INNOVARE JOURNAL OF SCIENCES
Vol 8, Special Issue 1, 2020
ISSN-2321-5496

Full Proceeding Paper

MATLAB PROGRAM CODES FOR BIDIRECTIONAL ASSOCIATIVE MEMORY NETWORKS

KALYAN V.A.

Department of Mathematics, Dnyanopasak College, Parbhani, India. Email: kalyankarved11@gmail.com

Received: 25 January 2020, Revised and Accepted: 17 March 2020

ABSTRACT

Objective Neural networks are being used for solving problems in various diverse areas including education, research, business, management, and many more. In this article, models describing the dynamics of bidirectional associative memory (BAM) neural networks are considered.

Methods: MATLAB, the numerical computing environment and programming language is used for solving certain problems associated with BAM.

Results: The concept of BAM networks is improved so that it can be applied to a wider class of networks. Algorithm for solving BAM problems is studied. And also the MATLAB program codes to find the weight matrix, to test the net with input, and to generate activation functions are accompanied.

Conclusion: MATLAB programming can be effectively used to solve the problems associated with BAM.

Keywords: Artificial neural networks, bidirectional associative memory, auto-association, hetero-association.

INTRODUCTION

The fundamental purpose of an associative memory is to correctly recall complete patterns from input patterns, which may be altered with additive, subtractive, or mixed noise [1]. Design of neural network is inspired by the design and functioning of human brain and its components. Bidirectional associative memory (BAM) is a type of recurrent neural network. Bart Kosko, in 1988, introduced bidirectionality in neural networks to produce two-way associative search for stored associations [2]. He developed several versions of the BAM. A BAM stores a set of pattern associations by summing bipolar correlation matrices (an $n \times m$ outer product matrix for each pattern to be stored). The architecture of BAM consists of two layers of neurons: input and output. Information can go in both directions - from input to output and back from output to input. These two layers are connected by directional weighted connection paths. The network iterates by sending signals back and forth between the two layers until all neurons reach equilibrium (i.e., until each neuron's activation remains constant for several steps). BAM neural networks can respond to input to either layer. The weights are bidirectional and the algorithm alternates between updating the activations for each layer. We call these layers as the X-layer and the Y-layer instead of the input and output layers. Kosko proposed a solution based on the same matrix representing the Hopfield model. With that solution he was capable of realizing the learning phase in both directions: to obtain an output pattern from an input pattern and to obtain an input pattern from an output pattern, and hence he named it bidirectional. Kosko's model was successful in obtaining a hetero-associative memory, but the disadvantage of Hopfield memory was not solved by the BAM [3]. Kosko's BAM has a very low pattern learning and recovering capacity, that depends on the minimum of the dimensions of the input and output patterns.

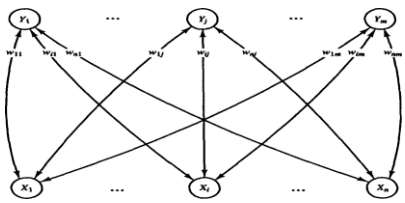

Fig.1: General Diagram of Bidirectional Associative Memory (BAM)

A single-layer nonlinear feedback BAM network has *n* units in its *X*-layer and *m* units in its *Y*-layer (figure). The connections between the layers are bidirectional i.e., if the weight matrix for signals sent from the X-layer to the Y-layer is $\boldsymbol{W}$, then the weight matrix for signals sent from the Y-layer to the X-layer is $\boldsymbol{W^T}$. The formulas for the entries depend on whether the training vectors are binary or bipolar. Associative memory is of two types, auto-associative and hetero-associative. BAM is hetero-associative, which means when a pattern is given, it can return another pattern which is potentially of a different size. An auto-associative memory can be considered as a particular case of a hetero-associative memory [2,4].

Neural network has many applications which include recognition, classification, association, information extraction and reasoning. Neural Networks are also used in self driving cars, image compression, machine translation, stock market prediction, and lots of other interesting applications. The Artificial Neural Networks have ability to learn so quickly, which makes them so powerful and useful for a variety of tasks. The BAM does hetero-associative processing in which association between pattern pairs is stored. A BAM contains two layers of neurons, which are fully connected to each other. Once the weights have been established, input into first layer presents the pattern in second layer, and vice versa.

During past decades, associative memories and neural networks have been developed with hand in hand. The first model of artificial neuron was the Threshold Logic Unit (TLU), or Linear Threshold Unit, created by Warren McCulloch and Walter Pitts in 1943. The model was specifically targeted as a computational model of the *nerve net* in the brain and was based on mathematics and algorithms [5]. In 1982 John J. Hopfield came with his associative memory model which also has the particularity of an iterative algorithm [4]. His work resulted in the renewal of researchers' interest on topics regarding both associative memories as well as neural networks, which has dominated past few years. Modern concepts of neural network models are based on morphological mathematics, graph theory and fuzzy logic [6].

