



Artificial Intelligence Tutorial

- AI - Home
- AI - Overview
- AI - Intelligent Systems
- AI - Research Areas
- AI - Agents and Environments
- AI - Popular Search Algorithms
- AI - Fuzzy Logic Systems
- AI - Natural Language Processing
- AI - Expert Systems
- AI - Robotics
- AI - Neural Networks
- AI - Issues
- AI - Terminology

Artificial Intelligence Resources

- Artificial Intelligence - Quick Guide
- AI - Useful Resources
- Artificial Intelligence - Discussion

Selected Reading

- Developer's Best Practices
- Questions and Answers
- Effective Resume Writing
- HR Interview Questions
- Computer Glossary
- Who is Who

Artificial Intelligence - Neural Networks

Advertisements

⌂ Previous PageNext Page ⌂

Yet another research area in AI, neural networks, is inspired from the natural neural network of human nervous system.

What are Artificial Neural Networks (ANNs)?

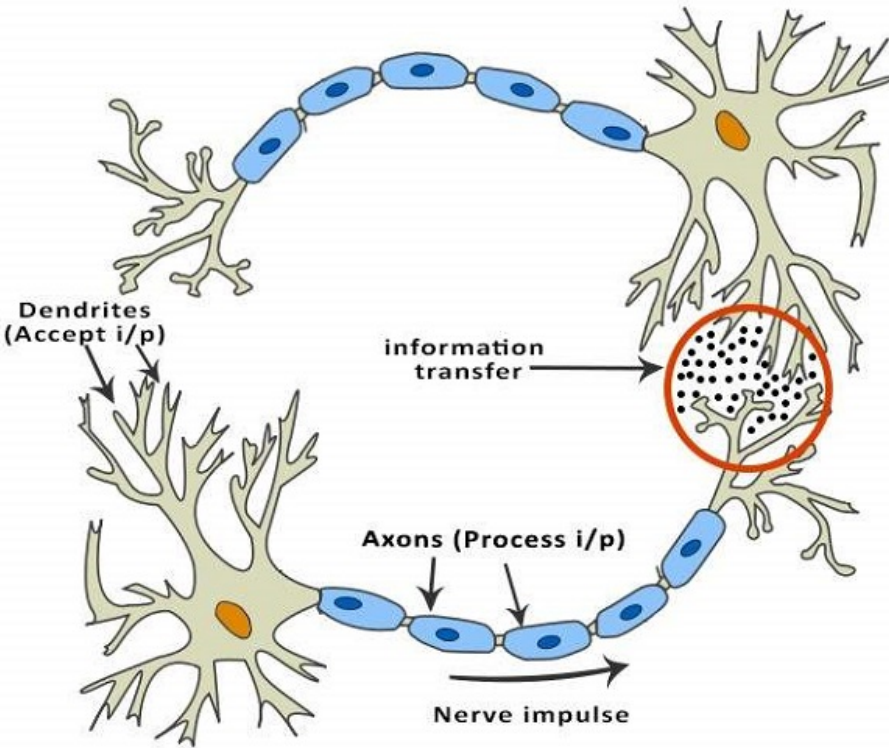
The inventor of the first neurocomputer, Dr. Robert Hecht-Nielsen, defines a neural network as –

"...a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs."

Basic Structure of ANNs

The idea of ANNs is based on the belief that working of human brain by making the right connections, can be imitated using silicon and wires as living **neurons** and **dendrites**.

