OPTOELECTRONIC IMPLEMENTATION OF ARTIFICIAL NEURAL NETWORK: PERCEPTRON LEARNING RULE AND M-CATEGORY CLASSIFIER

Asst. Prof. Dr. M. S. Abdul-Wahab Asst. Prof. Dr. Hanan A. Reda Akkar
O. Q. J. Al-Thahab
Department of Electrical And Electronic Engineering
University of Technology
Baghdad – Iraq

ABSTRACT

Single neuron perceptron is designed as a classifier of two different classes using the hard-limiter activation function (i.e. in the absence of light, and presence of light). An example is designed and tested so that the proposed circuit learned different categories and then used as a classifier for two different classes because of the use of single neuron. Additional electronic circuits were used for computation processes. The Computer simulation results indicate stable solution that compares with theoretical results.

Single layer perceptron M-category classifier is designed as a classifier for more than two classes. An example is designed and tested for the verification. The example learns after (5) iterations. Computer simulation results indicate stable solution that compares favorably with theoretical results.

الخلاصة

تم اعتماد نموذج برسبترون كأحد النماذج التي تطبق بوجود المشرف، وتم فحص مثال على الدائرة المقترحة، حيث تم تعليم الدائرة أنماط مختلفة ومن ثم استخدامها كمصنِّف لنمطين فقط بسبب استخدام خلية عصبية واحدة. لقد استُخدمت دوائر إلكترونية أخرى مساعدة لتحقيق العمليات الحسابية التي نحتاجها في تمثيل هذا النموذج، وقد أظهرت الدائرة المقترحة نتائج مقاربة للنتائج النظرية من خلال محاكاة الدائرة المقترحة حاسوبياً.

كذلك تم تصميم نموذج برسبترون لتصنيف الأنماط المتعددة من أجل تصنيف عدد أكبر من الأنماط (أكثر من اثنين)، وقد تم فحص مثال عليها وكانت النتائج مقاربة للنتائج النظرية، كما وتم استخدام دالة المحدد كدالة فعالة في الدائرة المقترحة.

INTRODUCTION

Learning theory develops models from data in an inductive framework. It is, therefore, no surprise that one of the critical issues of learning is generalization. But before generalization, the machine must learn from data [Principe 2000]. Learning can be viewed as maximizing the likelihood of observed data under the generative model, which is mathematically equivalent to discovering efficient ways of coding the sensory data [Chahramani 1999]. While neural networks can and are implemented entirely with electronic hardware, optoelectronics have a clear advantage in the task of interconnecting neurons.

Unlike electronic signal, light beams do not interact with one another, permitting many connections to be made in the same space. Since photons do not interact with one another, it is difficult to implement directly the processing operation of the neuron with optical devices, thus many systems combine optical synaptic interconnections with optoelectronic neurons [Wilamowski 2000].

In this paper, a variety of issues are discussed that have impact on the development of optoelectronic technology as applied to hardware implementation of neural networks with enhanced capabilities. Optoelectronics has the potential for the implementation of neural networks with large numbers of neurons (10^5 to 10^6) and high connectivity (approximately 10^{10} analog weighted interconnections) in one module. The approach taken here is to use electronics to implement the internal functions of each neuron unit, and to use optics to implement the connections, weight, and input and output. With this technique, most of the area of a two-dimensional “chip” can be used to implement the neuron units themselves [Keller 1993]. Analog optoelectronic hardware implementation of neural nets has been the focus of attention for several reasons. Primary among these is that the optoelectronic approach combines the best of two worlds: the massive interconnectivity and parallelism of optics, the flexibility, high gain, and decision making capability (non linearity) offered by electronics [Keller 1992].

OPTOELECTRONIC NEURAL NETWORK

Optical implementation of neural networks is promising for a variety of reasons. First of all, light offers the fastest possible communication channel, not requiring physical limiting conductors. Secondly, increasing light beams do not noticeably interfere with each other. This means that the large number of interconnections of artificial neural network can be optically implemented in a compact way of truly parallel implementation of large neural networks. Optical implementations of multilayer neural network perform nonlinear thresholding which is an essential constituent of all neural network models, and hence involve conversion of optical signals to electronic ones and vice versa. In order to avoid this conversion and to progress to all – optical forward propagation in multilayer neural networks, the use of optical activation function is essential [Al-Buhrezi 2001].

Desirable features for the field of optical neural networks are the ability of performing subtraction and an ideal optical linearity, which is spatially uniform in its response. In general, spatial non - uniformities are measured as a variation in the read out intensities.

The fan – out optics and other optical elements may have non-ideal behavior. These non-uniformities are expected to be compensated to a considerable extent in an adaptive optical neural network, the weights of which are updated during the training of the actual optical system.

Generally, in optoelectronic neural system each neuron is composed of an input summing port, nonlinear transfer device, and an output port. Differential pair of detectors is operated as the input to the neuron; signals with positive (excitatory) weights arrive at one detector and signals with negative (inhibitory) weights arrive at the other detector. These detectors sum the intensity of each optical signal arriving at the neuron. The neuron’s activation function is electronically applied to the detected signal to produce an output signal. The output signal drives either an optical source or pair of sources [Chew 1993].

PERCEPTRON LEARNING RULE

For the perceptron learning rule, the learning signal is the difference between the desired and the actual neuron's output. Thus, learning is supervised and the learning signal is equal to:-

$$r = d_i - o_i \qquad \dots (1)$$

where $o_i = f(w_i^t x)$, and (di) is the desired response as shown in **Fig.(1)**. Weight adjustment in this method, (Δwi), and (Δwij), is obtained as follows:-

$$\Delta w_i = c[d_i - f(w_i^t x)]x \qquad \dots (2a)$$

$$\Delta w_{ij} = c[d_i - f(w_i^t x)]x_j , \qquad \dots (2b)$$

for j=1, 2, …, n and t:- iteration

Under this rule, weights are adjusted if and only if (oi) is incorrect. Error as a necessary condition of learning is inherently included in this training rule. Obviously, since the desired response is either (1) or (-1), the weight adjustment (2a) reduces to:-

