AI
Neural Networks

Lecture 5

The simple arithmetic computing elements correspond to **neurons**—the cells that perform information processing in the brain—and the network as a whole corresponds to a collection of interconnected neurons.

For this reason, the networks are called **neural networks.**

The truly amazing thing is that *a collection of simple cells can lead to thought, action, and consciousness.*

- A neural network is composed of a number of nodes, or **units,** connected by **links.** Each link has a numeric **weight** associated with it.
- Weights are the primary means of long-term storage in neural networks, and learning usually takes place by updating the weights.
- Some of the units are connected to the external environment, and can be designated as input or output units.
- The weights are modified so as to try to bring the network's input/output behavior more into line with that of the environment providing the inputs.

To build a neural network to perform some task:

- How many units are to be used.
- What kind of units are appropriate.
- How the units are to be connected to form a

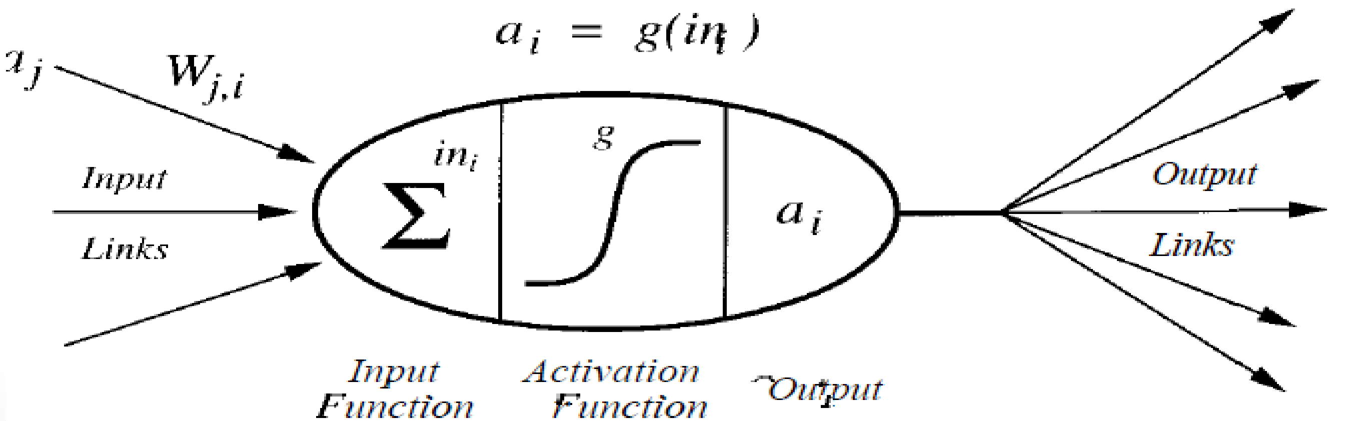
network.

Simple computing elements

- Each unit performs a simple computation it receives signals from its input links and computes a new activation level that it sends along each of its output links.
- The computation of the activation level is based on the values of each input signal received from a neighboring node, and the weights on each input link.

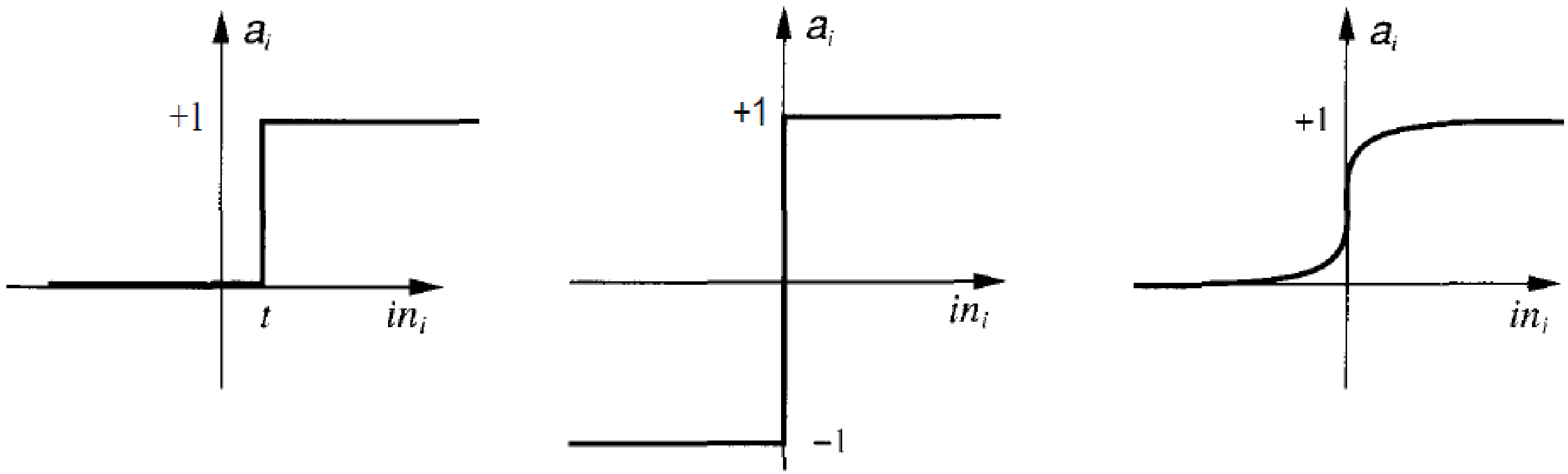
The computation is split into two components.