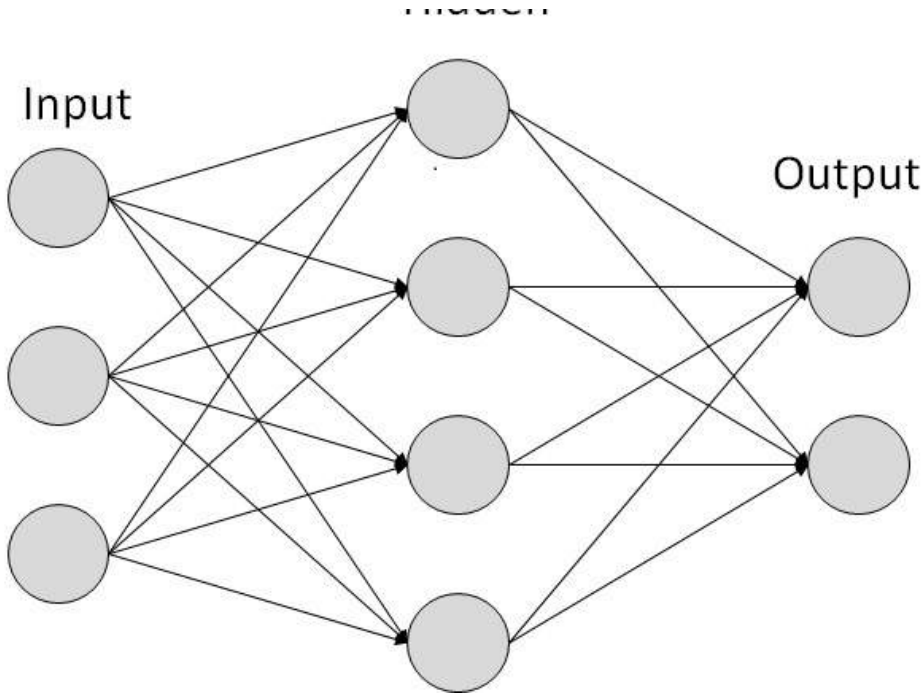
The human brain is composed of 100 billion nerve cells called **neurons**. They are connected to other thousand cells by **Axons**. Stimuli from external environment or inputs from sensory organs are accepted by dendrites. These inputs create electric impulses, which quickly travel through the neural network. A neuron can then send the message to other neuron to handle the issue or does not send it forward.



ANNs are composed of multiple **nodes**, which imitate biological **neurons** of human brain. The neurons are connected by links and they interact with each other. The nodes can take input data and perform simple operations on the data. The result of these operations is passed to other neurons. The output at each node is called its **activation** or **node value**.

Each link is associated with **weight**. ANNs are capable of learning, which takes place by altering weight values. The following illustration shows a simple ANN –

Hidden

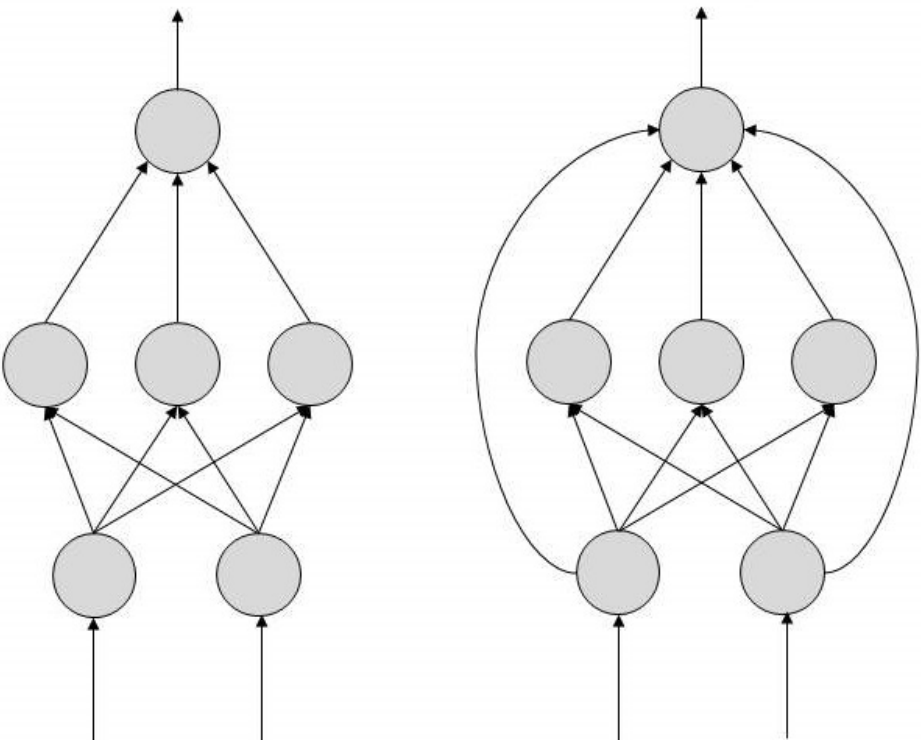


Types of Artificial Neural Networks

There are two Artificial Neural Network topologies – **FeedForward** and **Feedback**.