$$\Delta w_i = \pm 2cx \qquad \dots (3)$$

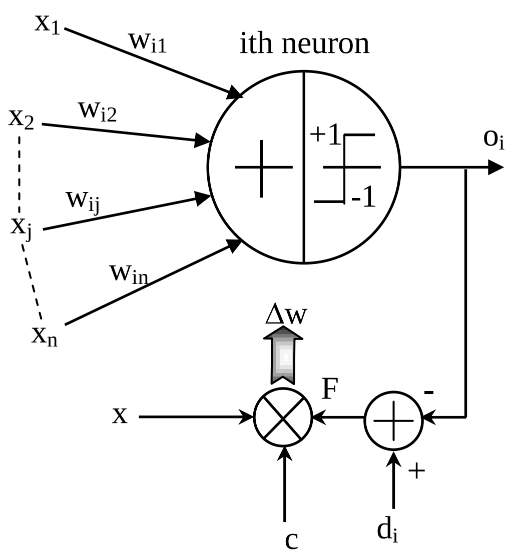


Fig. (1) Perceptron learning rule.

EQUIPMENT NEEDS FOR THE DESIGN OF PERCEPTRON LEARNING RULE

Even though neural networks are primarily implemented in software, their good approximation properties make them attractive in hardware [Moerland 1996], as seen from the previous section (i.e. from perceptron model). First the multiplication of the input signal by the stored weight must be performed. A simple multiplier can achieve this, and it must be four-quadrant multiplier, use of the four quadrant analog Gilbert multiplier satisfies this purpose. After this stage, a summing device is used. It can be seen that the summer is either for voltage or for currents, as seen from **Fig.(1).**

SUMMER-DRIVER CIRCUIT

The summing circuit is shown in **Fig.(2),** it can be seen that if the summation is grater than (zero) then the output will be in logic (0) because the transistor reaches the saturation region or the output will be logic (1) when the value of summation is less then (zero) because the transistor falls onto the cut off region.

This circuit will be used to drive the light emitting diode as shown in **Fig.(3)**. When the transistor turns (ON) and if (Ic) is greater than the turn (ON) current of the (LED), the (LED) will light up, and this can be represented as the (ON) state or logic (1). Else if (Ic) is less than the turn (ON) current of the (LED), the (LED) will not light up, so this case can be considered as logic (0). The turn (ON) current for the (LED) used in this circuit is (10 mA).

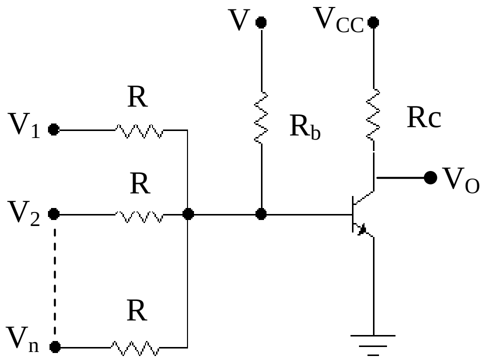


Fig. (2) The current summing circuit

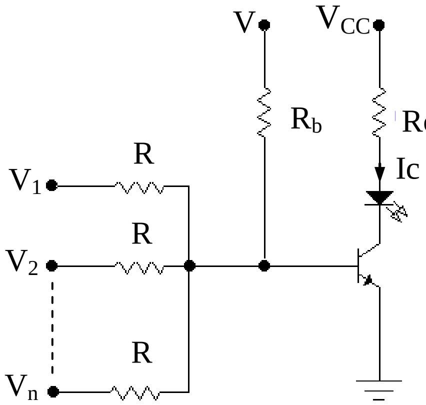


Fig.(3) The sum-driving circuit

FOUR QUADRANT ANALOG GILBERT MULTIPLIER

A Four Quadrant Analog multiplier is a device in which the output voltage is directly proportional to the product of the two input voltages regardless of the polarity of the inputs thus:-

$$V_O = k\,V_1 V_2 \qquad \text{… (4)}$$

where (V_1) and (V_2) can be either positive or negative.

The basic Four Quadrant Analog multiplier is Gilbert multiplier as shown in **Fig.(4),** having:-

$$V_{out} = Rc\,Ix\ \tanh\left(\frac{V_1}{2Vt}\right)\ \tanh\left(\frac{V_2}{2Vt}\right) \qquad \text{…(5)}$$

If (V_1) and (V_2) << V_t then equation (4) can be written as:-

$$V_{out} = Rc\,\frac{V_1 V_2}{4Vt^2} \qquad \text{…(6)}$$

Where V_t is the threshold voltage.

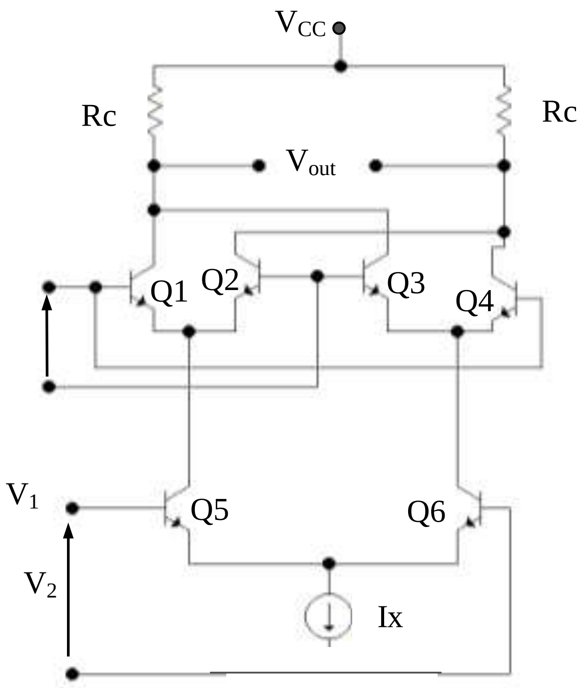


Fig. (4) Gilbert multiplier.

