Multi-Nary Content Addressable Memory Based on Artificial Neural Networks

Ahmad Abboud, Ali Kalakech, Seifedine Kadry, and Ibrahim Sayed Ahmad

***Abstract*—Using Simple Artificial Neural Networks, and away from strict Boolean logic, this paper propose a new design of memory array that has the ability to recognize erroneous and deformed data and specify the rate of error.**

To achieve this work, artificial neural network was exploited to be the actor responsible of representing the crude of the building. It's worth mentioning that simple neurons with binary step function and identity function were used, which will facilitate the way of implementation. The connection of few neurons in a simple network issues an exclusive X gate, which accepts only one value X (where X ∊ $\mathbb{R}^+$) with an acceptable error rate α. This gate will be the main core of designing a memory cell that can learn a value X and recognized this value when requested. After several stages of development, the final version of this memory cell will serve as a node unit of a large memory array which can recognize a data word or even a whole image with the ability to accept and recognize distorted data. Specific software that simulates the designed networks was developed in order to declare the efficiency of this memory. The obtained result will judge the Network.

***Index Terms*—Neural network, binary step function, identity function, content addressable memory (CAM).**

I. Introduction

The simplest is the unit, the more complex it can achieve. This is the key behind the dominance of digital systems worldwide. So, what about human brains, that great complex system which can store, recognize and solve complex problem. The construction unit of this system is the neuron. But how can this simple unit build like this complex system? And can we benefit from this concept in order to upgrade our artificial systems? Many studies have been conducted on artificial neural network. After the discovery of back propagation learning algorithm [1]-[3], neural networks resurge and become an important tool in the domain of artificial intelligence. Since then, the artificial neural network has been used in different areas of research, and among others, the field of image recognition [4]-[8]. Other effort was exploited on associative memories that were first developed by Kohonen and Anderson [9]. Auto-associative and Hetro-associative memory neural networks and used in order to store and classify patterns.

Other efforts were recruited on studying theoretical techniques of storage and retrieval of associative information [10]-[11]. Non monotone Neural Network (NNN) also, exploited to serve in learning sequential patterns and storing it to memory. All these researches play a directive role in the domain of recognizing patterns and storing it into memory. The interest when dealing with image object recognition is the tolerance of the classifier and the ability to recognize pattern even when adding or distort some of its pixels. A forward approach toward associative memories in Neural Network was introduced by Jinwen Ma [11], where the author has proposed a neural network, called simplex memory neural network (SMNN), which can learn and memorize (store) a single binary pattern with content-addressable memory function.

In this paper, a new Content Addressable Memory (CAM) network will be proposed as a tool in order to store and recognize any positive real number with an acceptable error rate. To achieve this memory, several stages were developed and used together to finally obtain a memory matrix which is able to recognize the distance between the tested pattern and the stored value.

II. Exclusive X

The simplest way to declare the role of Exclusive *X* net is to imagine a lock that has only one key *X* that can open this lock otherwise the lock will still block. Where big *X* is any input value to be tested. In order to achieve this network, we need two types of activation functions a binary step function with a threshold θ and an identity function. Fig. 1 will present this network.

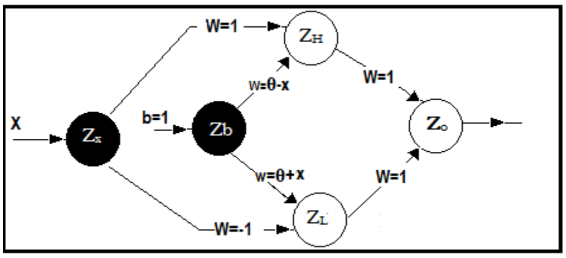


Fig. 1. Exclusive *X* gate

Z_H, Z_L, Z_O are neurons with activation function

$$f(Z_{in}) = \begin{cases} 0, Z_{in} < \theta \\ 1, Z_{in} \geq \theta \end{cases} \quad (\theta \in \mathbb{R}^+)$$

At Z_O the threshold is fixed to θ=2 in order to obtain a two input logical AND gate. Z_x and Z_b are dead neurons with an identity function $f(Z_{in}) = Z_{in}$. We note that, any input with no labeled weight, its weight will be equals to (1). In order to adjust this net to accept a value x (where $x \in \mathbb{R}^+$) you need just

Manuscript received December 1, 2012; revised January 31, 2013.