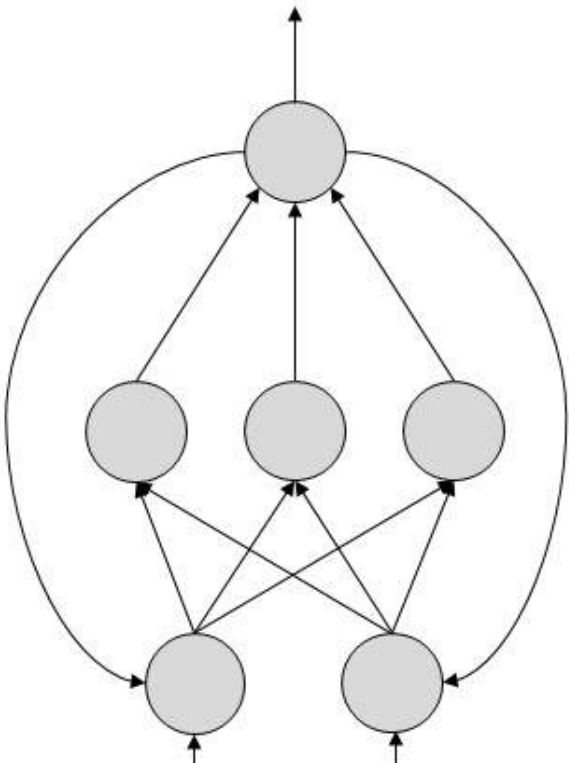
FeedForward ANN

The information flow is unidirectional. A unit sends information to other unit from which it does not receive any information. There are no feedback loops. They are used in pattern generation/recognition/classification. They have fixed inputs and outputs.



FeedBack ANN

Here, feedback loops are allowed. They are used in content addressable memories.





Working of ANNs

In the topology diagrams shown, each arrow represents a connection between two neurons and indicates the pathway for the flow of information. Each connection has a weight, an integer number that controls the signal between the two neurons.

If the network generates a “good or desired” output, there is no need to adjust the weights. However, if the network generates a “poor or undesired” output or an error, then the system alters the weights in order to improve subsequent results.

Machine Learning in ANNs

ANNs are capable of learning and they need to be trained. There are several learning strategies –

Supervised Learning – It involves a teacher that is scholar than the ANN itself. For example, the teacher feeds some example data about which the teacher already knows the answers.

For example, pattern recognizing. The ANN comes up with guesses while recognizing. Then the teacher provides the ANN with the answers. The network then compares it guesses with the teacher’s “correct” answers and makes adjustments according to errors.

Unsupervised Learning – It is required when there is no example data set with known answers. For example, searching for a hidden pattern. In this case, clustering i.e. dividing a set of elements into groups according to some unknown pattern is carried out based on the existing data sets present.

Reinforcement Learning – This strategy built on observation. The ANN makes a decision by observing its environment. If the observation is negative, the network adjusts its weights to be able to make a different required decision the next time.

Back Propagation Algorithm

It is the training or learning algorithm. It learns by example. If you submit to the algorithm the example of what you want the network to do, it changes the network’s weights so that it can produce desired output for a particular input on finishing the training.

Back Propagation networks are ideal for simple Pattern Recognition and Mapping Tasks.

Bayesian Networks (BN)

These are the graphical structures used to represent the probabilistic relationship among a set of random variables. Bayesian networks are also called **Belief Networks** or **Bayes Nets**. BNs reason about uncertain domain.

In these networks, each node represents a random variable with specific propositions. For example, in a medical diagnosis domain, the node Cancer represents the proposition that a patient has cancer.

The edges connecting the nodes represent probabilistic dependencies among those random variables. If out of two nodes, one is affecting the other then they must be directly connected in the directions of the effect. The strength of the relationship between variables is quantified by the probability associated with each node.

There is an only constraint on the arcs in a BN that you cannot return to a

node simply by following directed arcs. Hence the BNs are called Directed Acyclic Graphs (DAGs).

BNs are capable of handling multivalued variables simultaneously. The BN variables are composed of two dimensions –

- Range of prepositions
- Probability assigned to each of the prepositions.

Consider a finite set $X = \{X_1, X_2, \dots, X_n\}$ of discrete random variables, where each variable X_i may take values from a finite set, denoted by $Val(X_i)$. If there is a directed link from variable X_i to variable, X_j , then variable X_i will be a parent of variable X_j showing direct dependencies between the variables.

