SECTION 4

Neural-Type Logic Devices

The pursuit of Artificial Intelligence leads to the neuron and its functioning in the operation of a brain or an "intelligent system".

THE NEURON AND MAJORITY LOGIC

The *neuron* is a special type of biological cell which is the operating component in the nervous system of all life on Earth that has a nervous system, whether human, animal, insect or whatever. By *neural-type logic* is meant systems in which the principal operating component is the *neuron* or systems in which the principal operating component is a device, a man-made device, that operates *logically in the same way as a neuron.*

The logic technique used in such neural-type rational systems, including the human brain, is slightly different from the *and / or* logic examined so far. The basic logic function (logic procedure) used in biological systems is *majority logic*. Using the notation *M(…)*, where the *M* stands for *majority of* and the ellipsis *[…]* stands for the listing of the variables, the quantities, involved (e.g. array or retina element signals) then a translation between *majority* and *and / or* logic is, for example

(4-1) M(A,B,C) = AB + AC + BC + ABC

That is, a majority logic operator has an output of *on* if a majority of its inputs are *on* and otherwise an output of *off*. In the example of equation *4-1* any two of the three variables is a majority of them.

For convenience of notation, and because Boolean algebra employs binary logic [a logic based on the binary number system having base *2* instead of *10* and digits *0* through *1* instead of *0* through *9*], the binary digit *1* will be used to represent *on* or *yes* or *satisfied* hereafter with regard to Boolean algebra or Majority expressions and the digit *0* to represent the opposite. In those terms equation *4-1* states that the output is *1* if any two or all three of the inputs are *1*. Otherwise the output is *0*.

In addition to variables such as the *A*, *B*, etc. already used, majority logic can also use *logic constants*. Here a constant is like a variable in all respects except that it always has the same, fixed value. Since the system is binary there are only two values that a constant can have, *1* or *0*.

In *and / or* logic, constants are essentially meaningless as the following examples illustrate.

(4-2) A + B + 1 = 1 [In spite of the variables the result is always "1". The variables have no meaning because of the constant.)

A + B + 0 = A + B The constant has no effect.]

A·B·1 = A·B [The constant has no effect.]

A·B·0 = 0 [In spite of the variables the result is always "0". The variables have no meaning because of the constant.]

However, in *majority logic*, *constants* play a useful and important role; they enable *majority logic* to represent *Boolean logic*. For example:

(4-3) M(A,B,1) = A·B + A·1 + B·1 = A + B

M(A,B,C,1,1) = (analogous) = A + B + C

M(A,B,0) = A·B + A·0 + B·0 = A·B

M(A,B,C,0,0) = (analogous) = A·B·C

The *not* operation still applies in *majority logic*; that is, the majority operation may operate on *natural* or *not-ed* variables. For example

(4-4) M(A,$\underline{B}$,$\underline{C}$,1,1) = A + $\underline{B}$ + $\underline{C}$

M($\underline{A}$,B,0) = $\underline{A}$·B

Thus majority logic with both constants and variables can produce all of the fundamental type logical constructs that Boolean logic uses.

Likewise, a majority operation's output can be an input variable in another majority operation just as in Boolean logic. For example

(4-5) M(A, [M(B,C,0)] ,1)

where the bracket indicates the "*The Majority of B*, *C* and *0*" as one of the variables in the overall expression, which reads as "*The Majority of A*, *The Majority of B*, *C* and *0*, and *1*. Such complex majority operations, which can have many more levels than the two-level case illustrated in equation *4-5*, enable majority logic to implement any Boolean logic whatsoever.

In fact majority logic can do more than that. The very same physical structure, that is the same connection of inputs to a given majority processor, can yield controllably different logical constructs, logical results, depending on the value of the constants applied to that majority processor. Majority logic makes possible fixed "pre-wired" interconnections in a configuration where the logical effect of the physically

fixed structure can be controlled and varied by varying the values of the constants involved.