A. Abboud, A. Kalakech and I. Ahmad are with the Arts, Sciences and Technology University, Lebanon (e-mail: ahb.myemail@gmail.com, alikalakech@hotmail.com, ibrahimsayedahmad@gmail.com).

S. Kadry is with the Engineering and Technology School, American University of the Middle East, Kuwait (e-mail: skadry@gmail.com).

DOI: 10.7763/JACN.2013.V1.24

to set the weight between Z_b and Z_H to be $w=\theta-x$ and the weight between Z_b, Z_L to be $w=\theta+x$ (θ is the same threshold of Z_H, Z_L where the threshold of Z_O will be fixed to a value of 2). After adjusting the net to store a small x Z_H will serve as a high pass filter that accepts from Z_x only values more than or equal to x to be active. Z_L will serve as a low pass filter. This means that it will be active only for inputs at Z_x that are less than or equals to x. Both Z_H and Z_L will be active if and only if the tested value at Z_x is equal to x and this is the only case that will activate the output at Z_O. Adding an acceptable error value α to Z_H and Z_L will allow Z_O to be active at values between: $x - \alpha \leq X \leq x + \alpha$

Fig. 2 shows the connection of Z_α to the Exclusive X net.

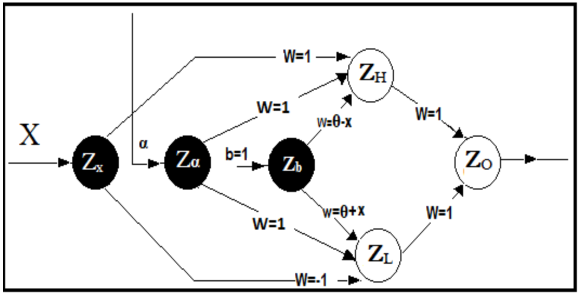


Fig. 2. Exclusive X gate with α

III. Exclusive X Memory Cell

The intention behind connecting the previous Exclusive X gate was to act as storage net. By the addition of control line to the network an Exclusive X memory net will be obtained. The new born storage network will possess the ability to store values (presented by small x) between Z_{RW} and Z_C. To control the behavior (Read, Write, and CAM) of this network, a control line presented by the input at Z_{Con} will help. Fig. 3 presents a memory cell with control line.

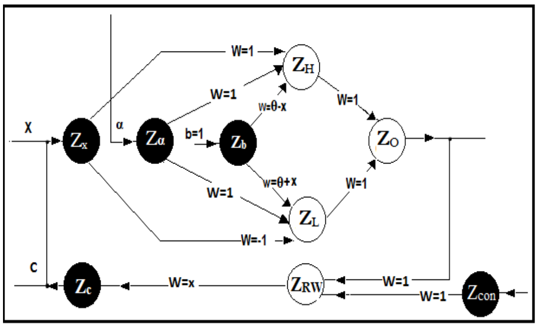


Fig. 3. CAM cell with α

The activation function and the threshold of Z_{RW} is the same as that of Z_O since it will behave as a two input AND gate.

The activation function of Z_C and Z_{Con} is the same identity function as that of Z_x, Z_b and Z_α.

In order to adjust the net to recognize a value x, we simply set the weights of the Exclusive X net as mentioned in the previous section, and set the weight between Z_{RW} and Z_C to x.

To read the content of the stored value of the memory just enter the value (2) to the control line at Z_{Con} and the output at Z_C will present the content.

To oblige the net to behave as a content addressable memory, apply to the control line at Z_{Con} the value (1) and input the value to be recognized at the input of Z_x, also you can input an α acceptable error value which force the network to accept the input value as an error. The output of the network will be found at Z_O and Z_C. While testing the CAM cell to recognize a doubted value X, a value of (1) at Z_O indicates that the tested pattern is accepted and the exact stored value (small x) will be found at Z_C, else if Z_O output a value of (0) this indicates that the tested pattern X is far from the stored value x by a value more than α.