The current source as shown in **Fig. (5)**, illustrate that:-

$$Ix = \frac{1}{R_3}\left(\frac{V_{EE}R_2}{R_1+R_2} + \frac{V_D R_1}{R_1+R_2} - V_{BE7}\right) \qquad \dots (7)$$

Since this current is independent of the signal voltages (V_1) and (V_2), then (Q7) acts a supply of constant current to the Gilbert multiplier.

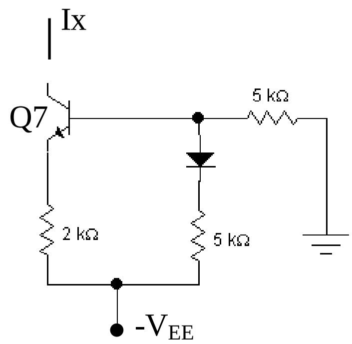


Fig. (5) The current source circuit

The CMOS Inverter

The (CMOS) inverter circuit is shown in **Fig.(6)**, which consists of complementary (NMOS) and (PMOS) transistors, and having (V_{SS}=-V_{DD}), Vthn= -Vthp (Vthn=1.4 V and Vthp= -1.4 V). If we want to use this circuit to be in logic mode then V_{SS} is set to zero.

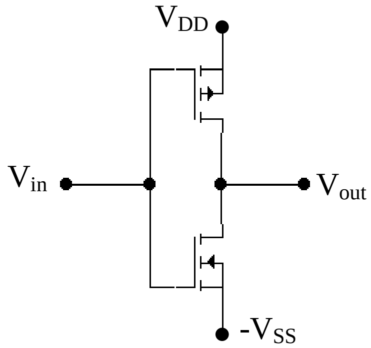


Fig.(6) The CMOS inverter circuit.

THE CMOS ANALOG SWITCH

Connecting an (NMOS) device in parallel with a (PMOS) forms a (CMOS) analog switch also called bilateral transmission gate. This combination acts like two parallel switches, both of which are simultaneously closed or simultaneously open. Actually when the switches are closed, both conduct small positive and negative signals, but only one conducts large positive signals, while other conducts large negative signals. **Fig.(7)** shows how the devices are connected.

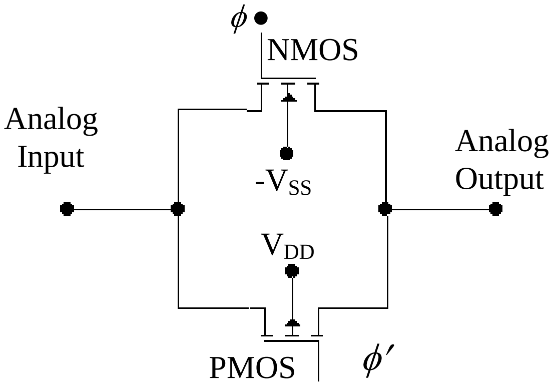


Fig. (7) The CMOS analog switch.

PROPOSED DESIGN OF SINGLE NEURON PERCEPTRON LEARNING RULE

The main problem in all learning rule designs is the storage element (i.e. where the updating weight is stored), and it can be taken as a capacitor. The synaptic weight can thus be stored as a charge on the capacitor (storage capacitor).

Fig.(8) illustrates the block diagram of the proposed circuit for the perceptron learning rule for (n) inputs. It includes elements of all above mentioned circuits. This circuit needs two storage capacitors: the first one, (C1) is used to store the updated weight w(t+1) then transfers its charge to the second capacitor (C2) which is used to store the weight (w(t)) as charge (C1=1 μF. and C2=10 μF.).

The photodiode used in the design of this circuit supplies a current (Iph=100 μA). Ones sees that when the photodiode is illuminated by an optical signal the output of the drive circuit will be (-1) in order to achieve the subtraction of the neuron output from the desired signal, and when there is no optical signal then (Iph= 0), the neuron output will be (1) for the same reason mentioned above. A simple subtraction circuit can be used to achieve this purpose.

The initialization of weights is stored on the storage capacitor before the learning operation takes place. The proposal presented above has been implemented with the aid of the software package (EWB) program, and verified by (MATLAB) programs.