- **First** is a *linear* component, called the **input function,** *int,* that computes the weighted sum of the unit's input values.
- **Second** is a *nonlinear* component called the **activation function,** *g,* that transforms the weighted sum into the final value that serves as the unit's activation value, *a,*.



A unit.

- Different models are obtained by using different mathematical functions for *g.* Three common choices are the step, sign, and sigmoid functions.
- The step function has a threshold *t* such that it outputs a 1 when the input is greater than its threshold, and outputs a 0 otherwise.
- The biological motivation is that a 1 represents the firing of a pulse down the axon, and a 0 represents no firing.

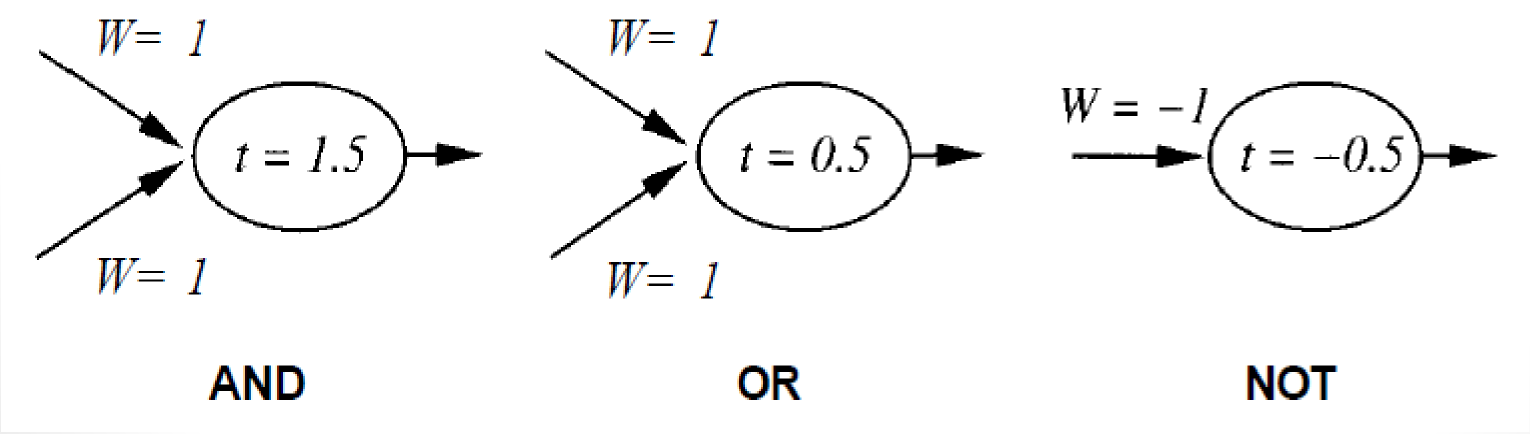


(a) Step function

(b) Sign function

(c) Sigmoid function

- One of the original motivations for the design of individual units was their ability to represent basic Boolean functions.
- Figure below shows how the Boolean functions *AND, OR,* and *NOT* can be represented by units with suitable weights and thresholds.
- This is important because it means we can use these units to build a network to compute any Boolean function of the inputs.



Network structures

There are a variety of kinds of network structure, each of which results in very different computational properties.