PRELIMINARIES

Definition 1: (X, Y) is *fixed point* of BAM iff $Y = sgn(XW)$ and $X^T = sgn(WY^T)$ [7,8].

Definition 2: A continuous BAM transforms input smoothly and continuously into output in the range [0 , 1] using the logistic sigmoid function as the activation function for all units.

Proposition 1: For binary input vectors, *the weight matrix* $W = \{w_{ij}\}$ is given by

$$w_{ij} = \sum_{p=1}^{P} (2s_i(p) - 1)(2t_j(p) - 1)$$

Whereas for bipolar input vectors, the *weight matrix* $W = \{w_{ij}\}$ is given by

$$w_{ij} = \sum_{p=1}^{P} (s_i(p))(t_j(p)).$$

Proposition 2: For the discrete BAM, *the activation function* is the appropriate step function depending on whether *binary or bipolar vectors* are used [3,8]. For *binary input vectors*, the activation function for the X-layer is given by

$$x_i = \begin{cases} 0 & if\ x_{in_i} < 0 \\ x_i & if\ x_{in_i} = 0 \\ 1 & if\ x_{in_i} > 0, \end{cases}$$

and the activation function for the Y-layer is given by

$$y_j = \begin{cases} 0 & if\ y_{in_j} < 0 \\ y_j & if\ y_{in_j} = 0 \\ 1 & if\ y_{in_j} > 0. \end{cases}$$

Whereas for *bipolar input vectors*, the activation function for the X-layer is given by

$$x_i = \begin{cases} -1 & if\ x_{in_i} < \theta_i \\ x_i & if\ x_{in_i} = \theta_i \\ 1 & if\ x_{in_i} > \theta_i, \end{cases}$$

and the activation function for the Y-layer is given by

$$y_j = \begin{cases} -1 & if\ y_{in_j} < \theta_i \\ y_j & if\ y_{in_j} = \theta_i \\ 1 & if\ y_{in_j} > \theta_i, \end{cases}$$

Proposition 3: For the continuous BAM, *the activation function* is the logistic sigmoid

$$f(y_{in_j}) = \frac{1}{1 + \exp(y_{in_j})}$$

where $y_{in_j} = b_j + \sum_i w_{ij} x_i$.

Proposition 4: BAM can be described by the formula:

$$Y = sgn(XW)$$

where sgn is sign function, the output of sgn function is always 1 or -1 [4].
X is input - $m \times n$ matrix, Y is output - $m \times l$ matrix, W is weight - $n \times l$ matrix. In other words we have m input - output pairs to train network, the input layer has n nodes and the output layer has l nodes. Since $sgn(1) = 1$ and $sgn(-1) = -1$, we can put that

$$Y = sgn(Y)$$

As all members of XX^T are positive, so we can put

$$Y = sgn(XX^T Y).$$

Comparing the last equation with the equation

$$Y = sgn(XW),$$

we get

$$W = X^T Y.$$

ALGORITHM

Step 1: The associations between pattern pairs are stored in the memory in the form of bipolar binary vectors with entries -1 and 1. Vector *X* is *n* - dimensional and it stores pattern, *Y* is *m* -dimensional and it stores associated output.

Step 2: Weights are calculated by using the formula

$$w = \sum_{i=1}^{p} (x^{[i]})(x^{[i]t})$$

Step 3: Test vector pair *X* and *Y* is given as input.

Step 4: In the forward pass, *Y* is given as input and *X* is calculated by using

$$X = \Gamma[WY]$$

Each element of vector X is given by

$$x_i' = sgn(\sum_{j=1}^{m} w_{ij}\, y_j), \qquad for\ i = 1,2,\dots,n$$

Step 5: Vector X is now given as input to the second layer during backward pass. Output of this layer is given by

$$Y' = \Gamma[WX]$$

Each element of vector Y is given by

$$y_j' = sgn(\sum_{i=1}^{n} w_{ij}\, x_i), \qquad for\ j = 1,2,\dots,m$$

Step 6: If there is further update then repeat steps 4 and 5 otherwise stop process.

RESULTS

MATLAB programs are developed to store vectors in auto associative networks, find the weight matrix, and test the net with input. Programs are also developed to generate activation functions and XOR function using McCulloch-Pitts neuron**.**

Program 1: MATLAB program to (I) store two vectors (-1 -1 -1 -1), (-1 1 1 1) in an auto associative network, (II) find the weight matrix and (III) test the net with (1 1 1 1) as input.

```
% Auotassociative net to store two vectors, find the weight matrix
and test the net

clc;
clear;
x=[–1 –1 –1 –1;–1 1 1 1];
t=[1 1 1 1];
w=zeros (4, 4);
for i=1:2
   w=w + x(i,1:4)'*x(i,1:4);
   end
yin = t*w;
   for i=1:4
     if yin(i)>0
       y(i)=1;
     else
       y(i)=–1;
     end
   end

disp('The calculated weight matrix is');

disp(w);

if x(1,1:4)==y(1:4)

  x(2,1:4)==y(1:4)

   disp('The vector is a Known Vector');
```

```
else
   disp('The vector is a unknown vector');
end
```

Output

The calculated weight matrix is

2 2 0 0
2 2 0 0
0 0 2 2
0 0 2 2

The vector is an unknown vector.