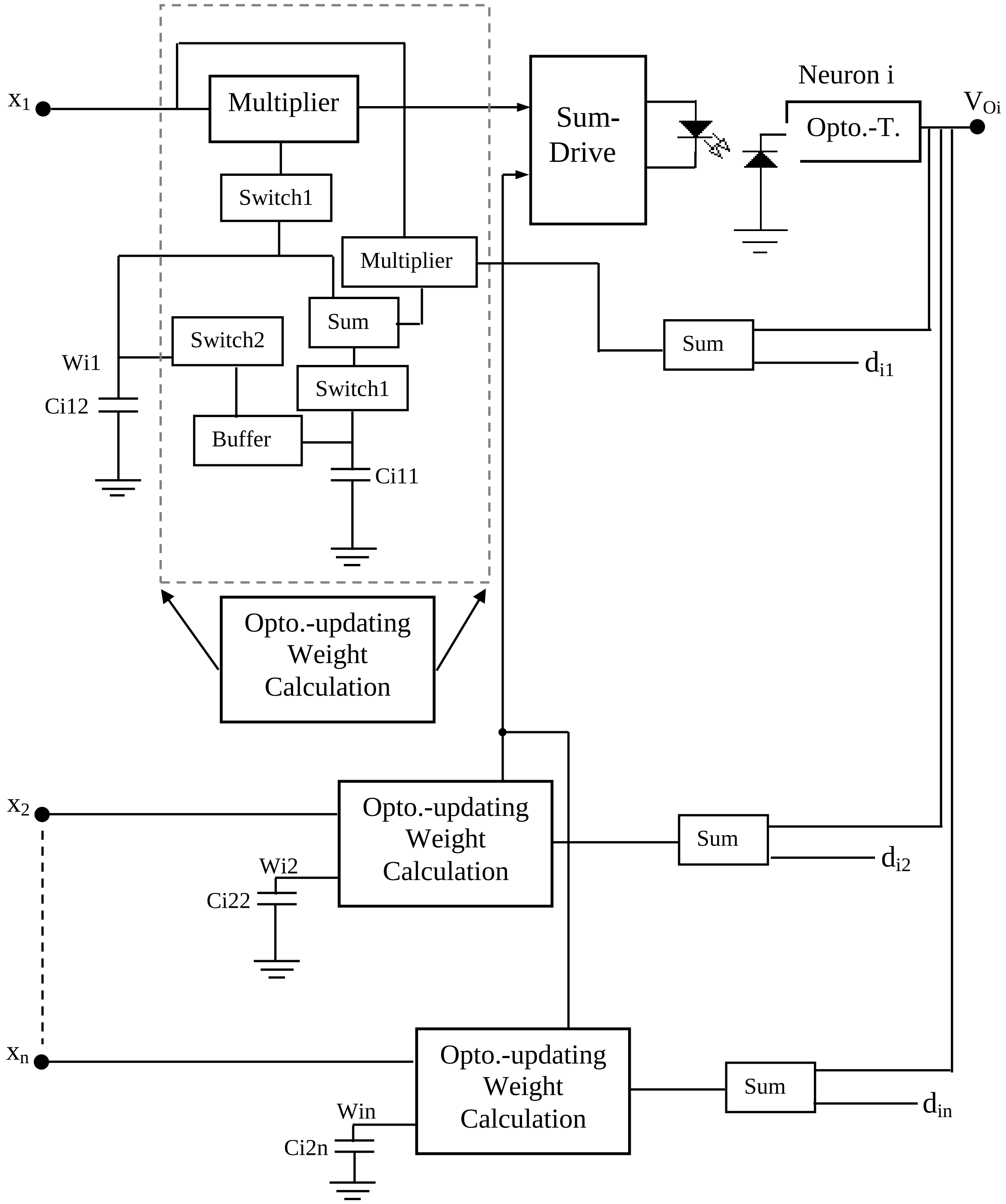


Fig.(8) The block diagram of the proposed implementation circuit for perceptron learning rule.

EXAMPLE 1

A single neuron can be designed to classify two different input patterns. **Fig. (9)** shows an example of two patterns (M=2 and number of nodes n=12), which are the characters (O, j) entered to the proposed circuit of the perceptron rule with the desire (1, -1) respectively, The circuit will learn until it reaches the final weight when w(t+1)=w(t).

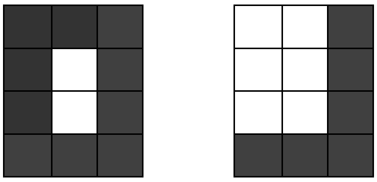


Fig. (9) Two different input patterns.

The results are illustrated in **table (1)** obtained from the simulation results of the (MATLAB) computer programs for (example 1). It can be seen that the learning process stopped after three iterations. As an examination, two distorted patterns are examinated. The vector of the distorted pattern shown in **table (1)**, has the shape seen in **Fig. (10)** together with the correct pattern. The proposed circuit block diagram of this example is shown in **Fig.(11).**

Table (1) Simulation result for example 1.

Pattern 1	Pattern 2	Initial Weight W^0	Updating Weight W^3	Distorted Input	
				Pattern 1	Pattern 2
1	-1	-.5	-2.5000	1	1
-1	1	-1	-3.0000	-1	1
-1	-1	.3	-1.7000	-1	1
1	-1	-.5	-2.5000	1	-1
-1	1	.1	-1.9000	-1	-1
-1	-1	0	-2.0000	1	1
1	-1	.3	-1.7000	-1	-1
-1	1	.1	-1.9000	-1	-1
-1	-1	-.4	-2.4000	1	1
1	-1	-.2	-2.2000	1	1
1	1	-.3	-2.3000	1	1
1	-1	1	-1.0000	-1	1