IV. Exclusive X with Absolute Distance B

Instead of the ability of the Exclusive memory cell to recognize pattern that lies between ($x + \alpha \leq X \leq x - \alpha$), there is a necessity to know how far is the tested pattern from the stored one.

For this purpose, a simple network called Absolute Distance Network (ADN) is used to calculate the absolute distance between two patterns. Fig. 4 will present this ADN.

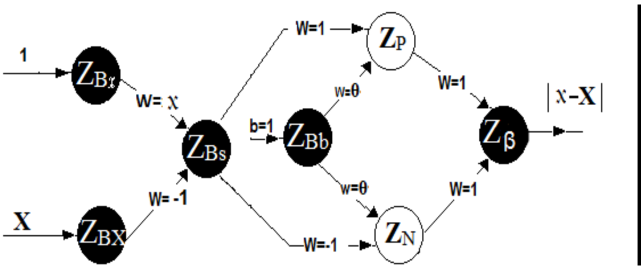


Fig. 4. Absolute distance network

Z_P and Z_N are neurons with activation function

$$f(Z_{in}) = \begin{cases} 0, Z_{in} < \theta \\ Z_{in}, Z_{in} \geq \theta \end{cases}$$

Z_{BX}, Z_{Bx}, Z_{BS}, Z_{Bb} and Z_β are dead neurons with identity function $f(Z_{in}) = Z_{in}$. The weights in this network are fixed to (1), except that between Z_{Bb}, Z_P and Z_N where $W=\theta$ (θ is the threshold at Z_P and $Z_{N)}$. If two patterns x and X are applied to the inputs at Z_{BX} and Z_{Bx} respectively, the output at Z_β will be the absolute distance between them ($\beta=|x-X|$).

The ADN will be connected to the Exclusive X memory cell in order to find the absolute distance between the tested pattern and the stored one. This distance will be called β value. Fig. 5 shows the connected ADN to the memory cell.

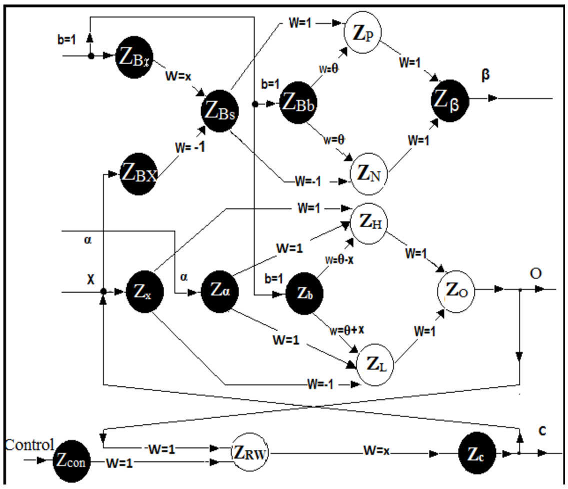


Fig. 5. CAM cell with β

While adjusting the memory cell to store a pattern *x*, the weight between Z_{Bx} and Z_{BS} must be set to be *w*=*x*. Other weights in the ADN are supposed to be fixed to the values shown in the Fig. 5.

V. Exclusive X Memory Cell with Congruency Rate $\overline{\beta}$

The addition of ADN network allows detecting the value of distance between, the stored pattern and the tested one. The new upgrade step introduces the rate of congruence $\overline{\beta}$.

$\overline{\beta}$ will presents a value that reflect the rate of similarity between the stored *x* and the tested pattern *X*. The new $\overline{\beta}$ will help detecting the nearest memory cell while searching among memory arrays. The equation of $\overline{\beta}$ will be as following:

$$\overline{\beta} = 1 - \frac{|X - x|}{X + x} = 1 - \frac{\beta}{X + x} \qquad (1)$$

for $\beta > X+x$ $\overline{\beta}<0$

Fig. 6 shows the connection of $\overline{\beta}$ to the Exclusive *X* memory.

