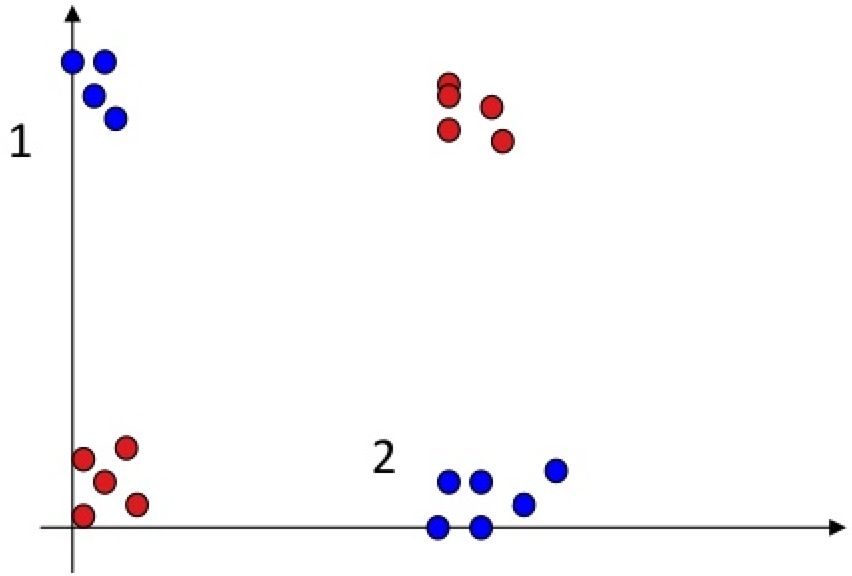


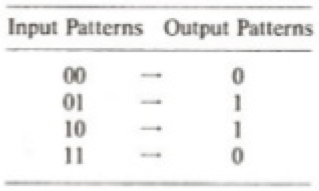
D-dimensions with
D-1 (under) classification

https://web.stanford.edu/group/pdplab/pdphandbook/handbookch6.html

XOR Problem

- XOR Problem
  cannot be solved by conventional D-1 linear function.