That is precisely the process that goes on in a rational system based on neurons, whether that system is in a human, a cow, an ant or whatever. The inputs to a neuron are the outputs of other neurons or of sensors (e.g. the retina of the eye). Those inputs are such that some act on the neuron in an *excitatory* fashion and some act on it in an *inhibitory* fashion. That is, *excitatory* inputs are analogous to `natural` variables (as opposed to `not-ed` ones) and have the logical effect of an input of `1` if activated and `0` if not. *Inhibitory* inputs are analogous to `not-ed` variables and have the logical effect of an input of `0` *if activated and* `1` if not.

In a neuron the presence or absence of a majority is not determined by counting the total possible inputs and comparing the number of them that are `1` to that count. Rather the effect is as if the `1` inputs are each `+1` (excitatory) and the `0` inputs are each `-1` (inhibitory). If the algebraic sum, the excitatory plus the inhibitory (the number of excitatory less the number of inhibitory), is greater than zero then a majority is present.

There is still another component of a neuron's operation, however. That *algebraic sum* of the excitatory `+1` and the inhibitory `-1` inputs is not compared to `zero` as such. Rather it is compared to a *threshold* level present in that neuron. If the *threshold* happens to be `zero` then the logical construct of the neuron is simply the majority of its inputs.

But, if the threshold is greater than `zero`, meaning that for the neuron to have an output of `1` the number of excitatory inputs must be that much (the *threshold* amount) greater in number than the number of inhibitory inputs, then the effect is the same as if there were as many constants equal to `0` present and acting as the level of the *threshold*. Likewise, a *threshold* less than `zero` corresponds to there being that many constants equal to `1` present and acting. Thus the value of the *threshold* represents the net value of constants in the input and variation of the *threshold* produces variation of the net value of the constants which produces variation in the Boolean logic that the majority operator is equivalent to.

For example, if the inputs to the neuron are `A`, `B`, `C`, … and all of them are excitatory (simply for this example), then:

(4-6)

With Threshold	The Neuron Performs
0	M(A,B,C,…)
+1	M(A,B,C,…,0)
+2	M(A,B,C,…,0,0)
-1	M(A,B,C,…,1)
-2	M(A,B,C,…,1,1)

The threshold is equivalent to the net number of constants involved, constants of `-1` for positive threshold and of `+1` for negative threshold. The output is `1` if the majority of the input variables and those constants is greater than `zero`.

But, the special power of the neuron is that its threshold can be changed. That means that its constants can be changed and that means that the logical effect, the Boolean logic that the neuron is implementing, can be changed. The neuron "remembers" the value of the threshold so that the threshold is, in that sense, some set number of majority logic constants operating as such in the logical construct that the neuron effects. However, that set value or level of the threshold can be changed, adjusted so that the logical construct that the neuron effects is slightly, gradually changed. It is that process that enables learning. Learning is, in effect, the directed adjustment of neural thresholds to achieve the desired result.

The input to the neuron from other neurons or from sensors is received by the neuron as various excitatory and inhibitory, *+1* and *–1*, inputs. The neuron emits an output that is *1* or *0* depending on the internal operation of the neuron. That output acting as an excitatory input to another neuron is a *+1* input to it if the output was *1*. That output acting as an inhibitory input to another neuron is a *–1* input to it if the output was *1*. The internal operation of the neuron simply determines whether the majority of the inputs plus the threshold is greater than zero (neuron output is *1*) or not (neuron output is *0*). (How the threshold changes occur will be treated shortly, in the next section of this work.)

Actual biological neurons operate in this manner. A single biological neuron consists of a central cell body, a number of input lines (filaments or fibers of cell material) called dendrites, and an output line (also a filament or fiber of cell material) called an axon. Output signals of neurons travel to the end of the axon where they then communicate, as inputs, with other neurons. The junction where the signal transmission from neuron to neuron takes place is called a synapse. Within a neuron some of the inputs are excitatory and some are inhibitory. The threshold, at the main cell body, determines whether the net effective input signal causes or fails to cause an output signal on the axon. The processes within the neurons and at the synapses are electrochemical in nature.

