Neural network learning rules – Supervised and Unsupervised Learning

1. The Categories of Neural Network Learning Rules

There are many types of Neural Network Learning Rules, they fall into two broad categories: supervised learning, and unsupervised learning. Block diagrams of the learning types are illustrated in Figures (1) and Figure(2).

1.1 Supervised Learning

The learning rule is provided with a set of examples (the training set) of proper network behavior:

$$\{x_1, d_1\} , \{x_2, d_2\} , \ldots, \{x_n, d_n\} \quad \ldots (1)$$

where (x_n) is an input to the network, (d_n) is the corresponding correct target (desired) output. As the inputs are applied to the network, the network outputs are compared with the targets. The learning rule is then used to adjust the weights and the biases of the network in order to move the network outputs closer to the targets (desired).

In supervised learning we assume that at each instant of time when the input is applied, the desired response (d) of the system is provided by the teacher. This is illustrated in figure (1), the distance between the actual and the desired response serves as an error measure and is used to correct network parameter externally.

For instance, in learning classifications of input patterns or situations with known responses, the error can be used to modify the weights so that the error decreases. This mode of learning is very pervasive. Also it is used in many situations of natural learning. A set of input and output patterns called training set is required for this learning mode .

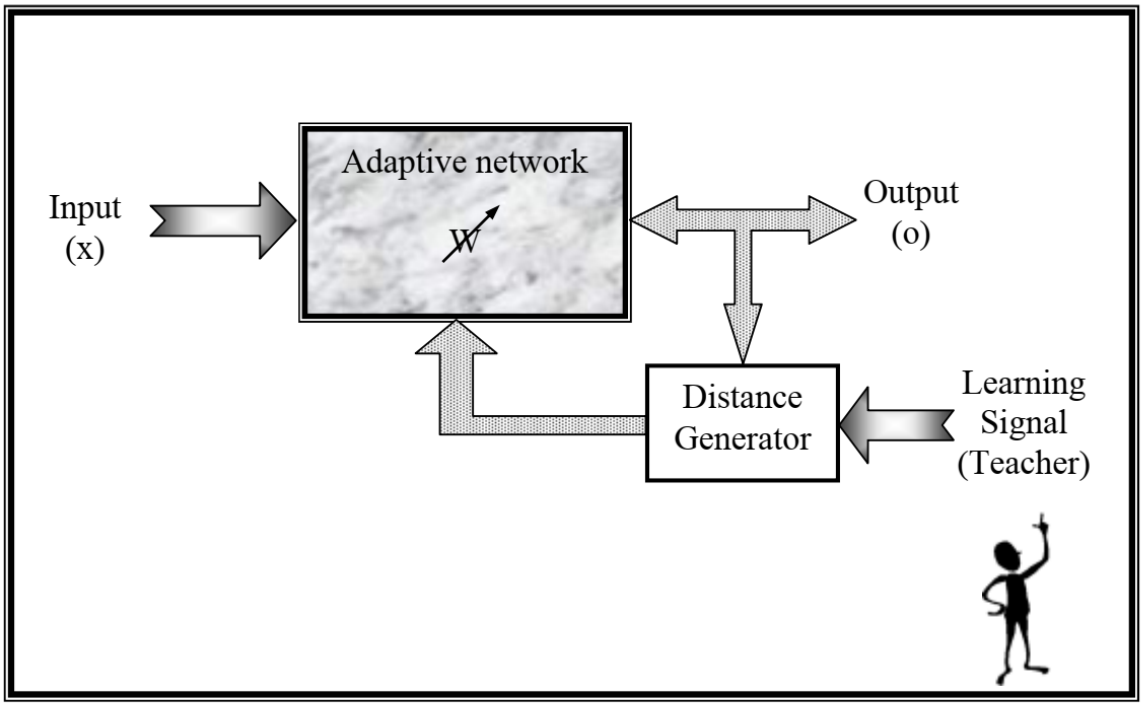

Figure (1) Block diagram for explaining of supervised learning.

