Assignment

Practical Assignment #4

Felix Strobel und Gabriel Cevallos

June 2, 2021

Examiner: Prof. Dr. Peter Krzystek

Contents

1 Input Data **6**
- 1.1 Problem 1: Simple network . . . 6
- 1.2 Problem 2: Backpropagation . . . 6
- 1.3 Problem 3: Artificial neural network . . . 7
- 1.4 Problem 4: Gradient Descent . . . 7

2 Methods **9**
- 2.1 Problem 1: Simple network . . . 9
- 2.2 Problem 2: Backpropagation . . . 9
- 2.3 Problem 3: Artificial neural network . . . 9
- 2.4 Problem 4: Gradient Descent . . . 9

3 Results **10**
- 3.1 Problem 1: Simple network . . . 10
- 3.2 Problem 2: Backpropagation . . . 10
- 3.3 Problem 3: Artificial neural network . . . 11
- 3.4 Problem 4: Gradient Descent . . . 11

4 Discussion **12**
- 4.1 Problem 1: Simple network . . . 12
- 4.2 Problem 2: Backpropagation . . . 12
- 4.3 Problem 3: Artificial neural network . . . 12
- 4.4 Problem 4: Gradient Descent . . . 12

5 Sourcecode **13**

6 Bib **14**

List of Figures

1.1 Simple neural network for problem 1 6
1.2 Data provided for problem 1 7
1.3 Single neuron for problem 2 8
1.4 Generated input data of problem 4 8

List of Tables

1 Input Data

1.1 Problem 1: Simple network

In the first problem a small neural network with 4 Layers was provided. In addition

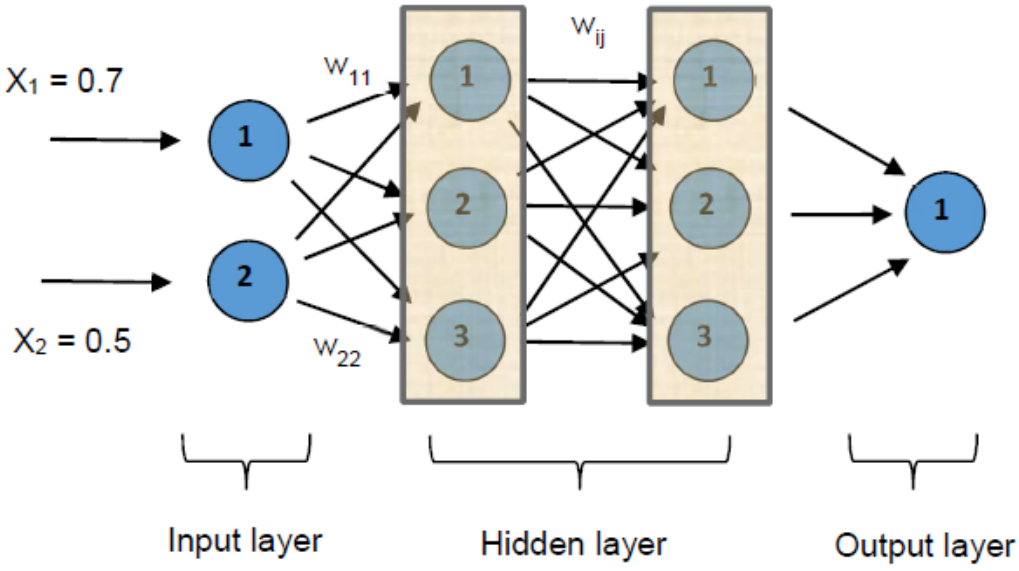


Figure 1.1: Simple neural network for problem 1

the input values and the weights for the connections of the fully connected layers were provided.

1.2 Problem 2: Backpropagation

Similarly to the first problem a small neural network was provided, in this case the network consists only of a single neuron with two inputs and an activation function Z.