The structure of BN is ideal for combining prior knowledge and observed data. BN can be used to learn the causal relationships and understand various problem domains and to predict future events, even in case of missing data.

Building a Bayesian Network

A knowledge engineer can build a Bayesian network. There are a number of steps the knowledge engineer needs to take while building it.

Example problem – Lung cancer. A patient has been suffering from breathlessness. He visits the doctor, suspecting he has lung cancer. The doctor knows that barring lung cancer, there are various other possible diseases the patient might have such as tuberculosis and bronchitis.

Gather Relevant Information of Problem

- Is the patient a smoker? If yes, then high chances of cancer and bronchitis.
- Is the patient exposed to air pollution? If yes, what sort of air pollution?
- Take an X-Ray positive X-ray would indicate either TB or lung cancer.

Identify Interesting Variables

The knowledge engineer tries to answer the questions –

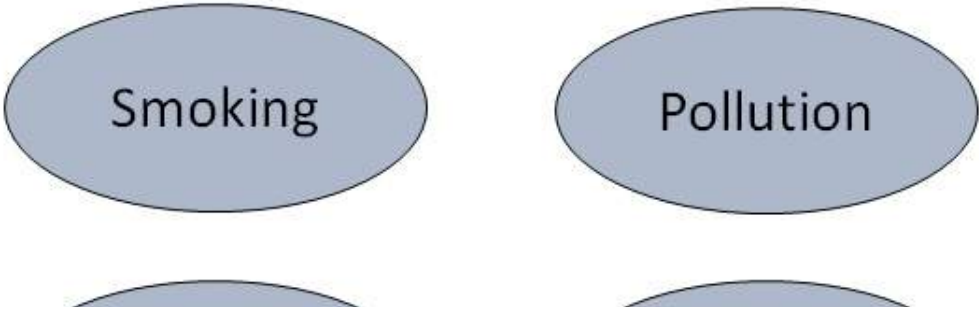
- Which nodes to represent?
- What values can they take? In which state can they be?

For now let us consider nodes, with only discrete values. The variable must take on exactly one of these values at a time.

Common types of discrete nodes are –

- Boolean nodes** – They represent propositions, taking binary values TRUE (T) and FALSE (F).
- Ordered values** – A node *Pollution* might represent and take values from {low, medium, high} describing degree of a patient’s exposure to pollution.
- Integral values** – A node called *Age* might represent patient’s age with possible values from 1 to 120. Even at this early stage, modeling choices are being made.

Possible nodes and values for the lung cancer example –

Node Name	Type	Value	Nodes Creation	
Polution	Binary	{LOW, HIGH, MEDIUM}		
Smoker	Boolean	{TRUE, FASLE}		