1.2 Unsupervised Learning

In unsupervised learning, the weights and biases are modified in response to network input only. There are no target outputs available. At first glance this might seem to be impractical. How can you train a network if you don't know what is supposed to do? Most of these algorithms perform some kind of clustering operation. They learn to categorize the input patterns into a finite number of classes. This is useful in such applications such as vector quantization .

Figure (2) shows the block diagram of unsupervised learning rule. In learning without supervision the desired response is not known; thus, explicit error information cannot be used to improve network behavior. Since no information is available as to correctness or incorrectness of responses, learning must somehow be accomplished based on observations of responses to inputs that we have marginal or no knowledge about.

Unsupervised learning algorithms use patterns that are typically redundant raw data having no label regarding their class membership, or

associations. In this mode of learning, a network must discover for itself any possibly existing patterns, regularities, separating properties, etc., while discovering these the network undergoes change in its parameters, unsupervised learning is sometimes called learning without teacher. This terminology is not the most appropriate because learning without a teacher is not possible at all. Although, the teacher does not have to be involved in every training step, he has to set goals even in an unsupervised learning mode.

Learning with feedback, either from the teacher or from environment, however, is more typical for neural network. Such learning is called incremental and is usually performed in steps. The concept of feedback plays a central role in learning.

The concept is highly elusive and somewhat paradoxical. In a broad sense it can be understood as an introduction of a pattern of relationships into the cause-and-effect path.

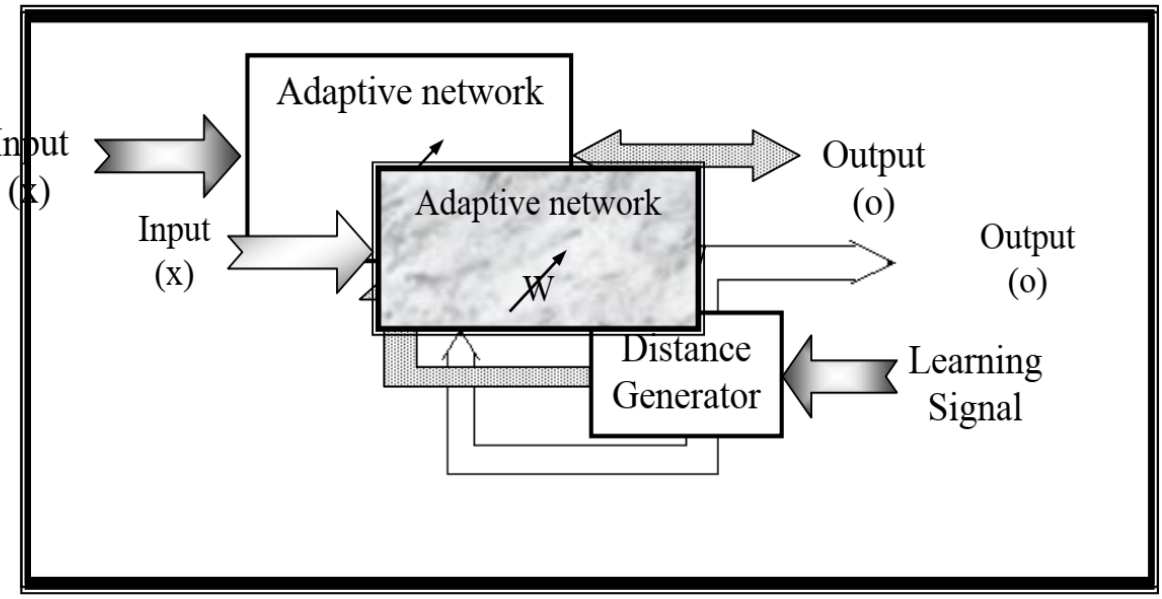

Figure (2) Block diagram for explaining of unsupervised learning.

2. Neural Network Learning Rules

Our focus in this section will be on artificial neural network learning rules. A neuron is considered to be an adaptive element. Its weights are modifiable depending on the input signal it receives, its output value, and the associated teacher response. In some cases the teacher signal is not available and no error information can be used, thus a neuron will modify its weights, based only on the input and / or output. This is the case for unsupervised learning.

