Deep Learning 2018/19 - Assignment 1

Abdul Faiz Punakkath, Marco Concetto Rudilosso, Matineh Akhlaghinia,
Matteo Cassia, Reemma Muthal Puredath

November 25, 2018

Exercise 1

In this exercise, σ_n is used to express $\sigma\left(\mathbf{w}^T\mathbf{x}_n\right)$.

Exercise 1.1

In this exercise it is required to determine the expression for the negative log likelihood for N pairs of data $(x_n, c_n), c_n \in 0, 1$. It is also reminded that the likelihood loss is $H(p_n, c_n) = -\mathbb{E}[\log p_n]_{c_n}$. Finally, it is known that p_n is in state 1 with probability σ_n and since the setting is binary, it is in state 0 with probability $1 - \sigma_n$.

$$\begin{aligned} E_{lik}(\mathbf{w}) &= H(p_n, c_n) \\ &= -\mathbb{E}[\log p_n]_{c_n} \\ &= -\sum_{n=1}^{N} c_n \log p_n \\ &= -\sum_{n=1}^{N} \begin{cases} \log(\sigma_n) & c_n = 1 \\ \log(1-\sigma_n) & c_n = 0 \end{cases} \\ &= -\sum_{n=1}^{N} \mathbb{I}[c_n = 1]\log(\sigma_n) + \mathbb{I}[c_n = 0]\log(1-\sigma_n) \end{aligned} \tag{1}$$

The operator $\mathbb{I}[\upsilon]$ is the indicator function: it is equal to 1 if the condition υ specified is true, 0 otherwise. However, the scaled loss is required, in order to normalise it for the number of pairs.

$$E_{lik}(\mathbf{w}) = -\frac{1}{N}\sum_{n=1}^{N} \mathbb{I}[c_n = 1]\log(\sigma_n) + \mathbb{I}[c_n = 0]\log(1-\sigma_n) \tag{2}$$

Exercise 1.2

The question requires to determine the convexity of E_{lik} with respect to $\mathbf{w}$. One way to do that is to determine whether the second derivative is ever negative.

$$E_{lik}(\mathbf{w}) = -\frac{1}{N}\sum_{n=1}^{n} \mathbb{I}[c_n = 1]\log\sigma_n + \mathbb{I}[c_n = 0]\log(1-\sigma_n) \tag{3}$$

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{w}} &= -\frac{1}{N}\sum_{n=1}^{n} \mathbb{I}[c_n = 1]\frac{x_n\sigma_n(1-\sigma_n)}{\sigma_n} - \mathbb{I}[c_n = 0]\frac{x_n\sigma_n(1-\sigma_n)}{1-\sigma_n} \\
&= -\frac{1}{N}\sum_{n=1}^{n} \mathbb{I}[c_n = 1]x_n(1-\sigma_n) - \mathbb{I}[c_n = 0]x_n\sigma_n \\
&= -\frac{1}{N}\sum_{n=1}^{n} x_n\left(\mathbb{I}[c_n = 1](1-\sigma_n) - \mathbb{I}[c_n = 0]\sigma_n\right) \\
&= -\frac{1}{N}\sum_{n=1}^{n} x_n(c_n - \sigma_n) \\
&= -\frac{1}{N}\sum_{n=1}^{n} x_n c_n - x_n\sigma_n \\
\frac{\partial^2 E}{\partial \mathbf{w}^2} &= -\frac{1}{N}\sum_{n=1}^{n} 0 - x_n x_n \sigma_n(1-\sigma_n) \\
&= \frac{1}{N}\sum_{n=1}^{n} x_n^2\sigma_n(1-\sigma_n)
\end{aligned}
\tag{4}
\tag{5}
$$

It can be observed that both x_n^2 and $\sigma_n(1-\sigma_n)$ are never negative, which indicates that E_{lik} is convex with respect to $\mathbf{w}$.

Exercise 1.3

The question requires to determine the convexity of E_{sq} with respect to $\mathbf{w}$. One way to do that is to determine whether the second derivative is ever negative.

$$
E_{sq}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{n} (c_n - \sigma_n)^2 \tag{6}
$$

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{w}} &= \frac{1}{N}\sum_{n=1}^{N} 2(c_n - \sigma_n)(-1)\sigma_n(1-\sigma_n)x_n \\
&= -\frac{2}{N}\sum_{n=1}^{N} (c_n - \sigma_n)\sigma_n(1-\sigma_n)x_n \\
&= -\frac{2}{N}\sum_{n=1}^{N} x_n c_n\sigma_n - \sigma_n^2 x_n - c_n\sigma_n^2 x_n + \sigma_n^3 x_n
\end{aligned}
\tag{7}
$$

$$
\begin{aligned}
\frac{\partial^2 E}{\partial \mathbf{w}^2} &= -\frac{2}{N}\sum_{n=1}^{N} x_n c_n\sigma_n(1-\sigma_n)x_n - 2\sigma_n\sigma_n(1-\sigma_n)x_n x_n - c_n x_n 2\sigma_n\sigma_n(1-\sigma_n)x_n + 3\sigma_n^2\sigma_n(1-\sigma_n)x_n x_n \\
&= -\frac{2}{N}\sum_{n=1}^{N} c_n x_n^2\sigma_n(1-\sigma_n) - 2x_n^2\sigma_n^2(1-\sigma_n) - 2c_n x_n^2\sigma_n^2(1-\sigma_n) + 3x_n^2\sigma_n^3(1-\sigma_n) \\
&= -\frac{2}{N}\sum_{n=1}^{N} x_n^2\sigma_n(1-\sigma_n)(c_n - 2\sigma_n - 2c_n\sigma_n + 3\sigma_n^2) \\
&= \frac{2}{N}\sum_{n=1}^{N} x_n^2\sigma_n(1-\sigma_n)(2\sigma_n + 2c_n\sigma_n - 3\sigma_n^2 - c_n)
\end{aligned}
$$

(8)

The factor of interest to determine convexity is then $(2\sigma_n + 2c_n\sigma_n - 3\sigma_n^2 - c_n)$.

For $c_n = 0$:

$$\begin{aligned} 2\sigma_n - 3\sigma_n^2 &< 0 \\ 3\sigma_n^2 - 2\sigma_n &> 0 \\ \sigma_n(3\sigma_n - 2) &> 0 \\ 3\sigma_n - 2 &> 0 \\ \sigma_n &> \frac{2}{3} \end{aligned} \tag{9}$$

For $c_n = 1$:

$$\begin{aligned} 2\sigma_n + 2\sigma_n - 3\sigma_n^2 - 1 &< 0 \\ 3\sigma_n^2 - 4\sigma_n + 1 &> 0 \\ \frac{1}{3} < \sigma_n &< 1 \end{aligned} \tag{10}$$

This determines that the second derivative of E_{sq} with respect to $\mathbf{w}$ is negative when $c_n = 0 \wedge \sigma_n \in [\frac{2}{3}, 1]$ and $c_n = 1 \wedge \sigma_n \in [\frac{1}{3}, 1]$. Due to this constraints, it cannot be convex.

Exercise 1.4

The sigmoid function is defined as

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{11}$$

An alternative form is

$$\sigma(x) = \frac{1}{1 + e^{-x}} = (1 + e^{-x})^{-1} = \left(1 + \frac{1}{e^x}\right)^{-1} = \frac{e^x}{1 + e^x} \tag{12}$$

The question asks to determine why using equation 11 for $x > 0$ and 12 for $x \leq 0$ is ideal to prevent numerical overflow.
On digital devices numerical overflow occurs when a computation for a value exceeds the maximum representable number for the system architecture: on a 32-bit machine, the greatest unsigned integer is $2^{32} - 1$, so when 1 is added to it, overflow will happen causing the value to be equal to 0. This is common when computation of great values are performed (such as e^x).
If equation 11 is used for large negative numbers, overflow is likely as the result of e^{-x} will be high, but it will be rather small (between 0 and 1) for large positive values, hence avoiding overflow. Similarly, when equation 12 is used for large positive values, the result of e^x will be high, whereas for large negatives it will not, preventing overflow.

Exercise 1.5

The logarithm of the sigmoid function is defined as

$$\log(\sigma(x)) = \log\left(\frac{1}{1 + e^{-x}}\right) = \log(1) - \log\left(1 + e^{-x}\right) = -\log\left(1 + e^{-x}\right) \tag{13}$$

It can alternatively be written as

$$-\log\left(1 + e^{-x}\right) = -\log\left(\frac{1 + e^x}{e^x}\right) = x - \log(1 + e^x) \tag{14}$$

Similarly to che conclusions drawn in the previous question, it is ideal to use equation 13 for positive values and equation 14 for negative values, as that would prevent e^{-x} and e^x to become too high and cause overflow.

Exercise 1.6 and 1.7

In this exercises, 1-dimensional and 2-dimensional slices of the error surface of the LogLoss and SquareLoss are plotted to observe whether they are convex or else. While it does not guarantee a definite answer, if a plot shows a function is not convex for the selected interval, then it is not convex. Results are observed in figure 1 and 2.

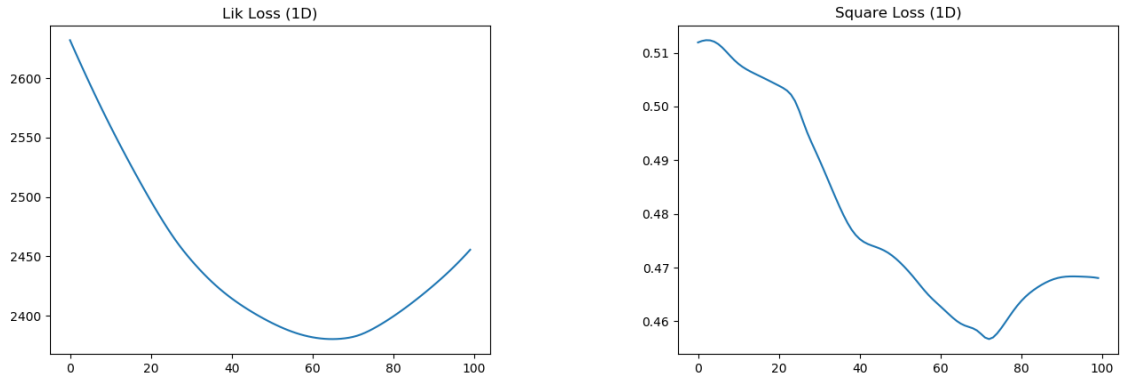

Figure 1: Slices of the 1-dimensional error surface of LikLoss and SquareLoss

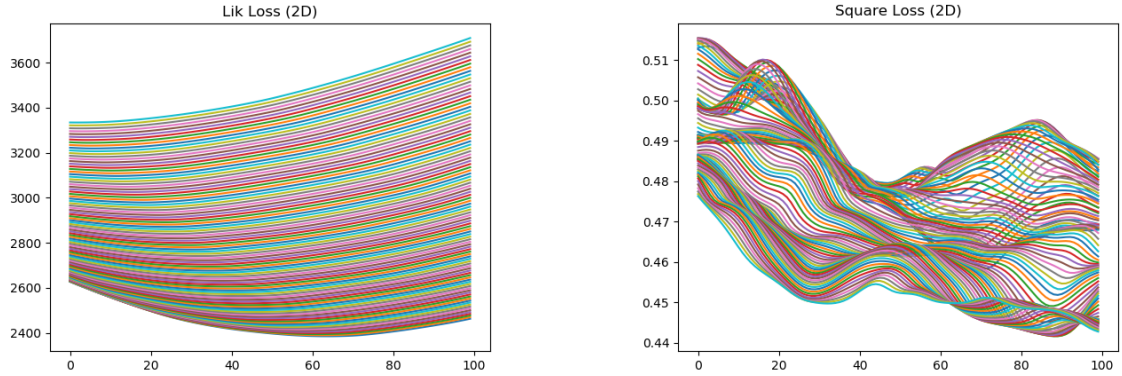

Figure 2: Slices of the 2-dimensional error surface of LikLoss and SquareLoss

For exercise 1.6, the supplied Julia code was used without any change, whereas in exercise 1.7 slight changes have been introduced to account for the additional dimension. The code in the appendix is the overall change to the given Julia to plot the four plots.

