Application of Associative Matrices to Recognize DNA Sequences in Bioinformatics

Jorge L. Ortiz
Department of Electrical and Computer Engineering
College of Engineering
University of Puerto Rico-Mayagüez, P.R. 00683
jortiz.ece.uprm.edu

February 19, 2008

1. Introduction.

Associative matrices are considered a type of neural network topology used to recall and recognize previously known or unknown patterns. For example, the Hebbian linear associative matrices can be trained to recognize a particular DNA sequence into another specimen sequence that may help biologist to identify similarities and other characteristics important in the knowledge and recognition of a particular sequence in another specimen. This approach may result especially beneficial in mutated sequences where mutations or other changes in the sequence as deletions and insertions are present. Associative matrices have been used to recognize characters, shapes, or specific objects from an image. Such changes are considered noisy patterns that are one of the important features of using associative matrices in this field.

2. Project Objectives.

The project introduces students to the use of associative matrices concepts in learning and pattern recognition. Students will experiment the use of the matrices to recognize DNA sequences and its possible mutations
The objectives of the project is that the students understand the use of associate matrices in learning and recognizing patterns, objects, and strings such as DNA sequences in bioinformatics.
After the completion of this project the student should be able to:

- Understand the use and implementation of associative matrices.
- Recognize the capability of these type of networks to associate new matrices with previously learned patterns allowing to recognize patterns with noise or different from the original patterns.
- Understand the use of associative networks and pattern recognition.
- Understand the mathematical foundations of matrix or linear algebra and its applications.

- Introduce the students to the applications of artificial intelligence, learning, and neural networks to the bioinformatics field.

3.Suppervised Hebbian Learning.

The Hebb rule was proposed by Donald Hebb in 1949 as a possible mechanism for synaptic activity in the brain and become one of the first neural network learning laws. This rule has been used extensively in several applications specially in pattern recognition.

This famous idea was included in the publication of "The Organization of Behaviors" and it was the postulate that became the Hebbian Learning Rule:

"When an axon of a cell A near enough to excite a cell B and repeatedly or persistely takes part of firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B is increased.

Hebb's learning has been used in a variety of neural network architectures to implement what is called in general a linear associator. These linear associator are also called associative memories. Associative memories learn to recognize Q pairs of prototypes input/output pair vectors. Thus, that given an input vector P_i the associative memory recognizes an output vector t_i. Generally, the associative memories are trained with a set of known input/output pairs:

$$\{p_1, t_1\}, \{p_2, t_2\}, \ldots \{p_Q, t_Q\} \qquad \text{Eq. 1}$$

Figure 1. Associative Matrix Model

The network operation is to receive an input P_i and in the recognition process identify the input pattern with its corresponding output t_i. In some cases where the input pattern is distorted by noise then most of the cases the output is recognized correctly or only small changes are noticed.

The trained linear associator using the supervised Hebb rule consists of a weight matrix constructed using the Q input/output pairs as in the equation below:

$$W = t_1 p_1^T + t_2 p_2^T + t_3 p_3^T + \cdots t_Q p_Q^T \qquad \text{Eq. 2}$$

$$W = \sum_{q=1}^{Q} t_q p_q^T \qquad \text{Eq. 3}$$

The simplified weight matrix used in Equation 3, can be expressed in vector form as:

$$W = [t_1 t_2 t_3 \cdots\cdots t_q] \begin{pmatrix} p_1^T \\ p_2^T \\ p_3^T \\ \cdot \\ \cdot \\ \cdot \\ p_Q^T \end{pmatrix} \qquad \text{Eq. 4}$$

Or in matrix form as:

$$\mathbf{W = TP^T} \qquad \text{Eq. 5}$$

Using this approach any input vector could be associated with its corresponding output vector at least to the **closest one**.

The Hebb rule is specially efficient when the input vectors are orthogonal:

$$P_q^T p_k = 1 \qquad \text{for } q = k$$

$$= 0 \qquad \text{for } q \neq k \qquad \text{Eq. 6}$$

In these cases the recalling operation of the W matrix is perfect. Unfortunately, in many cases this condition is not achieved but the results can be quite impressive anyway.