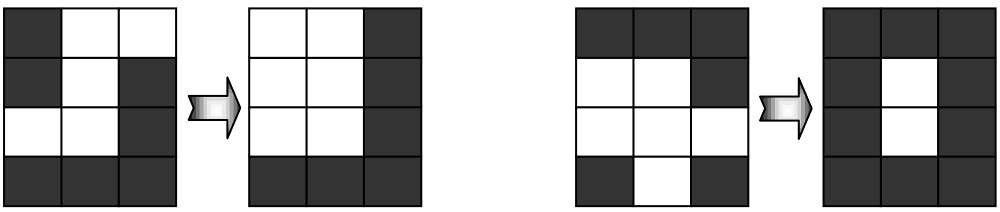


Fig. (10) Distorted pattern examination

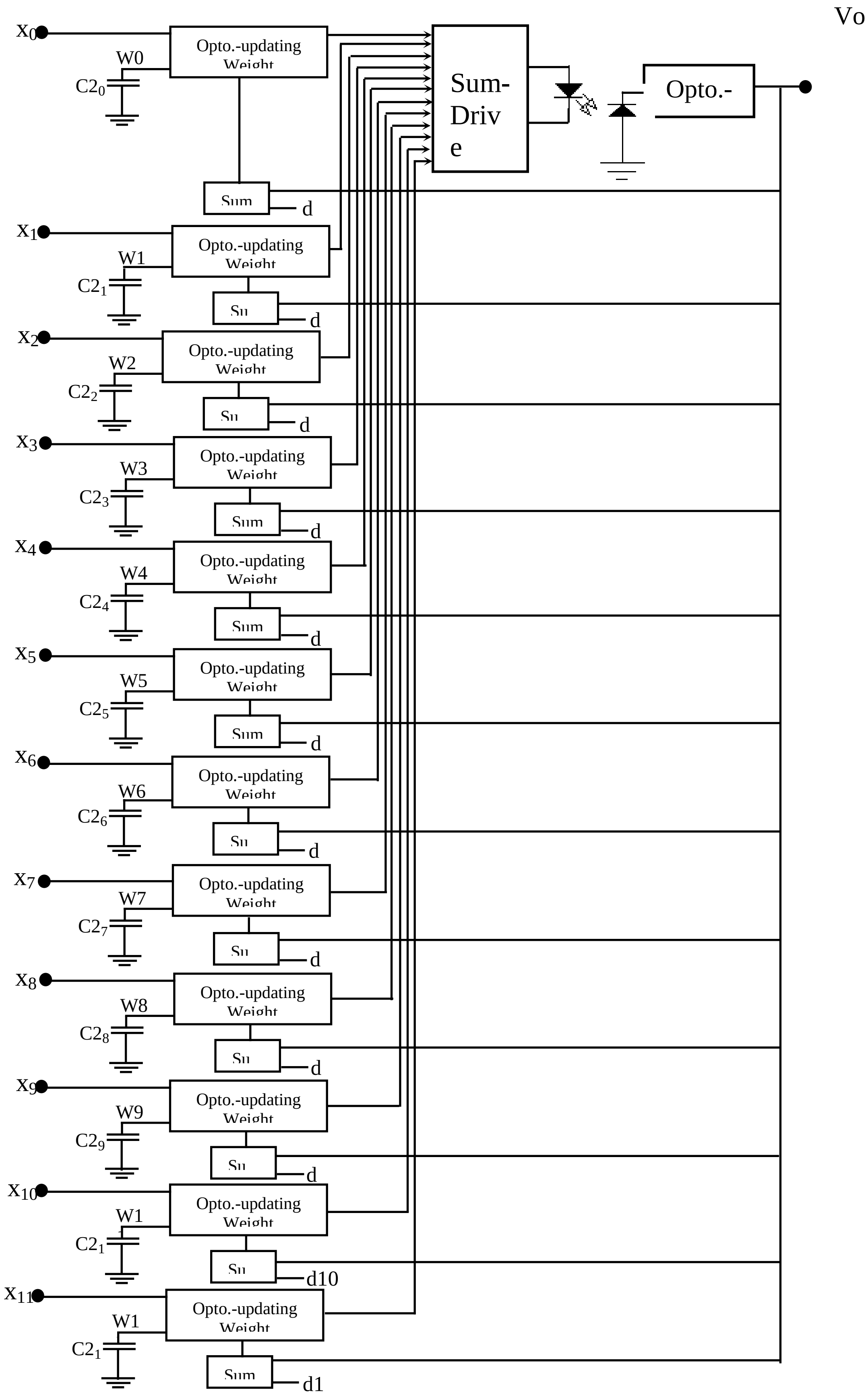


Fig. (11) Optoelectronic implementation of (3*4) input pattern

MULTICATEGORY SINGLE LAYER PERCEPTRON NETWORK

In this section an attempt to apply the error-correcting algorithm to the task of multicategory classification is made. The assumption needed is that classes are linearly pair wise separable, or that each class is linearly separable from other class. This assumption is equivalent to the fact there exist (M) linear discriminate functions such that:-

$$g_i(x) > g_j(x) \quad \text{for } i,j = 1, 2, \ldots, M, \; i \neq j \qquad \ldots (8)$$

where g(x) is the discriminate function and can be considered as an equation depending on the stored weight. Since the inputs here are not only (x), but is added to it a constant value called (bias) which always has the value (-1). This bias element makes the learning process faster so the discriminate function according to this input is given by:-

$$y = \begin{bmatrix} x \\ \\ -1 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ -1 \end{bmatrix} \begin{matrix} y_1 \\ y_2 \\ \\ \\ y_{n+1} \end{matrix}$$

$$g(x) = w_1x_1 + w_2x_2 + \ldots + w_nx_n - w_{n+1} \qquad \ldots (9)$$

$$w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \\ w_{n+1} \end{bmatrix}$$

In this paper the focus is (M) discrete perceptron. The network generated in this way is comprised of (M)-discrete perceptrons as shown in **Fig.(12).** The output will depend on the discriminate function g(x), so if g1(x) > gj(x) where j=1, 2, …, M, then (O_1=1) and (O_2, O_3, …, O_M) is equal to (-1). This theory exists when using a hard-limiter as an activation function shown in **Fig.(12).** It is called Threshold Logic Unit (TLU #M). This should indicate category (1) input. So the classifier using (M) individual (TLU) elements can be obtained. For this classifier the (k'th) (TLU) response of (1) is indicative of class (k) and all other (TLUs) respond with (-1).

The weight adjustment during the (k'th) step for this network is as follows:-

$$w_i^{k+1} = w_i^k + \frac{c^k}{2}(d_i^k - O_i^k)\, y^k \qquad \text{for } i=1, 2, \ldots, M \qquad \ldots (10)$$

where (di) and (Oi) are the desired and the actual response of the (i'th) discrete perceptron respectively. The desired response for the training pattern of the (i'th) category is:-

$$d_i = 1, \; d_j = -1 \qquad \text{for } j=1, 2, \ldots, M. \; i \neq j \qquad \ldots (11)$$

Example

Assume having three classes with random initial weights as shown in **table (3),** and the augment pattern are presented in the sequence (y_1, y_2, y_3, y_1, y_2, …) as shown in **Fig.(13a).**

Table (3) Input classes and initial weights.

Class 1	*Class 2*	*Class 3*	w_1^0	w_2^0	w_3^0
10	2	5	1	2	4
6	-4	-2	-1	-1	3
-1	-1	-1	0	2	0

Step 1: y_1 is input**: -**

$O_1=1, O_2=1^*, O_3=1^*$

Since the only incorrect response is provided by (TLU #2, 3), so for (c=1):

$$w_1^1 = w_1^0, \quad w_2^1 = \begin{bmatrix} 2 \\ -1 \\ 2 \end{bmatrix} - \begin{bmatrix} 10 \\ 6 \\ -1 \end{bmatrix} = \begin{bmatrix} -8 \\ -7 \\ 3 \end{bmatrix}$$

$$w_3^1 = \begin{bmatrix} 4 \\ 3 \\ 0 \end{bmatrix} - \begin{bmatrix} 10 \\ 6 \\ -1 \end{bmatrix} = \begin{bmatrix} -6 \\ -3 \\ 1 \end{bmatrix}$$

Step 2: y_2 is input**:-**

$O_1=1^*, O_2=1, O_3=-1$

The weight updates are:-

$$w_2^2 = w_2^1, \quad w_3^2 = w_3^1,$$

$$w_1^2 = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} - \begin{bmatrix} 2 \\ -4 \\ -1 \end{bmatrix} = \begin{bmatrix} -1 \\ 3 \\ 1 \end{bmatrix}$$

