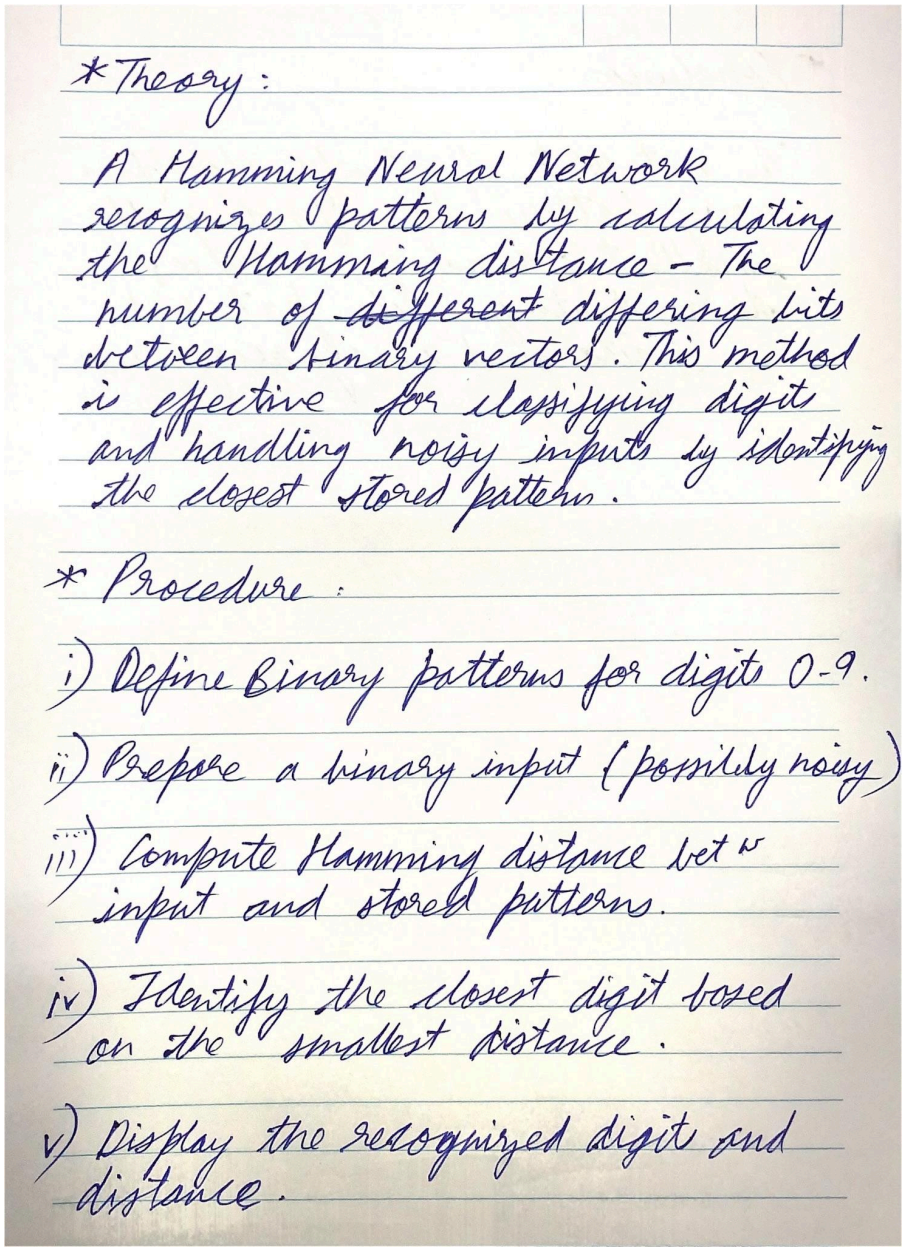
Assignment 7

Experiment 13:

Aim: Implement a Program for Pattern storage of 10 digits with Hamming Neural Network.

Objective: Students will be able to understand practical aspects of Hamming Neural Network model



*Theory :

A Hamming Neural Network recognizes patterns by calculating the Hamming distance - The number of ~~different~~ differing bits between binary vectors. This method is effective for classifying digits and handling noisy inputs by identifying the closest stored pattern.

* Procedure :

i) Define Binary patterns for digits 0-9.

ii) Prepare a binary input (possibly noisy)

iii) Compute Hamming distance bet^n input and stored patterns.

iv) Identify the closest digit based on the smallest distance.

v) Display the recognized digit and distance.