When neurons, whether biological or man made neural-type electronic devices, are interconnected so that the outputs of some neurons are inputs to other neurons then a multilevel neural network exists. Such a network makes possible neuron-implemented complex majority logic structures that can effect logic such as illustrated in equation *4-5*. Multilevel networks of neurons use the neuron's majority logic, modified by the individual neuron's thresholds, to represent the equivalent of complex Boolean logical descriptions. Such descriptions are the logical representation of universals. Complex neural networks can represent specific universals if the individual neural thresholds are correctly set to make them do so.

HOW NEURONS CAN BE TAUGHT; HOW THEY LEARN

Let us now operate a simple such neural network using as its input the sample four-by-four, *16* element, array used in Section 2. That array was there used to illustrate the universal *cross-ness* among the various possible images that could appear as input on the array.

An individual neuron or neural-type device will be symbolized here as in Figure 4-1, below.

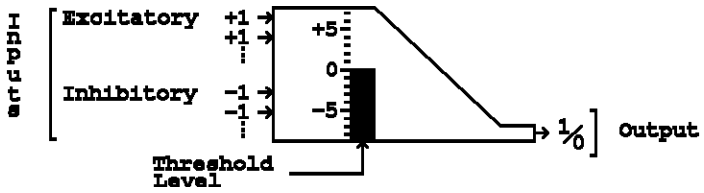

Figure 4-1

The outputs of the four-by-four array will be interconnected to the inputs of a number of such neurons and then the outputs of those *first level* neurons will be interconnected to the inputs of one more neuron. The output of that final, single, neuron will be deemed the representation of the action of the entire neural network. (See Figure 4-2 below.)

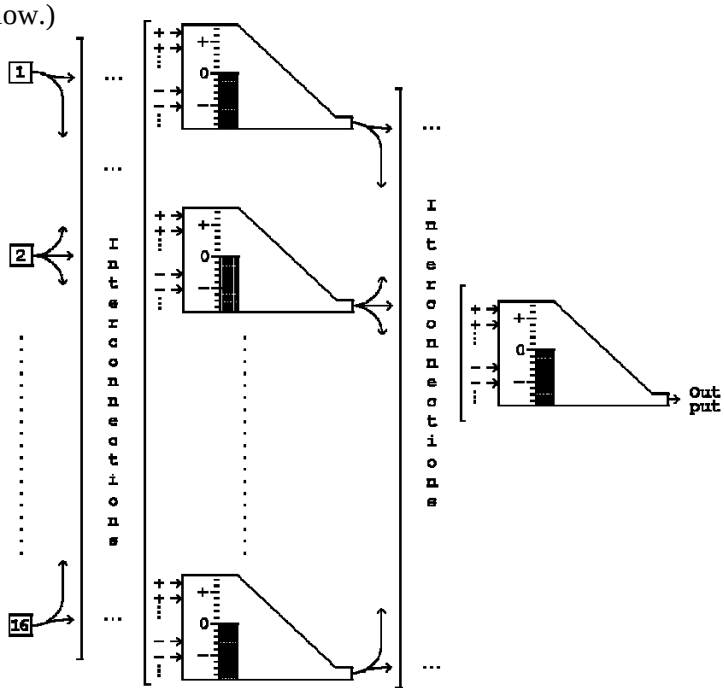

Figure 4-2

But, how should the interconnections be made; that is, which sensors should be connected to which inputs of which neurons and which excitatory and which inhibitory ? This question is quite fundamental to neural networks as is the matter of how threshold changes occur. As with the control of threshold changes, the subject will be treated fully in the following sections. For the moment let us assume that those aspects of the problem have been correctly implemented in the sample neural network being used.

Let us now teach the neural network to recognize the universal *cross-ness*; that is, let us cause it to learn how to discriminate between input images exhibiting *cross-ness* and those lacking it. Our objective is that the neural network should give an output of *1* if the image has *cross-ness* and *0* otherwise.

We Use the Following Procedure.

(1) Show the input array an input image (project an image onto the four-by-four, *16* element array). That is, cause various of the *16* elements in the array to be *on* and others *off* so that the desired pattern is represented on the array.

