Boolean Functions - Matlab Code

Lukas Fu

Matlab Code

```
clear
clc
n=4; % #dimensions 2,3,4,5
nTrials = 10000; nEpochs = 20;
eta = 0.05; % learning rate
counter = 0;
%booleanInputs = [0 0 ; 0 1 ; 1 0 ; 1 1];
booleanInputs = [];
for D = 1:2^n
    binNum = dec2bin(D-1,n);
    binNumArray = regexp(binNum,'\d','match');
    binVec = [];
    for i = 1:n
        binDigit = str2double(binNumArray{i});
        binVec = [binVec binDigit];
    end
    booleanInputs = [booleanInputs ; binVec];
end
for i = 1:numel(booleanInputs) % set 0 to -1
   if booleanInputs(i) == 0
       booleanInputs(i) = -1;
   end
end
usedBool = {};
duplicateCounter = 0;
for trial = 1:nTrials
    %sample boolean function
    booleanOutput = randi([0 1],2^n,1); % random vector of 1 and 0
    for i = 1:length(booleanOutput) % set 0 to -1
       if booleanOutput(i) == 0
           booleanOutput(i) = -1;
       end
    end
    isNotMember = true;
    for l = 1:length(usedBool)
        if usedBool{l} == booleanOutput
            isNotMember = false;
            duplicateCounter = duplicateCounter +1 ;
            break
        end
    end
    if isNotMember %if output not in usedBool
        w = randn(1,n) / sqrt(n); %weight
```

```
44          th = 0; %threshold
45
46          for epoch = 1:nEpochs
47              totalError = 0;
48              for mu = 1:2^n % compute output
49                  b = 0;
50                  for j = 1:n
51                      b = b + w(j) * booleanInputs(mu,j);
52                  end
53                  if b == 0
54                      y = 1;
55                  else
56                      y = sign(b-th);
57                  end
58                  error = booleanOutput(mu) - y;
59                  %update weight and threshold
60                  dw = eta * (error) * booleanInputs(mu,:);
61                  dth = -eta * (error);
62                  w = w + dw;
63                  th = th + dth;
64                  totalError = totalError + abs(error);
65              end
66              if totalError == 0
67                  counter = counter+1;
68                  break
69              end
70          end
71      end
72      usedBool{end+1} = booleanOutput; % add booleanOutput array to usedBool
            array
73  end
74  numberOfBoolFunc = nTrials - duplicateCounter;
75  disp(numberOfBoolFunc)
76  disp(counter)
```