The objective is to select a weight matrix that minimizes the error produced by the absence of orthogonality. Lets define a performance index:

$$\mathbf{F(W)} = \sum_{q=1}^{Q} \left\| t_q - p_q \right\|^2 \qquad \text{Eq. 7}$$

When P_q vectors are orthogonal then:

$$\mathbf{F(W)} = 0$$

Defining the associative memory matrix in the form:

$$\mathbf{WP = T} \qquad \text{Eq. 8}$$

Where,

$$\mathbf{P} = [\, p_1 p_2 p_3 \ldots\ldots p_q \,] \qquad \text{Eq. 9}$$

and

$$\mathbf{T} = [t_1 t_2 t_3 \ldots\ldots t_q] \qquad \text{Eq. 10}$$

The performance function to minimize the error function $\mathbf{F(W)}$ is:

$$\mathbf{F(W)} = \left\| \mathbf{T - WP} \right\|^2 = \left\| \mathbf{E} \right\|^2 \qquad \text{Eq. 11}$$

Where E is the error function defined in matrix form by:

$$\mathbf{E = T - WP} \qquad \text{Eq. 12}$$

Since the objective of the error function E is to make it equal to zero,

$$\mathbf{E = 0}$$

From Eq. 12:

$$\mathbf{E = 0 = T - WP} \qquad \text{Eq. 13}$$

Assuming P matrix has an inverse, then:

$$\mathbf{W = TP^{-1}} \qquad \text{Eq. 14}$$

This new W matrix is designed to minimize errors when non orthogonal input vectors are used.

Sometimes, the problem getting the inverse of matrix P is that a matrix with no orthogonal vectors has no inverse and the inverse of a matrix is only defined for square matrices. To solve this problem the Moore_Penrose Pseudoinverse definition is used and discussed in the next section.

4. Moore_Penrose Pseudoinverse.

The Moore Penrose Pseudoinverse method is used to obtain the inverse of a singular and/or non square matrix.

Lets call the pseudoinverse matrix of $\mathbf{P}$ as $\mathbf{P^+}$, where:

$$\mathbf{W = TP^+} \qquad \text{Eq. 15}$$

Then $\mathbf{P^+}$ is obtained by:

$$\mathbf{P^+ = [P^T P]^{-1}} \qquad \text{Eq. 16}$$

Therefore for this application of pattern recognition $\mathbf{P}^{+}$ will be used to warrantee a feasible solution for the matrix inversion problem.

5. Bioinformatics

Bioinformatics is a new science that is about searching biological databases to look at protein structures using a computer. This task was done by biologist using long protein lists written on paper before computers became available. Computers had help to develop an extraordinary important field that combines biology sciences with computer engineering and computer science. Protein sequences databases in DNA (deoxyribonucleic acid) sequences are important to be compared. Bioinformatics is the combination of biotechnology and computation technology with the objective of revealing new insights and principles in biology.

Analyzing protein sequences is one of the most important tasks used in bioinformatics. DNA is a large macromolecule consisting of a chain of four constituents that are called nucleotides. A DNA sequence is what make the genes of any animal or plant different.

A nucleotide is made up of 4 types of nitrogenous organic bases symbolized by four letters A, C, G, and T. It is very important for molecular biologist to identify or find DNA sequences in a large amount of character data to analyze similarities between organism and also, to find out possible mutations. Mutations are cases when one of the nucleotides is changed or may be absent. In bioinformatics pattern matching or recognition is concerned with the automatic classification of character sequences representative of nucleotide basis or molecular structures.

This project is intended to implement a sequence alignment or recognition method using associative matrices. Sequence alignment in bioinformatics is fundamental to infer homology or common ancestry. If two or more sequences are aligned partially or completely to all of the pattern nucleotides then they are similar and may be homologous.

The program will compare a query sequence to all other sequences specified in the database. Comparisons are made in a pairwise method and a score may be assigned reflecting the degree of similarity for each sequence. For the purpose of this project the alignment of the tests sequence will be the same and the best score or percent of pixel similarity will represent the best match. Similarity is the extend to which nucleotide or protein sequences are related.

Another heuristic used by biologists, is that if a sequence matches significantly the sequence of another known structure, then the molecules may share the same structure and function.

The pairwise sequence alignment involves matching of two sequences, one pair of the elements at the time The challenge consist of finding the best alignment of two sequences. A scoring system is designed that reflects the number of paired characters in two

sequences and the number and length of gaps required to adjust the sequence to the maximum number of matches.

