CS-E4850 Computer Vision, Answers to Exercise Round 9

Yangzhe Kong, Student number: 765756

November 14, 2019

Exercise 1. Neural networks and backpropagation.

1)

$$E(t, y) = \frac{1}{n_2} \sum_{i}^{n_2} -t_i \, log(y_i)$$

2) Let's take a look at each individual z_i

$$E(z_i^{(2)}) = -t_i \, log(y_i))$$

Then

$$\begin{aligned}
\frac{\partial E}{\partial z_i^{(2)}} &= -\sum_{j=1}^{n_2} t_j \frac{\partial log(y_j)}{\partial z_i} \\
&= -\sum_{j=1}^{n_2} t_j \frac{1}{y_j} \frac{\partial y_j}{\partial z_i} \\
&= -\frac{t_i}{y_i} \frac{\partial y_i}{\partial z_i} - \sum_{j \neq i}^{n_2} \frac{t_j}{y_j} \frac{\partial y_j}{\partial z_i} \\
&= -\frac{t_i}{y_i} y_i(1 - y_i) - \sum_{j \neq i}^{n_2} \frac{t_j}{y_j}(-y_i y_j) \\
&= -t_i + t_i y_i + \sum_{j \neq i}^{n_2} t_j y_i \\
&= -t_i + \sum_{j=1}^{n_2} t_j y_i \\
&= -t_i + y_i \sum_{j=1}^{n_2} t_j \\
&= y_i - t_i
\end{aligned}$$

Thus for the vector $\mathbf{z^{(2)}}$, we have:

$$\frac{\partial E}{\partial \mathbf{z^{(2)}}} = \mathbf{y_i} - \mathbf{t_i}$$

3) Still we will prove the formula in the case of individual $y_i^{(1)}$

$$\begin{aligned}
\frac{\partial E}{\partial y_i^{(1)}} &= \frac{\partial E(t, s(z_i^{(2)})}{\partial y_i^{(1)}} \\
&= \frac{\partial E(t, s(y_i^{(1)}\mathbf{W}_i^{(2)})}{\partial y_i^{(1)}} \\
&= \frac{\partial E(t, s(z_i))}{\partial z_i} \frac{\partial y_i^{(1)}\mathbf{W}_i^{(2)}}{\partial y_i^{(1)}} \\
&= (y_i^{(2)} - t_i)^\top \frac{\partial y_i^{(1)}\mathbf{W}_i^{(2)}}{\partial y_i^{(1)}} \qquad \text{(Chain Rule Applied)} \\
&= (y_i^{(2)} - t_i)^\top \mathbf{W}_i^{(2)}
\end{aligned}$$

Note that we transpose the first term simply because the derivative of matrix should have the same size of the matrix itself (can also be viewed as to operate the matrix multiplication correctly). This trick also applies to the following formulas.

4)

$$\begin{aligned}
\frac{\partial E(t, y_u^{(2)})}{\partial w_{uv}^{(2)}} &= \frac{\partial E(t, s(z_i))}{\partial z_i} \frac{\partial y_v^{(1)} w_{uv}^{(2)}}{\partial w_{uv}^{(2)}} \\
&= (y_u^{(2)} - t_u) \frac{\partial y_v^{(1)} w_{uv}^{(2)}}{\partial w_{uv}^{(2)}} \\
&= (y_u^{(2)} - t_u) y_v^{(1)}
\end{aligned}$$

If we write all the results using the formula we derived just now with respect to u and v in a matrix $\nabla \mathbf{W}^2$, we can derive:

$$\begin{aligned}
\frac{\partial E}{\partial \mathbf{W}} &= \frac{\partial E(t, s(\mathbf{y^{(1)}}\mathbf{W}))}{\partial \mathbf{W}} \\
&= \frac{\partial E(t, s(\sum_{v=0}^{n^2} \sum_{u=0}^{n^1} y_i^{(1)} w_{uv}^{(2)}))}{\partial w_{uv}} \\
&= (\mathbf{y^{(2)}} - \mathbf{t})\mathbf{y^{(1)\top}}
\end{aligned}$$

5) Since,

$$\frac{\partial \sigma(z)}{\partial z} = \partial(z)(1 - \partial(z))$$

We can get:

$$\frac{\partial y^{(1)}}{\partial z^{(1)}} = \frac{\partial \sigma(z^{(1)})}{\partial z^{(1)}} = \sigma(z^{(1)})(1-\sigma(z^{(1)})) = y^{(1)}(1-y^{(1)})$$

This can obviously be written as a diagonal matrix as proposed.