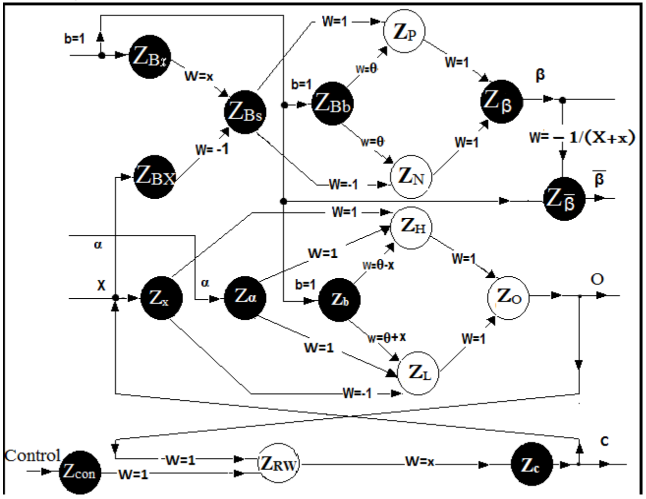


Fig. 6. CAM Cell with $\overline{\beta}$

In order to adjust the net to detect $\overline{\beta}$, two steps are required.

Step 1: adjust the net to store *x* pattern.

Step 2: the tested pattern *X* and the stored value *x* will be the variants to use in order to set the weight between $Z_{\overline{\beta}}$ and Z_β, where W=-1/(X+x).

After a complete adjusting, this memory cell will be able to store any positive real number, recognize this number with anα acceptance value, detect the distance between the tested pattern and the stored one (β) and the rate of congruence between them ($\overline{\beta}$).

VI. Complete Exclusive X Content Addressable Memory Array

The consecutive upgrading steps of the content addressable memory cell leads to the connection of complete memory array. This array possesses the ability to recognize and hold a complete word or even a matrix of colored image pixels.

In order to not complicate the view of the network, a box shape illustration shown in Fig. 7 will represent the Exclusive *X* memory cell.

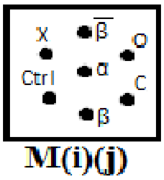


Fig. 7. CAM symbol

The unconnected lines of the cell will not be drawn (in later networks). Using the representation of memory cell in Fig. 7, an over view of the array will be presented in Fig. 8. Where Z_s at the input of each αis a dead neuron with identity function as an activation function, i and j are any positive natural numbers. This connection array represents 9 neighbors (3X3) cells sharing the same α, the most top inputs are the X input pattern. At the left of the array, each memory cell has its own control line, to the right of the array $\overline{\beta}$ for each cell and at the bottom the output *C* of each cell is represented.

In order to read data from this memory array using the memory as a CAM, an algorithm called **Blind Calling** will take place:

Step 1: Initialize α error value that can be accepted and set all control lines to the value of 1.

Step 2: Broadcast the doubted pattern to each of the X Inputs, thus β can be obtained.

Step 3: The cell with the highest positive $\overline{\beta}$ will be elected as the true blind and the content at its output C line will be considered as the true stored pattern.

If there is more than one cell having the same $\overline{\beta}$ then the best cell will be randomly elected.

To store any pattern into the array:

Step1: Select the cell to store the pattern into and initialize its control line to 0.

Step 2: Set the weights of the selected cell according to Fig. 6 where *x* is the pattern to be stored, *X* =0.1 and *θ* is the threshold of none dead neurons.

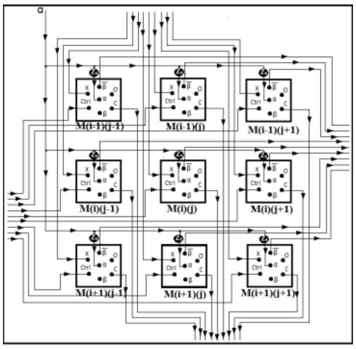


Fig. 8. CAM Array

To read the stored pattern from any cell, just assign value (2) to the control line of the intended cell and the output will be obtained at its C line.