To make this project simpler we will identify a sequence of interest to identify among several other unknown sequences. A score system is based on the correct dot matrices obtained after processing the sequence using the associative matrices. This is a simplified version of the real problem but provides an introduction to the students interested in machine learning and bioinformatics. We can assume that the sequence that wants to be identified is:

ATTCG

And other unknown sequences will be used to identify their similarity with the original sequence, for example:

ATCCG ; mutated sequence at the third nucleotide or character
AT-CG ;where "-" means a sequence gab.
ACCGC
A-C-G
ATTTC
, and

ATGCG

The student will utilize the initial sequence and the set of test sequences to search for similarities with the associative matrix algorithm.

5.1 Deliverable 1

Using the four patterns defined before write a computer program using C++ or Matlab to process the prototype inputs and obtain their corresponding association with the output patterns after processed with the associative weight matrix. Figure 2 shows several cases where the described patterns for letters A,C,G, and T are used. Also, distorted patters are inputted to the matrix to watch the results of the association using Hebb rule. Test your prototype patterns with distorted inputs and look at the outputs. How distorted can the input be? Test the patterns individually.

Example 1:

Lets define four input/output vectors using 7 rows and 5 columns patterns for the letters A, C,G, and T.

Lets make the input and output and input the same pattern p_i and t_i, the same in such a matter that when p_1 is the input to the weight matrix $\mathbf{W}$ weight matrix the output should be the same $p_1 = t_1$.

To represent the input and output vectors, the four patterns are converted to column vectors $\mathbf{P}_{mX1}$ and $\mathbf{t}_{mX1}$, where m is the number of rows and in this example is equal to 7x5 = 35 rows.

Assign a numerical “1” to the dark pixels (picture points) and a “0” to the pale pixels, then a column vector of dimension 35 X 1 is obtained.

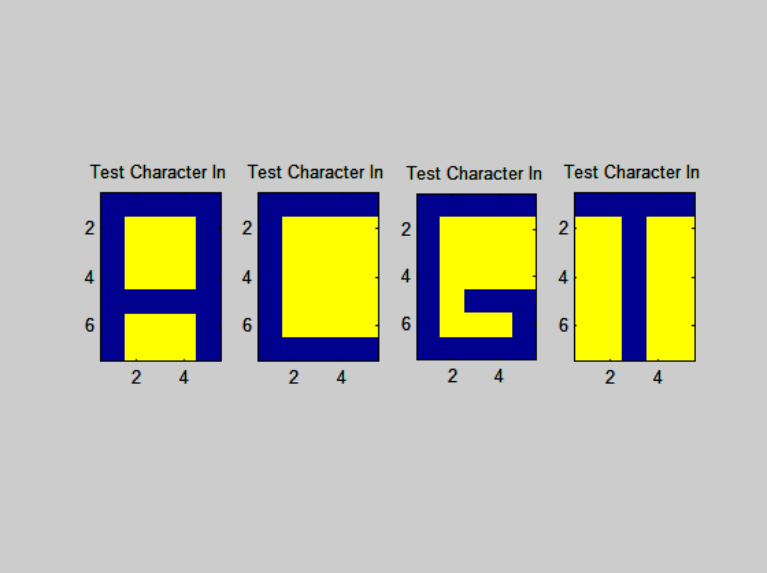

Figure 2. Four nucleotides models for A, C, G and T.

The four transposed vectors are the following:

```
A=[1 1 1 1 1 1 1  1 0 0 0 1 0 0  1 0 0 0 1 0 0  1 0 0 0 1 0 0  1 1 1 1 1 1 1]'
C= [1 1 1 1 1 1 1  1 0 0 0 0 0 1 1 0 0 0 0 0 1  1 0 0 0 0 0 1  1 0 0 0 0 0 1]'
G=[1 1 1 1 1 1 1  1 0 0 0 0 0 1  1 0 0 0 1 0 1  1 0 0 0 1 0 1  1 0 0 0 1 1 1]'
T=[1 0 0 0 0 0 0  1 0 0 0 0 0 0  1 1 1 1 1 1 1  1 0 0 0 0 0 0  1 0 0 0 0 0 0]'
```

The objective is to select a weight matrix that minimizes the error produced by the absence of orthogonality. Lets define a performance index **F(W)** using Equations 15 and 16:

Using the initial characters in Figure 2. The first test characters were inputted as shown in Figure 3.