6)

$$\begin{aligned}
\frac{\partial E}{\partial \mathbf{z^{(1)}}} &= \frac{\partial E(t, s(\mathbf{W^{(2)}}\sigma(\mathbf{z^{(1)}})))}{\partial \mathbf{z^{(1)}}} \\
&= \frac{\partial E(t, s(\mathbf{W^{(2)}}\sigma(\mathbf{z^{(1)}})))}{\partial \mathbf{z^{(2)}}} \frac{\partial \mathbf{z^{(2)}}}{\partial \mathbf{y^{(1)}}} \frac{\partial \mathbf{y^{(1)}}}{\partial \mathbf{z^{(1)}}} \\
&= (\mathbf{y^{(2)}} - \mathbf{t})^\top \mathbf{W^{(2)}} \mathrm{diag}(\mathbf{y^{(1)}}(\mathbf{1} - \mathbf{y^{(1)}}))
\end{aligned}$$

7)

$$\begin{aligned}
\frac{\partial E}{\partial w_{uv}^{(1)}} &= \frac{\partial E}{\partial z_u^{(1)}} \frac{\partial W_u^{(1)} x_v}{\partial w_{uv}^{(1)}} \\
&= \frac{\partial E}{\partial z_u^{(1)}} x_v
\end{aligned}$$

Again we can write the results in the matrix $\nabla \mathbf{W}^{(1)}$

$$\frac{\partial E}{\partial \mathbf{W^{(1)}}} = \left(\frac{\partial E}{\partial \mathbf{z}^{(1)}}\right)^\top \mathbf{x}^\top$$

8) Average the gradients:

$$\frac{\partial E}{\partial \mathbf{W^{(2)}}} = \frac{1}{m} \sum_i^m \frac{\partial E_i}{\partial \mathbf{W^{(2)}}}$$

$$\frac{\partial E}{\partial \mathbf{W^{(1)}}} = \frac{1}{m} \sum_i^m \frac{\partial E_i}{\partial \mathbf{W^{(1)}}}$$

9) $\frac{\partial \frac{\lambda}{2} \| \mathbf{w}\|^2}{\partial \mathbf{w}} = \lambda \mathbf{w}$

Exercise 2. Image classification using a neural network

The code in the function **d_loss_by_d_model** goes as follows: [frame=single]

```
ret.input_to_hid = (((model.hid_to_class' * (class_prob - data.targets)) .* hid_output .* (1 - hid_output)) * data.inputs') / size(data.targets, 2);
ret.hid_to_class=((class_prob-data.targets)*hid_output')/size(data.targets, 2);
```

And here's the resulting training data classification loss:
The total loss on the training data is 2.301907
The classification loss (i.e. without weight decay) on the training data is 2.301907
The classification error rate on the training data is 0.889000
And the plot of training loss and validation loss during training is shown in Figure 1

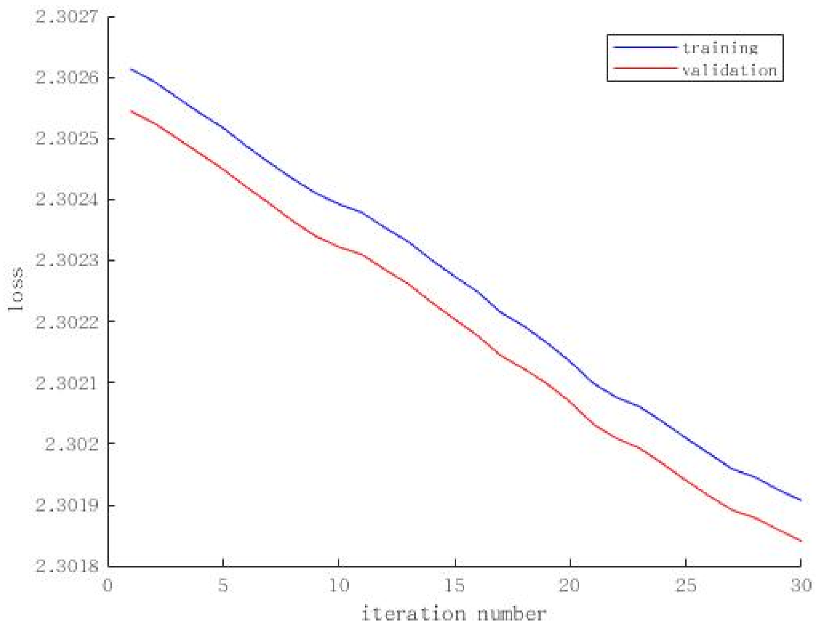

Figure 1: Training loss and validation loss during training

Exercise 3. Optimisation using backpropagation

We first try with $a2(0, 10, 70, 0.005, 0, false, 4)$, and the resulting training data classification loss is:
The total loss on the training data is 2.301771
The classification loss (i.e. without weight decay) on the training data is 2.301771
The classification error rate on the training data is 0.888000
And the plot of training loss and validation loss during training is shown in Figure 2