Step 3: y_3 is input**:-**

$O_1=-1, O_2=-1, O_3=-1^*$

The weight updates are

$w_2^3 = w_2^2 \quad w_1^3 = w_1^2$

$$w_3^3 = \begin{bmatrix} -6 \\ -3 \\ 1 \end{bmatrix} + \begin{bmatrix} 5 \\ -2 \\ -1 \end{bmatrix} = \begin{bmatrix} -1 \\ -5 \\ 0 \end{bmatrix}$$

This terminates the first learning cycle and the third step of weight adjustment. The only adjusted weights as from now will be those of the third perceptron. The outcome of the subsequent training is

$$w_3^{77} = \begin{bmatrix} 3 \\ -3 \\ 20 \end{bmatrix}$$

The three-perceptron network obtained as a result of the training is shown in **Fig.(13b).** It perform the following classification:

$O_1 = f(-x_1 + 3x_2 - 1)$

$O_2 = f(-8x_1 - 7x_2 - 3)$

$O_3 = f(3x_1 - 3x_2 - 20)$

The resulting decision surfaces are shown in **Fig.(14).** It can be seen that the three – perceptron classifier produces three decision surfaces (shown as lines here) and they are:

$-x_1 + 3x_2 - 1 = 0$

$-8x_1 - 7x_2 - 3 = 0$

$3x_1 - 3x_2 - 20 = 0$

The lines are shown in **Fig.(14)** along with their corresponding normal vectors directed toward their positive sides. It can be seen that there are several indecision regions thus patterns in shaded areas are not assigned any reasonable classification. One such patterns may be (Q). The corresponding linear discriminate functions are shown in **Fig.(15).**

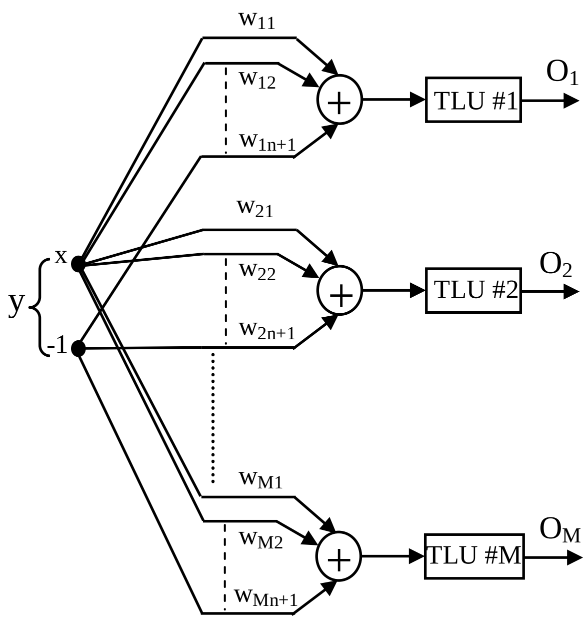
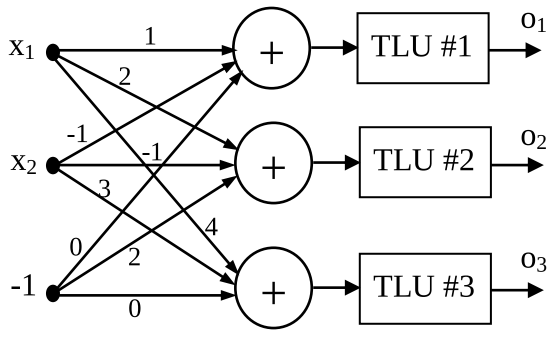
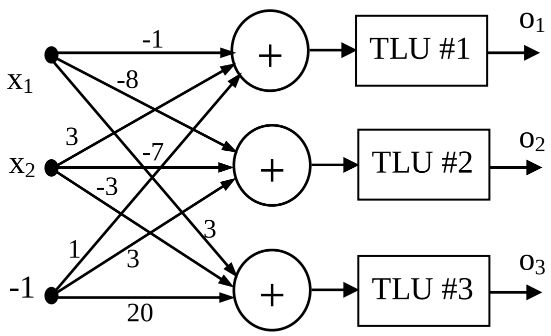


Fig.(12) M-category linear classifier using M-discrete perceptron



(a)



(b)

Fig. (13) Three-class classifier(a) three perceptron untrained classifier. (b) three perceptron trained classifier.

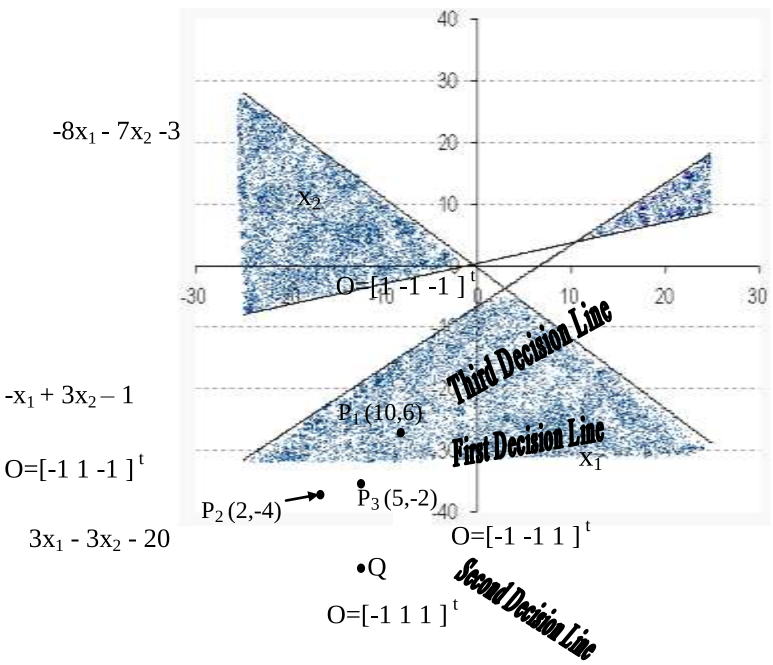


Fig.(14) Decision regions.