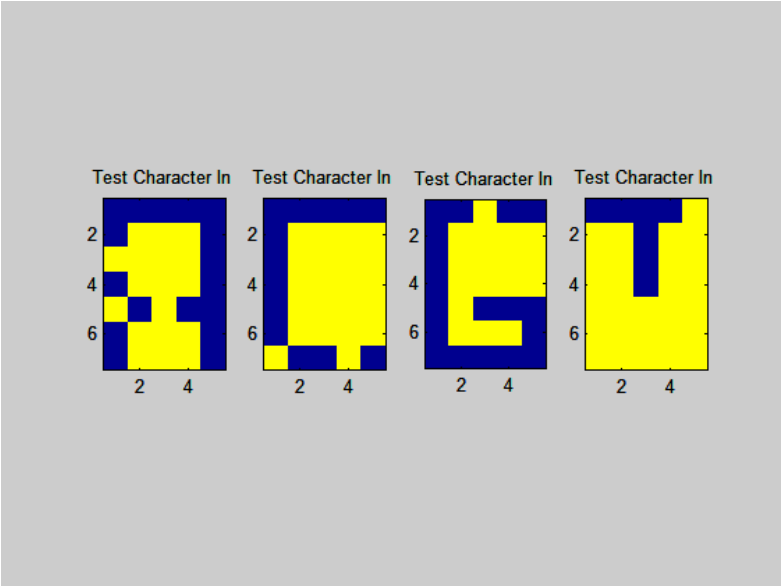

Figure 3. Test characters used to check the algorithm using characters with missing pixels.

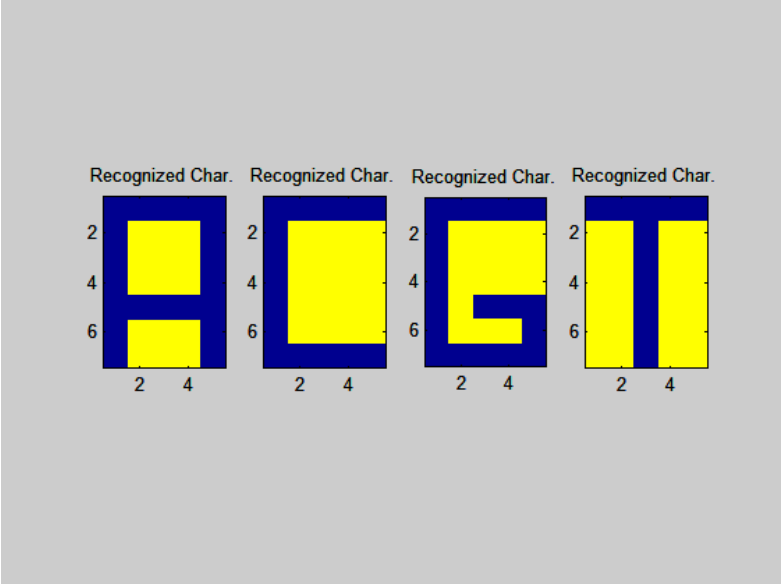

Figure 4. Output of the Associative Matrix, recognizing the characters as similar to the prototype characters.

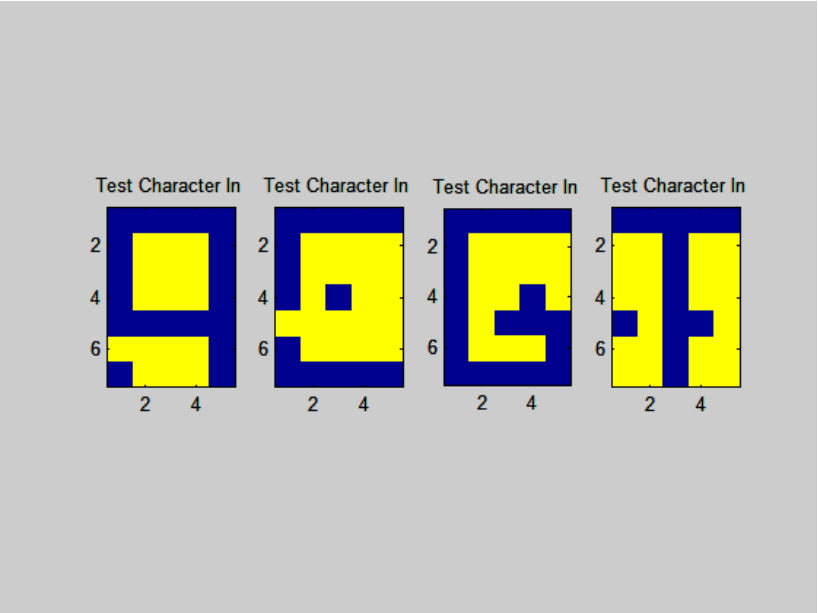

Figure 5. Another set of test characters with missing and additional pixels.

