

# Einführung in die Neuroinformatik

Tim Luchterhand, Paul Nykiel (Gruppe P)

10. Juli 2018

## 1 Dropout

### 1.1

- a) Durch Dropout wird die Relevanz einzelner Neuronen reduziert, da das trainierte Netzwerk auch nur mit einer Teilmenge aller Neuronen funktioniert. Es werden quasi mehrere Netze mit weniger Neuronen trainiert und dann der Durchschnitt verwendet. Im Gegensatz zur naiven Implementierung mit mehreren Netzen wird allerdings deutlich weniger Rechenkapazität benötigt.
- b) Die Gewichte müssen mit dem Faktor  $1 - p = \frac{2}{3}$  skaliert werden.

### 1.2

- a)

```
1 function [netDropout] = networkDropout(input, target,
    dropoutRate)
2 %networkDropout creates and trains a neural network for a
    regression task
3 % Arguments:
4 %     - input: one-dimensional input values (first data
    dimension)
5 %     - target: one-dimensional target values (second
    data dimension)
6 %     - dropoutRate: probability for dropping out neurons
    in the second and third layer
7 %
8 % Returns:
9 %     - netDropout: the trained network object which can
    be used directly for prediction
```

```

10 %
11     assert(isrow(input), 'The input values must be passed
12           as a row-vector');
12     assert(isrow(target), 'The target values must be passed
13           as a row-vector');
13     assert(length(input) == length(target), 'Each input
14           value must have an associated target value');
14     assert(isscalar(dropoutRate), 'The dropout rate must be
15           a scalar value');
15     assert(dropoutRate >= 0 && dropoutRate <= 1, 'The
16           dropout rate must be in the range [0;1]');
16
17     rng(1337, 'combRecursive');
18
19     % Define options for training
20     options = trainingOptions('adam', ...
21           'MaxEpochs', 3000, ...
22           'Shuffle', 'never', ...
23           'L2Regularization', 0.0, ...
24           'InitialLearnRate', 0.01, ...
25           'GradientDecayFactor', 0.999, ...
26           'ValidationPatience', inf, ...
27           'Plots', 'training-progress', ...
28           'Verbose', false);
29
30     layers = [
31         sequenceInputLayer(1)
32         fullyConnectedLayer(100)
33         leakyReluLayer(0.01)
34         dropoutLayer(dropoutRate)
35         fullyConnectedLayer(100)
36         leakyReluLayer(0.01)
37         dropoutLayer(dropoutRate)
38         fullyConnectedLayer(1)
39         regressionLayer()
40     ];
41
42     netDropout = trainNetwork(input, target, layers, options)
43         ;
43 end

```

b)

```

1 load('data.mat');

```

```

2 inputs = data(:,1);
3 outputs = data(:,2);

c)

5 netDropout = networkDropout(transpose(inputs), transpose(
    outputs), 0.1);
6 netWithoutDropout = networkDropout(transpose(inputs),
    transpose(outputs), 0.0);

d)

8 x = -10:0.01:10;
9 yDropout = predict(netDropout,x);
10 yWithoutDropout = predict(netWithoutDropout, x);

e)

12 figure();
13 plot(x, yDropout);
14 xlabel("x");
15 ylabel("y");
16 hold on;
17 plot(x, yWithoutDropout);
18 scatter(inputs, outputs);
19 legend("Mit Dropout", "Ohne Dropout", "Datenpunkte");
20
21 print("b10a01.eps", "-depsc");

```

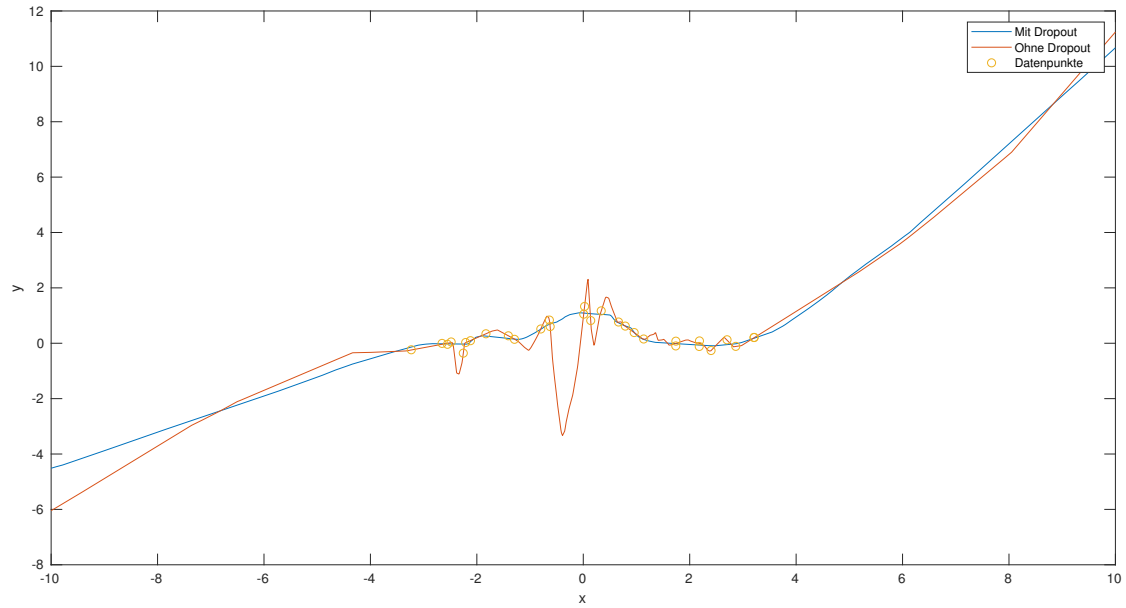


Abbildung 1: Ausgabe des Matlab-Skripts

f)

```

23 save('b10a01_y', 'yDropout');
24 save('b10a01_net', 'netDropout');

