

Artificial Neural Networks

Neural Networks	2
Biological Neural Networks	2
Artificial Neural Networks	3
ANN Structure	4
ANN Illustration	5
Perceptrons	6
Perceptron Learning Rule	6
Perceptrons Continued	7
Example of Perceptron Learning ($\alpha = 1$)	8
Gradient Descent	9
Gradient Descent (Linear)	9
The Delta Rule	10
Multilayer Networks	11
Illustration	11
Sigmoid Activation	12
Plot of Sigmoid Function	13
Applying The Chain Rule	14
Backpropagation	15
Hypertangent Activation Function	16
Gaussian Activation Function	17
Issues in Neural Networks	18

Neural Networks

Biological Neural Networks

- Neural networks are inspired by our brains.
- The human brain has about 10^{11} neurons and 10^{14} synapses.
- A neuron consists of a soma (cell body), axons (sends signals), and dendrites (receives signals).
- A synapse connects an axon to a dendrite.
- Given a signal, a synapse might increase (excite) or decrease (inhibit) electrical potential. A neuron fires when its electrical potential reaches a threshold.
- Learning might occur by changes to synapses.

Artificial Neural Networks

An (artificial) neural network consists of units, connections, and weights. Inputs and outputs are numeric.

Biological NN	Artificial NN
soma	unit
axon, dendrite	connection
synapse	weight
potential	weighted sum
threshold	bias weight
signal	activation

ANN Structure

- A typical unit i receives inputs a_{j1}, a_{j2}, \dots from other units and performs a weighted sum:

$$in_i = W_{0,i} + \sum_j W_{j,i} a_j$$

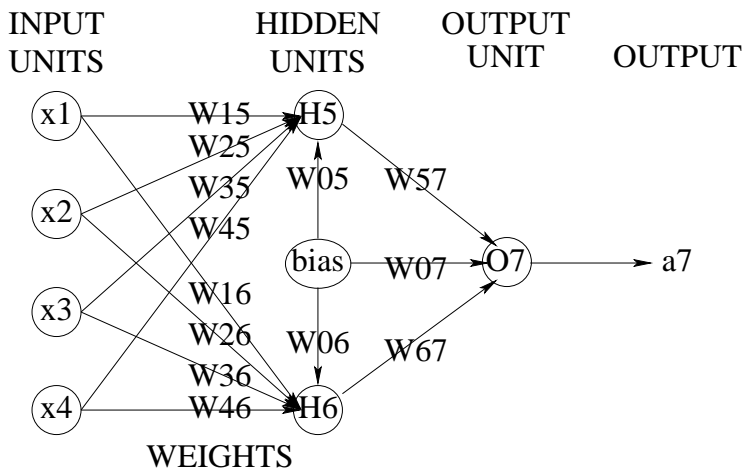
and outputs activation $a_i = g(in_i)$.

- Typically, *input units* store the inputs, *hidden units* transform the inputs into an internal numeric vector, and an *output unit* transforms the hidden values into the prediction.
- An ANN is a function $f(\mathbf{x}, \mathbf{W}) = a$, where \mathbf{x} is an example, \mathbf{W} is the weights, and a is the prediction (activation value from output unit).
- Learning is finding a \mathbf{W} that minimizes error.

CS 5233 Artificial Intelligence

Artificial Neural Networks – 4

ANN Illustration



CS 5233 Artificial Intelligence

Artificial Neural Networks – 5

Perceptrons

Perceptron Learning Rule

- A perceptron is a single unit with activation:

$$a = \text{sign} \left(W_0 + \sum_j W_j * x_j \right)$$

sign returns -1 or 1 . W_0 is the *bias* weight.

- One version of the perceptron learning rule is:

$$in \leftarrow W_0 + \sum W_j * x_j$$

$$E \leftarrow \max(0, 1 - y * in)$$

if $E > 0$ **then**

$$W_0 \leftarrow W_0 + \alpha * y$$

$$W_j \leftarrow W_j + \alpha * x_j * y \text{ for each input } x_j$$

\mathbf{x} is the inputs, $y \in \{1, -1\}$ is the target, and $\alpha > 0$ is the learning rate.