Program 2: MATLAB program to calculate the weights for mapping four input vectors to two output vectors (using hetero associative neural net).

Input vectors				Output vectors	
x1	x2	x3	x4	t1	t2
1	1	0	0	1	0
1	0	1	0	1	0
1	1	1	0	0	1
0	1	1	0	0	1

```
% Hetero associative neural network for mapping input vectors to output vectors
clc;
clear;
x= [1 1 0 0; 1 0 1 0; 1 1 1 0; 0 1 1 0];
t= [1 0; 1 0; 0 1; 0 1];
w=zeros (4, 2);
for i=1:4
   w=w+x (i, 1:4)'*t(i,1:2);
end
disp('The calculated weight matrix is');
disp(w);
```

Output

The calculated weight matrix is

2 1
1 2
1 2
0 0

Program 3: MATLAB program for calculating the weight matrix using BAM network.

```
% Bidirectional Associative Memory neural net
clc;
clear;
s=[1 1 0;1 0 1];
t=[1 0;0 1];
x=2*s–1
y=2*t–1
w=zeros(3,2);
for i=1:2
   w=w+x(i,:)'*y(i,:);
end
disp('The calculated weight matrix is');
disp(w);
```

Output

The calculated weight matrix

0 0
2 –2
–2 2

Program 4: MATLAB program to cluster two vectors.

```
% Kohonen self organizing maps
clc;
clear;
x=[1 1 0 0;0 0 0 1;1 0 0 0;0 0 1 1];
alpha=0.7;
% initial weight matrix
w=rand(4,2);
disp('Initial weight matrix');
disp(w);
contr=1;
epoch=0;
while contr
  for i=1:4
    for j=1:2
      D(j)=0;
      for k=1:4
        D(j)=D(j)+(w(k,j)-x(I,k))^4;
      end
    end
for j=1:2
  if D(j)==min(D)
  J=j;
end
  w(:,J)=w(:,J)+alpha*(x(i,:)'-w(:,J));
end
alpha=0.5*alpha;
epoch=epoch+1;
    if epoch==200
    contr=0;
    end
    end
    disp('Weight matrix after 200 epochs')
    disp(w);
```

Output

Initial weight matrix

0.8427	0.5271
0.5169	0.9578
0.8469	0.7492
0.4182	0.4639

Weight matrix after 200 epochs

0.0427	0.9467
0.0127	0.4525
0.5941	0.0312
0.9789	0.0093

DISCUSSION

In BAM, a distorted input pattern may also cause correct hetero-association at the output. The efficiency of BAM in pattern storage and recall capability can be severely affected by the logical symmetry of interconnection. It also limits their use for knowledge representation and inference. There is limitation on number of pattern pairs, which can be stored and successfully retrieved. The BAM is unconditionally stable which means that any set of associations can be learned without risk of instability. The maximum number of associations to be stored in the BAM should not exceed the number of neurons in the smaller layer. The BAM may not always produce the closest association. A stable association may be only slightly related to the initial input vector. The results for Kosko's

BAM are drastically reduced in comparison to the previous results on auto-associations.

REFERENCES

1. Schalkoff R J. Pattern Recognition: Statistical Structured and Neural Approach. New York: John Wiley and Sons Inc.; 1992.
2. Bart Kosko. Bidirectional Associative Memories. IEEE Transactions on Systems, Man, and Cybernetics. 1988; 18:49-60.
3. Chartier S, Boukadoum M. Encoding static and temporal patterns with a bidirectional heteroassociative memory. J of Applied Mathematics. 2011; 68:1-34.
4. Govindan V K, Sivaprasad A P. Character Recognition- A review. Pattern recognition 1999; 23:671-83.
5. Haikin S. Neural Networks: A comprehensive Foundation. New York: Macmillan College Publishing Company; 2014.
6. Zurada J M. Introduction to Artificial Neural Systems with Applications. Mumbai, India: Jaico Publication House; 2004.
7. Xu Z B, Leung Y, He X W. Asymmetrical Bidirectional Associative Memories. IEEE Transactions on Systems, Man and Cybernetics. 2008; 38: 1558-64.
8. Kalyan V A. On the recent developments and programming of bidirectional associative memory network. Int Research J Printing Area. 2018;68:152-61