VII. Neighbors Sharing Connection

In the headway of immunity to noise and decreasing the access time to memory cells, two types of sharing were implemented. In most cases, the content of neighbor cells of a memory array is reliant. For example, colored pixels that form an image are stored in a way where neighbors in most cases have approximately analogous colors. In other cases, it is recommended to store analogous data closely to each other, which will decrease the access time to read, classify and search this data. For this purpose, each memory cell must have an impact on the decision of who will be its neighbor.

The first type of sharing serves as memory arrays that can hold values where data order is not significant. This sharing requires the addition of part of $\overline{\beta}$ of each cell to $\overline{\beta}$ of all its neighbor cells. This will results in, attracting analogous data to the neighbors of the hosting cell after applying a specific storing algorithm which will be presented in the rest of this paper. Fig. 9 will declare the case. At the most right of the array, $Z\overline{\beta}$ does no long exist. It will be one of the inputs of the dead neuron $Z\overline{\beta}_C$ which represent the addition of all outputs of Z $\overline{\beta}$ from neighbors.

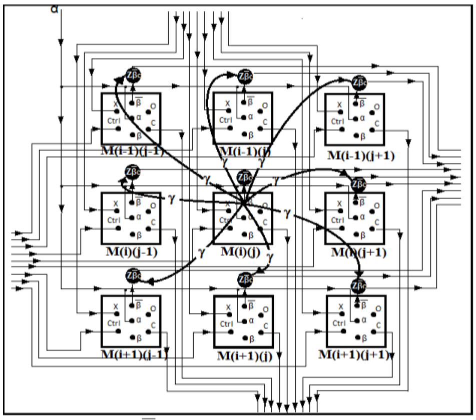


Fig. 9. Fisrt order $\overline{\beta}$ sharing

Fig. 9 presents a sharing of only one cell *M*(*i*)(*j*) to its first order neighbors with a weight=γ(0 <γ< 1).

First order neighbor are cells that are in touch with it, and second order neighbors are separated from *M*(*i*)(*j*) by one cell. *M*(*i*)(*j*) is any cell of position (*i*,*j*) in the array, where *i* and *j* are two positive integers.

Reading data from this memory array is the same as that before sharing $\overline{\beta}$**,** but in order to write data to this array an algorithm similar to Blind Calling algorithm will be used. Applying this algorithm will result in attracting analogous data to each other.

The following steps represent this algorithm.

Step 1: Initialize α to a given value according to the error value that can be accepted and set all control lines to the value of 1.

Step 2: Broadcast the pattern to be stored to each of the *X* inputs, thus $Z\overline{\beta}_C$ can be obtained.

Step 3: Set all control lines to 2 in order to recognize the empty cells.

Step 4: The empty cell with the highest positive $Z\overline{\beta}_C$ will be elected as the worthier cell that deserves to store the pattern *X*.

Step 5: Select the cell to store the pattern into and initialize the control line to 0.

Step 6: Set the weights of the selected cell according to Fig. 5 where *x* is the pattern to be stored, *X* =0.1 and θ is the threshold of none dead neurons.

It is obvious that the elected cell was previously commended by its neighbors where this recommendation is the result of the acceptance of the neighbors to the new pattern to be stored. The second type of neighbors sharing is more recommended to be used in memory array that stores images, where each data unit is related to its neighbors with specific order and position. Each memory cell shares a rate of $\overline{\beta}$ with neighbors. The emitted $\overline{\beta}$ will be multiplied by a weight equal toγ, then added to the acceptance error value α for specific order neighbors using a summation dead neuron Z_S. Fig. 10 shows the sharing of $\overline{\beta}$ with first order neighbors.