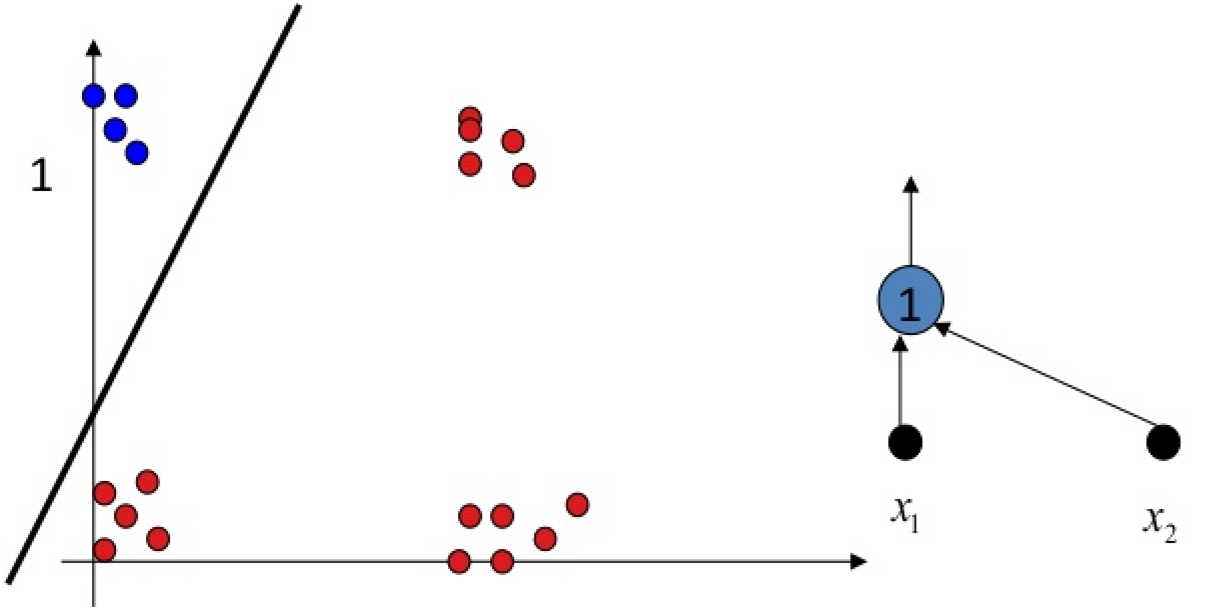


Input Patterns		Output Patterns
00	→	0
01	→	1
10	→	1
11	→	0

https://web.stanford.edu/group/pdplab/pdphandbook/handbookch6.html

XOR Problem

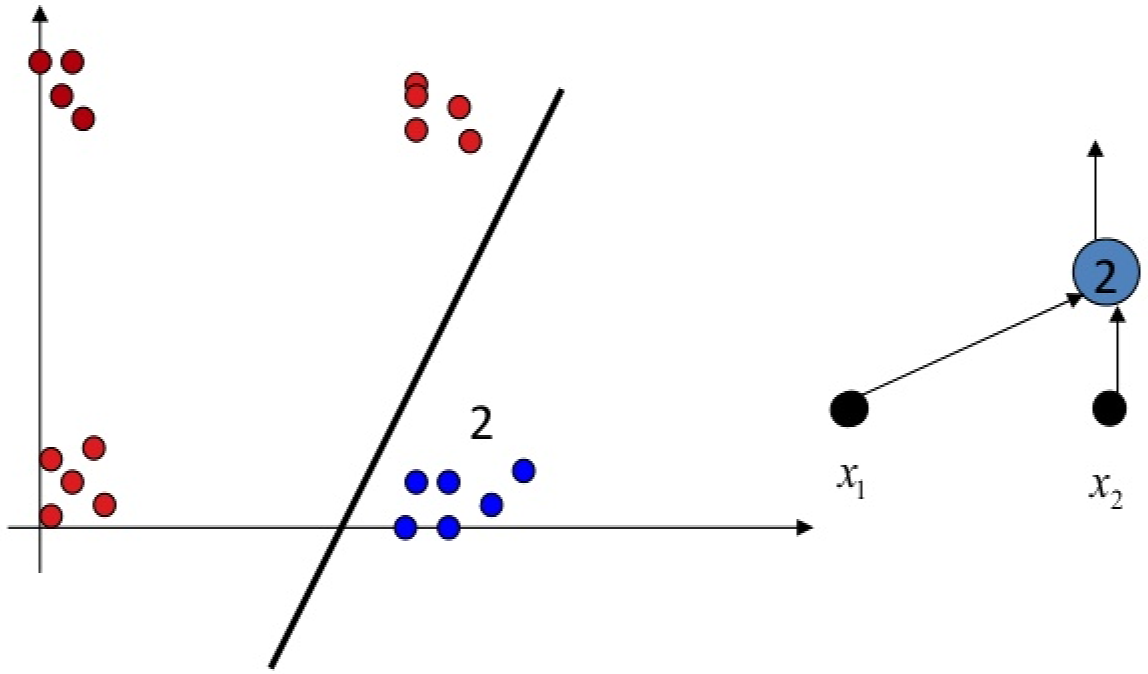
- XOR Problem



https://web.stanford.edu/group/pdplab/pdphandbook/handbookch6.html

XOR Problem

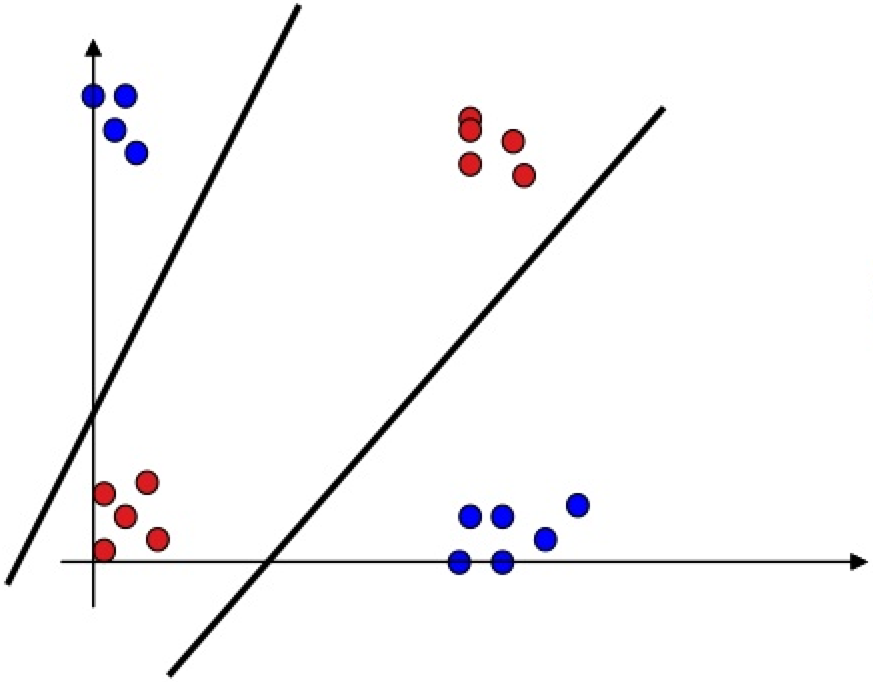
- XOR Problem

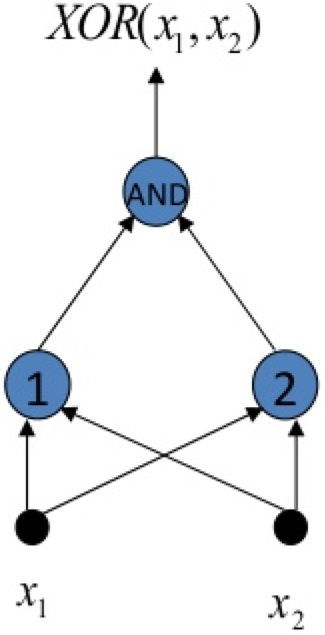


https://web.stanford.edu/group/pdplab/pdphandbook/handbookch6.html

XOR Problem

- XOR Problem

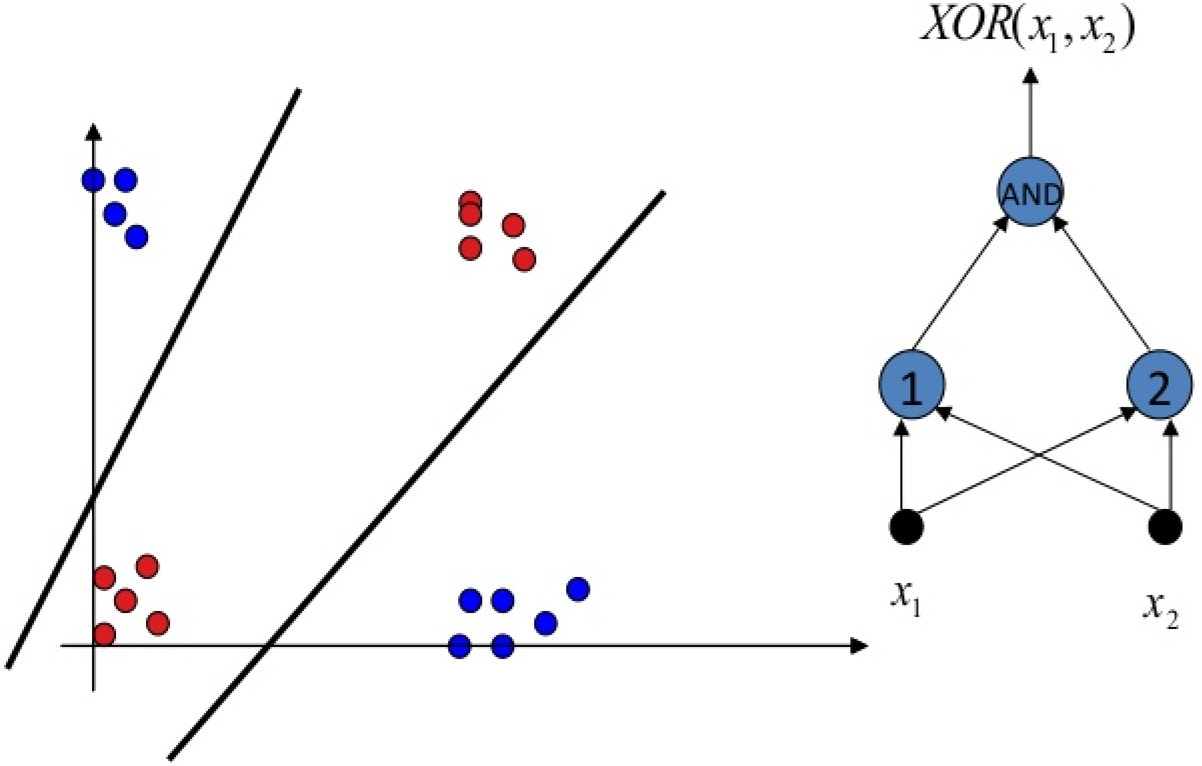




