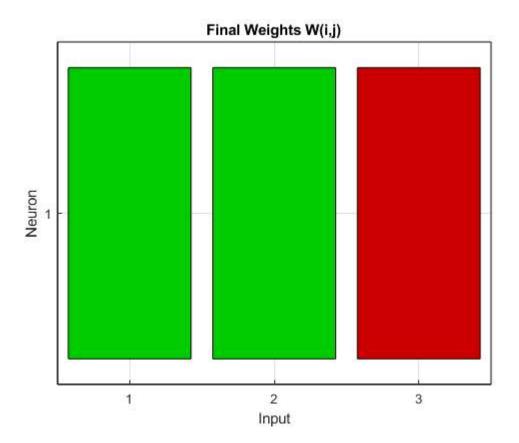
```
%David Melanson
clf reset; clearvars; clc
pausetime = 0;
pauseflag = 0;
maxw = 10;
minw = 0;
%Problem definition - bipolar AND function
%define three 4 element input vectors -- one element in a vector for each
%state
P = [
   1 1 -1 -1;
                    %X1
   1 -1 1 -1;
                  %X2
   1 1 1 1
                   %Bias
   ];
%each row is one input set
%each column is one set of training data for network
%define 4 1-element output vectors
% Ni01 Ni02 Ni03 Ni04
T = [1 -1 -1 -1];
%Initialize network
[R, Q] = size(P); [S, Q] = size(T);
W0 = zeros(S, R);
B0 = ones(S, 1);
fprintf('R =%2i Q =%2i S =%2i\n', R, Q, S);
% Plot original values
disp(['pause for ', num2str(pausetime), 'sec']);
pause(pausetime);
figure(1);
hintonw(W0, maxw, minw);
title('Original Weights W(i,j)');
% TRAIN THE NETWORK
% TRAINING PARAMETERS
disp_freq = 1;
max_epoch = 4;
% lr = 0.1;
% dr = 1r/3;
% lp Learning Parameter
% lr Learning Rate
% dr Decay Rate
lr = 1;
lp.lr = lr;
lp.dr = 0;
W = W0;
B = B0;
for epoch = 1:max_epoch
   for q = 1:Q
      % PRESENTATION PHASE
      A = T(:, q);
      % LEARNING PHASE
```

```
dW = learnhd(W, P(:, q), [], [], A, [], [], [], [], lp, []);
       fprintf("\npause\n");
       W = W + dW;
       fprintf('input = %2i W = %2i %2i %2i', epoch, W)
       if pauseflag == 1
           pause(pausetime)
           figure(1)
       end
   end
   % DISPLAY PROGRESS
   if rem(epoch, disp freq) == 0
       pause(pausetime)
       hintonw(W, maxw, minw)
       title('Weights W(i,j)');
   end
end
% PLOT FINAL VALUES
hintonw(W, maxw, minw);
title('Final Weights W(i,j)');
pause(pausetime);
% SUMMARIZE RESULTS
fprintf("\nWith inputs of \n%2i %2i \%2i\n%2i %2i \n%2i %2i\n%2i %2i\n", P);
fprintf("The network applies weights of %2i %2i %2i.\n", W)
A_nonbin = hardlims(W*P);
A_bin = hardlim(W*P);
disp('The network responds with binary outputs')
disp(A_bin)
disp('Thus the network successfully replicated an AND gate.')
%input, target, and output are necessary for submission of HW
```

```
R = 3 Q = 4 S = 1
pause for Osec
pause
input = 1 W = 1 1 1
pause
input = 1 W = 0 2 0
pause
input = 1 W = 1 1 -1
pause
input = 1 W = 2 2 - 2
pause
input = 2 W = 3 3 -1
pause
input = 2 W = 2 4 - 2
pause
input = 2 W = 3 3 - 3
pause
input = 2 W = 4 4 -4
pause
input = 3 W = 5 5 - 3
pause
input = 3 W = 4 6 - 4
pause
```

```
input = 3 W = 5 5 - 5
pause
input = 3 W = 6 6 - 6
pause
input = 4 W = 7 7 -5
pause
input = 4 W = 6 8 - 6
pause
input = 4 W = 7 7 -7
pause
input = 4 W = 8 8 - 8
With inputs of
1 1 1
1 -1 1
-1 1 1
-1 -1 1
The network applies weights of 8 8 -8.
The network responds with binary outputs
         0
            0 0
```

Thus the network successfully replicated an AND gate.



Published with MATLAB® R2022a