Boolean functions 2022

Erik Norlin

All equations are taken from the course book Machine Learning With Neural Networks.

$$O = \mathrm{sgn}(w_1 x_1 + w_2 x_2 - \theta) = \mathrm{sgn}(\boldsymbol{w} \cdot \boldsymbol{x} - \theta). \qquad (5.9)$$

$$\delta w_{ij}^{(\mu)} = \eta (t_i^{(\mu)} - O_i^{(\mu)}) x_j^{(\mu)}. \qquad (5.18)$$

```
nDimensions = 5;
counter = zeros(nDimensions,1);

for iDimension = 2:nDimensions
    nTrials = 10^4;
    eta = 0.05;
    nEpochs = 20;
    booleanInputs = zeros(2^iDimension,iDimension);
    booleanOutputs = zeros(2^iDimension,1);
    usedBoolean = [];

    % Generating boolean inputs based on given dimensions
    booleanInputs(1,:) = -1;
    for i = 1:2^iDimension - 1
        binary = dec2bin(i,iDimension);
        for j = 1:iDimension
            booleanInputs(i+1,j) = str2num(binary(j));
            if booleanInputs(i+1,j) == 0
                booleanInputs(i+1,j) = -1;
            end
        end
    end

    for iTrial = 1:nTrials

        % Sampling a random boolean function based on given dimensions
        for j = 1:2^iDimension
            outputState = rand;
            if outputState < 0.5
                outputState = -1;
            else
                outputState = 1;
            end
            booleanOutputs(j) = outputState;
        end

        % Checking if sampled boolean function already has been used
        validBoolean = true;
        if width(usedBoolean) > 0
            for jCol = 1:width(usedBoolean)
                if isequal(booleanOutputs, usedBoolean(:,jCol))
                    validBoolean = false;
```

```
                    break
                end
            end
        end

        if validBoolean
            weight = randn(1,iDimension)/sqrt(iDimension);
            theta = 0;

            % Training the network for given number of epochs
            for jEpoch = 1:nEpochs
                errorCounter = 0;

                % Updating weight and threshold for each boolean output
                for muPattern = 1:2^iDimension

                    input = booleanInputs(muPattern,:)';
                    output = sign(weight*input - theta);
                    if output == 0
                        output = 1;
                    end

                    target = booleanOutputs(muPattern);
                    xMu = booleanInputs(muPattern,:);

                    deltaWeight = eta*(target - output)*xMu;
                    deltaTheta = -eta*(target - output);

                    weight = weight + deltaWeight;
                    theta = theta + deltaTheta;

                    error = target - output;
                    errorCounter = errorCounter + abs(error);
                end
                if errorCounter == 0
                    counter(iDimension,1) = counter(iDimension,1) + 1;
                    break
                end
            end
            % Adding sampled boolean function to used boolean functions
            iCol = width(usedBoolean) + 1;
            usedBoolean(:,iCol) = booleanOutputs;
        end
    end
end

% Printing results
fprintf("Based on the training of the network. ..." + ...
    "The number of linearly separable boolean functions in 'n' dimensions:");
```

```
Based on the training of the network. ...The number of linearly separable boolean functions in 'n' dimensions:
```

```
for iDimension = 2:nDimensions
    fprintf("%d dimensions: %d\n", iDimension, counter(iDimension));
```

```
end
```

```
2 dimensions: 14
3 dimensions: 104
4 dimensions: 262
5 dimensions: 0
```