https://web.stanford.edu/group/pdplab/pdphandbook/handbookch6.html

XOR Problem

- XOR Problem

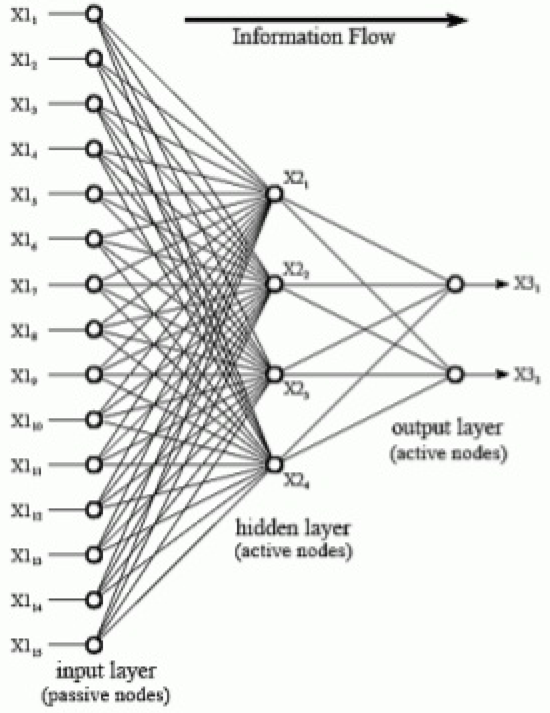


Linear combination needed to solve a nonlinear discrimination problem (sometimes)

https://web.stanford.edu/group/pdplab/pdphandbook/handbookch6.html

Structure of neural networks (1)

FIGURE 26-5
Neural network architecture. This is the most common structure for neural networks: three layers with full interconnection. The input layer nodes are passive, doing nothing but relaying the values from their single input to their multiple outputs. In comparison, the nodes of the hidden and output layers are active, modifying the signals in accordance with Fig. 26-6. The action of this neural network is determined by the weights applied in the hidden and output nodes.



- Combining perceptrons
- Feed forward Information flow
- Passive node
  - without weighted sum input
- Active node
  - with weighted sum

The Scientist and Engineer's Guide to Digital Signal Processing, Ch 26, Steven W. Smith, ISBN 0-7506-7444 X

Structure of neural networks (2)