CS 5233 Artificial Intelligence

Artificial Neural Networks – 6

Perceptrons Continued

- This learning rule tends to minimize E .
- The perceptron convergence theorem states that if some \mathbf{W} classifies all the training examples correctly, then the perceptron learning rule will converge to zero error on the training examples.
- Usually, many *epochs* (passes over the training examples) are needed until convergence.
- If zero error is not possible, use $\alpha \approx 0.1/n$, where n is the number of normalized or standardized inputs.

CS 5233 Artificial Intelligence

Artificial Neural Networks – 7

Example of Perceptron Learning ($\alpha = 1$)

Using $\alpha = 1$:

Inputs							Weights				
x_1	x_2	x_3	x_4	y	in	E	W_0	W_1	W_2	W_3	W_4
0	0	0	1	-1	0	1	-1	0	0	0	-1
1	1	1	0	1	-1	2	0	1	1	1	-1
1	1	1	1	1	2	0	0	1	1	1	-1
0	0	1	1	-1	0	1	-1	1	1	0	-2
0	0	0	0	1	-1	2	0	1	1	0	-2
0	1	0	1	-1	-1	0	0	1	1	0	-2
1	0	0	0	1	1	0	0	1	1	0	-2
1	0	1	1	1	-1	2	1	2	1	1	-1
0	1	0	0	-1	2	3	0	2	0	1	-1

CS 5233 Artificial Intelligence

Artificial Neural Networks - 8

Gradient Descent

9

Gradient Descent (Linear)

- Suppose activation is a linear function:

$$a = W_0 + \sum W_j * x_j$$

- The Widrow-Hoff (LMS) update rule is:

$$diff \leftarrow y - a$$

$$W_0 \leftarrow W_0 + \alpha * diff$$

$$W_j \leftarrow W_j + \alpha * x_j * diff \text{ for each input } j$$

where α is the learning rate.

- This update rule tends to minimize squared error.

$$E(\mathbf{W}) = \sum_{(x,y)} (y - a)^2$$

CS 5233 Artificial Intelligence

Artificial Neural Networks - 9

5

The Delta Rule

- Given error $E(\mathbf{W})$, obtain the gradient:

$$\nabla E(\mathbf{W}) = \left[\frac{\partial E}{\partial W_0}, \frac{\partial E}{\partial W_1}, \dots, \frac{\partial E}{\partial W_n} \right]$$

- To decrease error, use the update rule:

$$W_j \leftarrow W_j - \alpha \frac{\partial E}{\partial W_j}$$

- The LMS update rule can be derived using:

$$E(\mathbf{W}) = (y - (W_0 + \sum W_j * x_j))^2 / 2$$

- For the perceptron learning rule, use:

$$E(\mathbf{W}) = \max(0, 1 - y * (W_0 + \sum W_j * x_j))$$

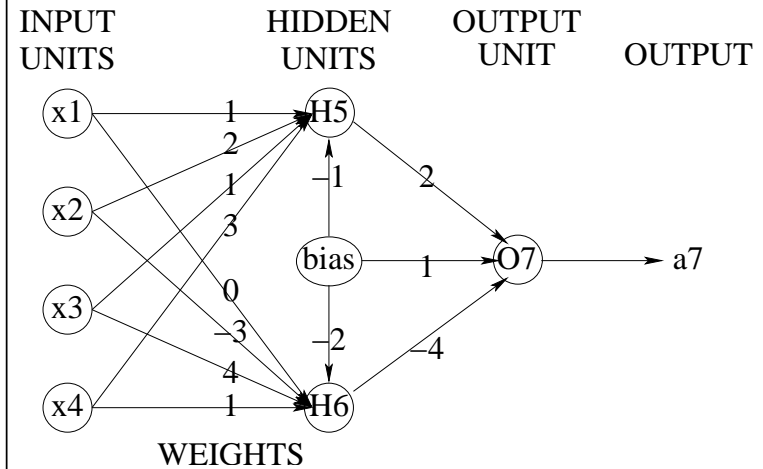
CS 5233 Artificial Intelligence

Artificial Neural Networks - 10

Multilayer Networks

11

Illustration



CS 5233 Artificial Intelligence

Artificial Neural Networks - 11

6

Sigmoid Activation

- The sigmoid function is defined as:

$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

- It is commonly used for ANN activation functions:

$$\begin{aligned} a_i &= \text{sigmoid}(in_i) \\ &= \text{sigmoid}(W_{0,i} + \sum_j W_{j,i} a_j) \end{aligned}$$