Lung-Cancer	Boolean	{TRUE, FASLE}
X-Ray	Binary	{Positive, Negative}

Cancer

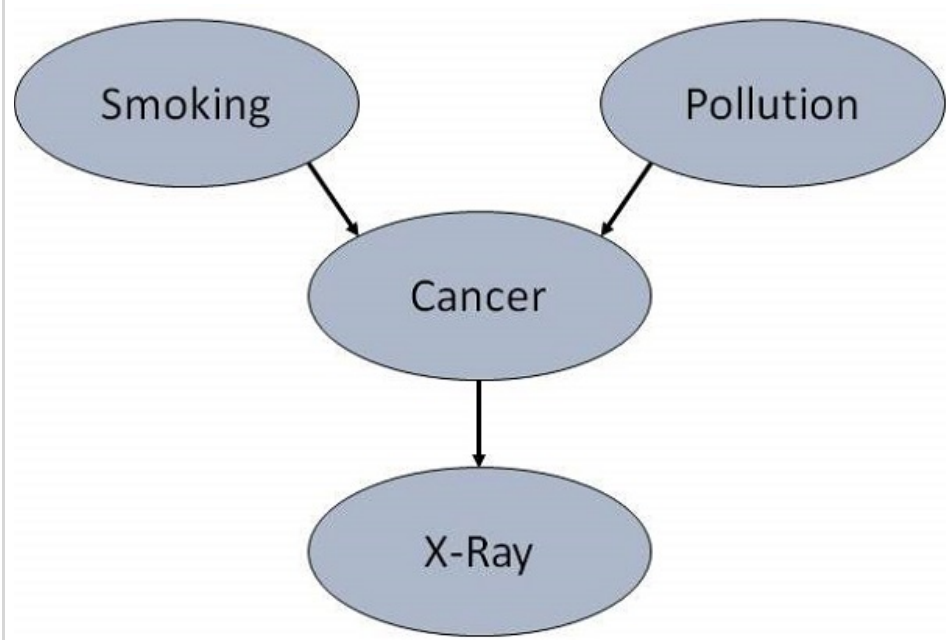
X-Ray

Create Arcs between Nodes

Topology of the network should capture qualitative relationships between variables.

For example, what causes a patient to have lung cancer? - Pollution and smoking. Then add arcs from node *Pollution* and node *Smoker* to node *Lung-Cancer*.

Similarly if patient has lung cancer, then X-ray result will be positive. Then add arcs from node *Lung-Cancer* to node *X-Ray*.



Specify Topology

Conventionally, BNs are laid out so that the arcs point from top to bottom. The set of parent nodes of a node X is given by Parents(X).

The *Lung-Cancer* node has two parents (reasons or causes): *Pollution* and *Smoker*, while node *Smoker* is an **ancestor** of node *X-Ray*. Similarly, *X-Ray* is a child (consequence or effects) of node *Lung-Cancer* and **successor** of nodes *Smoker* and *Pollution*.

Conditional Probabilities

Now quantify the relationships between connected nodes: this is done by specifying a conditional probability distribution for each node. As only discrete variables are considered here, this takes the form of a **Conditional Probability Table (CPT)**.

First, for each node we need to look at all the possible combinations of values of those parent nodes. Each such combination is called an **instantiation** of the parent set. For each distinct instantiation of parent node values, we need to specify the probability that the child will take.

For example, the *Lung-Cancer* node's parents are *Pollution* and *Smoking*. They take the possible values = { (H,T), (H,F), (L,T), (L,F)}. The CPT specifies the probability of cancer for each of these cases as <0.05, 0.02, 0.03, 0.001> respectively.

Each node will have conditional probability associated as follows –

Smoking	Pollution
P(S = T)	P(P = L)
0.30	0.90

Lung-Cancer

Tung Tung		
P	S	P (C = T P, S)
H	T	0.05
H	F	0.02
L	T	0.03
L	F	0.001


X-Ray	
C	X = (Pos C)
T	0.90
F	0.20

Applications of Neural Networks

They can perform tasks that are easy for a human but difficult for a machine –

- Aerospace** – Autopilot aircrafts, aircraft fault detection.
- Automotive** – Automobile guidance systems.
- Military** – Weapon orientation and steering, target tracking, object discrimination, facial recognition, signal/image identification.
- Electronics** – Code sequence prediction, IC chip layout, chip failure analysis, machine vision, voice synthesis.
- Financial** – Real estate appraisal, loan advisor, mortgage screening, corporate bond rating, portfolio trading program, corporate financial analysis, currency value prediction, document readers, credit application evaluators.
- Industrial** – Manufacturing process control, product design and analysis, quality inspection systems, welding quality analysis, paper quality prediction, chemical product design analysis, dynamic modeling of chemical process systems, machine maintenance analysis, project bidding, planning, and management.
- Medical** – Cancer cell analysis, EEG and ECG analysis, prosthetic design, transplant time optimizer.
- Speech** – Speech recognition, speech classification, text to speech conversion.
- Telecommunications** – Image and data compression, automated information services, real-time spoken language translation.
- Transportation** – Truck Brake system diagnosis, vehicle scheduling, routing systems.
- Software** – Pattern Recognition in facial recognition, optical character recognition, etc.
- Time Series Prediction** – ANNs are used to make predictions on stocks and natural calamities.
- Signal Processing** – Neural networks can be trained to process an audio signal and filter it appropriately in the hearing aids.
- Control** – ANNs are often used to make steering decisions of physical vehicles.
- Anomaly Detection** – As ANNs are expert at recognizing patterns, they can also be trained to generate an output when

something unusual occurs that misfits the pattern.

⬅ Previous Page  Print Next Page ➡

Advertisements



Write for us FAQ's Helping
Contact

© Copyright 2017. All Rights Reserved.

Enter email for newsletter

go