The trained network is show in Figure (3), It study the weight vector (w_i) or its component (w_{ij}), which connecting the (j'th) input with neuron (i). The output of another neurons can be the (j'th) input to the neuron (i). Our discussion in this section will cover single neuron and single layer network supervised learning and simple cases of unsupervised learning. The form of neuron activation function may be different when different learning rules is considered.

The learning ability of human beings is properly incorporated in the facility of the changing the transmission efficiency of the synapses which corresponding to adaptation of the weight. The convergence has always been a major problem in the neural network learning algorithms. In most cases, to avoid this, impractical applications different initial conditions are used until one case would converge to the desirable target.

The weight vector $w_i=[w_{i1}\ w_{i2}\ w_{i3}\ \ldots.\ w_{in}]^t$ increases in proportion to the product of input (x) and learning signal (r). The learning signal (r) is, in general, a function of (w_i,x), and sometimes of the teacher's signal (d_i). So for the network shown in Figure (3.3): -

$$r = r(w_i,x,d_i) \qquad \ldots (2)$$

The increment of the weight vector (w_i) product by the learning step at time (t) according to the general learning rule is given by

$$\Delta w_i(t) = c\ r[w_i(t), x(t), d_i(t)]\ x(t) \qquad \ldots (3)$$

where (c) is constant called the learning constant that determines the rate of learning. At the next instant learning step, the weight vector adapt at time (t) becomes:

$$w_i(t+1) = w_i(t) + c\, r[w_i(t), x(t), d_i(t)]\, x(t) \quad \ldots (4)$$

The superscript convention will be used in this context to index the discrete – time training steps as in equation (3.4). For the (k'th) step it can be had from equation (3.4) using this convention:

$$w_i^{k+1} = w_i^k + cr\,(w_i^k, x^k, d_i^k)\, x^k \quad \ldots (5)$$

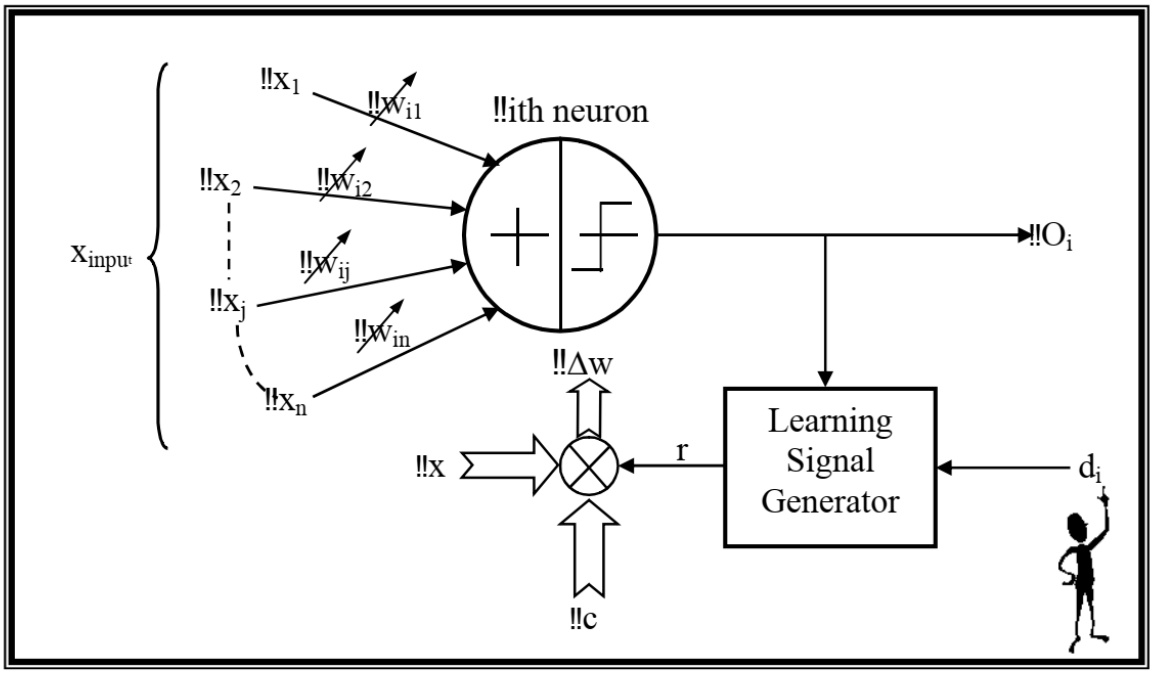

Figure (3) Illustration of weight learning values
(d_i) provided only for supervised learning mode)

3 Hebbian Learning Rule

For the Hebbian learning rule the learning is equal simply to the neuron's output as shown in Figure (3.4), so it can be seen that:

$$r = f(w_i^t x) \quad \ldots (6)$$

The increment (Δw_i)of the weight vector becomes:

$$\Delta w_i = c\, f(w_i^t x) x \quad \ldots (7)$$

The single weight (w_i) is adapted using the following increment:

$$\Delta w_{ij} = c\, f(w_i^t x) x_j \quad \ldots (8)$$

This can be written briefly as:

$$\Delta w_{ij} = co_i x_j \quad , \quad \text{For j= 1, 2, 3, …, n.} \quad \ldots (9)$$

This learning rule requires the weight initialization at small random values around (w_{ij}=0) prior to learning. The Hebbian learning rule represents a purely unsupervised learning. The rule implements the interpretation of the classic statement; “when an axon of cell (a) is near enough to the exit of a cell (b) and repeatedly or persistently takes place in firing it, some growth process or metabolic change takes place in one or both cells such that cell (a) efficiency, as one of the cells firing cell (b), is increased”.

The rule states that if the cross product of output and input, or correlation term ($o_i x_j$) is positive, this results in an increase of weight w_{ij}; otherwise the weight decreases. It can be seen that the output is strengthened in turn for each input presented. Therefore, frequent input patterns will have most influence at the neurons weight vectors and will eventually produce the largest output.

A persistence worry with computational model of unsupervised learning is that learning will become more difficult as problem is scaled. The Hebbian rule has evolved in a number of directions, in some cases, the Hebbian rule needs to be modified to counteract unconstrained growth of weight values, which takes place when excitations and responses consistently agree in sign. This corresponds to Hebbian learning rule with saturation of the weights at certain, preset level.