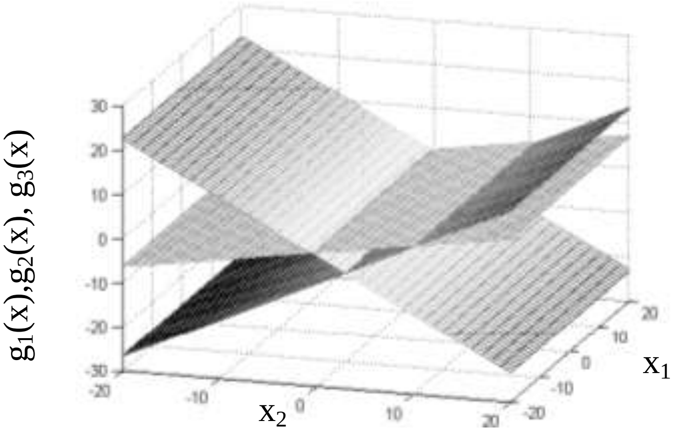


Fig. (15) Discriminate functions.

PROPOSED OPTOELECTRONIC CIRCUIT FOR MULTICATEGORY SINGLE LAYER PERCEPTRON

The sub-circuits needed for the designs of M-category single layer perceptron are similar to those used for the design of single neuron perceptron learning rule. The M-category depends on the same model of the single neuron perceptron except the learning constant in M-category perceptron is half that of the single layer perceptron model used. This can be achieved by multiplying the subtraction between the output and the desired by (0.5), by using the same operational-amplifier circuit.

The block diagram of the proposed circuit for (n) input and (M) category is shown in **Fig. (16).**

EXAMPLE 3

To examine the proposed circuit, three classes are applied to it as follows: the three classes are three characters (T, F, I), as shown in **Fig.(17),** are entered to the circuit. The input signals and the final training weights are tabulated in **table (4)**.

The results were obtained by using (MATLAB) programs for computer simulation as shown in **table (4).** These results were obtained when the desired input was of the form:

$$d = \begin{bmatrix} 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \end{bmatrix}$$

entered into the circuit.

The learning process stopped when the output signal was equal to the desired even though the input classes contained training because the training process is not affected by the updating of weight.

The problem in this classifier model is the value of the bias so that the patterns that are used to classify it must have the value of (-1) in its last term, but this problem can be solved by increasing the number of pixels used to divide the input pattern or adding an additional column so that it takes the value of (-1).

The entire components used in the design of the proposed circuits were designed using (Electronic Work Bench) program in computer simulation. As a test, three different classes were entered to the proposed circuit for their recognition them. The results are shown in **Fig. (18).**