Exercise 2

Exercise 2.1

The following loss function is given

$$E = \sum_i \left(y_i - \sum_j W_{ij} x_j \right)^2 \tag{15}$$

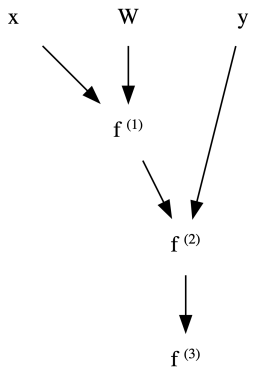

Figure 3: The computation graph of the loss function in Exercise 2.1

and the exercise requires do compute the derivative

$$\frac{\partial E}{\partial W_{ab}} \tag{16}$$

The computation graph is drawn (figure 3) and the functions are determined

$$f_i^{(1)} = \sum_j W_{ij} x_j \tag{17}$$

$$f_i^{(2)} = y_i - f_i^{(1)} \tag{18}$$

$$f^{(3)} = \sum_i \left(f_i^{(2)} \right)^2 \tag{19}$$

The required partial derivatives are calculated.

$$\begin{aligned} \frac{\partial f_i^{(1)}}{\partial W_{ab}} &= \frac{\partial}{\partial W_{ab}} \sum_j W_{ij} x_j \\ &= \frac{\partial}{\partial W_{ab}} \left(W_{i0} x_0 + W_{i1} x_1 + \cdots + W_{ib} x_b + \cdots + W_{in} x_n \right) \\ &= \delta_{ia} x_b \end{aligned} \tag{20}$$

$$\begin{aligned} \frac{\partial f_i^{(2)}}{\partial f_j^{(1)}} &= \frac{\partial}{\partial f_j^{(1)}} \left(y_i - f_i^{(1)} \right) \\ &= -\delta_{ij} \end{aligned} \tag{21}$$

$$\begin{aligned} \frac{\partial f^{(3)}}{\partial f_i^{(2)}} &= \frac{\partial}{\partial f_i^{(2)}} \sum_i \left(f_i^{(2)} \right)^2 \\ &= \frac{\partial}{\partial f_i^{(2)}} \left(2 f_0^{(2)} + 2 f_1^{(2)} + \cdots + 2 f_i^{(2)} + \cdots + 2 f_n^{(2)} \right) \\ &= 2 f_i^{(2)} \end{aligned} \tag{22}$$

The following reverse propagation schedules are defined.

$$t^{(3)} = 1 \tag{23}$$

$$t_i^{(2)} = \frac{\partial f^{(3)}}{\partial f_i^{(2)}} t^{(3)} \tag{24}$$

$$t_i^{(1)} = \sum_j \frac{\partial f_j^{(2)}}{\partial f_i^{(1)}} t_j^{(2)} \tag{25}$$

$$t_{ab}^{W} = \sum_i \frac{\partial f_i^{(1)}}{\partial W_{ab}} t_i^{(1)} \tag{26}$$

From which the required derivative can be calculated.

$$\begin{aligned}
\frac{\partial E}{\partial W_{ab}} &= t_{ab}^{W} \\
&= \sum_i \frac{\partial f_i^{(1)}}{\partial W_{ab}} t_i^{(1)} \\
&= \sum_i \delta_{ia} x_b t_i^{(1)} \\
&= x_b t_a^{(1)} \\
&= x_b \sum_j \frac{\partial f_j^{(2)}}{\partial f_a^{(1)}} t_j^{(2)} \\
&= x_b \sum_j -\delta_{ja} t_j^{(2)} \\
&= -x_b t_a^{(2)} \\
&= -x_b \frac{\partial f^{(3)}}{\partial f_a^{(2)}} t^{(3)} \\
&= -2 x_b f_a^{(2)} \\
&= -2 x_b \left(y_a - f_a^{(1)} \right) \\
&= -2 x_b \left(y_a - \sum_i W_{ai} x_i \right)
\end{aligned} \tag{27}$$

Exercise 2.2

An extended loss function (compared to the previous exercise) is given.

$$E = \sum_i \left(y_i - \phi \left(\sum_j W_{ij} x_j \right) \right)^2 \tag{28}$$

and the exercise requires do compute the derivative

$$\frac{\partial E}{\partial W_{ab}} \tag{29}$$

The computation graph is drawn (figure 4) the functions are derived

$$f_i^{(1)} = \sum_j W_{ij} x_j \tag{30}$$

$$f_i^{(2)} = \phi(f_i^{(1)}) \tag{31}$$

$$f_i^{(3)} = y_i - f_i^{(2)} \tag{32}$$

$$f^{(4)} = \sum_i \left(f_i^{(2)} \right)^2 \tag{33}$$

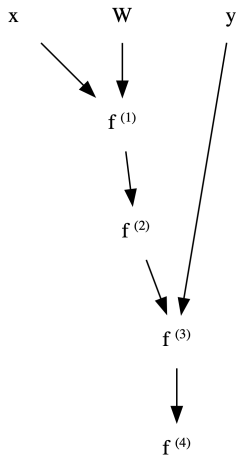

Figure 4: The computation graph of the loss function in Exercise 2.2

The required partial derivatives are calculated.

$$\frac{\partial f_i^{(1)}}{\partial W_{ab}} = \delta_{ia} x_b \tag{34}$$

$$\frac{\partial f_i^{(2)}}{\partial f_j^{(1)}} = \phi'(f_j^{(1)})\delta_{ij} \tag{35}$$

$$\frac{\partial f_i^{(3)}}{\partial f_j^{(2)}} = -\delta_{ij} \tag{36}$$

$$\frac{\partial f^{(4)}}{\partial f_i^{(3)}} = 2 f_i^{(3)} \tag{37}$$

The following reverse propagation schedules are defined.

$$t^{(4)} = 1 \tag{38}$$

$$t_i^{(3)} = \frac{\partial f^{(4)}}{\partial f_i^{(3)}} t^{(4)} \tag{39}$$

$$t_i^{(2)} = \sum_j \frac{\partial f_j^{(3)}}{\partial f_i^{(2)}} t_j^{(3)} \tag{40}$$

$$t_i^{(1)} = \sum_j \frac{\partial f_j^{(2)}}{\partial f_i^{(1)}} t_j^{(2)} \tag{41}$$

$$t_{ab}^{W} = \sum_i \frac{\partial f_i^{(1)}}{\partial W_{ab}} t_i^{(1)} \tag{42}$$

From which the required derivative can be calculated.

$$\begin{aligned}
\frac{\partial E}{\partial W_{ab}} &= t_{ab}^{W} \\
&= \sum_i \frac{\partial f_i^{(1)}}{\partial W_{ab}} t_i^{(1)} \\
&= \sum_i \delta_{ia} x_b t_i^{(1)} \\
&= x_b t_a^{(1)} \\
&= x_b \sum_j \frac{\partial f_j^{(2)}}{\partial f_a^{(1)}} t_j^{(2)} \\
&= x_b \sum_j \phi'(f_j^{(1)}) \delta_{aj} t_j^{(2)} \\
&= x_b \phi'(f_a^{(1)}) t_a^{(2)} \\
&= x_b \phi'(f_a^{(1)}) \sum_j \frac{\partial f_j^{(3)}}{\partial f_a^{(2)}} t_j^{(3)} \\
&= x_b \phi'(f_a^{(1)}) \sum_j -\delta_{ja} t_j^{(3)} \\
&= -x_b \phi'(f_a^{(1)}) t_a^{(3)} \\
&= -x_b \phi'(f_a^{(1)}) \frac{\partial f^{(4)}}{\partial f_a^{(3)}} t^{(4)} \\
&= -2 f_a^{(3)} x_b \phi'(f_a^{(1)})
\end{aligned} \tag{43}$$

This result can be proved by differentiating traditionally by applying the chain rule.

$$\frac{\partial E}{\partial W_{ab}} = \sum_i \left(2 f_i^{(3)}\right)(-1)\left(\phi'\left(f_i^{(1)}\right)\right)(x_b \delta_{ia}) = -2 f_a^{(3)} \phi'\left(f_a^{(1)}\right) x_b \tag{44}$$

Exercise 2.3

The following loss function is given.

$$E = \sum_i \left(y_i - \phi_2 \left(\sum_k W_{ik}^{(2)} \phi_1 \left(\sum_j W_{kj}^{(1)} x_j \right) \right) \right)^2 \tag{45}$$

The computation graph is drawn (figure 5) The functions are defined.

$$f_k^{(1)} = \sum_j W_{kj}^{(1)} x_j \tag{46}$$

$$f_k^{(2)} = \phi_1(f_k^{(1)}) \tag{47}$$

$$f_i^{(3)} = \sum_k W_{ik}^{(2)} f_k^{(2)} \tag{48}$$

$$f_i^{(4)} = \phi_2(f_i^{(3)}) \tag{49}$$

$$f_i^{(5)} = y_i - f_i^{(4)} \tag{50}$$

$$f^{(6)} = E = \sum_i \left(f_i^{(5)}\right)^2 \tag{51}$$

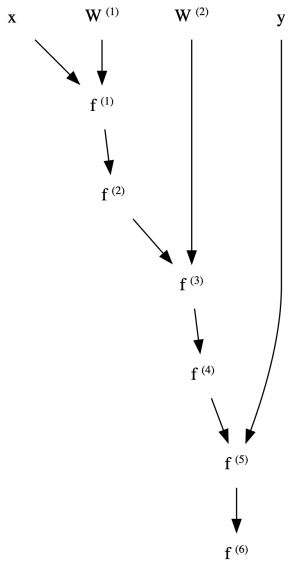

Figure 5: The computation graph of the loss function in Exercise 2.3

The partial derivatives are calculated.

$$\frac{\partial f_i^{(1)}}{\partial W_{ab}^{(1)}} = x_b \delta_{ia} \tag{52}$$

$$\frac{\partial f_i^{(2)}}{\partial f_j^{(1)}} = \phi_1' \left(f_i^{(1)} \right) \delta_{ij} \tag{53}$$

$$\frac{\partial f_i^{(3)}}{\partial f_j^{(2)}} = W_{ij}^{(2)} \tag{54}$$

$$\frac{\partial f_i^{(3)}}{\partial W_{ab}^{(2)}} = f_b^{(2)} \delta_{ia} \tag{55}$$

$$\frac{\partial f_i^{(4)}}{\partial f_j^{(3)}} = \phi_2' \left(f_i^{(3)} \right) \delta_{ij} \tag{56}$$

$$\frac{\partial f_i^{(5)}}{\partial f_j^{(4)}} = -\delta_{ij} \tag{57}$$

$$\frac{\partial f^{(6)}}{\partial f_i^{(5)}} = 2 f_i^{(5)} \tag{58}$$

The propagation schedule is determined and messages calculated.