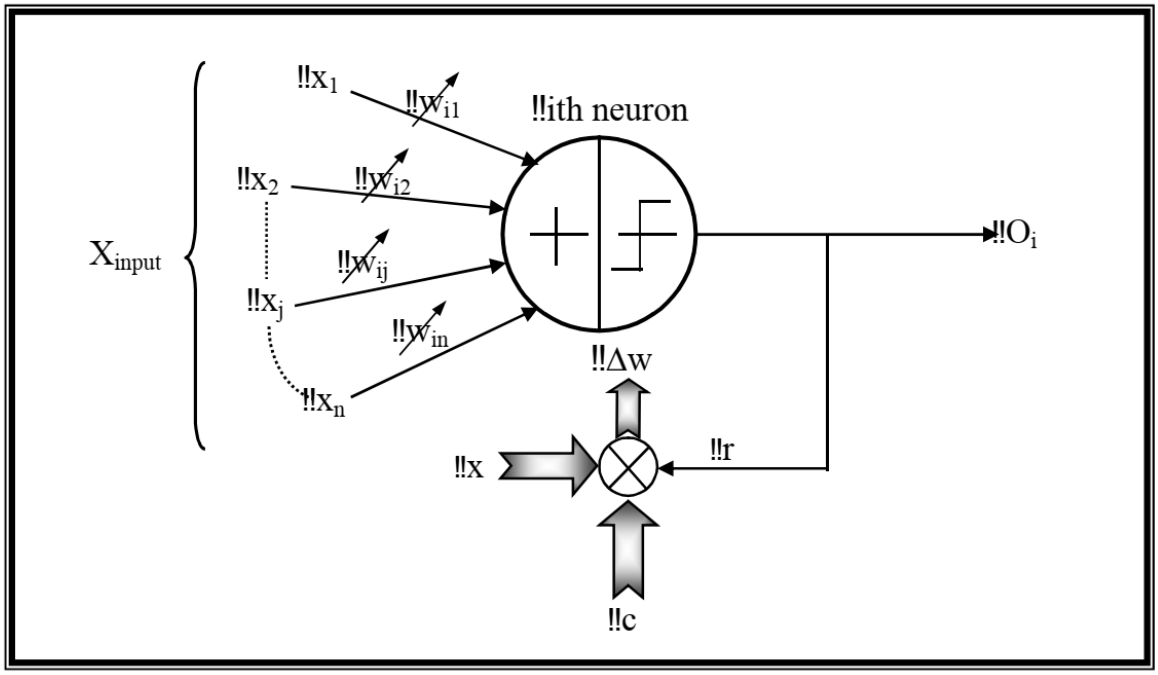

Figure (4) The Hebbian learning rule structure.

It can be seen from Figure (4) that:

$$w_i^1 = w_i^0 + \Delta w_i^0 \quad \ldots (10)$$

$$w_i^2 = w_i^1 + \Delta w_i^1 \quad \ldots (11)$$

$$\vdots$$

$$w_i^k = w_i^{k-1} + \Delta w_i^{k-1} \quad \ldots (12)$$

where (k) is the number of steps that the Hebbian learning rule need to learn the input signals.

A DIFFERENT VIEW

Neural networks are members of a family of computational architectures inspired by biological brains. Traditionally a neuron operates by receiving signals from other neurons through connections, called synapses. The combination of these signals, in excess of a certain threshold or activation level, will result in the neuron firing, i.e sending a signal to other neurons connected to it. Some signals act as excitations and others as inhibitions to a neuron firing which is the collective effect of the presence or absence of firings in the pattern of synaptic connections between neurons. Figure 1 shows the biological neuron.

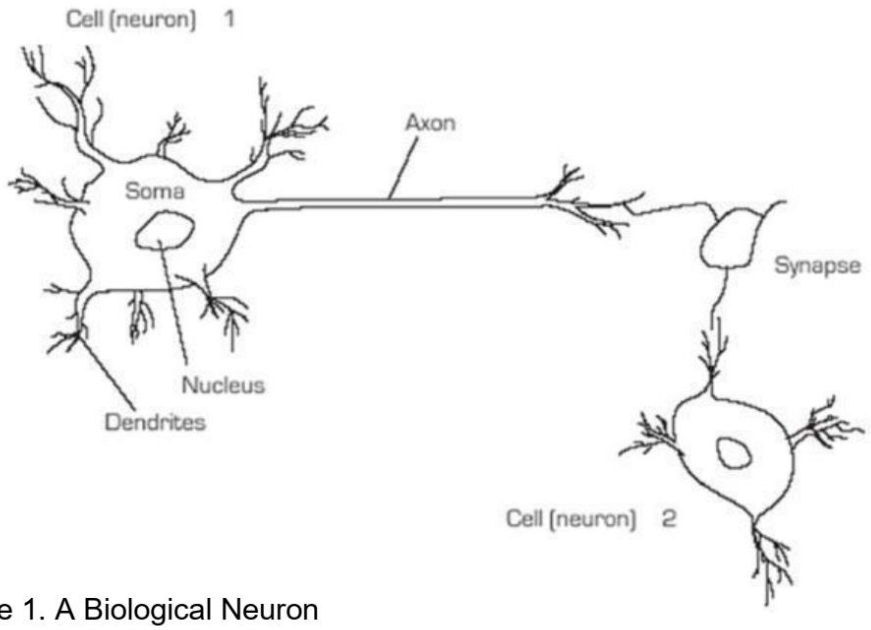

Figure 1. A Biological Neuron

There are approximately one hundred billion (100,000,000,000) neurons each connected to as many as one thousand (1,000) others in the human brain. The massive number of neurons and the complexity of their interconnections results in a "thinking machine", the brain.

