Recognising digits (2020)

For each of the three experiments you are asked two questions: (A) To which pattern does your network converge? (B) Classify this pattern using the following scheme: if the pattern you obtain correspods to any of the stored patterns ***x***​​​(*μ*)​​, enter the pattern index *μ*. If your network retrieves an inverted stored pattern, then enter −*μ*. If you get anything else, enter 6.

**Functions Used:**

function [new\_pattern, isSame] = aSynchronousUpdate(s,W,N)

%Outputs [new\_pattern, isSame] where new\_pattern is an asynchronously

% updated pattern s according to matrix W and bit-length N and isSame=1 if

% steady state is reached, 0 otherwise

new\_pattern = s;

neuronsChecked = zeros(1,N); % 1 if neuron at index i has been checked, 0 otherwise

while ismember(0,neuronsChecked)

i = randi(N);

if neuronsChecked(i) == 0

neuronsChecked(i) = 1;

end

b = W(i,:)\*new\_pattern';

new\_pattern(i) = sgn(b);

end

isSame = isequal(new\_pattern,s);

end

function out = sgn(num)

%Outputs 1 if input >=0 and -1 if <0

if num >= 0

out = 1;

else

out = -1;

end

end

**Scripts Used:**

X = readmatrix('X.txt'); % A matrix (csv format) file where each row is a pattern i.e. 1st row is pattern "0", 2nd row is pattern "1", ...

% These are in csv format, typewriter

% test\_pattern = readmatrix('test\_pattern1.txt');

% test\_pattern = readmatrix('test\_pattern2.txt');

test\_pattern = readmatrix('test\_pattern3.txt');

sizeX = size(X);

p = sizeX(1);

N = sizeX(2);

W = (X'\*X - p\*eye(N))/N;

converged = 0;

cnt = 0;

while converged == 0

[test\_pattern, converged] = aSynchronousUpdate(test\_pattern,W,N);

end

state = 6;

digit = NaN;

for i=1:p

if isequal(X(i,:),test\_pattern)

formatted\_pattern = reshape(test\_pattern,10,16)';

state = i;

digit = i - 1;

writematrix(formatted\_pattern,'formatted\_pattern.csv');

break

elseif isequal(-1\*X(i,:),test\_pattern)

formatted\_pattern = reshape(test\_pattern,10,16)';

state = -i;

digit = i - 1;

writematrix(formatted\_pattern,'formatted\_pattern.csv');

break

end

end

disp('The pattern is classified as state:')

disp(state)

if ~isnan(digit)

if state > 0

disp('The pattern converged to the digit:')

else

disp('The pattern converged to the INVERSE of digit:')

end

disp(digit)

else

disp('The pattern did not converge to any stored pattern or its inverse')

end