$$t^{(6)} = 1 \tag{59}$$

$$t_i^{(5)} = \frac{\partial f^{(6)}}{\partial f_i^{(5)}} t^{(6)} = 2 f_i^{(5)} t^{(6)} \tag{60}$$

$$t_i^{(4)} = \sum_j \frac{\partial f_j^{(5)}}{\partial f_i^{(4)}} t_j^{(5)} = \sum_j -\delta_{ji} t_j^{(5)} = -t_i^{(5)} \tag{61}$$

$$t_i^{(3)} = \sum_j \frac{\partial f_j^{(4)}}{\partial f_i^{(3)}} t_j^{(4)} = \sum_j \phi_2' \left(f_j^{(3)} \right) \delta_{ji} t_j^{(4)} = \phi_2' \left(f_i^{(3)} \right) t_i^{(4)} \tag{62}$$

$$t_i^{(2)} = \sum_j \frac{\partial f_j^{(3)}}{\partial f_i^{(2)}} t_j^{(3)} = \sum_j W_{ji}^{(2)} t_j^{(3)} \tag{63}$$

$$t_i^{(1)} = \sum_j \frac{\partial f_j^{(2)}}{\partial f_i^{(1)}} t_j^{(2)} = \sum_j \phi_1' \left(f_j^{(1)} \right) \delta_{ji} t_j^{(2)} = \phi_1' \left(f_i^{(1)} \right) t_i^{(2)} \tag{64}$$

$$t_{ab}^{W^{(1)}} = \sum_i \frac{\partial f_i^{(1)}}{\partial W_{ab}^{(1)}} t_i^{(1)} = \sum_i x_b \delta_{ia} t_i^{(1)} = x_b t_a^{(1)} \tag{65}$$

$$t_{ab}^{W^{(2)}} = \sum_i \frac{\partial f_i^{(3)}}{\partial W_{ab}^{(2)}} t_i^{(1)} = \sum_i f_b^{(2)} \delta_{ia} t_i^{(3)} = f_b^{(2)} t_a^{(3)} \tag{66}$$

The required derivatives are calculated by joining messages.

$$\begin{aligned}
\frac{\partial E}{\partial W_{ab}^{(1)}} &= t_{ab}^{W^{(1)}} \\
&= x_b t_a^{(1)} \\
&= x_b \phi_1' \left(f_a^{(1)} \right) t_a^{(2)} \\
&= x_b \phi_1' \left(f_a^{(1)} \right) \sum_j W_{ja}^{(2)} t_j^{(3)} \\
&= x_b \phi_1' \left(f_a^{(1)} \right) \sum_j W_{ja}^{(2)} \phi_2' \left(f_j^{(3)} \right) t_j^{(4)} \\
&= -x_b \phi_1' \left(f_a^{(1)} \right) \sum_j W_{ja}^{(2)} \phi_2' \left(f_j^{(3)} \right) t_j^{(5)} \\
&= -2 x_b \phi_1' \left(f_a^{(1)} \right) \sum_j W_{ja}^{(2)} \phi_2' \left(f_j^{(3)} \right) f_j^{(5)}
\end{aligned} \tag{67}$$

$$\begin{aligned}
\frac{\partial E}{\partial W_{ab}^{(2)}} &= t_{ab}^{W^{(2)}} \\
&= f_b^{(2)} t_a^{(3)} \\
&= f_b^{(2)} \phi_2' \left(f_a^{(3)} \right) t_a^{(4)} \\
&= -f_b^{(2)} \phi_2' \left(f_a^{(3)} \right) t_a^{(5)} \\
&= -2 f_b^{(2)} \phi_2' \left(f_a^{(3)} \right) f_a^{(5)}
\end{aligned} \tag{68}$$

These results are proved by differentiating in the traditional way with the chain rule.

$$\begin{aligned}
\frac{\partial E}{\partial W_{ab}^{(1)}} &= \sum_i \left(2 f_i^{(5)} \right) (-1) \left(\phi_2' \left(f_i^{(3)} \right) \right) \sum_k W_{ik}^{(2)} \left(\phi_1' \left(f_k^{(1)} \right) \right) (x_b \delta_{ka}) \\
&= -2 x_b \phi_1' \left(f_a^{(1)} \right) \sum_i f_i^{(5)} \phi_2' \left(f_i^{(3)} \right) W_{ia}^{(2)}
\end{aligned} \tag{69}$$

$$\begin{aligned}
\frac{\partial E}{\partial W_{ab}^{(2)}} &= \sum_i \left(2 f_i^{(5)} \right) (-1) \left(\phi_2' \left(f_i^{(3)} \right) \right) f_b^{(2)} \delta_{ia} \\
&= -2 f_a^{(5)} \phi_2' \left(f_a^{(3)} \right) f_b^{(2)}
\end{aligned} \tag{70}$$

Exercise 2.4

The loss functions observed in the previous exercises compute the loss for a given unit of data. This question, instead, requires to extend this scheme for all the units of data. In other words, rather than computing the loss for a single pair (x, y), it requires to do so for the n pairs (x^n, y^n). Consequently, the loss function analysed in the previous exercise is extended by introducing a sum for each pair and the actual loss sees each occurrence y and x being replaced by y^n and x^n, respectively. This is equivalent to computing the loss seen in the previous exercise for each unit of data and summing up the individual losses.

$$E = \sum_n \sum_i \left(y_i^n - \phi_2 \left(\sum_k W_{ik}^{(2)} \phi_1 \left(\sum_j W_{kj}^{(1)} x_j^n \right) \right) \right)^2 \tag{71}$$

Consequently, the individual functions are redefined to account for this change: all function previously defined will take an additional parameter n which is passed down by the newly introduced function $f^{(7)} = E = \sum_n f_n^{(6)}$. Partial derivatives which are both passing down the unit index are updated accordingly, that is, a bind δ_{mn} is introduced for each derivative to ensure that within the summation for n derivation is being carried out with respect to the same unit of data; the partial derivative $\frac{\partial f^{(7)}}{\partial f_n^{(6)}} = 1$ is added. The schedule is updated to reflect the introduction of the new parameter. Eventually, the two required derivatives are eventually calculated.

$$\frac{\partial E}{\partial W_{ab}^{(1)}} = -2 \sum_n x_b^n \phi_1' \left(f_{na}^{(1)} \right) \sum_i f_{ni}^{(5)} \phi_2' \left(f_{ni}^{(3)} \right) W_{ia}^{(2)} \tag{72}$$

$$\frac{\partial E}{\partial W_{ab}^{(2)}} = -2 \sum_n f_{na}^{(5)} \phi_2' \left(f_{na}^{(3)} \right) f_{nb}^{(2)} \tag{73}$$

Exercise 2.5

This exercise is similar to exercise 2.3 in that the two weights matrices are identical, such that $W^{(1)} = W^{(2)} = W$. Consequently, the loss function becomes

$$E = \sum_i \left(y_i - \phi_2 \left(\sum_k W_{ik} \phi_1 \left(\sum_j W_{kj} x_j \right) \right) \right)^2 \tag{74}$$

The computation graph is drawn (figure 6) the individual functions are updated.

$$f_k^{(1)} = \sum_j W_{kj} x_j \tag{75}$$

$$f_k^{(2)} = \phi_1(f_k^{(1)}) \tag{76}$$

$$f_i^{(3)} = \sum_k W_{ik} f_k^{(2)} \tag{77}$$

$$f_i^{(4)} = \phi_2(f_i^{(3)}) \tag{78}$$

$$f_i^{(5)} = y_i - f_i^{(4)} \tag{79}$$

$$f^{(6)} = E = \sum_i \left(f_i^{(5)} \right)^2 \tag{80}$$

Partial derivatives are updated as well.

$$\frac{\partial f_i^{(1)}}{\partial W_{ab}} = x_b \delta_{ia} \tag{81}$$

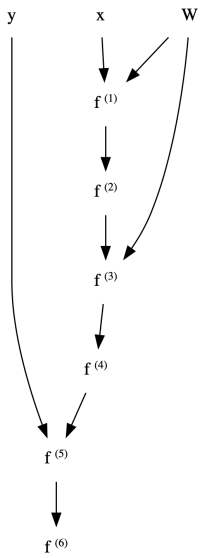

Figure 6: The computation graph of the loss function in Exercise 2.5

$$\frac{\partial f_i^{(2)}}{\partial f_j^{(1)}} = \phi_1' \left(f_i^{(1)} \right) \delta_{ij} \tag{82}$$

$$\frac{\partial f_i^{(3)}}{\partial f_j^{(2)}} = W_{ij} \tag{83}$$

$$\frac{\partial f_i^{(3)}}{\partial W_{ab}} = f_b^{(2)} \delta_{ia} \tag{84}$$

$$\frac{\partial f_i^{(4)}}{\partial f_j^{(3)}} = \phi_2' \left(f_i^{(3)} \right) \delta_{ij} \tag{85}$$

$$\frac{\partial f_i^{(5)}}{\partial f_j^{(4)}} = -\delta_{ij} \tag{86}$$

$$\frac{\partial f^{(6)}}{\partial f_i^{(5)}} = 2 f_i^{(5)} \tag{87}$$

Reverse schedules and messages are updated too.

$$t^{(6)} = 1 \tag{88}$$

$$t_i^{(5)} = \frac{\partial f^{(6)}}{\partial f_i^{(5)}} t^{(6)} = 2 f_i^{(5)} t^{(6)} \tag{89}$$

$$t_i^{(4)} = \sum_j \frac{\partial f_j^{(5)}}{\partial f_i^{(4)}} t_j^{(5)} = \sum_j -\delta_{ji} t_j^{(5)} = -t_i^{(5)} \tag{90}$$

$$t_i^{(3)} = \sum_j \frac{\partial f_j^{(4)}}{\partial f_i^{(3)}} t_j^{(4)} = \sum_j \phi_2' \left(f_j^{(3)} \right) \delta_{ji} t_j^{(4)} = \phi_2' \left(f_i^{(3)} \right) t_i^{(4)} \tag{91}$$

$$t_i^{(2)} = \sum_j \frac{\partial f_j^{(3)}}{\partial f_i^{(2)}} t_j^{(3)} = \sum_j W_{ji} t_j^{(3)} \tag{92}$$

$$t_i^{(1)} = \sum_j \frac{\partial f_j^{(2)}}{\partial f_i^{(1)}} t_j^{(2)} = \sum_j \phi_1' \left(f_j^{(1)} \right) \delta_{ji} t_j^{(2)} = \phi_1' \left(f_i^{(1)} \right) t_i^{(2)} \tag{93}$$

$$t_{ab}^W = \sum_i \frac{\partial f_i^{(1)}}{\partial W_{ab}} t_i^{(1)} + \sum_j \frac{\partial f_j^{(3)}}{\partial W_{ab}} t_j^{(3)} \tag{94}$$

The main change is observed in t_{ab}^W as it is updated to account for both *paths*, as seen in the computation graph. The required derivative is then calculated with the new schedule.