Each neuron has a body, called the soma. The soma is much like the body of any other cell. It contains the cell nucleus, various bio-chemical factories and other components that support ongoing activity. Surrounding the soma are dendrites. The dendrites are receptors for signals generated by other neurons. These signals may be excitatory or inhibitory. All signals present at the dendrites of a neuron are combined and the result determines whether that neuron is fired or not.

If a neuron fires, an electrical impulse is generated. This impulse starts at the base, called the hillock, of a long cellular extension, called the axon, and proceeds down the axon to its ends. The end of the axon is actually split into multiple ends, called the boutons. The boutons are connected to the dendrites of other neurons and the resulting interconnections are through synapses. If a neuron has fired, the electrical impulse that has been generated stimulates the boutons and results in electrochemical activity which transmits the signal across the synapses to the receiving dendrites.

ARTIFICIAL NEURAL NETWORK

Artificial Neural Network (ANN) is an interconnected group of artificial neurons that uses

a mathematical model or computational model for information processing based on the human brain that tries to simulate its learning process. Neural networks are models of biological neural structures. The starting point for most neural networks is a model neuron, as in Figure 2. This neuron consists of multiple inputs and a single output. Each input is modified by a weight, which multiplies with the input value. The neuron will combine these weighted inputs and, with reference to a threshold value and activation function, use these to determine its output. Voice recognition, video compression, steering wheel and speed control, image processing, biomedical imaging, signal processing are some of the applications of ANN.

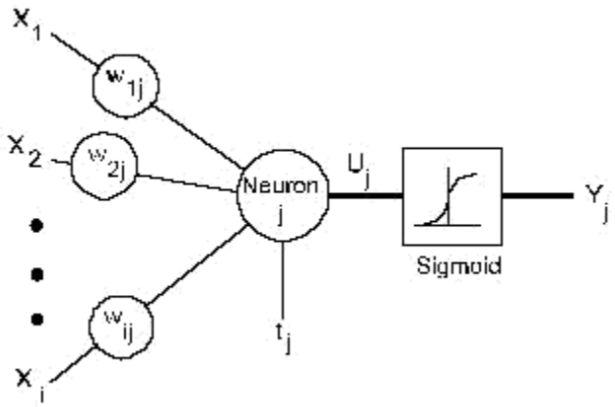

Figure 2. A Model Neuron

SUPERVISED AND UNSUPERVISED LEARNING

Supervised learning is the type of learning that takes place when the training instances are labelled with the correct result, which gives feedback about how learning is progressing. This is akin to having a supervisor who can tell the agent whether or not it was correct. In unsupervised learning, the goal is harder because there are no pre-determined categorizations. Figure 3 shows the classification of ANN based on their learning algorithms.