```

## 2 Uncertainty via Dropout

### 2.1

```

1 % Initialization
2 rng(1337, 'combRecursive');
3
4 load('data.mat');
5 load('b10a01_y.mat');
6 load('b10a01_net.mat');

```

### 2.2

```

8 x = -10:0.01:10;
9 W1 = netDropout.Layers(2).Weights;
10 b1 = netDropout.Layers(2).Bias;
11 W2 = netDropout.Layers(5).Weights;
12 b2 = netDropout.Layers(5).Bias;

```

```

13 W3 = netDropout.Layers(8).Weights;
14 b3 = netDropout.Layers(8).Bias;
15
16 Y = zeros(1000, size(x,2));
17
18 for c=1:1000
19     Y1 = leakyRelu(W1 * x + b1);
20     for i=1:size(Y1,1)
21         Y1(i,randperm(size(Y1,2), round(size(Y1,2)*0.1))) = 0;
22     end
23     Y2 = leakyRelu(W2 * Y1 + b2);
24     for i=1:size(Y2,1)
25         Y2(i,randperm(size(Y2,2), round(size(Y2,2)*0.1))) = 0;
26     end
27     Y3 = W3 * Y2 + b3;
28
29     Y(c,:) = Y3;
30 end

```

### 2.3

```

32 means = mean(Y);
33 stds = std(Y);

43 function y= leakyRelu(x)
44     if x>0
45         y = x;
46     else
47         y = 0.01*x;
48     end
49 end

```

### 2.4

```

35 plot(x, yDropout);
36 hold on;
37 plot(x, means);
38 plot(x, means+stds);
39 plot(x, means-stds);

```

## 2.5

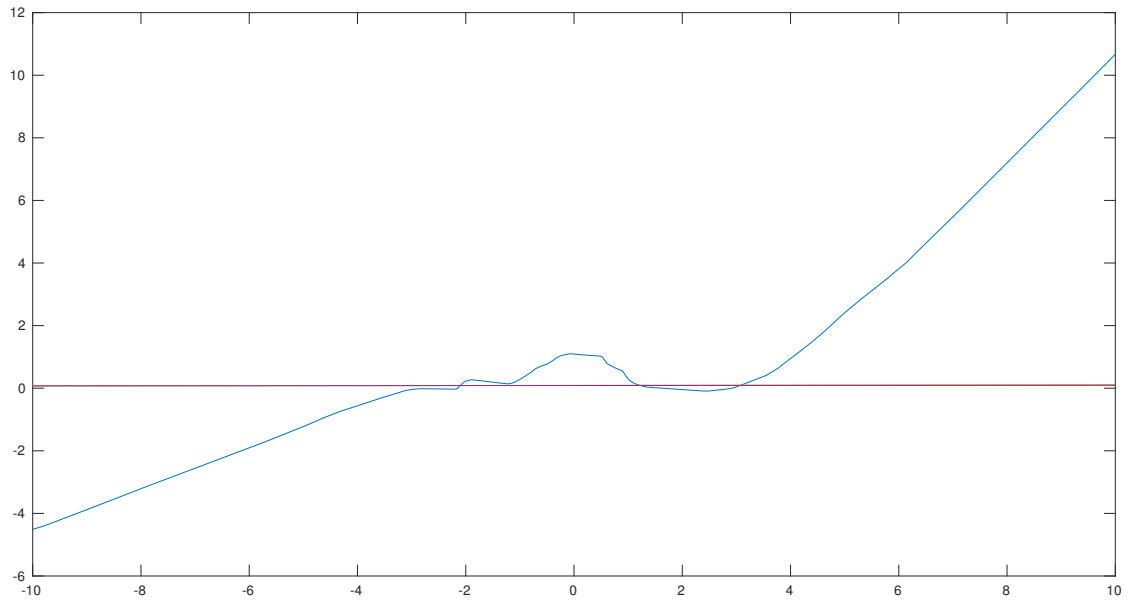


Abbildung 2: Ausgabe des Matlab Scripts

## 3 Radiale Basisfunktionen

### 3.1

$$H = \begin{pmatrix} h(0) & h(1) & h(2) \\ h(1) & h(0) & h(1) \\ 4.001 & h(1) & h(0) \end{pmatrix} = \begin{pmatrix} 0.001 & 1.001 & 4.001 \\ 1.001 & 0.001 & 1.001 \\ 4.001 & 1.001 & 0.001 \end{pmatrix}$$

### 3.2

$$\begin{pmatrix} 0.001 & 1.001 & 4.001 \\ 1.001 & 0.001 & 1.001 \\ 4.001 & 1.001 & 0.001 \end{pmatrix} \begin{pmatrix} 1.752 \\ -8.004 \\ w_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \\ -1 \end{pmatrix}$$
$$\Rightarrow w_3 = 2.252$$

### 3.3

a)

$$g(3) = 1.752 \cdot h(3) - 8.004 \cdot h(2) + 2.252 \cdot h(1) = -14$$

b) Ja, da der Datenpunkt bei  $x = 3$  nicht bekannt ist.

### 3.4

$\alpha = -1.0 \Leftrightarrow$  Grafik B.  $h$  steigt Quadratisch  $\Rightarrow$  Parabelförmig.

$\alpha = -0.3 \Leftrightarrow$  Grafik C. Da  $h(0) \ll 1$ .

$\alpha = 0.15 \Leftrightarrow$  Grafik A. Da  $h(0) \gg 1$ .