$$\begin{aligned} \frac{\partial E}{\partial W_{ab}} &= t_{ab}^W \\ &= \sum_i \frac{\partial f_i^{(1)}}{\partial W_{ab}} t_i^{(1)} + \sum_k \frac{\partial f_k^{(3)}}{\partial W_{ab}} t_k^{(3)} \\ &= \left(-2 x_b \phi_1' \left(f_a^{(1)} \right) \sum_j W_{ja}^{(2)} \phi_2' \left(f_j^{(3)} \right) f_j^{(5)} \right) + \left(-2 f_a^{(5)} \phi_2' \left(f_a^{(3)} \right) f_b^{(2)} \right) \\ &= -2 \left(f_a^{(5)} \phi_2' \left(f_a^{(3)} \right) f_b^{(2)} + x_b \phi_1' \left(f_a^{(1)} \right) \sum_j W_{ja}^{(2)} \phi_2' \left(f_j^{(3)} \right) f_j^{(5)} \right) \end{aligned} \tag{95}$$

This is confirmed by differentiating traditionally by applying the chain rule.

$$\begin{aligned} \frac{\partial E}{\partial W_{ab}} &= \left(\sum_i \left(2 f_i^{(5)} \right) (-1) \left(\phi_2' \left(f_i^{(3)} \right) \right) \sum_k W_{ik} \left(\phi_1' \left(f_k^{(1)} \right) \right) (x_b \delta_{ka}) \right) + \\ &+ \left(\sum_j \left(2 f_j^{(5)} \right) (-1) \left(\phi_2' \left(f_j^{(3)} \right) \right) f_b^{(2)} \delta_{ia} \right) \\ &= -2 f_a^{(5)} \phi_2' \left(f_a^{(3)} \right) f_b^{(2)} - 2 x_b \phi_1' \left(f_a^{(1)} \right) \sum_i f_i^{(5)} \phi_2' \left(f_i^{(3)} \right) W_{ia}^{(2)} \\ &= -2 \left(f_a^{(5)} \phi_2' \left(f_a^{(3)} \right) f_b^{(2)} + x_b \phi_1' \left(f_a^{(1)} \right) \sum_i f_i^{(5)} \phi_2' \left(f_i^{(3)} \right) W_{ia}^{(2)} \right) \end{aligned} \tag{96}$$

Exercise 3

A neural network with a single hidden layer is given, representing an autoencoder for the 28×28 MNIST images.

$$f_i(x) = \sigma \left(b_i^{(2)} + \sum_{k=1}^{H} W_{ik}^{(2)} \phi \left(b_k^{(1)} + \sum_{j=1}^{784} W_{kj}^{(1)} x_j \right) \right) \tag{97}$$

The error function E is defined for a generic loss function L.

$$E = \frac{1}{N} \sum_n^N \sum_i^{784} L \left(x_i^n, f_i \left(x^n \right) \right) \tag{98}$$

Exercise 3.1

It is given that

$$2(x-y)^2 \leq x \log x + (1-x) \log(1-x) - x \log y - (1-x) \log(1-y) \tag{99}$$

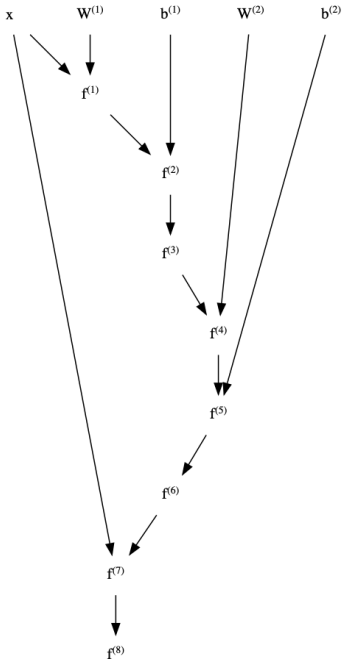

Figure 7: The computation graph of the loss function in Exercise 3.2

From which

$$2(x-y)^2 \leq x \log\left(\frac{x}{y}\right) + (1-x)\log\left(\frac{1-x}{1-y}\right) \tag{100}$$

$$f(x,y) = x \log\left(\frac{x}{y}\right) + (1-x)\log\left(\frac{1-x}{1-y}\right) - 2(x-y)^2 \tag{101}$$

It is required to prove that $f(x,y) \geq 0$ for $0 < x < 1$ and $0 < y < 1$. Differentiate

$$\frac{\partial f}{\partial y} = -\frac{x}{y} + (1-x)\frac{1}{1-y} + 4(x-y) = (x-y)\left(4 - \frac{1}{y(1-y)}\right) \tag{102}$$

Given that $0 < y < 1$, the maximum value of the product $y(1-y)$ is $\frac{1}{4}$ when $y = \frac{1}{2}$, therefore the factor $\left(4 - \frac{1}{y(1-y)}\right)$ is never positive; $f(x,y)$ is then a parabola due to the other factor $(x-y)$. Then, it can be observed that the lowest value of $f(x,y)$ occurs when $x = y$, since it is observed that $f(x,y)$ is a parabola. By substituting the result $x = y$ for the minimum into $f(x,y)$, it is noted that the minimum value of $f(x,y)$ is 0, proving the original inequality.

Exercises 3.2 and 3.3

The individual functions are defined and the related computation graph is drawn (figure 7).

$$f_{nk}^{(1)} = \sum_{j=1}^{784} W_{kj}^{(1)} x_j^n \tag{103}$$

$$f_{nk}^{(2)} = b_k^{(1)} + f_{nk}^{(1)} \tag{104}$$

$$f_{nk}^{(3)} = \phi\left(f_{nk}^{(2)}\right) \tag{105}$$

$$f_{ni}^{(4)} = \sum_{k=1}^{H} W_{ik}^{(2)} f_{nk}^{(3)} \tag{106}$$

$$f_{ni}^{(5)} = b_i^{(2)} + f_{ni}^{(4)} \tag{107}$$

$$f_{ni}^{(6)} = \sigma\left(f_{ni}^{(5)}\right) \tag{108}$$

$$f_{ni}^{(7)} = L\left(x_i^n, f_{ni}^{(6)}\right) \tag{109}$$

$$f^{(8)} = \frac{1}{N} \sum_{n}^{N} \sum_{i}^{784} f_{ni}^{(7)} \tag{110}$$

The partial derivatives are calculated.

$$\frac{\partial f_{ni}^{(1)}}{\partial W_{ab}^{(1)}} = \delta_{ia} x_b \tag{111}$$

$$\frac{\partial f_{ni}^{(2)}}{\partial f_{mj}^{(1)}} = \delta_{ij}\delta_{mn} \tag{112}$$

$$\frac{\partial f_{ni}^{(2)}}{\partial b_a^{(1)}} = \delta_{ia} \tag{113}$$

$$\frac{\partial f_{ni}^{(3)}}{\partial f_{mj}^{(2)}} = \phi'(f_{ni}^{(2)})\delta_{ij}\delta_{mn} \tag{114}$$

$$\frac{\partial f_{ni}^{(4)}}{\partial f_{mj}^{(3)}} = W_{ij}^{(2)}\delta_{mn} \tag{115}$$

$$\frac{\partial f_{ni}^{(4)}}{\partial W_{ab}^{(2)}} = \delta_{ia} f_{nb}^{(3)} \tag{116}$$

$$\frac{\partial f_{ni}^{(5)}}{\partial f_{mj}^{(4)}} = \delta_{ij}\delta_{mn} \tag{117}$$

$$\frac{\partial f_{ni}^{(5)}}{\partial b_a^{(2)}} = \delta_{ia} \tag{118}$$

$$\frac{\partial f_{ni}^{(6)}}{\partial f_{mj}^{(5)}} = \sigma'\left(f_{ni}^{(5)}\right)\delta_{ij}\delta_{mn} \tag{119}$$

$$\frac{\partial f_{ni}^{(7)}}{\partial f_{mj}^{(6)}} = L'\left(x_i^n, f_{ni}^{(6)}\right)\delta_{ij}\delta_{mn} \tag{120}$$

$$\frac{\partial f^{(8)}}{\partial f_{ni}^{(7)}} = \frac{1}{N} \tag{121}$$

The reverse schedule is determined along with the messages.

$$t^{(8)} = 1 \tag{122}$$

$$t_{ni}^{(7)} = \frac{\partial f^{(8)}}{\partial f_n^{(7)} i} t^{(8)} = \frac{1}{N} t^{(8)} \tag{123}$$

$$t_{ni}^{(6)} = \sum_m \sum_j \frac{\partial f_{mj}^{(7)}}{\partial f_n^{(6)} i} t_{mj}^{(7)} = \sum_m \sum_j L'\left(x_j^m, f_{mj}^{(6)}\right) \delta_{ij}\delta_{mn} t_{mj}^{(7)} = L'\left(x_i^n, f_{ni}^{(6)}\right) t_{ni}^{(7)} \tag{124}$$

$$t_{ni}^{(5)} = \sum_m \sum_j \frac{\partial f_{mj}^{(6)}}{\partial f_n^{(5)} i} t_{mj}^{(6)} = \sum_m \sum_j \sigma'\left(f_{mj}^{(5)}\right) \delta_{ij}\delta_{mn} t_{mj}^{(6)} = \sigma'\left(f_{ni}^{(5)}\right) t_{ni}^{(6)} \tag{125}$$

$$t_a^{b^{(2)}} = \sum_m \sum_j \frac{\partial f_{mj}^{(5)}}{\partial b_a^{(2)}} t_{mj}^{(5)} = \sum_m \sum_j \delta_{ja} t_{mj}^{(5)} = \sum_m t_{ma}^{(5)} \tag{126}$$

$$t_{ni}^{(4)} = \sum_m \sum_j \frac{\partial f_{mj}^{(5)}}{\partial f_n^{(4)} i} t_{mj}^{(5)} = \sum_m \sum_j \delta_{ij}\delta_{mn} t_{mj}^{(5)} = t_{ni}^{(5)} \tag{127}$$

$$t_{ab}^{W^{(2)}} = \sum_n \sum_i \frac{\partial f_{ni}^{(4)}}{\partial W_{ab}^{(2)}} t_{ni}^{(4)} = \sum_n \sum_i \delta_{ia} f_{nb}^{(3)} t_{ni}^{(4)} = \sum_n f_{nb}^{(3)} t_{na}^{(4)} \tag{128}$$

$$t_{ni}^{(3)} = \sum_m \sum_j \frac{\partial f_{mj}^{(4)}}{\partial f_n^{(3)} i} t_{mj}^{(4)} = \sum_m \sum_j W_{ji}^{(2)} \delta_{mn} t_{mj}^{(4)} = \sum_j W_{ji}^{(2)} t_{nj}^{(4)} \tag{129}$$

$$t_{ni}^{(2)} = \sum_m \sum_j \frac{\partial f_{mj}^{(3)}}{\partial f_n^{(2)} i} t_{mj}^{(3)} = \sum_m \sum_j \phi'\left(f_{mj}^{(2)}\right) \delta_{ij}\delta_{mn} t_{mj}^{(3)} = \phi'\left(f_{ni}^{(2)}\right) t_{ni}^{(3)} \tag{130}$$

$$t_{ni}^{(1)} = \sum_m \sum_j \frac{\partial f_{mj}^{(2)}}{\partial f_n^{(1)} i} t_{mj}^{(2)} = \sum_m \sum_j \delta_{ij}\delta_{mn} t_{mj}^{(2)} = t_{ni}^{(2)} \tag{131}$$

$$t_a^{b^{(1)}} = \sum_m \sum_j \frac{\partial f_{mj}^{(2)}}{\partial b_a^{(1)}} t_{mj}^{(2)} = \sum_m \sum_j \delta_{ja} t_{mj}^{(2)} = \sum_m t_{ma}^{(2)} \tag{132}$$

$$t_{ab}^{W^{(1)}} = \sum_m \sum_j \frac{\partial f_{mj}^{(1)}}{\partial W_{ab}^{(1)}} t_{mj}^{(1)} = \sum_m \sum_j x_b \delta_{ja} t_{mj}^{(2)} = \sum_m x_b t_{ma}^{(1)} \tag{133}$$