Input layer:

$X_1 = 0.7, X_2 = 0.7$

Weights between Input layer – 1st hidden layer:

$W_{11} = 0.9, W_{12} = 0.3, W_{13} = 0.9$
$W_{21} = 0.1, W_{22} = 0.2, W_{23} = 0.4$

Weights between 1st hidden layer – 2nd hidden layer:

$W_{11} = 0.1, W_{12} = 0.8, W_{13} = 0.4$
$W_{21} = 0.5, W_{22} = 0.1, W_{23} = 0.6$
$W_{31} = 0.6, W_{32} = 0.7, W_{33} = 0.3$

Weights between 2nd hidden layer and output layer:

$W_{11} = 0.5, W_{21} = 0.7, W_{31} = 0.3$

Figure 1.2: Data provided for problem 1

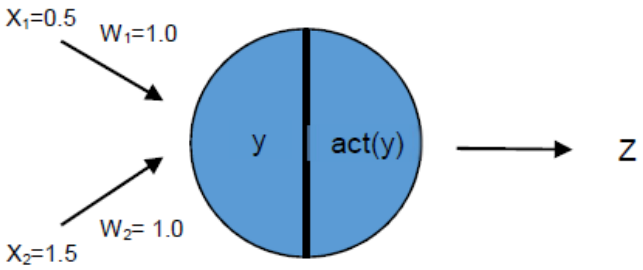
1.3 Problem 3: Artificial neural network

For the third problem a small neural network with 3 Layers was provided in form of a python script using the Pytorch library, in addition to the network a dataset called **One-hundred plant species leaves data set Data Set**. This dataset contains the leaves of 100 different plant species each represented with 16 examples of real leaves, which are themselves represented by a feature vector with 64 different elements representing different features.

1.4 Problem 4: Gradient Descent

The input data set consists of 1000 generated points with a X value between -5 and 5. The Y value is calculated with the formula $y = mx + c$ where m and c are assumed by us. Applying a random noise leads to the dataset which is shown in Figure 1.4

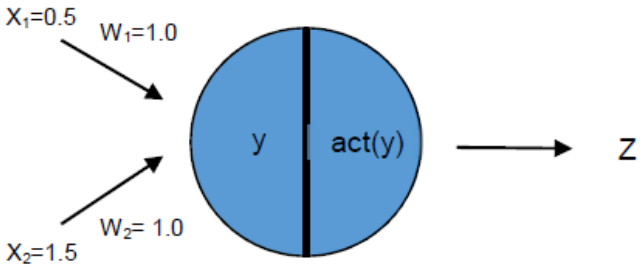
Further a jupyter notebook is provided which generate these data and provides a script which only need to be completed with the calculation of the gradient and updating and weight update for m and c.



$$y = w_1 \cdot X_1 + w_2 \cdot X_2 + b$$

Z = sigmoid(y)
b = 2.0
Z' = 1.0

Figure 1.3: Single neuron for problem 2



$$y = w_1 \cdot X_1 + w_2 \cdot X_2 + b$$

Z = sigmoid(y)
b = 2.0
Z' = 1.0

Figure 1.4: Generated input data of problem 4

2 Methods

2.1 Problem 1: Simple network

The task of problem 1 is to calculate the output of the given network using matrix operations. In order to execute the operations firstly the operation should be defined. It can be formulated as $x_j = f(x_i * w_{ij})$ with x_j being the output of layer j, x_i being the output of layer i, w_{ij} being the weight matrix for the connections between the layers i and j and $f()$ being the activation function of layer j which in our case is the sigmoid function.

2.2 Problem 2: Backpropagation

The task of problem 1 is to calculate the back propagation of the given network (section 1.2) for two different cost functions. This is done by first finding the gradient using the partial derivatives of the forward propagation function, taking advantage of the chain rule this can be done layer by layer from end to beginning. This is use to determine the influence of each single weight and bias to the final error, which are updated accordingly to minimize the error.

2.3 Problem 3: Artificial neural network

In problem 3 both of the tasks are combined and put to use for the training of a simple neural network with the help of the Pytorch library. The best suiting combination of network architecture and hyperparameters have to be found empirically by making assumptions and testing them by comparing the results of the different tests.