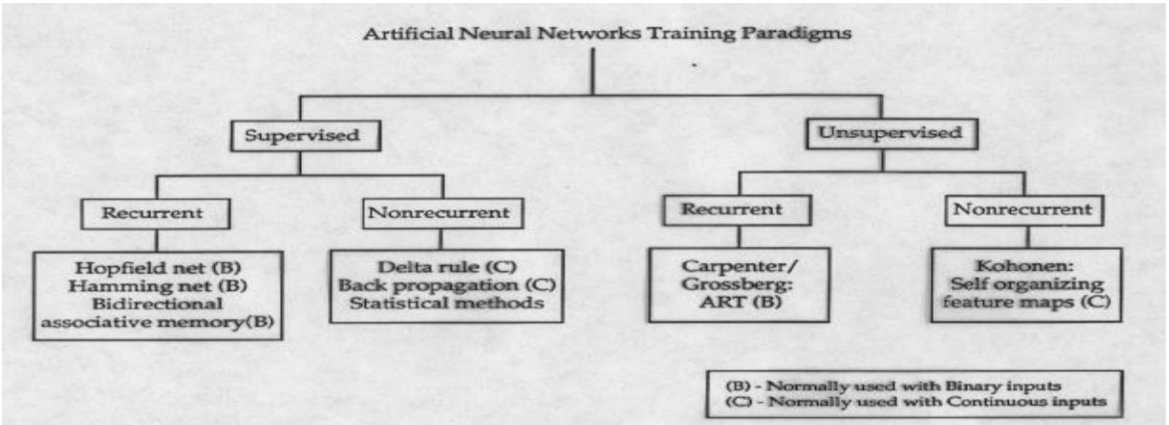

Figure 3. Classification of ANN

Supervised learning

Supervised learning is fairly common in classification problems because the goal is often to get the computer to learn a classification system that we have created. Digit recognition, once again, is a common example of classification learning. More generally, classification learning is appropriate for any problem where deducing a classification is useful and the classification is easy to determine. In some cases, it might not even be necessary to give pre-determined classifications to every instance of a problem if the agent can work out the classifications for itself. This would be an example of unsupervised learning in a classification context.

Supervised learning is the most common technique for training neural networks and decision trees. Both of these techniques are highly dependent on the information given by the pre-determined classifications. In the case of neural networks, the classification is used to determine the error of the network and then adjust the network to minimize it, and in decision trees, the classifications are used to determine what attributes provide the most information that can be used to solve the classification puzzle.

The goal of the learning algorithm is to minimize the error with respect to the given inputs. These inputs, often called the "training set", are the examples from which the agent tries to learn. With machine learning algorithms, a common problem is over-fitting the data and essentially memorizing the training set rather than learning a more general classification technique. Not all training sets have the inputs classified correctly. This can lead to problems if the algorithm used is powerful enough to memorize even the apparently "special cases" that don't fit the more general principles. This, too, can lead to overfitting, and it is a challenge to find algorithms that are both powerful enough to learn complex functions and robust enough to produce generalizable results. Pattern recognition and voice recognition are examples of supervised learning.

Unsupervised learning

Unsupervised learning seems much harder: the goal is to have the computer learn how to do something that we don't tell it how to do! There are actually two approaches to

unsupervised learning. The first approach is to teach the agent not by giving explicit categorizations, but by using some sort of reward system to indicate success. This type of training will generally fit into the decision problem framework because the goal is not to produce a classification but to make decisions that maximize rewards. This approach nicely generalizes to the real world, where agents might be rewarded for doing certain actions and punished for doing others.

Often, a form of reinforcement learning can be used for unsupervised learning, where the agent bases its actions on the previous rewards and punishments without necessarily even learning any information about the exact ways that its actions affect the world. In a way, all of this information is unnecessary because by learning a reward function, the agent simply knows what to do without any processing because it knows the exact reward it expects to achieve for each action it could take. This can be extremely beneficial in cases where calculating every possibility is very time consuming. But this kind of learning can be powerful because it assumes no pre-discovered classification of examples.

A second type of unsupervised learning is called clustering. In this type of learning, the goal is not to maximize a utility function, but simply to find similarities in the training data. The assumption is often that the clusters discovered will match reasonably well with an intuitive classification. For instance, clustering individuals based on demographics might result in a clustering of the wealthy in one group and the poor in another.

Although the algorithm won't have names to assign to these clusters, it can produce them and then use those clusters to assign new examples into one or the other of the clusters. This is a data-driven approach that can work well when there is sufficient data; for instance, social information filtering algorithms, such as those that Amazon.com use to recommend books, are based on the principle of finding similar groups of people and

then assigning new users to groups. In some cases, such as with social information filtering, the information about other members of a cluster (such as what books they read) can be sufficient for the algorithm to produce meaningful results. In other cases, it may be the case that the clusters are merely a useful tool for a human analyst. Unfortunately, even unsupervised learning suffers from the problem of overfitting the training data.

REFERENCES

1. Stuart J. Russell, Peter Norvig (2010) Artificial Intelligence: A Modern Approach, Third Edition, Prentice Hall ISBN 9780136042594.