The required derivatives are then computed using the schedule and messages.

$$\begin{aligned}
\frac{\partial E}{\partial W_{ab}^{(1)}} &= t_{ab}^{W^{(1)}} \\
&= \sum_n x_b t_{na}^{(1)} \\
&= \sum_n x_b t_{na}^{(2)} \\
&= \sum_n x_b \phi'\left(f_{na}^{(2)}\right) t_{na}^{(3)} \\
&= \sum_n x_b \phi'\left(f_{na}^{(2)}\right) \sum_i W_{ia}^{(2)} t_{ni}^{(4)} \\
&= \sum_n x_b \phi'\left(f_{na}^{(2)}\right) \sum_i W_{ia}^{(2)} t_{ni}^{(5)} \\
&= \sum_n x_b \phi'\left(f_{na}^{(2)}\right) \sum_i W_{ia}^{(2)} \sigma'\left(f_{ni}^{(5)}\right) t_{ni}^{(6)} \\
&= \sum_n x_b \phi'\left(f_{na}^{(2)}\right) \sum_i W_{ia}^{(2)} \sigma'\left(f_{ni}^{(5)}\right) L'\left(x_i^n, f_{ni}^{(6)}\right) t_{ni}^{(7)} \\
&= \frac{1}{N} \sum_n x_b \phi'\left(f_{na}^{(2)}\right) \sum_i W_{ia}^{(2)} \sigma'\left(f_{ni}^{(5)}\right) L'\left(x_i^n, f_{ni}^{(6)}\right)
\end{aligned} \tag{134}$$

$$
\begin{aligned}
\frac{\partial E}{\partial b_a^{(1)}} &= t_a^{b^{(1)}} \\
&= \sum_n t_{na}^{(2)} \\
&= \sum_n \phi'\left(f_{na}^{(2)}\right) t_{na}^{(3)} \\
&= \sum_n \phi'\left(f_{na}^{(2)}\right) \sum_i W_{ia}^{(2)} t_{ni}^{(4)} \\
&= \sum_n \phi'\left(f_{na}^{(2)}\right) \sum_i W_{ia}^{(2)} t_{ni}^{(5)} \\
&= \sum_n \phi'\left(f_{na}^{(2)}\right) \sum_i W_{ia}^{(2)} \sigma'\left(f_{ni}^{(5)}\right) L'\left(x_i^n, f_{ni}^{(6)}\right) t_{ni}^{(7)} \\
&= \frac{1}{N} \sum_n \phi'\left(f_{na}^{(2)}\right) \sum_i W_{ia}^{(2)} \sigma'\left(f_{ni}^{(5)}\right) L'\left(x_i^n, f_{ni}^{(6)}\right)
\end{aligned} \tag{135}
$$

$$
\begin{aligned}
\frac{\partial E}{\partial W_{ab}^{(2)}} &= t_{ab}^{W^{(2)}} \\
&= \sum_n f_{nb}^{(3)} t_{na}^{(4)} \\
&= \sum_n f_{nb}^{(3)} t_{na}^{(5)} \\
&= \sum_n f_{nb}^{(3)} \sigma'\left(f_{na}^{(5)}\right) t_{na}^{(6)} \\
&= \sum_n f_{nb}^{(3)} \sigma'\left(f_{na}^{(5)}\right) L'\left(x_a^n, f_{na}^{(6)}\right) t_{na}^{(7)} \\
&= \frac{1}{N} \sum_n f_{nb}^{(3)} \sigma'\left(f_{na}^{(5)}\right) L'\left(x_a^n, f_{na}^{(6)}\right)
\end{aligned} \tag{136}
$$

$$
\begin{aligned}
\frac{\partial E}{\partial b_a^{(2)}} &= t_a^{b^{(2)}} \\
&= \sum_n t_{na}^{(5)} \\
&= \sum_n \sigma'\left(f_{na}^{(5)}\right) t_{na}^{(6)} \\
&= \sum_n \sigma'\left(f_{na}^{(5)}\right) L'\left(x_a^n, f_{na}^{(6)}\right) t_{na}^{(7)} \\
&= \frac{1}{N} \sum_n \sigma'\left(f_{na}^{(5)}\right) L'\left(x_a^n, f_{na}^{(6)}\right)
\end{aligned} \tag{137}
$$

Exercise 3.4 and 3.5

For this question, Python was used to implement the ADAM optimiser, along with the previously defined AutoDiff schedule. Different transfer functions were tested and the relu was chosen; initial weights are picked randomly from a zero-mean unit-variance normal distribution. Learning rate was set to 0.05, while the three ADAM parameters were tuned as suggested; 100 epochs of optimisation were performed. The implementation can be observed in the appendix. The error decrease can be observed in figure 8; results of reconstruction with the autoencoder can be observed in figure 9.

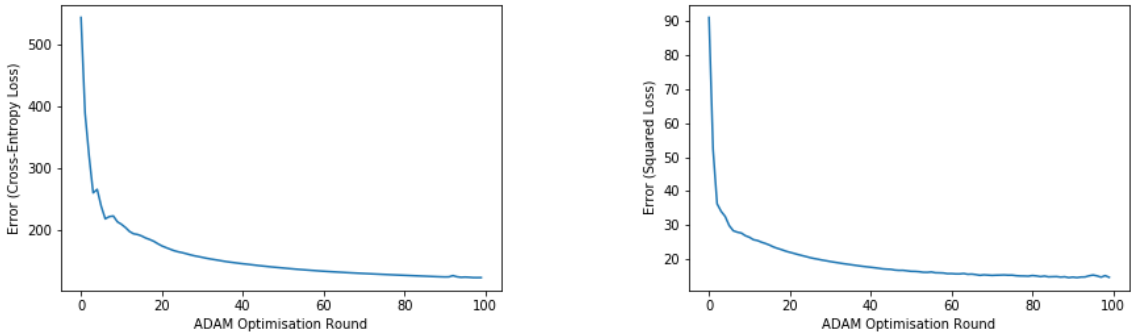

Figure 8: Training error at each ADAM optimisation round with Square Loss (left) and Cross Entropy Loss (right).

Exercise 3.6

This question requires to determine whether the error function is convex with respect to W^2 and b^2 for both the LikLoss and SquareLoss. As usual, convexity is determined by whether the second derivative is always positive. For convenience, throughout this exercise, σ is used to represent $f_{na}^{(6)} = \sigma\left(f_{na}^{(5)}\right)$ and L to represent $L\left(x_a^n, f_{na}^{(6)}\right)$; the single apex $'$ and double apex $''$ represent the first and second derivative of the function they follow with respect to either of the two variables of interest. The loss functions and their derivatives with respect to the second parameter y (which will be the one of interest in the calculations) are

$$L_{lik}(x,y) = -x\log(y) - (1-x)\log(1-y) \tag{138}$$

$$L'_{lik}(x,y) = \frac{y-x}{y(1-y)} \tag{139}$$

$$L''_{lik}(x,y) = \frac{y^2+x-2xy}{y^2(1-y)^2} \tag{140}$$

$$L_{sq}(x,y) = \frac{1}{2}(x-y)^2 \tag{141}$$

$$L'_{sq}(x,y) = y-x \tag{142}$$

$$L''_{sq}(x,y) = 1 \tag{143}$$

With regard to $b^{(2)}$, the first and second derivatives are:

$$\frac{\partial E}{\partial b_a^{(2)}} = \frac{1}{N}\sum_n \sigma'\left(f_{na}^{(5)}\right) L'\left(x_a^n, f_{na}^{(6)}\right) = \frac{1}{N}\sum_n \sigma' L' \tag{144}$$

$$\begin{aligned}\frac{\partial^2 E}{\partial {b_a^{(2)}}^2} &= \frac{1}{N}\sum_n \sigma' L'' \sigma' + \sigma'' L' \\ &= \frac{1}{N}\sum_n \sigma^2(1-\sigma)^2 L'' + \sigma(1-\sigma)(1-2\sigma)L' \\ &= \frac{1}{N}\sum_n \sigma(1-\sigma)[\sigma(1-\sigma)L'' + (1-2\sigma)L']\end{aligned} \tag{145}$$

It is known that $\sigma(1-\sigma)$ is never negative, hence the factor to assess for positivity is $\sigma(1-\sigma)L'' + (1-2\sigma)L'$. For the LikLoss:

$$\sigma(1-\sigma)\frac{\sigma^2+x-2\sigma x}{\sigma^2(1-\sigma)^2} + (1-2\sigma)\frac{\sigma-x}{\sigma(1-\sigma)} = \frac{x+\sigma^2-2\sigma x+\sigma-x-2\sigma^2+2\sigma x}{\sigma(1-\sigma)} = 1 \tag{146}$$