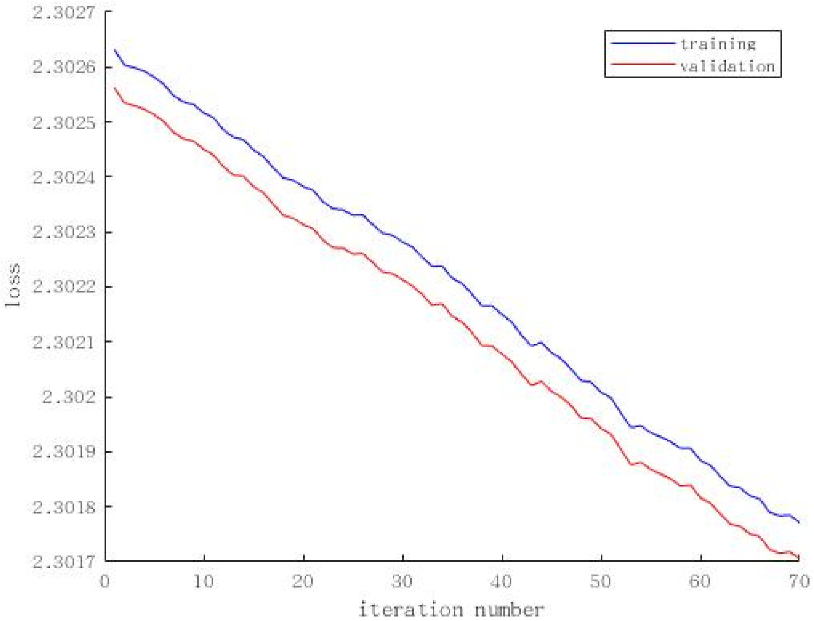

Figure 2: Training loss and validation loss during training

We can see that training data loss and validation data loss are both decreasing, but they're still going down steadily after those 70 optimization iterations. We can try with a larger learning rate or with different momentum. And the setting for the experiment goes like this: we try different learning rates 0.002, 0.01, 0.05, 0.2, 1.0, 5.0, and 20.0. We'll try all of those both without momentum (i.e. momentum=0.0 in the program) and with momentum (i.e. momentum=0.9 in the program), so we have a total of 7 x 2 = 14 experiments to run. And here are the result:

The learning rate for this experiment is 0.002 and momentum 0

The total loss on the training data is 2.302293
The classification loss (i.e. without weight decay) on the training data is 2.302293
The classification error rate on the training data is 0.897000

The total loss on the validation data is 2.302226
The classification loss (i.e. without weight decay) on the validation data is 2.302226
The classification error rate on the validation data is 0.898000

The total loss on the test data is 2.302253
The classification loss (i.e. without weight decay) on the test data is 2.302253
The classification error rate on the test data is 0.895000

The learning rate for this experiment is 0.002 and momentum 0.9

The total loss on the training data is 2.299567
The classification loss (i.e. without weight decay) on the training data is 2.299567
The classification error rate on the training data is 0.848000

The total loss on the validation data is 2.299502
The classification loss (i.e. without weight decay) on the validation data is 2.299502
The classification error rate on the validation data is 0.839000

The total loss on the test data is 2.299510
The classification loss (i.e. without weight decay) on the test data is 2.299510
The classification error rate on the test data is 0.850778

The learning rate for this experiment is 0.01 and momentum 0

The total loss on the training data is 2.300901
The classification loss (i.e. without weight decay) on the training data is 2.300901
The classification error rate on the training data is 0.886000

The total loss on the validation data is 2.300839
The classification loss (i.e. without weight decay) on the validation data is 2.300839
The classification error rate on the validation data is 0.887000

The total loss on the test data is 2.300854
The classification loss (i.e. without weight decay) on the test data is 2.300854
The classification error rate on the test data is 0.881667

The learning rate for this experiment is 0.01 and momentum 0.9

The total loss on the training data is 2.285062
The classification loss (i.e. without weight decay) on the training data is 2.285062
The classification error rate on the training data is 0.751000

The total loss on the validation data is 2.285036
The classification loss (i.e. without weight decay) on the validation data is 2.285036
The classification error rate on the validation data is 0.756000