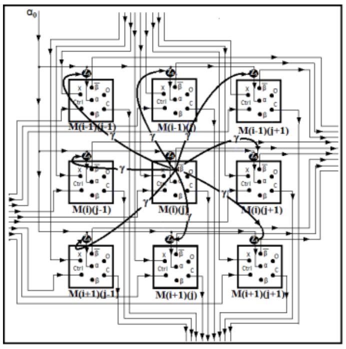


Fig. 10. First order Alfa sharing

The new *α* at each cell of first order sharing will be calculated according to the following equation:

$$\alpha = \alpha_0 + \sum \overline{\beta} x \qquad (2)$$

Knowing that $\overline{\beta}$ is the output of each neighbor, α_0 is the initial acceptance error value andγ is the weight between intended cell and its neighbors. Reading and writing to this memory is the same as that of Fig. 7 (Complete Exclusive *X* Content Addressable Memory Array). The advantage gained from this connection is the ability of this memory to increase the acceptance error value in a way which makes it able to recognize erroneous, noisy and deformed images. The result in the next section will prove the efficiency of this kind of sharing to accept erroneous and deformed images.

VIII. Experimental Results

In order to test the efficiency of this CAM, special software was developed for this purpose. Because of the variety of upgrade stages of this memory, the last memory array with $\overline{\beta}$ to α sharing will be the test agent. An array of dimension 200x250 memory cells was built to serve the aimed tests. Each cell is connected to its neighbors using second type of sharing, where $\overline{\beta}$ of each cell is connected to Z_S of neighbor cells. The weights at sharing lines are settled according to the distance between the donor cell and the receiving one. Supposing that MN represents the maximum sharing order and N presents the order number which is positive integer. The weight W_γ will be set as following:

$$W_\gamma = \frac{\gamma}{N}$$

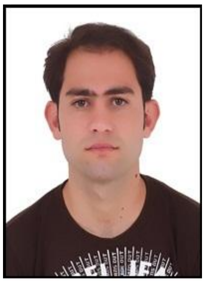
Let η_i be the neighbor number at each sharing level. For example, at N=1 each cell must have 8 neighbor cells starting from η_i=1 till 8, at *N*=2 each cell has 16 neighbor cells, thus for *MN*=2 the number of neighbors *TN* will be equal to N_1+N_2=24.

$$TN = 8\sum_{n_i=1}^{MN} n_i = 4MN(MN+1) \qquad (3)$$

The total acceptance value α will be determined according to the following equation.

$$\alpha = \alpha_0 + \sum_{n_i=1}^{MN} n_i \times \frac{\gamma}{N} \qquad (4)$$

Taking into consideration the memory cells at the edges of the array, many tests had been done with variant values of *α*, *γ* and *MN*. For example, Fig. 11 represents an image that was stored in the array for the first time where each pixel of the image was stored by mapping into its facing cell. The image in Fig. 12 is the same as the stored one but with addition of 50% noise.



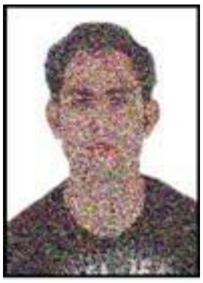


Fig. 11. Stored image Fig. 12. Image with noise

Test one is done as following: image in Fig. 11 was stored in the memory array, where each pixel color is stored as one number *x*. This value represents the average of adding red, green and blue colors. The image with noise of Fig. 12 presents the data to be tested. The following graph of Fig. 13 is the result of the first test where no neighbor sharing was applied.

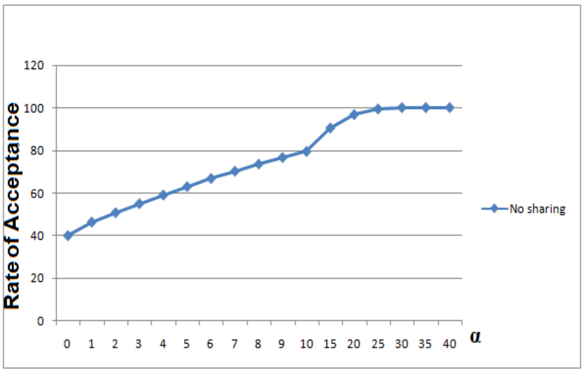


Fig. 13. Effect of *α* to rate of acceptance

Referring to the chart of Fig. 13, the increase of *α* will result in definitely increase in the rate of acceptance. The rate of acceptance shown in the graph is the average value taken from Z_O of each cell expressed in percent. After applying different sharing order and at a range of *γ*, the following chart shown in Fig. 14 presents the results.