Code:

```
% Hamming Neural Model

% Binary representations of digits (simplified example)
patterns = [
   0 0 0 0 0; % Digit 0 (binary pattern)
   0 0 0 0 1; % Digit 1 (binary pattern)
   0 0 0 1 0; % Digit 2 (binary pattern)
   0 0 0 1 1; % Digit 3 (binary pattern)
   0 0 1 0 0; % Digit 4 (binary pattern)
   0 0 1 0 1; % Digit 5 (binary pattern)
   0 0 1 1 0; % Digit 6 (binary pattern)
   0 0 1 1 1; % Digit 7 (binary pattern)
   0 1 0 0 0; % Digit 8 (binary pattern)
   0 1 0 0 1; % Digit 9 (binary pattern)
];

% Input pattern (example: input for which you want to find the closest
match)
input_pattern = [0 0 0 1 0]; % Input could be a noisy version of digit
2

% Initialize variables
min_hamming_distance = inf; % Start with a large number
closest_digit = -1;

% Compute Hamming distance for each stored pattern
for digit = 1:10
   % Calculate Hamming distance (number of differing bits)
   hamming_distance = sum(input_pattern ~= patterns(digit, :));

   % If the current pattern is closer, update the closest digit
   if hamming_distance < min_hamming_distance
       min_hamming_distance = hamming_distance;
       closest_digit = digit - 1; % Because digit index is 1-based
   end
End
```

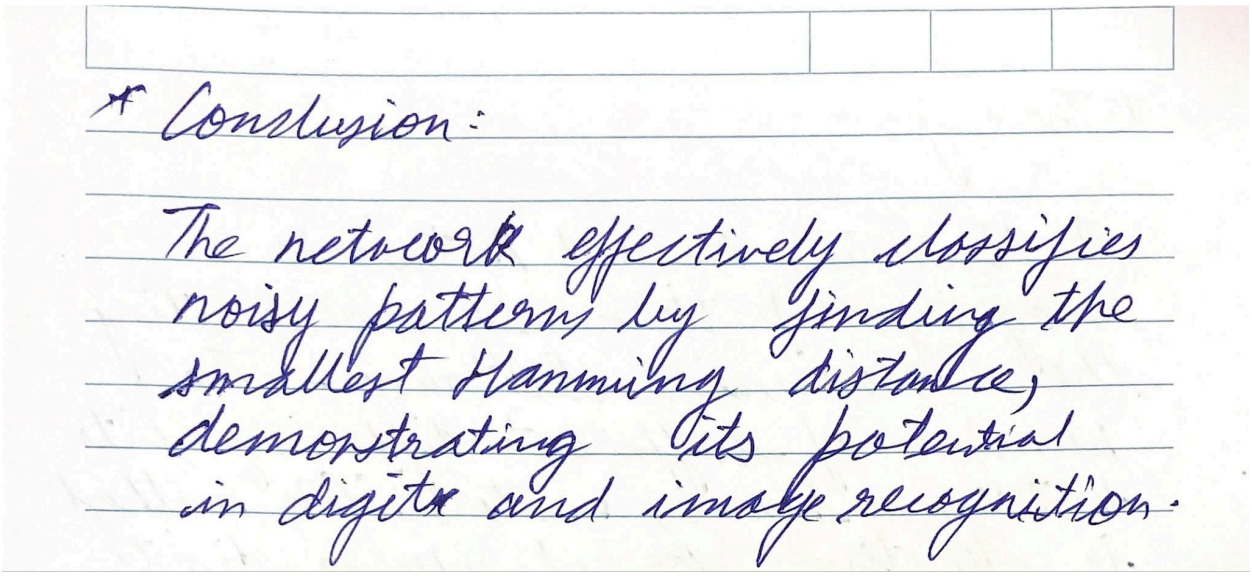
```
% Output the recognized digit
disp("Recognized Digit:")
disp(closest_digit)
disp("Hamming Distance:")
disp(min_hamming_distance)
```

Output:

```
Command Window
>> Experiment_13
Recognized Digit:
     2

Hamming Distance:
     0

>>
```



* Conclusion:

The network effectively classifies noisy patterns by finding the smallest Hamming distance, demonstrating its potential in digits and image recognition.