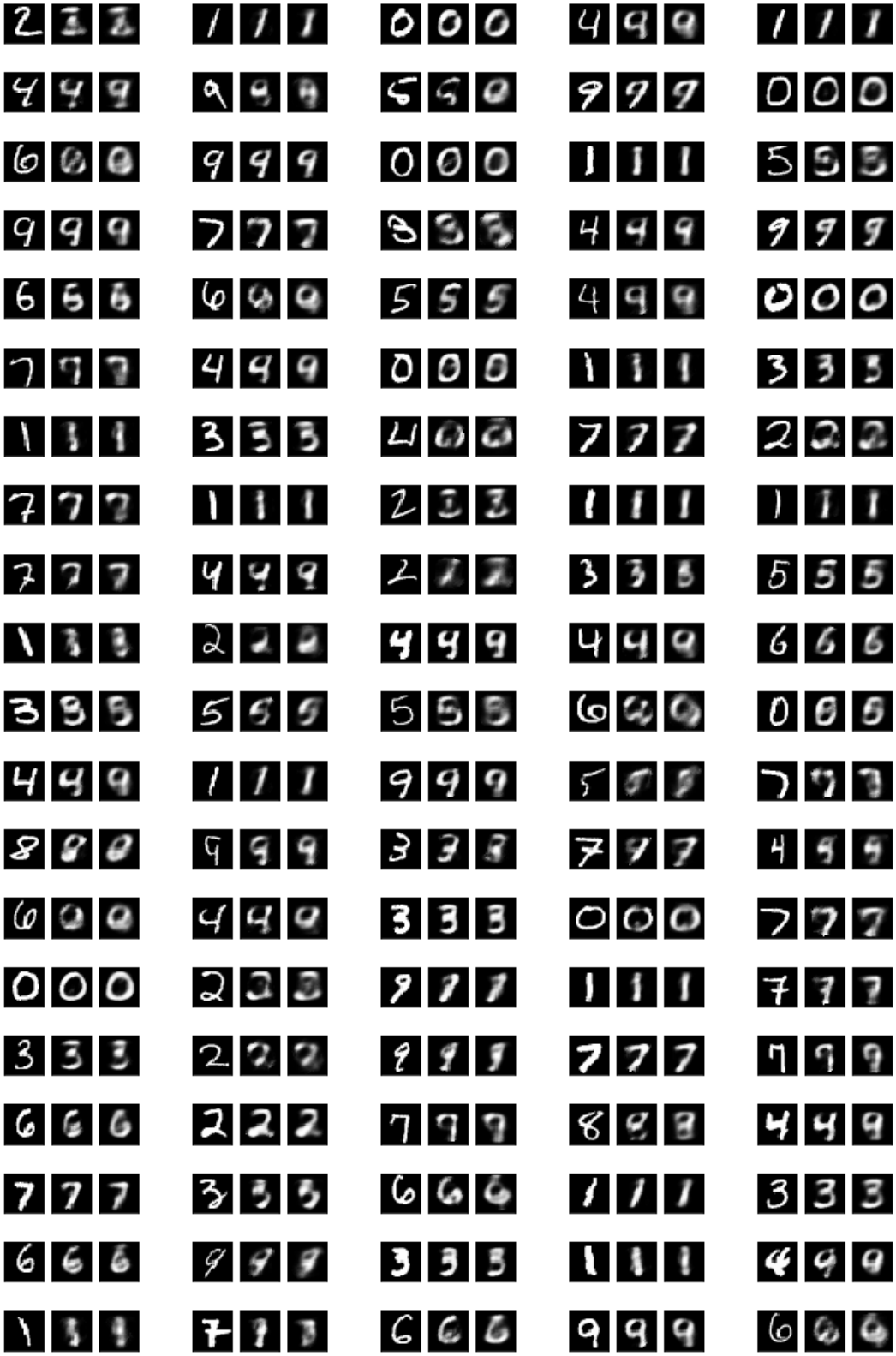

Figure 9: The reconstructions of 100 MNIST digits; each triplet contains (from left to right) the original image, the reconstruction using squared loss and cross-entropy loss.

This proves that the second derivative of the error function with respect to $b^{(2)}$ using the LikLoss is always positive, proving convexity. Similarly, for the SquareLoss:

$$\sigma(1-\sigma)+(1-2\sigma)(\sigma-x)=\sigma-\sigma^2+\sigma-x-2\sigma^2+2\sigma x=-3\sigma^2+\sigma(2+2x)-x \tag{147}$$

For the error function to be convex, this factor has to be positive for each value of x, but that is not the case as the function can be negative (e.g. for $\sigma=0.75, x=0$ the factor is equal to -0.1875). Due to this inconsistency in the sign of the factor, it is determined that the error function is not convex with respect to $b^{(2)}$ when the SquareLoss is used.

The process is now repeated for $W^{(2)}$. The derivative of the error function with respect to $W^{(2)}_{ab}$ is:

$$\frac{\partial E}{\partial W^{(2)}_{ab}}=\frac{1}{N}\sum_n f^{(3)}_{nb}\sigma'\left(f^{(5)}_{na}\right)L'\left(x^n_a,f^{(6)}_{na}\right) \tag{148}$$

The factor $f^{(3)}$ does not depend on $W^{(2)}$ and can therefore be treated as a constant. The second derivative is:

$$\begin{aligned}
\frac{\partial^2 E}{\partial {W^{(2)}_{ab}}^2} &= \frac{1}{N}\sum_n f^{(3)}_{nb}\left[\sigma'\left(f^{(5)}_{na}\right)L''\left(x^n_a,f^{(6)}_{na}\right)\sigma'\left(f^{(5)}_{na}\right)f^{(3)}_{nb}+L'\left(x^n_a,f^{(6)}_{na}\right)\sigma''\left(f^{(5)}_{na}\right)f^{(3)}_{nb}\right]\\
&= \frac{1}{N}\sum_n \left(f^{(3)}_{nb}\right)^2\left[\left(\sigma'\left(f^{(5)}_{na}\right)\right)^2 L''\left(x^n_a,f^{(6)}_{na}\right)+L'\left(x^n_a,f^{(6)}_{na}\right)\sigma''\left(f^{(5)}_{na}\right)\right]\\
&= \frac{1}{N}\sum_n \left(f^{(3)}_{nb}\right)^2\left[(\sigma')^2L''+\sigma''L'\right] \qquad (149)\\
&= \frac{1}{N}\sum_n \left(f^{(3)}_{nb}\right)^2\left[\sigma^2(1-\sigma)^2L''+\sigma(1-\sigma)(1-2\sigma)L'\right]\\
&= \frac{1}{N}\sum_n \left(f^{(3)}_{nb}\right)^2\sigma(1-\sigma)\left[\sigma(1-\sigma)L''+\sigma(1-2\sigma)L'\right]
\end{aligned}$$

The only factor to assess is once again $\sigma(1-\sigma)L''+\sigma(1-2\sigma)L'$. The conclusions on its sign are similar to the ones in the first half of this exercise. This means that for the LikLoss, the error function is concave with respect to $W^{(2)}$, but it is not for the SquareLoss.

Exercise 3.7 and 3.8

In these questions, slices of the error function with the Square Loss first and Cross-Entropy Loss then are plotted to observe whether they are convex or not. In particular, the functions plotted are

$$F_{11}(z_1,z_2)=E(W^{(1)}+z_1U^{(1)}+z_2^{(2)}U^{(2)},b^{(1)},W^{(2)},b^{(2)}) \tag{150}$$

$$F_{12}(z_1,z_2)=E(W^{(1)}+z_1U^{(1)},b^{(1)},W^{(2)}+z_2U^{(2)},b^{(2)}) \tag{151}$$

$$F_{22}(z_1,z_2)=E(W^{(1)},b^{(1)},W^{(2)}+z_1U^{(1)}+z_2^{(2)}U^{(2)},b^{(2)}) \tag{152}$$

for both loss functions, results are in figures 10, 12 and 14. It can be observed that with cross-entropy loss, the error function appears to be convex, whereas with square loss, the error function is not convex.

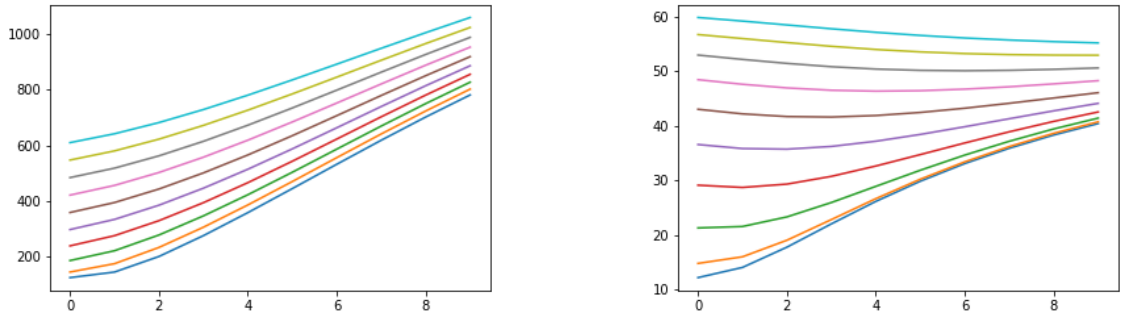

Figure 10: $F_{1,1}$ for Cross-Entropy Loss (left) and Square Loss (Right)

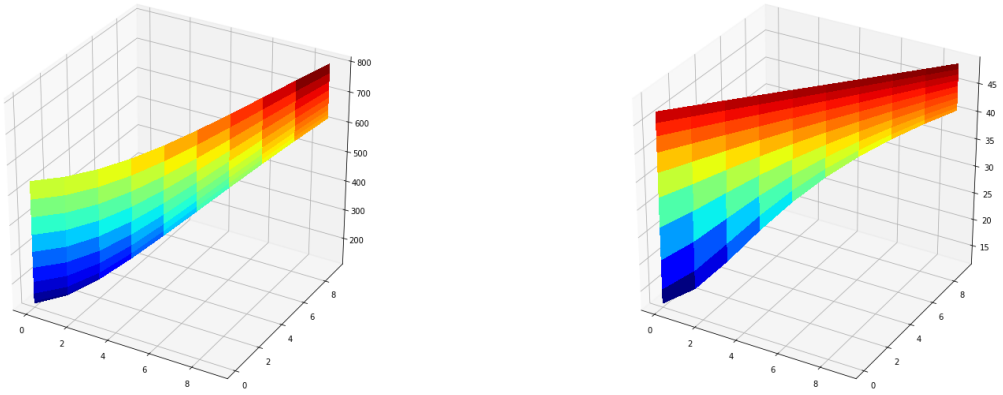

Figure 11: $F_{1,1}$ for Cross-Entropy Loss (left) and Square Loss (Right)

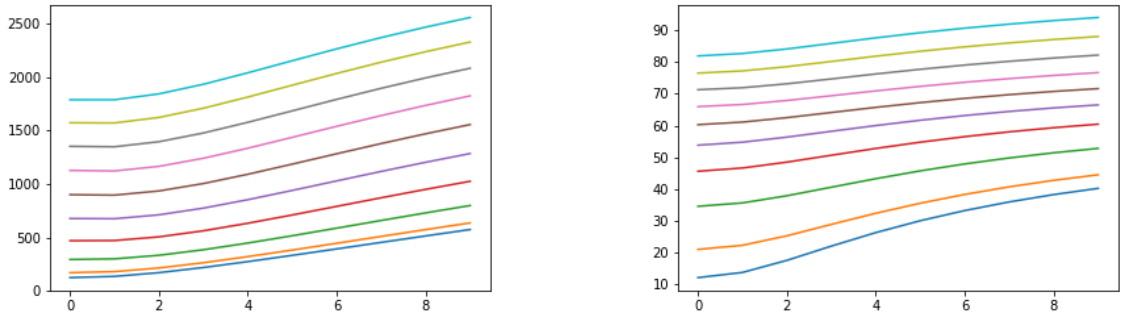

Figure 12: $F_{1,2}$ for Cross-Entropy Loss (left) and Square Loss (Right)

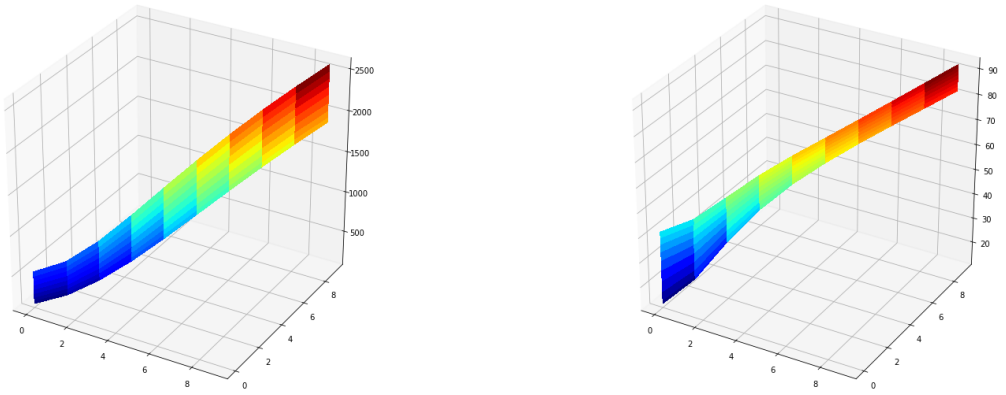

Figure 13: $F_{1,2}$ for Cross-Entropy Loss (left) and Square Loss (Right)