2.4 Problem 4: Gradient Descent

3 Results

3.1 Problem 1: Simple network

In order to be able to use our forward propagation function $x_j = f(x_i * w_{ij})$(from section 2.1), the given data (from section 1.1) were formulated as matrices resulting in the following data:

- The input vector $x_0 = \begin{bmatrix} 0.7 & 0.5 \end{bmatrix}$
- The weight matrix $w_01 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
- The weight matrix $w_12 = \begin{bmatrix} 0.9 & 0.3 & 0.9 \\ 0.1 & 0.2 & 0.4 \end{bmatrix}$
- The weight matrix $w_23 = \begin{bmatrix} 0.1 & 0.8 & 0.4 \\ 0.5 & 0.1 & 0.6 \\ 0.6 & 0.7 & 0.3 \end{bmatrix}$
- The weight matrix $w_34 = \begin{bmatrix} 0.5 & 0.7 & 0.3 \end{bmatrix}$

Thus the final equation can be summarized as:
$f(\, f\, (\, f(\, f(x_0 * w_01) * w_12) * w_23) * w_34$

The output of the neural network is 0.7451673339899871.

Alternatively the input vector $x_0 = \begin{bmatrix} 0.7 & 0.5 \end{bmatrix}$ was used, resulting in the value 0.7453676512649436.

3.2 Problem 2: Backpropagation

Using the given equations from section 1.2 the following gradients were calculated for a:

- $\partial sigmoid = sigmoid * (1 - sigmoid)$
- $\frac{\partial cost}{\partial prediction} = -1$
- $\frac{\partial prediction}{\partial y} = \partial sigmoid(y)$

- $\frac{\partial y}{\partial w_1} = x_1$
- $\frac{\partial y}{\partial w_2} = x_2$
- $\frac{\partial y}{\partial b} = 1$
- $\frac{\partial cost}{\partial w_1} = \frac{\partial cost}{\partial prediction} * \frac{\partial prediction}{\partial y} * \frac{\partial y}{\partial w_1}$
- $\frac{\partial cost}{\partial w_2} = \frac{\partial cost}{\partial prediction} * \frac{\partial prediction}{\partial y} * \frac{\partial y}{\partial w_2}$
- $\frac{\partial cost}{\partial b} = \frac{\partial cost}{\partial prediction} * \frac{\partial prediction}{\partial y} * \frac{\partial y}{\partial b}$

This results in the following weights (assuming a learning rate of 1):

- $w1 = w1 - \frac{\partial cost}{\partial w_1} = 1.0088313531066455$
- $w2 = w2 - \frac{\partial cost}{\partial w_2} = 1.0264940593199368$

- $b = b - \frac{\partial cost}{\partial b} = 2.017662706213291$

For part b only the cost function(see section 1.2 b)is different with its derivative being:

- $\frac{\partial cost}{\partial prediction} = -(z' - z)$
-

And giving the following weight values: This results in the following weights (assuming a learning rate of 1):

- $w1 = w1 - \frac{\partial cost}{\partial w_1} = 1.0001588425712256$
- $w2 = w2 - \frac{\partial cost}{\partial w_2} = 1.0004765277136765$

- $b = b - \frac{\partial cost}{\partial b} = 2.000317685142451$

3.3 Problem 3: Artificial neural network

3.4 Problem 4: Gradient Descent

4 Discussion

4.1 Problem 1: Simple network

The concatenation of the results of the define propagation function is very similar to the Pytorch setup of a neural network with the biggest difference being the extra efficiency provided by the Pytorch implementation of the tensor operations with possible access to a GPU if available on the system.

4.2 Problem 2: Backpropagation

When comparing the two error functions given for the problem it becomes evident that the minimum of the nonlinear equation b) is more suitable to fit the model to the data than the linear function a).

4.3 Problem 3: Artificial neural network

As can be seen in the number of neurons of the network it appears that the input features are strongly correlated since they can only be accurately classified by a layer with a number of neurons which is only a fraction of the number of input features.

4.4 Problem 4: Gradient Descent

5 Sourcecode

6 Bib

42 - the answer [1]

Bibliography

[1] Douglas Adams. *The hitchhiker's guide to the galaxy.* Pan Books, London, 2009.