LINEAR SEPARABILITY OF 4-DIMENSIONAL BOOLEAN FUNCTIONS

Table of Contents

Load data

```
clear
inputData = load("input_data_numeric.csv")';
targetDataA = [1, -1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1];
targetDataB = [1, -1, 1, 1, -1, 1, 1, 1, 1, -1, -1, 1, -1, 1, 1, -1];
targetDataC = [-1, 1, -1, 1, -1, 1, -1, 1, -1, 1, 1, 1, -1, -1, 1,
 -1];
targetDataD = [1, 1, 1, -1, 1, 1, -1, 1, 1, 1, -1, 1, -1, -1, -1, 1];
targetDataE = [1, 1, 1, -1, -1, 1, -1, -1, 1, 1, -1, 1, 1, -1, -1,
 -1];
targetDataF = [1, 1, 1, -1, 1, 1, 1, 1, -1, -1, -1, -1, 1, 1, -1, -1];
targetData = [targetDataA; targetDataB; targetDataC; ...
    targetDataD; targetDataE; targetDataF];
```

Check linear separability

```
T = 1e5;
learningRate = 0.02;
alphabet = ['A', 'B', 'C', 'D', 'E', 'F'];
for i = 1:6
    currentTargetData = targetData(i,:);
    threshold = rand();
    weights = randomWithinRange([1 4], -0.2, 0.2);
    energies = zeros(1,T);
    for t = 1:T
        patternNumber = randi([1 16]);
        input = inputData(2:5, patternNumber);
        output = calculateOutput(input, weights, threshold);
        target = currentTargetData(patternNumber);
        b = -threshold + weights*input;
        delta = learningRate*(target - output)*gPrime(b);
        deltaWeights = delta*input';
        weights = weights + deltaWeights;
        deltaThreshold = -delta;
        threshold = threshold + deltaThreshold;
        outputs = calculateOutputs(inputData(2:5,:), weights,
 threshold);
```

```
        energies(t) = energyFunction(outputs, currentTargetData);
    end
    success = (all(sign(outputs) == currentTargetData));
    if success
        disp(['Function ', alphabet(i), ' is linearly separable.'])
    else
        disp(['Function ', alphabet(i), ' is not linearly
 separable.'])
    end
end
```

```
Function A is linearly separable.
Function B is not linearly separable.
Function C is not linearly separable.
Function D is not linearly separable.
Function E is linearly separable.
Function F is linearly separable.
```

Functions

```
disp('')
function r = randomWithinRange(size, min, max)
    r = (max - min).*rand(size) + min;
end

function g = g(x)
    g = tanh(x);
end

function gPrime = gPrime(x)
    gPrime = 1-(tanh(x)).^2;
end

function output = calculateOutput(input, weights, threshold)
    output = g(1./2.*(-threshold + weights*input));
end

function outputs = calculateOutputs(inputs, weights, threshold)
    numberOfInputs = size(inputs,2);
    outputs = zeros(1, numberOfInputs);
    for i = 1:numberOfInputs
        input = inputs(:, i);
        outputs(i) = calculateOutput(input, weights, threshold);
    end
end

function energy = energyFunction(outputs, targets)
    energy = 1/2*sum((outputs-targets).^2);
end
```

Published with MATLAB® R2020a