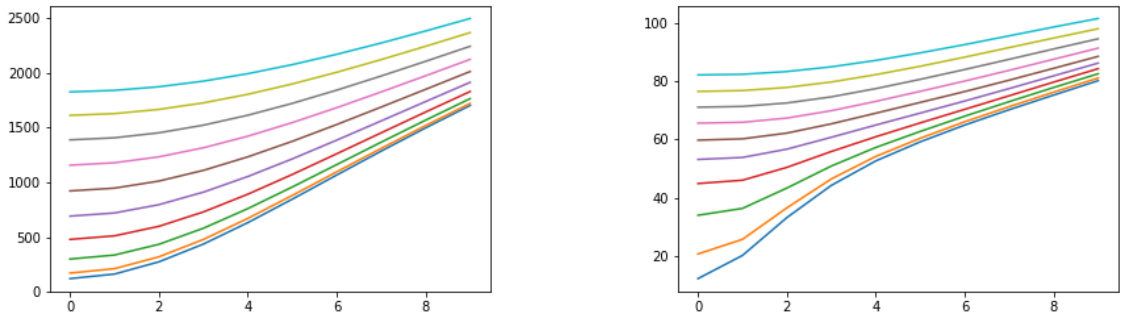

Figure 14: $F_{2,2}$ for Cross-Entropy Loss (left) and Square Loss (Right)

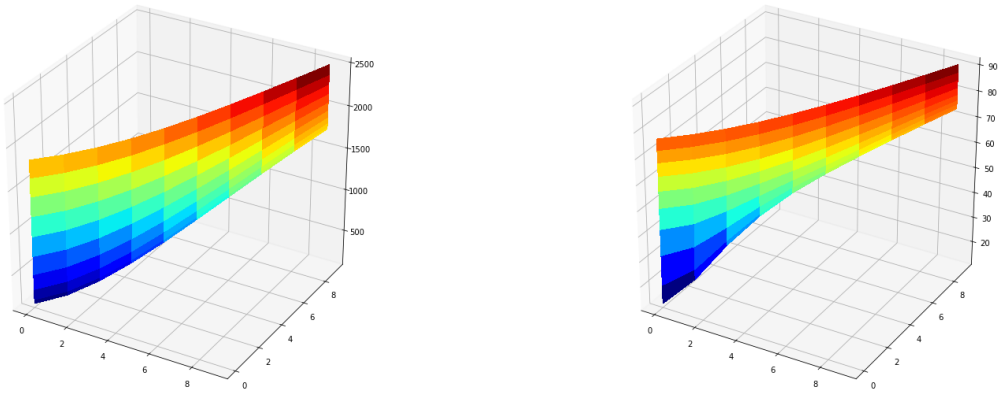

Figure 15: $F_{2,2}$ for Cross-Entropy Loss (left) and Square Loss (Right)

Exercise 4

Exercise 4.1

It is given that

$$\begin{aligned} KL(q\|p) &= \int q(x) \log\left(\frac{q(x)}{p(x)}\right) dx \\ &= -\int q(x) \log\left(\frac{q(x)}{p(x)}\right) dx \end{aligned} \tag{153}$$

Using the bound $\log(x) \leqslant x - 1$:

$$\begin{aligned} KL(q\|p) &= -\int q(x) \log\left(\frac{p(x)}{q(x)} - 1\right) dx \\ &= -\int p(x) - q(x) dx \\ &= -\int p(x) dx + \int q(x) dx \end{aligned} \tag{154}$$

It is known that $\int p(x)dx = 1$ and $\int q(x)dx = 1$, showing that $KL(q\|p) = 0$.

Exercise 4.2

It is asked to show that

$$\begin{aligned} \log p(x) &\geq -\int q(z) \log q(z) dz + \int q(z) \log p(x|z,\theta) p(z) dz \\ &\geq \int q(z) \log\left(\frac{p(x|z,\theta)p(z)}{q(z)}\right) dz \\ &\geq \int q(z) \log\left(\frac{p(z)}{q(z)}\right) dz + \int q(z) p(x|z,\theta) dz \\ &\geq \langle \log p(x|z,\theta) \rangle_{q(z)} - KL \end{aligned} \tag{155}$$

Given the result in question 4.1, the inequality holds.

Exercise 4.3

The multivariate Gaussian distribution is given

$$q(z) = \frac{1}{\sqrt{2\pi|\Sigma|}} e^{-\frac{1}{2}(z-\mu)^T\Sigma^{-1}(z-\mu)} \tag{156}$$

For convenience:

$$\frac{1}{\sqrt{2\pi|\Sigma|}} = (2\pi|\Sigma|)^{-\frac{1}{2}} = \alpha \tag{157}$$

$$e^{-\frac{1}{2}(z-\mu)^T\Sigma^{-1}(z-\mu)} = e^{-\frac{(z-\mu)^2}{2\Sigma}} = e^{\beta} \tag{158}$$

$$q(z) = \alpha e^{\beta} \tag{159}$$

$$\int q(z)dz = \int \alpha e^{\beta} dz = 1 \tag{160}$$

Therefore

$$\begin{aligned}
-\int q(z)\log q(z)dz &= -\int q(z)\log\left(\alpha e^{\beta}\right)dz \\
&= -\int q(z)\left[\log(\alpha)+\beta\right]dz \\
&= -\log(a)\int q(z)dz - \int \beta q(z)dz \\
&= -\log(a) - \int -\frac{1}{2}\frac{(z-\mu)^2}{\Sigma}q(z)dz \\
&= -\log(a) + \frac{1}{2\Sigma}\int (z-\mu)^2 q(z)dz \\
&= -\log(a) + \frac{1}{2\Sigma}\Sigma \\
&= -\log(a) + \frac{1}{2} \\
&= -\log\left((2\pi|\Sigma|)^{-\frac{1}{2}}\right) + \frac{1}{2}\log(e) \\
&= \frac{1}{2}\log\left(2\pi|\Sigma|\right) + \frac{1}{2}\log(e) \\
&= \frac{1}{2}\log\left(2\pi e|\Sigma|\right)
\end{aligned} \tag{161}$$

Exercise 4.4

For a distribution $p(x|\theta) = \int p(x|\theta,z)p(z)dz$, thus

$$\log(p(x)) \geq -\int q(z|x,\phi)\log(q(z|x,\phi))dz + \int q(z|x,\phi)\log(p(x|z,\theta)p(z))dz \tag{162}$$

Then

$$
\begin{aligned}
&\int q(z|x,\phi)\log(p(x|z,\theta)p(z))dz \\
&= \int q(z|x,\phi)[\log(p(x|z,\theta)) + \log(p(z))]dz \\
&= \int q(z|x,\phi)\log(p(x|z,\theta))dz + \int q(z|x,\phi)\log(p(z))dz \\
&= \int q(z|x,\phi)\log\left[\left(2\pi\sigma_x^2\right)^{-\frac{X}{2}}\exp\left(-\frac{1}{2\sigma_x^2}(x-\mu_x(x,\theta))^2\right)\right]dz + \int q(z|x,\phi)\log(p(z))dz \\
&= -\int q(z|x,\phi)\frac{X}{2}\log\left(2\pi\sigma_x^2\right)dz - \int q(z|x,\phi)\frac{1}{2\sigma_x^2}(x-\mu_x(x,\theta))^2dz + \int q(z|x,\phi)\log(p(z))dz \\
&= -\frac{X}{2}\log\left(2\pi\sigma_x^2\right)\int q(z|x,\phi)dz - \frac{1}{2\sigma_x^2}\int q(z|x,\phi)(x-\mu_x(x,\theta))^2dz + \int q(z|x,\phi)\log(p(z))dz \\
&= -\frac{X}{2}\log(\sigma_x^2) - \frac{1}{2\sigma_x^2}\int q(z|x,\phi)(x-\mu_x(x,\theta))^2dz + \int q(z|x,\phi)\log(p(z))dz + \text{const.} \\
&= -\frac{X}{2}\log\left(\sigma_x^2\right) - \frac{1}{2\sigma_x^2}\int q(z|x,\phi)(x-\mu_x(x,\theta))^2dz \\
&\quad + \int q(z|x,\phi)\log(2\pi|\Sigma|)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}(z-\mu)^T\Sigma^{-1}(z-\mu)\right)dz + \text{const.} \\
&= -\frac{X}{2}\log\left(\sigma_x^2\right) - \frac{1}{2\sigma_x^2}\int q(z|x,\phi)(x-\mu_x(x,\theta))^2dz + \int q(z|x,\phi)\log(2\pi|\Sigma|)^{-\frac{1}{2}}dz \\
&\quad + \int q(z|x,\phi)\left(-\frac{1}{2}(z-\mu)^T\Sigma^{-1}(z-\mu)\right)dz + \text{const.}
\end{aligned}
\tag{163}
$$

Since $\mu = 0$ and $\Sigma = I$:

$$
\begin{aligned}
&= -\frac{X}{2}\log\left(\sigma_x^2\right) - \frac{1}{2\sigma_x^2}\int q(z|x,\phi)(x-\mu_x(x,\theta))^2dz - \frac{1}{2}\int q(z|x,\phi)(z)^T(z)dz + \text{const.} \\
&= -\frac{X}{2}\log\left(\sigma_x^2\right) - \frac{1}{2\sigma_x^2}\int q(z|x,\phi)(x-\mu_x(x,\theta))^2dz - \frac{1}{2}\mathbb{E}\left[z^2\right] + \text{const.}
\end{aligned}
\tag{164}
$$

It is known that $Var(z) = \sigma_z^2 = \mathbb{E}\left[z^2\right] - \mathbb{E}\left[z\right]^2$, therefore:

$$
= -\frac{X}{2}\log\left(\sigma_x^2\right) - \frac{1}{2\sigma_x^2}\int q(z|x,\phi)(x-\mu_x(x,\theta))^2dz - \frac{1}{2}\left(\sigma_z^2 + \mu_z^2(x,\theta)\right) + \text{const.} \tag{165}
$$

For the other term in the inequality:

$$
\begin{aligned}
&\int q(z|x,\phi)\log(q(z|x,\phi))dz \\
&= \int q(z|x,\phi)\log\left(2\pi\sigma_z^2\right)^{-\frac{Z}{2}}\exp\left(-\frac{1}{2\sigma_z^2}(z-\mu_z(x,\phi))^2)\right)dz \\
&= -\frac{Z}{2}\int q(z|x,\phi)\log\left(2\pi\sigma_z^2\right)dz - \frac{1}{2\sigma_z^2}\int q(z|x,\phi)(z-\mu_z(x,\phi))^2dz \\
&= -\frac{Z}{2}\log\left(2\pi\sigma_z^2\right)dz - \frac{1}{2} \\
&= -\frac{Z}{2}\log\left(\sigma_z^2\right) + \text{const.}
\end{aligned}
\tag{166}
$$

In conclusion

$$
\begin{aligned}
&-\int q(z|x,\phi)\log(q(z|x,\phi))dz + \int q(z|x,\phi)\log(p(x|z,\theta)p(z))dz = \\
&= \frac{Z}{2}\log\left(\sigma_z^2\right) - \frac{X}{2}\log\left(\sigma_x^2\right) - \frac{1}{2\sigma_x^2}\int q(z|x,\phi)(x-\mu_x(x,\theta))^2dz - \frac{1}{2}\left(\sigma_z^2 + \mu_z^2(x,\theta)\right) + \text{const.}
\end{aligned}
\tag{167}
$$

Proving the given inequality.