(2) We (being in this case the teacher, the authority) note whether the image exhibits the universal *cross-ness* or not. (The problem of where, in general, the teacher comes from is also addressed in the next sections.)

(3) Observe the output of the neural network (whether it is *1* or *0*).

(4) Evaluate the performance of the network which could be any of the following four possible cases.

Input Image	Output	Result
cross	1	correct
cross	0	wrong
not cross	1	wrong
not cross	0	correct

(5) Change the threshold of each neuron of the neural network as follows:

- If the neural network output was correct reinforce that behavior by adjusting each neuron's threshold in the direction that makes that neuron's output result more likely.
    - If its output was *1* lower its threshold by *1* unit (making even more likely an output of *1* for another input like this one).
    - If its output was *0* raise its threshold by *1* unit (making even more likely an output of *0* for another input like this one).
- If the neural network output was wrong discourage that behavior by adjusting each neuron's threshold in the direction that makes that neuron's output result less likely.
    - If its output was *1* raise its threshold by *1* unit (making less likely an output of *1* for another input like this one).
    - If its output was *0* lower its threshold by *1* unit (making less likely an output of *0* for another input like this one).

(6) Repeat the above five steps using a different input image each time until the neural network's performance is sufficiently consistently correct.

This has the appearance of a reward-and-punishment type procedure but that is not the case here. The neurons do not understand anything, certainly not reward and punishment. The procedure simply changes the thresholds in a direction tending to increase the chances that for input images similar to the one just processed the neural network's operation on the input variables, with its now changed thresholds, will more likely yield the desired correct output.

But, whether the neurons "understand" this or not is irrelevant. The end result of the process is that the neural network actually becomes able to discriminate *cross-ness* even though at the start of the process it could not do so. The neural network has learned, has been taught by the teacher, to discriminate. It effectively *perceives* the *universal* that was taught, *cross-ness* in this example, having *learned* to do so.

That learning was accomplished by directed, logical adjustments to each neuron's threshold level. Such adjustments have already been shown to change the Boolean logical construct that is effected by each neuron's majority operation in conjunction with the constants represented by its threshold.

In other words, the above described learning process causes the Boolean logical construct or operation that the neural network performs on the input variables to gradually change until it is identical to, or it sufficiently resembles, the Boolean logical construct that corresponds to the universal being taught.

The accomplishment of that **is** the learning to perceive that universal. The subsequent using of that to make correct outputs in response to input images **is** the perceiving of that universal.

[This concept and laboratory research with regard to it were first developed and pursued at the Cornell Aeronautical Laboratory in the latter 1950's. The research was reported in the Proceedings of the Electronic Computers Group of the (then) Institute of Radio Engineers, IRE, (now the Institute of Electrical and Electronic Engineers, IEEE) circa 1960. The neuron simulator device, operating as herein described, was called the "perceptron". Laboratory development demonstrated that the type device does learn and operate as here described.

[The first generation of commercially produced machines using these principles, appearing on the market, and being used was in the mid 1990's. The machines employed neural networks similar to those described above. The machines were used to perceive patterns in data in situations where humans may be too slow or unable to perceive the pattern.]

IN GENERAL SUMMARY SO FAR

- Perception is the correlating of an experienced specific example with a universal, a class to which it belongs.

- Learning is the developing of the ability to so perceive.

- The perception is accomplished by using

and the learning is the process of constructing

a logical mechanism that operates on the experienced example

in a fashion that detects the presence or absence of the universal.

- That "logical mechanism" is a physical implementation of, in effect, a Boolean logical expression that conforms to the universal.

- The "logical mechanism" is "constructed", exists and operates, by means of majority logic with constants as implemented by neurons or neural-type devices having majority logic and adjustable thresholds.

While this process has been discussed in terms of our sense of vision the same process operates with regard to all of the senses: hearing, smell, touch, etc. Hearing involves the universals in sounds and hearing and understanding language involves universals just as numerous and complex in their effect as in the case of vision. The blind read by their sense of touch and process a similarly numerous and complex set of universals through their fingertips. And some of the animals, unlike we humans, derive quite extensive information from their sense of smell.

Next: The complex perception mechanism