The S-shape lines for each sharing order shows the rapid increase of acceptance rate to *γ*, which is obvious in (3).

The increase in the order of sharing also results in rapid increase of the acceptance ratio. Referring to the graph of Fig. 14 all lines starts from an acceptance ratio equal to 46.5% which is the same ratio of Fig. 13 at *α*=1. This is done because the tests of Fig. 14 were realized with constant *α*=1.

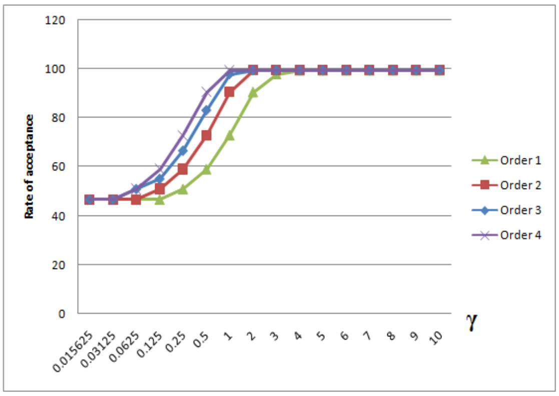


Fig. 14. Rate of acceptance for multiple sharing order

Thus, at *γ*<0.125, the effect of the sharing order can be ignored, and can be applied for *γ*>2. This observation leads to defining the range of the initial sharing weight to be 0.125<*γ*<2. This optimization is affected by the value and type of data being stored in the CAM array. Another test was done on the ability of this memory to accept deformed image, so same parameters were tuned in order to best declare differences. The first step presents the effect of α on the acceptance ratio. The same image of Fig. 11 was stored into the array of 200x250 cells where no sharing takes place.

The deformed image to be tested is shown in Fig. 15.

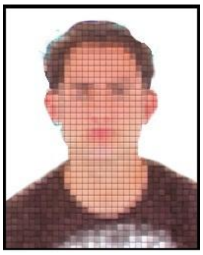


Fig. 15. Deformed image

The graph of Fig. 16 declares the rate of deformation at *α*=0 and will show the effect of increasing of *α* on the acceptance ratio.

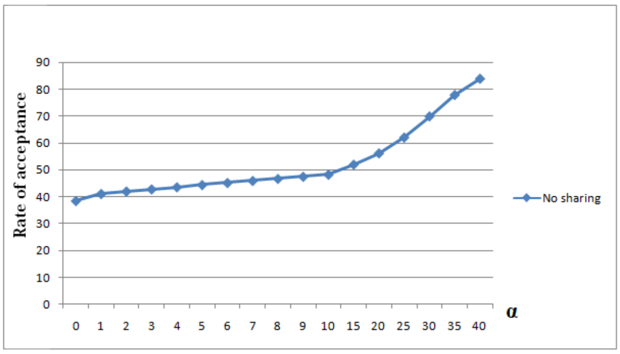


Fig. 16. Effect of *α* to rate of acceptance

The graph of Fig. 16 shows that the acceptance ratio at *α*=0 reaches 38.5% which means that the rate of deformation is approximately 61.5% .This kind of deformation is called patch deformation where each patch of the image was deformed apart. Referring to the graph, even at high α of 40, the acceptance rate didn't reach the 90%. This may give better understood of the effect of this type of deformation.