The total loss on the test data is 2.284913
The classification loss (i.e. without weight decay) on the test data is 2.284913
The classification error rate on the test data is 0.761667

The learning rate for this experiment is 0.05 and momentum 0

The total loss on the training data is 2.293412
The classification loss (i.e. without weight decay) on the training data is 2.293412
The classification error rate on the training data is 0.765000

The total loss on the validation data is 2.293384
The classification loss (i.e. without weight decay) on the validation data is 2.293384
The classification error rate on the validation data is 0.766000

The total loss on the test data is 2.293329
The classification loss (i.e. without weight decay) on the test data is 2.293329
The classification error rate on the test data is 0.757667

The learning rate for this experiment is 0.05 and momentum 0.9

The total loss on the training data is 1.994052
The classification loss (i.e. without weight decay) on the training data is 1.994052
The classification error rate on the training data is 0.708000

The total loss on the validation data is 1.996559
The classification loss (i.e. without weight decay) on the validation data is 1.996559
The classification error rate on the validation data is 0.700000

The total loss on the test data is 1.989121
The classification loss (i.e. without weight decay) on the test data is 1.989121
The classification error rate on the test data is 0.699889

The learning rate for this experiment is 0.2 and momentum 0

The total loss on the training data is 2.223581
The classification loss (i.e. without weight decay) on the training data is 2.223581
The classification error rate on the training data is 0.801000

The total loss on the validation data is 2.224201
The classification loss (i.e. without weight decay) on the validation data is 2.224201
The classification error rate on the validation data is 0.797000

The total loss on the test data is 2.222849
The classification loss (i.e. without weight decay) on the test data is 2.222849
The classification error rate on the test data is 0.795667

The learning rate for this experiment is 0.2 and momentum 0.9

The total loss on the training data is 1.292496
The classification loss (i.e. without weight decay) on the training data is 1.292496
The classification error rate on the training data is 0.409000

The total loss on the validation data is 1.337265
The classification loss (i.e. without weight decay) on the validation data is 1.337265
The classification error rate on the validation data is 0.423000

The total loss on the test data is 1.301506
The classification loss (i.e. without weight decay) on the test data is 1.301506
The classification error rate on the test data is 0.407000

The learning rate for this experiment is 1 and momentum 0

The total loss on the training data is 1.674933
The classification loss (i.e. without weight decay) on the training data is 1.674933
The classification error rate on the training data is 0.697000

The total loss on the validation data is 1.690025
The classification loss (i.e. without weight decay) on the validation data is 1.690025
The classification error rate on the validation data is 0.714000

The total loss on the test data is 1.676163
The classification loss (i.e. without weight decay) on the test data is 1.676163
The classification error rate on the test data is 0.703222

The learning rate for this experiment is 1 and momentum 0.9

The total loss on the training data is 1.877051
The classification loss (i.e. without weight decay) on the training data is 1.877051
The classification error rate on the training data is 0.721000

The total loss on the validation data is 1.920913
The classification loss (i.e. without weight decay) on the validation data is 1.920913
The classification error rate on the validation data is 0.737000

The total loss on the test data is 1.897451
The classification loss (i.e. without weight decay) on the test data is 1.897451
The classification error rate on the test data is 0.724222

The learning rate for this experiment is 5 and momentum 0

The total loss on the training data is 2.301619
The classification loss (i.e. without weight decay) on the training data is 2.301619
The classification error rate on the training data is 0.899000

The total loss on the validation data is 2.302043
The classification loss (i.e. without weight decay) on the validation data is 2.302043
The classification error rate on the validation data is 0.900000

total loss on the test data is 2.302443
The classification loss (i.e. without weight decay) on the test data is 2.302443
The classification error rate on the test data is 0.899333

The learning rate for this experiment is 5 and momentum 0.9

The total loss on the training data is 2.302585
The classification loss (i.e. without weight decay) on the training data is 2.302585
The classification error rate on the training data is 0.886000

The total loss on the validation data is 2.302585
The classification loss (i.e. without weight decay) on the validation data is 2.302585
The classification error rate on the validation data is 0.897000

The total loss on the test data is 2.302585
The classification loss (i.e. without weight decay) on the test data is 2.302585
The classification error rate on the test data is 0.887778

The learning rate for this experiment is 20 and momentum 0

The total loss on the training data is 2.302585
The classification loss (i.e. without weight decay) on the training data is 2.302585
The classification error rate on the training data is 0.900000

The total loss on the validation data is 2.302585
The classification loss (i.e. without weight decay) on the validation data is 2.302585
The classification error rate on the validation data is 0.900000

