Hand-in 2

Mads-Peter Verner Christiansen, au616397
Zeyuan Tang, au597881
Douwe Tjeerd Schotanus, au600876

October 25, 2018

1 Part I: Derivative

We want to find an expression for the derivative of the cost function

$$L(z) = -\sum_{i=1}^{k} y_i \log(\mathrm{softmax(z)}_i) = -\log(\mathrm{softmax(z)}_j) \tag{1}$$

This function can be written as

$$L(z) = g(f(z)_j) \tag{2}$$

Where

$$g(f) = -\log(f) \tag{3}$$

$$f(z)_j = \mathrm{softmax(z)}_j \tag{4}$$

With these definitions the derivative can be written as

$$\frac{dL}{dz_i} = \frac{dL}{dg}\frac{dg}{df}\frac{df}{dz_i} \tag{5}$$

Each of these derivatives can be solved as follows

$$\frac{dL}{dg} = \frac{d}{dg}(g) = 1 \tag{6}$$

$$\frac{dg}{df} = -\frac{d}{df}\log(f) = -\frac{1}{f} \tag{7}$$

$$\frac{df}{dz_i} = \frac{d}{dz_i}\mathrm{softmax(z)}_j = (\delta_{i,j} - \mathrm{softmax(z)}_i)\mathrm{softmax(z)}_j \tag{8}$$

Combining these terms results in the wanted expression

$$\frac{dL}{dz_i} = -\frac{1}{\mathrm{softmax(z)}_j}(\delta_{i,j} - \mathrm{softmax(z)}_i)\mathrm{softmax(z)}_j = -\delta_{i,j} + \mathrm{softmax(z)}_i \tag{9}$$

2 Part II: Implementation and test

The forward pass and backpropagation code has been implemented as follows

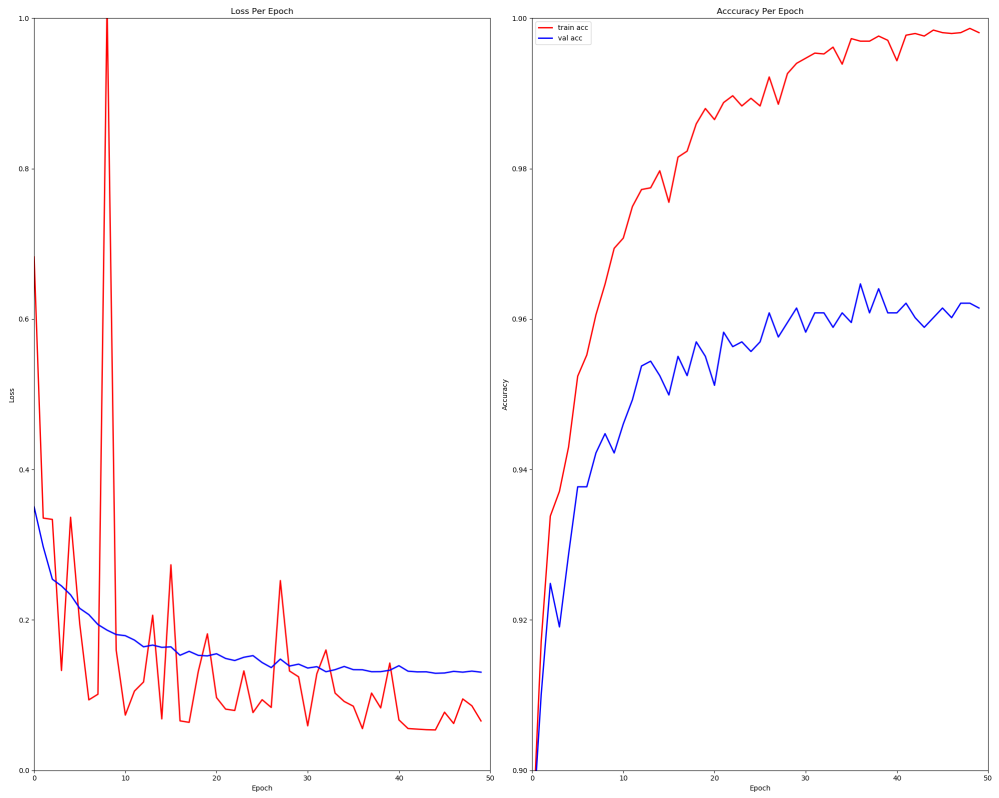
```
a1 = X @ W1 + b1
h1 = relu(a1)
a2 = h1 @ W2 + b2
C = np.sum(np.sum(-labels*np.log(softmax(a2)), axis=1))
```

Listing 1: Forward pass

```
g2 = -labels + softmax(a2)
d_b2 += np.sum(g2, axis=0)
d_w2 += h1.T @ g2
g1 = (W2 @ g2.T).T * (a1 > 0)
d_b1 += np.sum(g1, axis=0)
d_w1 += X.T @ g1
```

Listing 2: Backpropagation

The cost and the derivatives are divided by the number of training examples at the end of the calculation where the weight decay regularization is also added. The plot below shows the cost and the accuracy as a function of epoch for a run on the digits data set.


Loss Per Epoch
Loss
Epoch
Acccuracy Per Epoch
train acc
val acc
Accuracy
Epoch