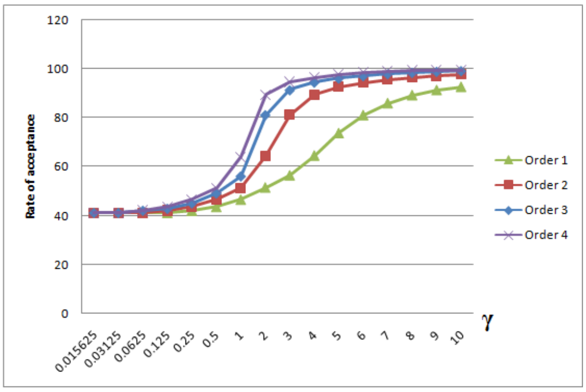


Fig. 17. Rate of acceptance for multiple sharing order

The results of Fig. 17 shows that deformation can be accepted by a high rates for sharing order more than 1 and initial γ of more than 0.5. Comparing the effect of γ on noise and deformation shows that the effective range of γ applied to deformed content is wider than when applied to noise effect. In the same way increase in order has more effect on deformed content than that on noise.

IX. Conclusion

In this paper, a content addressable memory based on artificial neural network was designed and tested to be able to accept erroneous ad deformed data and to be able to indicate the distance between the stored data and the data to be recognized. The results shown in the previous section proves the efficiency of this memory where at sharing order more than 4 and initial sharing weight greater than 4, acceptance rate can exceed 98%. This result can be considered cogent to judge the tested data.

A variety of tests were done on many deformed data and in major cases the results are convincing. Other papers will be issued later exhibiting the efficiency of this memory to recognize sound and illustrations, where the alignment and size of the shape are not important.

References

[1] D. B. Parker, "Optimal algorithms for adaptive networks: second order back propagation, second order direct propagation," in *Proc. IEEE International Conf. on Neural Networks*, vol. 2, pp. 593-600, 1987.

[2] B. Verma, "Recognition of rotating images using an automatic feature extraction technique and neural networks," *International Journal of Neural Systems*, vol. 8, no. 2, pp. 201-207, 1997.

[3] M. I. Quraishi, J. P. Choudhury, and M. De, "Image recognition and processing using artificial neural network," in *Proc. 1st Int'l Conf. on Recent Advances in Information Technology, IEEE*, pp. 15-17, March 2012.

[4] M. E. Petersena, D. de Ridderb, H. Handelsc, L. Beaurepaire, K.Chehdi, and B.Vozel "Image processing with neural networks—a review," *Pattern Recognition* , pp. 2280-2288, 2002.

[5] B. J van der Zwaag, C. H. Slump, and L. Spaanenburg, "Extracting knowledge from supervised neural networks in image processing," *Innovations in Knowledge Engineering, Advanced Knowledge International,* Adelaide, Australia, pp. 107-127, 2003.

[6] T. Kohonen, *Content-addressable memories*, 2nd ed. Berlin: Springer-Verlag, 1987, ch. 3.

[7] D. L. Wang and X. Liu, "Scene analysis by integrating primitive segmentation and associative memory," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 32, no. 3, pp. 254-268, 2002.

[8] J. MA, "A hybrid neural network of addressable and content-addressable memory," *International Journal of Neural Systems*, vol. 13, pp. 205-213, 2003.

[9] B. H. Wang, S. B. Koh, and S. K. Ahn, "A new neural network: Hybrid location-content addressablememory," *Neurocomputing*, vol. 11, no. 1, pp. 55-74, 1996.

[10] L. Tarassenko, J. N. Tombs, and J. H. Reynolds, "Neural network architectures for content-addressable memory," *IEE PROCEEDINGS-F*, vol. 138, no. 1, 1991.

[11] M. Morita, "Memory and learning of sequential pattern by nonmonotone neural networks," *Neural Networks journal*, vol. 9, pp. 1477-1489, 1996.

Ali Kalakech is an associate professor at the Information Systems Department in the Lebanese University, Faculty of Economics and Business Administration. He got his Master Degree in Computer Systems from the National Institute of Applied Sciences, Toulouse, France in 2001. He received the Doctor degree from the National Polytechnic Institute, Toulouse, France in 2005. His Research interests include testing and evaluating the dependability and the performance of computer systems.

Seifedine Kadry is an associate professor at the American university of the Middle East, Faculty of Engineering. He got his Master Degree in Computer Science and Applied Math from AUF-EPFL-Inria, Lebanon in 2002. He received the Doctor degree from the Clermont Ferrand II University, France in 2007. His Research interests include software testing and security.