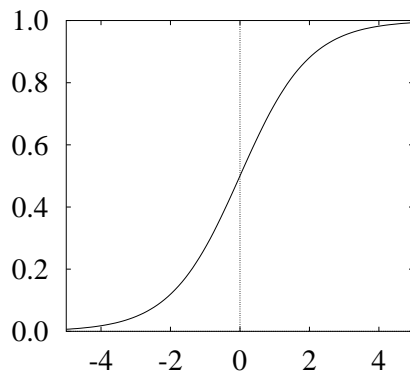
- Note that

$$\frac{\partial \text{sigmoid}(x)}{\partial x} = \text{sigmoid}(x)(1 - \text{sigmoid}(x))$$

CS 5233 Artificial Intelligence

Artificial Neural Networks – 12

Plot of Sigmoid Function



CS 5233 Artificial Intelligence

Artificial Neural Networks – 13

Applying The Chain Rule

- Using $E = (y_i - a_i)^2$ for output unit i :

$$\begin{aligned} \frac{\partial E}{\partial W_{j,i}} &= \frac{\partial E}{\partial a_i} \frac{\partial a_i}{\partial in_i} \frac{\partial in_i}{\partial W_{j,i}} \\ &= -2(y_i - a_i) a_i(1 - a_i) a_j \end{aligned}$$

- For weights from input to hidden units:

$$\begin{aligned} \frac{\partial E}{\partial W_{k,j}} &= \frac{\partial E}{\partial a_i} \frac{\partial a_i}{\partial in_i} \frac{\partial in_i}{\partial a_j} \frac{\partial a_j}{\partial in_j} \frac{\partial in_j}{\partial W_{k,j}} \\ &= -2(y_i - a_i) a_i(1 - a_i) W_{j,i} \\ &\quad a_j(1 - a_j) x_k \end{aligned}$$

CS 5233 Artificial Intelligence

Artificial Neural Networks – 14

Backpropagation

- Backpropagation is an application of the delta rule.
- Update each weight W using the rule:

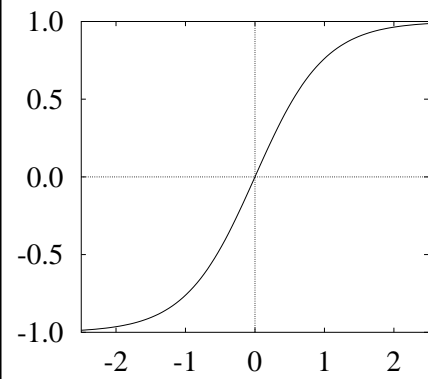
$$W \leftarrow W - \alpha \frac{\partial E}{\partial W}$$

where α is the learning rate.

CS 5233 Artificial Intelligence

Artificial Neural Networks – 15

Hypertangent Activation Function

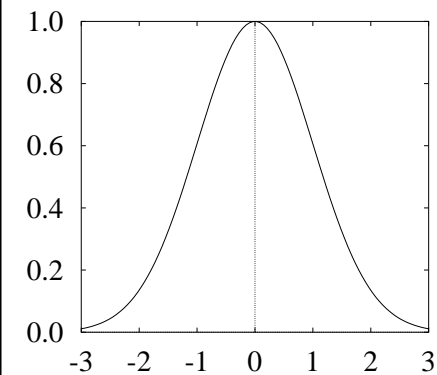


$\tanh(x)$ —

CS 5233 Artificial Intelligence

Artificial Neural Networks – 16

Gaussian Activation Function



$\text{gaussian}(x,1)$ —

CS 5233 Artificial Intelligence

Artificial Neural Networks – 17

Issues in Neural Networks

- Sufficiently complex ANNs can approximate any “reasonable” function. ANNs approx. a preference bias for interpolation.
- How many hidden units and layers?
What are good initial values?
- One approach to avoid overfitting is:
Remove “validation” exs. from training exs.
Train neural network using training exs.
Choose weights that are best on validation set.
- Faster algorithms might rely on:
Momentum, a running average of deltas.
Conjugant Gradient, a second deriv. method,
or RPROP, update based only on sign.

CS 5233 Artificial Intelligence

Artificial Neural Networks – 18