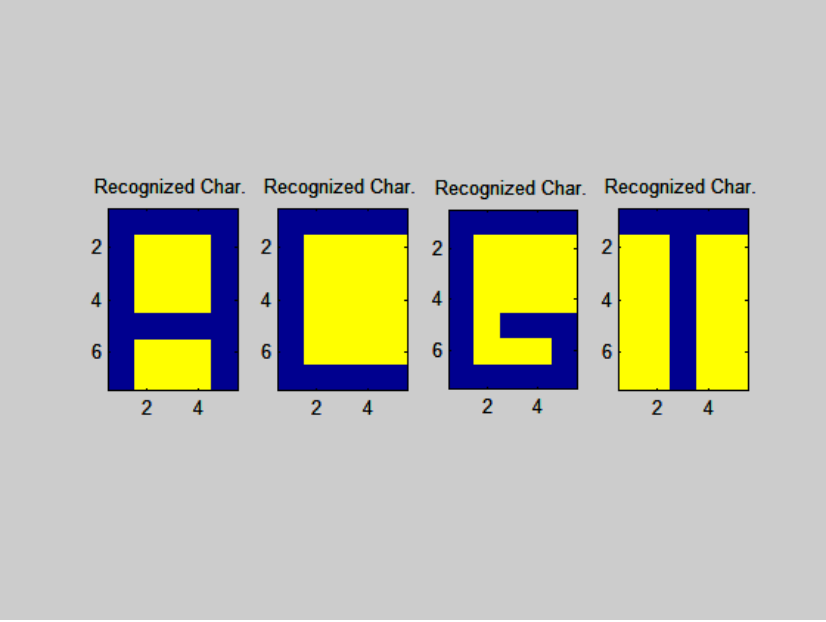

Figure 6. Recognized characters from Figure 5.

5.2 Deliverable 2.

Use the Hebbian Learning rule to associate DNA sequences as a set of characters. These characters could represent a DNA sequence that is required to be searched into a database that could be mutated or identical to the initial sequence. The sequence could be constructed graphically character after character separated by a column of pale characters as shown in Figure 7.

The following is an example of an input sequence used as a prototype.

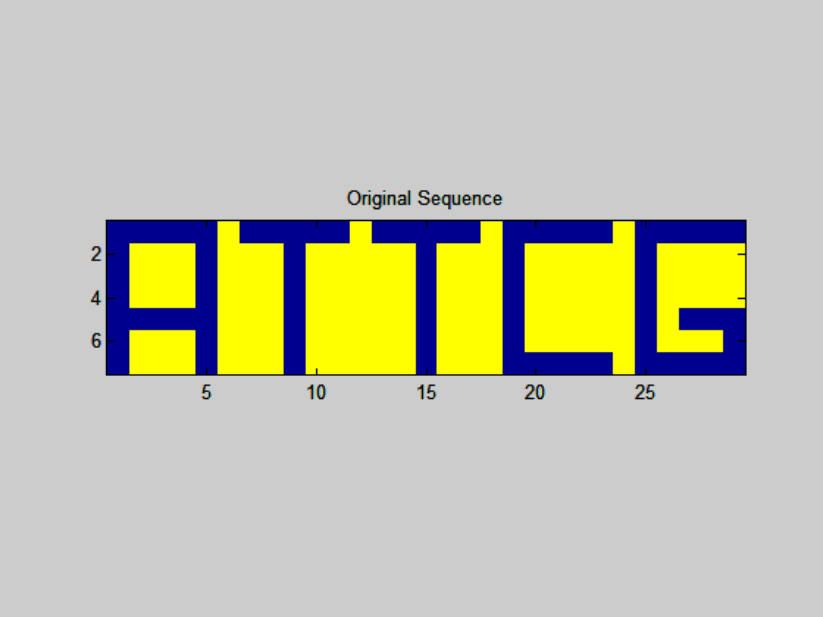

Figure 7. Original sequence to search among data available.