The total loss on the test data is 2.302585
The classification loss (i.e. without weight decay) on the test data is 2.302585
The classification error rate on the test data is 0.900000

The learning rate for this experiment is 20 and momentum 0.9

The total loss on the training data is 2.302585
The classification loss (i.e. without weight decay) on the training data is 2.302585
The classification error rate on the training data is 0.874000

The total loss on the validation data is 2.302585
The classification loss (i.e. without weight decay) on the validation data is 2.302585
The classification error rate on the validation data is 0.882000

The total loss on the test data is 2.302585
The classification loss (i.e. without weight decay) on the test data is 2.302585
The classification error rate on the test data is 0.877889

And all training set and validation set results are shown in the following table:

	Without momentum				With momentum 0.9			
Learning rate	training loss	training error	validation loss	validation error	training loss	training error	validation loss	validation error
0.002	2.302	0.897	2.302	0.898	2.300	0.848	2.300	0.839
0.01	2.301	0.886	2.301	0.887	2.285	0.751	2.285	0.756
0.05	2.293	0.765	2.293	0.766	1.994	0.708	1.997	0.700
0.2	2.224	0.801	2.224	0.797	1.292	0.409	1.337	0.423
1.0	1.675	0.697	1.690	0.714	1.877	0.721	1.921	0.737
5.0	2.302	0.899	2.302	0.900	2.303	0.886	2.303	0.897
20.0	2.303	0.900	2.303	0.900	2.303	0.874	2.303	0.882

We can observe from the results that if we choose a reasonable learning rate (0.2) and implement with momentum, we can achieve a overall better result. If we use extreme small or big learning rate, the performance will be worse. But one thing worth noting is that although small learning rate may perform worse compared with larger learning rate under the same number of iterations, if we train for larger iterations, small learning rate may still able to converge. Thus, number of training iterations is also an important parameter. So, in practice even if we build a very good model, we still have to tune our hyperparameters to train it efficiently to achieve best performance.

Finally we can train our model for larger iterations and see what we can get with $a2(0, 10, 200, 0.2, 0.9, false, 4)$. The plot is shown in Figure 3

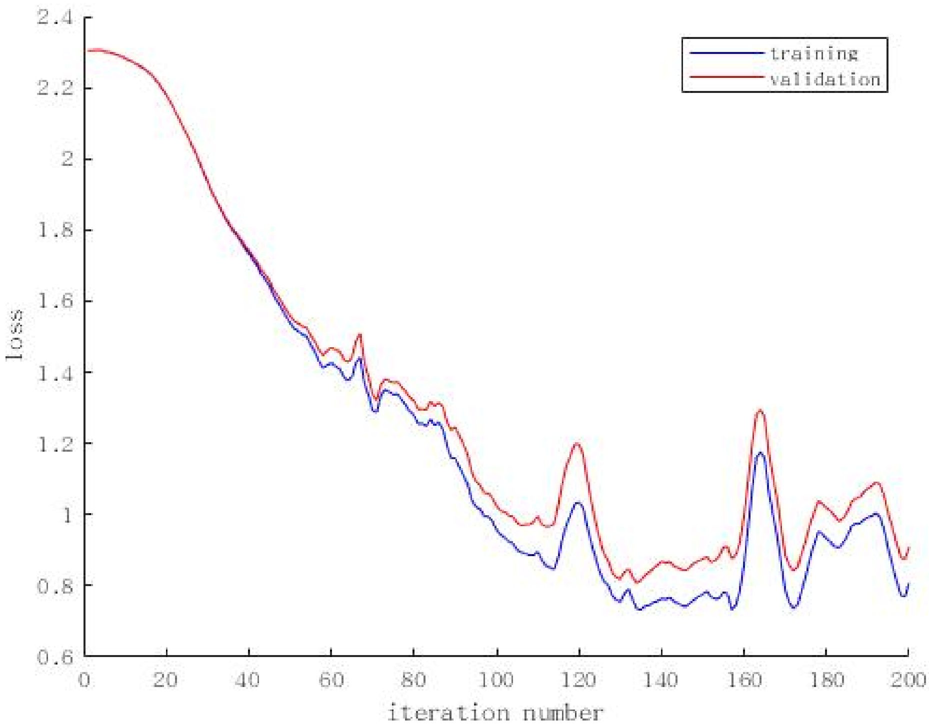

Figure 3: The training loss and validation loss during training with $a2(0, 10, 200, 0.2, 0.9, false, 4)$

We can see that at iteration 140 the model achieves best performance.