The main distinction to be made is between **feed-forward** and **recurrent** networks.

- In a **feed-forward network**, links are unidirectional, and there are no cycles.
- In **a recurrent network**, the links can form arbitrary topologies.

- We will usually be dealing with networks that are arranged in layers.
- In a layered feed-forward network, each unit is linked only to units in the next layer; there are no links between units in the same layer, no links backward to a previous layer, and no links that skip a layer.

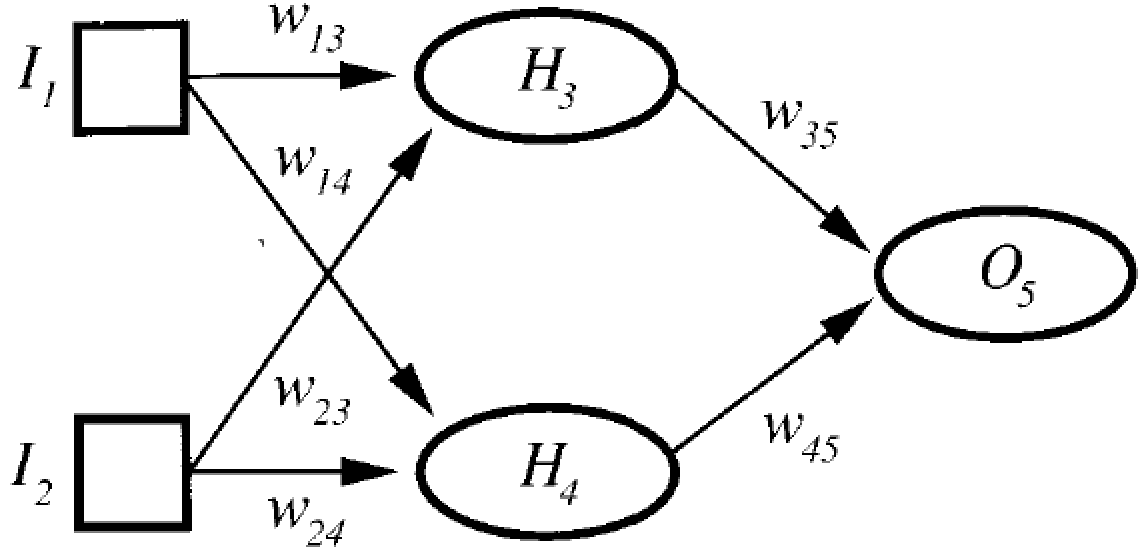
The significance of the lack of cycles is that computation can proceed uniformly from input units to output units.

- The activation from the previous time step plays no part in the computation, because it is not fed back to an earlier unit.
- Hence, a feed-forward network simply computes a function of the input values that depends on the weight settings—it has *no internal state* other than the weights themselves.

Obviously, the brain cannot be a feed-forward network, else we would have no short-term memory.

- Because activation is fed back to the units that caused it, **recurrent networks** have internal state stored in the activation levels of the units.
- **Recurrent networks** can become unstable, or oscillate, or exhibit chaotic behavior.
- Given some input values, it can take a long time to compute a stable output, and learning is made more difficult.
- On the other hand**, recurrent networks** can implement more complex agent designs and can model systems with state.

- Returning to feed-forward networks, there is one more important distinction to be made. Examine Figure below, which shows the topology of a very simple neural network



- On the left are the **input units.** The activation value of each of these units is determined by the environment.
- At the right-hand end of the network are four **output units.**
- In between, the nodes labelled *H3* and H4 have no direct connection to the outside world. These are called **hidden units,** because they cannot be directly observed by noting the input/output behavior of the network.
- Some networks, called **perceptrons,** have no hidden units. This makes the learning problem much simpler, but it means that perceptrons are very limited in what they can represent.
- Networks with one or more layers of hidden units are called **multilayer networks.**

Optimal network structure

Considering networks with a fixed structure is a potential weak point, because the wrong choice of network structure can lead to poor performance.

- If we choose a network that is too small, then the model will be incapable of representing the desired function.
- If we choose a network that is too big, it will be able to memorize all the examples by forming a large lookup table, but will not generalize well to inputs that have not been seen before.

Therefore, it is more common to see hill-climbing searches that selectively modify an existing network structure. There are two ways to do this: start with a big network and make it smaller, or start with a small one and make it bigger.