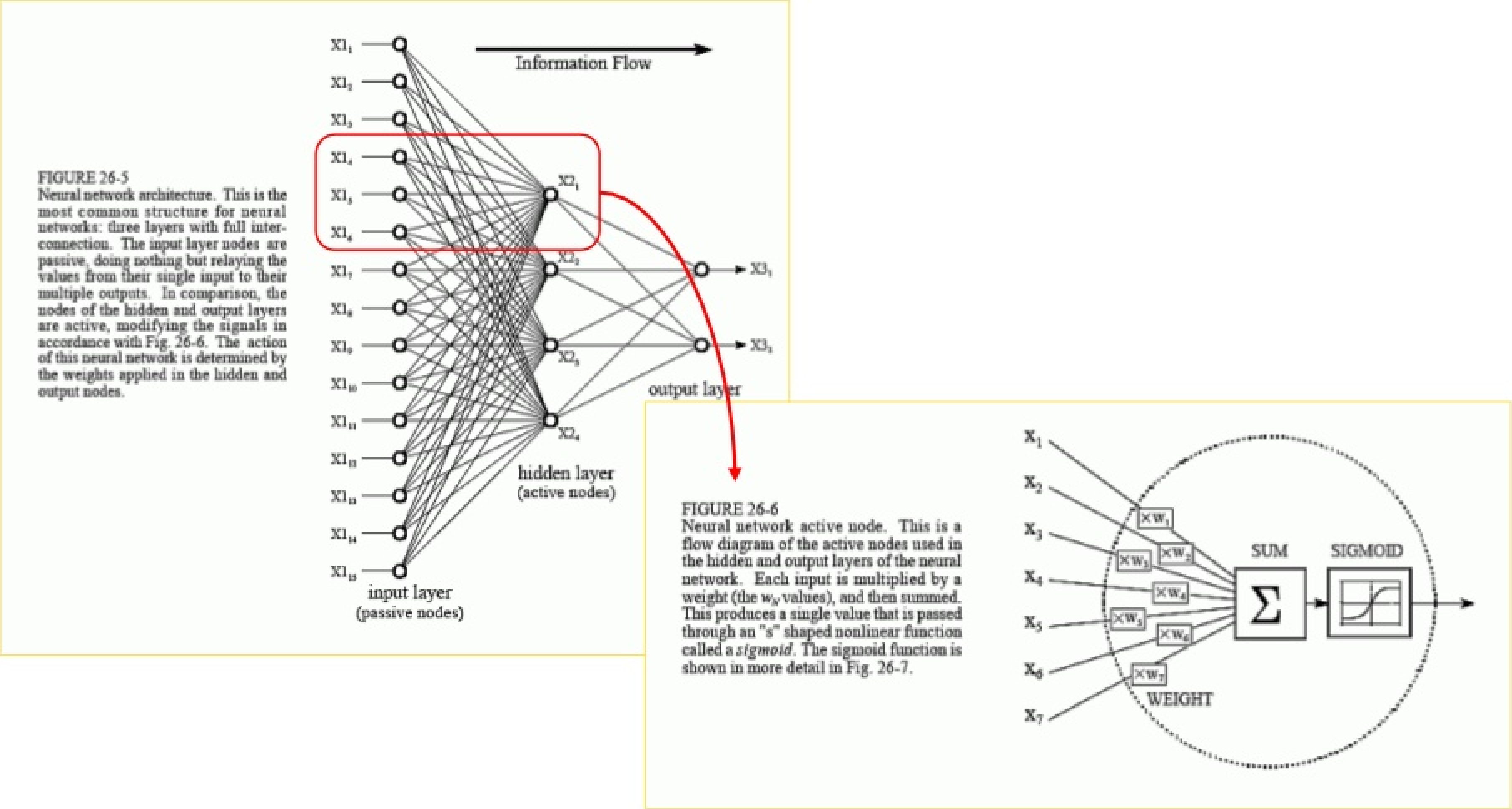
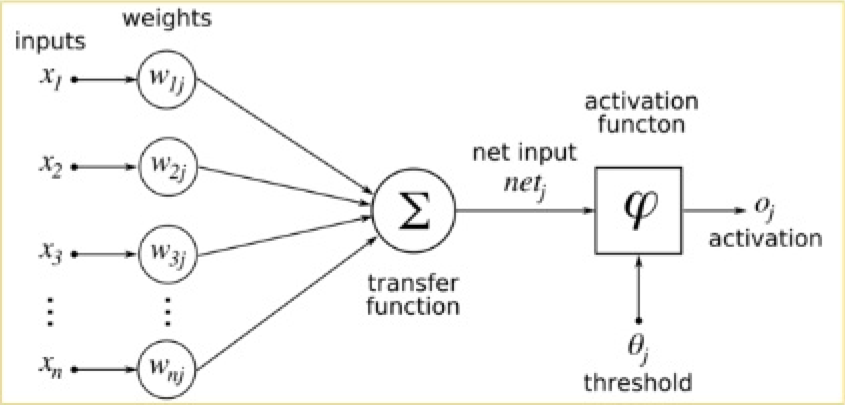