2. Mehryar Mohri, Afshin Rostamizadeh, Ameet Talwalkar (2012) Foundations of Machine Learning, The MIT Press ISBN 9780262018258.

3. Cabannes, Vivien; Rudi, Alessandro; Bach, Francis (2021). "Fast rates in Structured Prediction". CoRR. arXiv:2102.00760.

4. S. Geman, E. Bienenstock, and R. Doursat (1992). Neural networks and the bias/variance dilemma. Neural Computation 4, 1–58.

5. G. James (2003) Variance and Bias for General Loss Functions, Machine Learning 51, 115-135. (http://www-bcf.usc.edu/~gareth/research/bv.pdf)

6. C.E. Brodely and M.A. Friedl (1999). Identifying and Eliminating Mislabeled Training Instances, Journal of Artificial Intelligence Research 11, 131-167. (http://jair.org/media/606/live-606-1803-jair.pdf)

7. M.R. Smith and T. Martinez (2011). "Improving Classification Accuracy by Identifying and Removing Instances that Should Be Misclassified". Proceedings of International Joint Conference on Neural Networks (IJCNN 2011). pp. 2690–2697.

8. Vapnik, V. N. The Nature of Statistical Learning Theory (2nd Ed.), Springer Verlag, 2000.

9. A. Maity (2016). "Supervised Classification of RADARSAT-2 Polarimetric Data for Different Land Features".

10. Hinton, Geoffrey; Sejnowski, Terrence (1999). Unsupervised Learning: Foundations of Neural Computation. MIT Press. ISBN 978-0262581684.

11. Roman, Victor (2019-04-21). "Unsupervised Machine Learning: Clustering Analysis". Medium.

12. Snow, Dr Derek (2020-03-26). "Machine Learning in Asset Management: Part 2: Portfolio Construction—Weight Optimization". Journal of Financial Data Science. 2 (2): 17–24.

13. Jordan, Michael I.; Bishop, Christopher M. (2004). "Neural Networks". In Allen B. Tucker (ed.). Computer Science Handbook, Second Edition (Section VII: Intelligent Systems). Boca Raton, Florida: Chapman & Hall/CRC Press LLC. ISBN 1-58488-360-X.

14. Hastie, Trevor, Robert Tibshirani, Friedman, Jerome (2009). The Elements of Statistical Learning: Data mining, Inference, and Prediction. New York: Springer. pp. 485–586. ISBN 978-0-387-84857-0.

15. Garbade, Dr Michael J. (2018-09-12). "Understanding K-means Clustering in Machine Learning". Medium. Retrieved 2019-10-31.

16. Anandkumar, Animashree; Ge, Rong; Hsu, Daniel; Kakade, Sham; Telgarsky, Matus (2014). "Tensor Decompositions for Learning Latent Variable Models" (PDF). Journal of Machine Learning Research.

17. Hinton, G (2010-08-02). "A Practical Guide to Training Restricted Boltzmann Machines".

18. Buhmann, J.; Kuhnel, H. (1992). "Unsupervised and supervised data clustering with competitive neural networks". [Proceedings 1992] IJCNN International Joint Conference on Neural Networks. 4. IEEE. pp. 796–801.

19. Comesaña-Campos, Alberto; Bouza-Rodríguez, José Benito (June 2016). "An application of Hebbian learning in the design process decision-making". Journal of Intelligent Manufacturing.

20. Carpenter, G.A. & Grossberg, S. (1988). "The ART of adaptive pattern recognition by a self-organizing neural network" (PDF). Computer. 21 (3): 77–88. doi:10.1109/2.33. S2CID 14625094.

21. Bousquet, O.; von Luxburg, U.; Raetsch, G., eds. (2004). Advanced Lectures on Machine Learning. Springer-Verlag. ISBN 978-3540231226.

22. Duda, Richard O.; Hart, Peter E.; Stork, David G. (2001). "Unsupervised Learning and Clustering". Pattern classification (2nd ed.). Wiley. ISBN 0-471-05669-3.