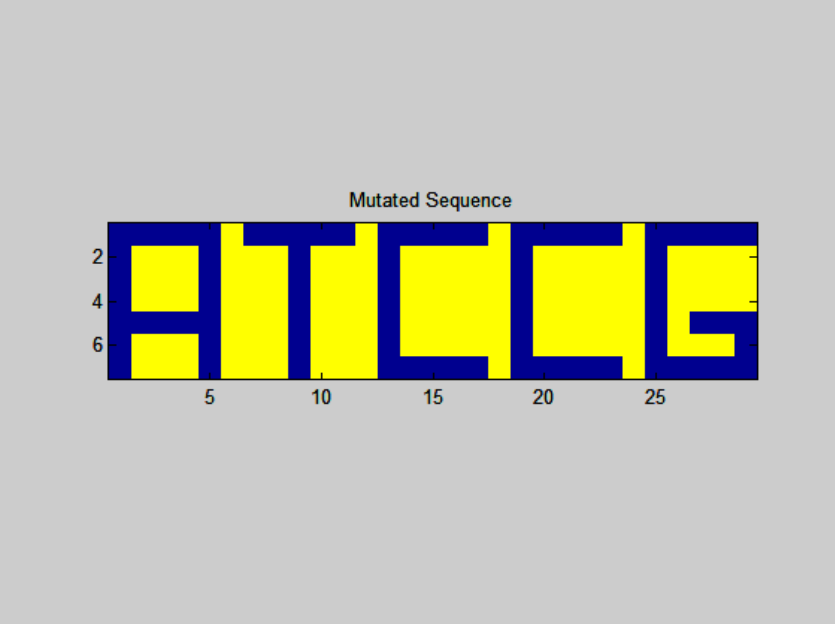

Figure 8. Mutated test sequence in the third character. T is exchanged for a C.

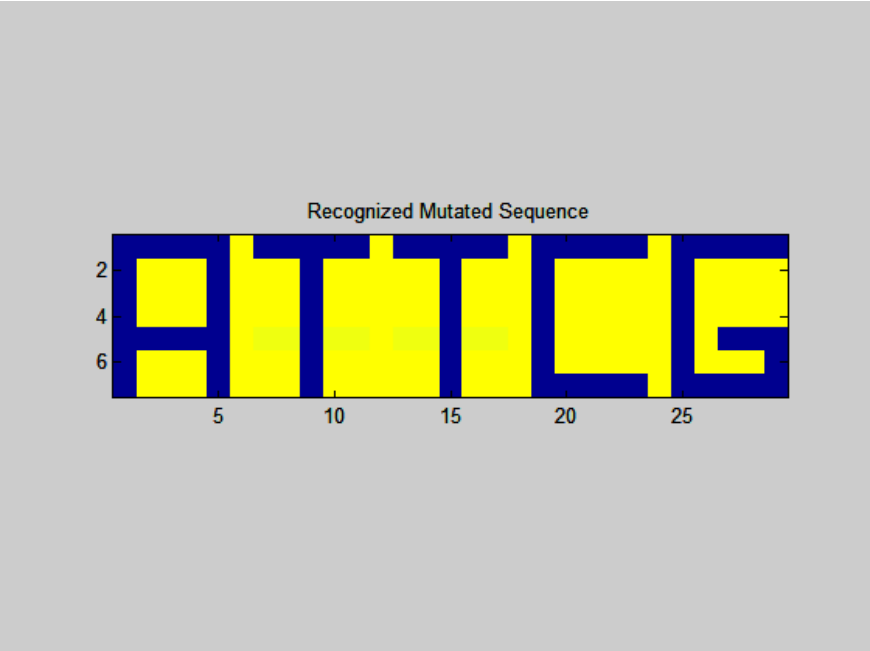

Figure 9. Sequence in Figure 8 recognized after processed by the Associative Matrix.

5.3 Delverable 3.

Look for other associative matrix techniques used in neural networks. What are the differences between Hebb rule and the others?

6. Additional Activities.

1. Try to check how distorted can be the input pattern to be able to be recognized by the associative matrix.
2. Look at any reference more information about the pseudoinverse matrix technique.
3. What could be the benefit obtained if the patterns used have a better resolution (more pixels)?

7. References.

1. Stuart Russell and Peter Norvig."Artificial Intelligence: A Modern Approach." 2nd Edition. Prentice Hall, Upper Saddle River, NJ, USA, 2003. Chapter 20 and Appendix A.
2. Jean-Michel Claverie, Cedric Nothedame, "Bionformatics for Dummies." For Dummies, 1st Edition. 2003. Chapters 1and 2.
3. Bergeron, Bryan. "Bioinformatics Computing."Pearson Education,Inc. 2003.
4. Hagan, Martin T., Demuth, Howard B., Beale, Mark."Neural network design." PWS Publishing Company. 1995.