FIGURE 26-5
Neural network architecture. This is the most common structure for neural networks: three layers with full interconnection. The input layer nodes are passive, doing nothing but relaying the values from their single input to their multiple outputs. In comparison, the nodes of the hidden and output layers are active, modifying the signals in accordance with Fig. 26-6. The action of this neural network is determined by the weights applied in the hidden and output nodes.

FIGURE 26-6
Neural network active node. This is a flow diagram of the active nodes used in the hidden and output layers of the neural network. Each input is multiplied by a weight (the w_N values), and then summed. This produces a single value that is passed through an "s" shaped nonlinear function called a *sigmoid*. The sigmoid function is shown in more detail in Fig. 26-7.

The Scientist and Engineer's Guide to Digital Signal Processing, Ch 26, Steven W. Smith, ISBN 0-7506-7444 X

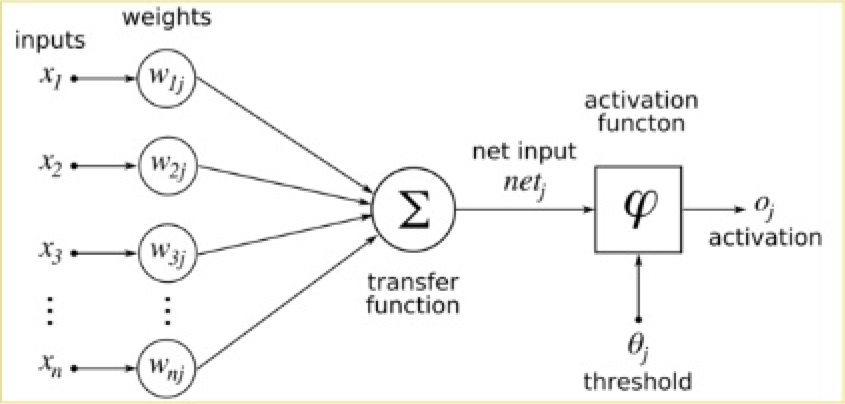
Why do sigmoid functions work in Neural Nets?



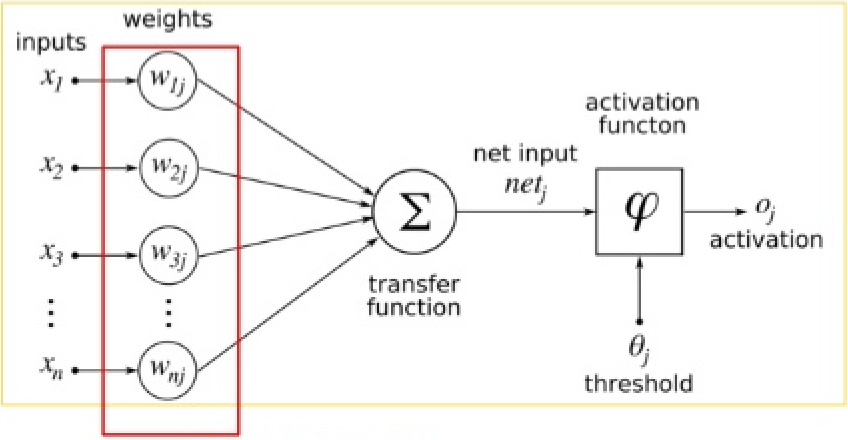
1. **To overcome nonlinearity** (Gradient Descent problem)
   a) Inputs are discrete (images, signals and etc.)
   b) Sigmoid is 'bounded' linear (0 to 1)
      - 1-to-1 relation between weighted sum and output value
      - Differentiable, its derivative is very fast to compute
2. **Make output probabilistic** (Activation function)
   a) Probabilistic output is more generous about the error
      a) Linear-combinations (=linear classifier) to avoid overfitting problem