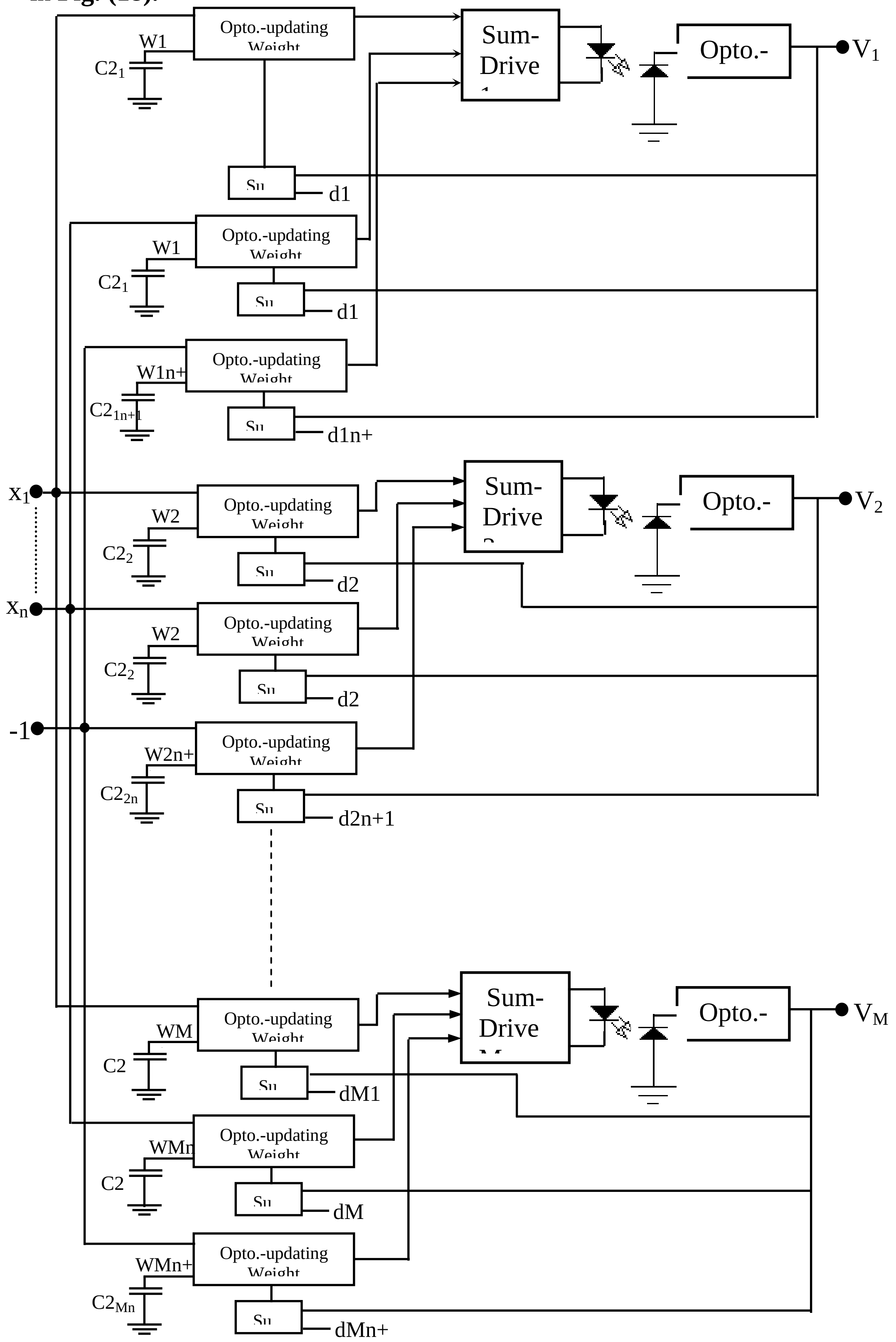


Fig. (16) M-category classifier for n-input signals block diagram

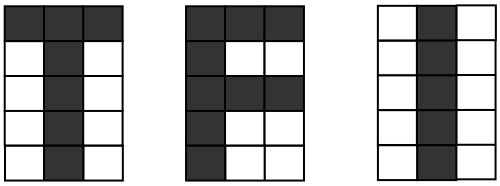


Fig. (17) Three input characters.

Table (4) Input signals, initial weight, and final training.

Class 1	Class 2	Class 3	w_1^0	w_2^0	w_3^0	w_1^5	w_2^5	w_3^5
1	1	-1	1	.5	1	4	-0.5	0
1	1	1	.5	1	-1	-0.5	0	-2
1	1	-1	0	-1	0	3	-2	-1
-1	1	-1	-1	0	.5	-2	1	-0.5
1	-1	1	-1	-.8	0	0	-1.8	1
-1	-1	-1	0	-1	.3	1		1.3
-1	1	-1	.4	.2	0	-0.6	1.2	-1
1	1	1	2	.6	-.2	1	-0.4	-1.2
-1	1	-1	.1	-.1	-.2	-0.9 -	0.9	-1.2
-1	1	-1	.4	.9	-.2	0.6	1.9	-1.2
1	-1	1	.2	.7	-.5	1.2	-0.3	0.5
-1	-1	-1	-.3	-.6	.7	0.7	0.4	1.7
-1	1	-1	1	-.1	.35	0	0.9	-0.65
1	-1	1	.5	-.25	-1	1.5	-1.25	0
-1	-1	-1	.35	1	.6	1.35	2	1.6

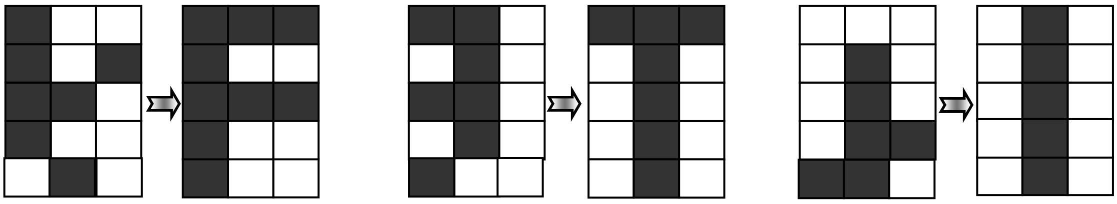


Fig. (18) Three different distorted classes and their correct class.

DISCUSSION

Neural networks and classical optical processing can be combined together to create optical neural networks, which offer great and fundamental advantages over electronic neural networks in various well- defined cases.

The fundamental advantage of optics is solving the massive interconnection between processors required in most neural network models. Since in electrical systems the electrical signals must travel on physical wire, then the massive interconnectivity is very difficult to be achieved. It requires a large area and careful design to minimize the interference between the physical wires and crosstalk. The perceptron learning rule is an example of supervised learning rule. Single neuron perceptron can be used to classify only two different classes, so the output may be taken as (1) for excitatory case and (-1) for inhibitory case. This is achieved by the LED driving circuit and it can be considered as hard limiter activation function, and the desired sums with output.

The optoelectronic representation of M-category single layer perceptron was realized. It depends on the single neuron perceptron model, it can be used as a classifier with its output depending on the desire that the output must reach it, such that the number of neurons is equal to the number of classes. In this design another element, which is very important in the classification process, is seen, it is the bias that speeds the learning process.

CONCLUSION

The availability of the desired speeds the learning process and lowers the number of iterations needed for reaching the final weight. This can be seen from the example taken on the proposed design of single neuron perceptron by using the same initial weight used for Hebbian learning rule so that the updating weight reaches the final weight after (3) iterations only. The designed M-category classifier can be used to classify the input pattern into one of a very large number of categories due to the large number of synapses capacity per neuron.

REFERENCES

B.M Wilamowski, J. Binfet, and M. O. Kaynak, **VLSI Implementations of Neural Networks**, www.com/ Goggle/ Neural Networks VLSI Implementation, 2000.

C. P. Chew, R. W. Newcomb, and J. D. Yuh, **VLSI Circuits For Optoelectronic Neural Network Weight Setting**, Microsystems Laboratory, Electrical Engineering Department, University of Maryland, IEEE, pp. 751-754, 1993.

J.C. Principe, D. Xu, Q. Zhao, and J. W. Fisher III, **Learning From Examples with Information Theoretic Criteria**, University of Florida Gainesville, www.com/ Goggle/ Neural Network Learning rules, 2000.

P. E. Keller and A. F. Gmitro, **Design of Fixed Planer Holographic Interconnects for Optical Neural Networks**, Applied Optics Vol. 31, pp. 5517-5526, 10 September 1992.

P. E. Keller and A. F. Gmitro, **Operational Parameters of An Optoelectronic Neural Net Employing Fixed Planer Holographic Interconnections**, the World Congress of Neural Network, July 1993 (WCNN 93).

P. Moerland, E. Fiesler, and I. Saxena, **Incorporating LCLV Non- Linearities in Optical Multilayer Neural Networks**, Applied Optics VoL. 35, No. 26, pp. 1-10, September 1996.

W. H. Al-Buhrezi, "**Computer Implementation of Optoelectronic Artificial Neural Networks**", M.Sc Thesis, University of Technology, 2001.

Z. Chahramani, A. T. Korenberg and G. E. Henton, **Scaling in A Hierarchical Unsupervised Network**, Ninth International Conference on Artificial Neural Networks-University College London, 1999.