http://stackoverflow.com/questions/11677508/why-do-sigmoid-functions-work-in-neural-nets

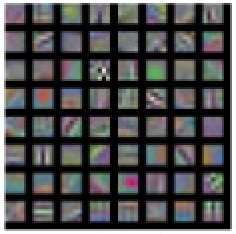
Value of Differentiable function



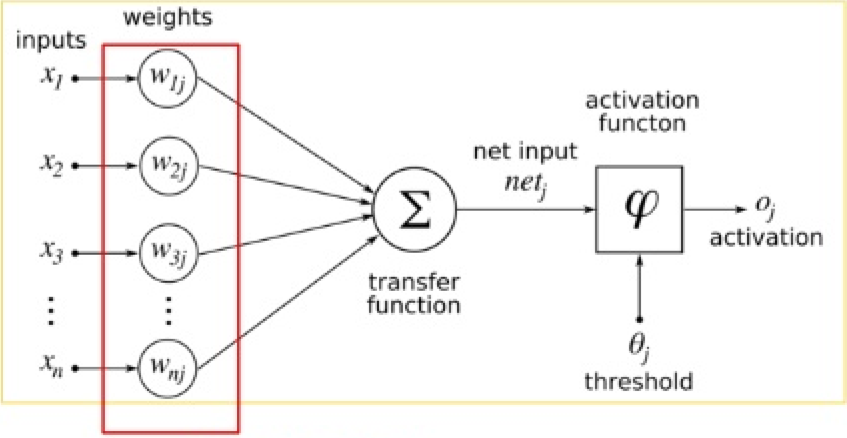
Value of Differentiable function



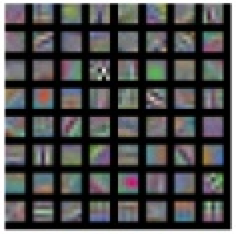
Tunable stimuli



Value of Differentiable function

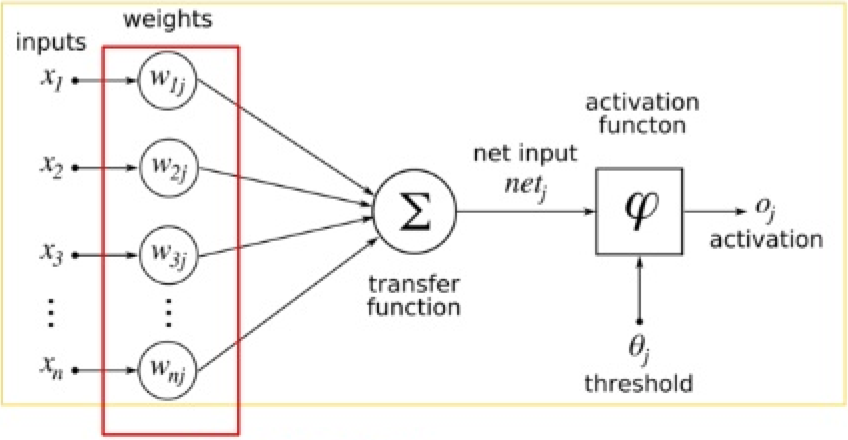


Tunable stimuli

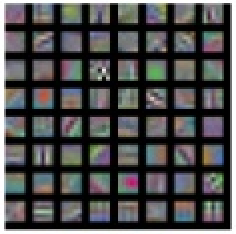


How to tune?

Value of Differentiable function



Tunable stimuli



How to tune?

Backpropagation method