Exercise 4.5

It is given

$$\int f(x)\frac{1}{\sqrt{(2\pi)^X|\Sigma|}}\exp\left(-\frac{(x-\mu)^2}{2\Sigma}\right)dx \tag{168}$$

By substituting $x=\mu+\epsilon\Sigma^{\frac{1}{2}}$, such that $x-\mu=\epsilon\Sigma^{\frac{1}{2}}$ and $dx=\Sigma^{\frac{1}{2}}d\epsilon$. Therefore

$$\int f(x)\frac{1}{\sqrt{(2\pi)^X|\Sigma|}}\exp\left(-\frac{(x-\mu)^2}{2\Sigma}\right)dx=\int f(\mu+\epsilon\Sigma^{\frac{1}{2}})\frac{1}{\sqrt{(2\pi)^X|\Sigma|}}\exp\left(-\frac{\epsilon^2}{2}\right)\Sigma^{\frac{1}{2}}d\epsilon \tag{169}$$

Exercise 4.6

From exercise 4.4, it is known that the following holds:

$$\log p(x)\geq\frac{Z}{2}\log(\sigma_z^2)-\frac{X}{2}\log(\sigma_x^2)-\frac{1}{2\sigma_x^2}\int_z q(z|x,\phi)(x-\mu_x(z,\theta))^2dz-\frac{1}{2}(Z\sigma_z^2+\mu_z^2(x,\phi))+\text{const.} \tag{170}$$

By applying the reparameterisation trick:

$$\int_z q(z|x,\phi)(-\frac{1}{2\sigma_x^2}(x-\mu_x(z,\theta))^2dz=\int_z q(z|x,\phi)\psi(z,x,\theta)dz \tag{171}$$

where

$$\psi(z,x,\theta)=(x-\mu_x(z,\theta))^2 \tag{172}$$

It follows that, for the requirements of the question:

$$\begin{aligned}&\int_z q(z|x,\phi)\psi(z,x,\theta)dz=\\&=\int_\epsilon\psi(\mu_z(x,\phi)+\sigma_z\epsilon,z,\theta)\frac{1}{(2\pi)^{\frac{Z}{2}}}\exp^{-\frac{1}{2}\epsilon^2}d\epsilon=\\&=\int_\epsilon(x-\mu_x(\mu_z(x,\phi)+\sigma_z\epsilon,\theta))^2\frac{1}{(2\pi)^{\frac{Z}{2}}}\exp^{-\frac{1}{2}\epsilon^2}d\epsilon=\\&=\mathbb{E}\left[(x-\mu_x(\mu_z(x,\phi)+\sigma_z\epsilon,\theta))^2\right]_\epsilon\end{aligned} \tag{173}$$

Which proves the following holds:

$$\log p(x)\geq\frac{Z}{2}\log(\sigma_z^2)-\frac{X}{2}\log(\sigma_x^2)-\frac{1}{2\sigma_x^2}\mathbb{E}\left[(x-\mu_x(\mu_z(x,\phi)+\sigma_z\epsilon,\theta))^2\right]_\epsilon-\frac{1}{2}(Z\sigma_z^2+\mu_z^2(x,\phi))+\text{const.} \tag{174}$$

Exercise 4.7

It is given

$$\mathbb{E}(\theta,\phi)=\frac{1}{N}\sum_{n=1}^{N}\mathbb{E}\left[(x^n-\mu_x(\mu_z(x^n,\phi)+\sigma_z\epsilon,\theta)\right]_\epsilon^2+\sigma_x^2\mu_z^2(x^n,\phi)) \tag{175}$$

We want to show that this is a fit objective to minimize for a generative model $p(x)$. What we want to do is maximise $p(x)$. We know from the previous answers that:

$$\log p(x)\geq\frac{Z}{2}\log(\sigma_z^2)-\frac{X}{2}\log(\sigma_x^2)-\frac{1}{2\sigma_x^2}\mathbb{E}\left[(x-\mu_x(\mu_z(x,\phi)+\sigma_z\epsilon,\theta))^2\right]_\epsilon-\frac{1}{2}(Z\sigma_z^2+\mu_z^2(x,\phi))+\text{const.} \tag{176}$$

We further know σ_x^2 and σ_z^2 are constant, thus the only terms in the lower bound that can change are $\mathbb{E}[(x-\mu_x(\mu_z(x,\phi)+\sigma_z\epsilon,\theta))^2]_\epsilon$ and $\mu_z^2(x,\phi)$. Due the fact that these terms appear as negative in the lower bound, minimizing these 2 will maximize it. By maximizing the lower bound we also maximize $p(x)$, thus achieving our initial goal for a genererative model.

Exercise 4.8

In this question, a generative model was fit to the MNIST dataset using the VAE method. The samples image of the network are seen in figure 16, the progress of the training objective in figure 17. PyTorch was used, as seen in the appendix.

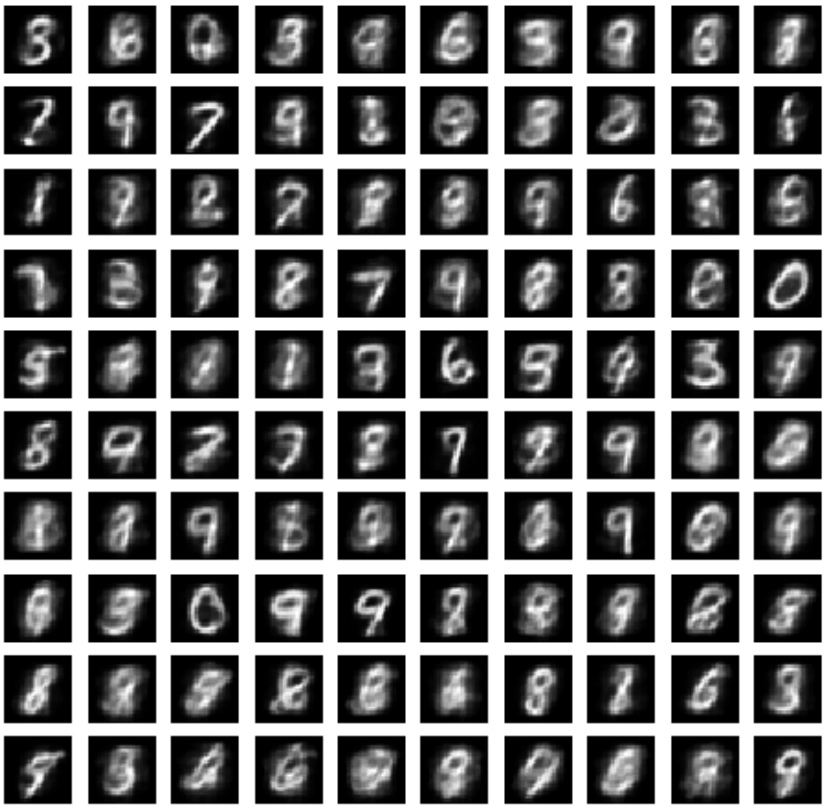

Figure 16: 100 sample images drawn from the trained network.

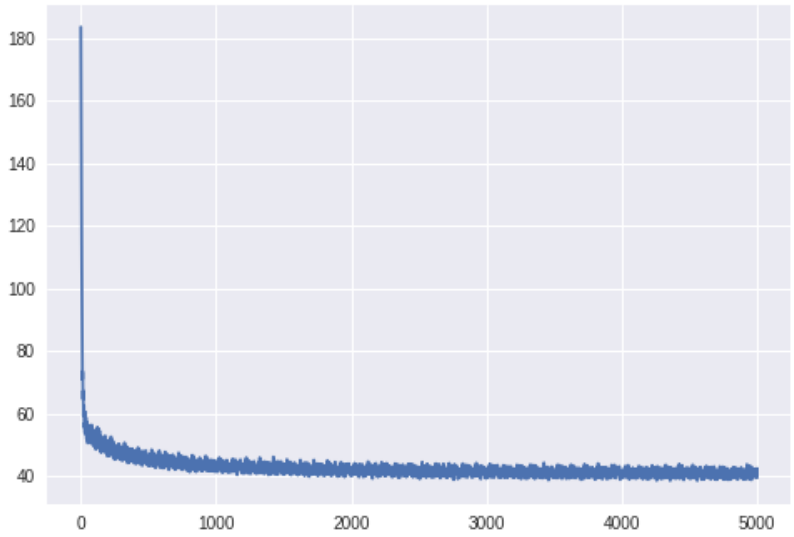

Figure 17: Progress of the training objective over rounds of training.

Code Appendix

Plotting of Loss Function Slices

The code below was used to plot 1-dimensional and 2-dimensional slices of the two loss functions to assess convexity.

```
# The values of the three coefficients are selected at random
vec0=10*randn(D)
vec1=10*randn(D)
vec2=10*randn(D)

# The number of steps in the [0, 1] interval to take
I=100

# The values for the two loss functions on 1 dimension
Loss11=zeros(I);
Loss12=zeros(I)

# The values for the two loss functions on 2 dimensions
Loss21=zeros(I,I);
Loss22=zeros(I,I)

for i=1:I
    # Compute the 1-dimensional error value for both losses
    lambda_i=i/I
    Loss11[i] = SquareLoss(c,NNpred(vec0+lambda_i*vec1,x))
    Loss12[i] = LikLoss(c,vec0+lambda_i*vec1,x)
    for j=1:I
        # Compute the 2-dimensional error value for both losses
```

```
        lambda_j=j/I
        Loss21[i,j] = SquareLoss(c,NNpred(vec0+lambda_i*vec1+lambda_j*vec2,x))
        Loss22[i,j] = LikLoss(c,vec0+lambda_i*vec1+lambda_j*vec2,x)
    end
end
```

MNIST Autoencoder

The following code was used in Question 3.

```
def sigmoid(x):
    return 1 / (1 + np.exp(0 - x))

def relu(x):
    return np.maximum(x,0)

def d_relu(x):
    return np.where(x > 0, np.full(np.shape(x), 1), np.full(np.shape(x), 0))

def squared_loss(y, x):
    return (y - x) ** 2 / 2

def d_squared_loss(y, x):
    return - (y - x)

def cross_entropy_loss(x,y):
    return -x*np.log(y) - (1-x)*np.log(1-y)

def d_cross_entropy_loss(x,y):
    return ((1-x)/(1-y)) - (x/y)

def create_weight(shape):
    stdv = 1. / np.sqrt(shape[-1])
    return np.random.uniform(-stdv, stdv, shape).astype(np.float32)

def autodiff(xs, w1, w2, b1, b2):
    """ Performs automatic differntiation for the required network.

    :param xs: the matrix of features
    :param w1, w2, b1, b2: the matrices and vectors for the pairs of weights and biases
    :return: the matrix representing the loss for each value in xs, the matrices
    representing the total derivative of the error function for each value
    in xs for w1, w2, b1 and b2, the matrix representing the error for each value in xs """
    N = xs.shape[1]
    D = xs.shape[0]
    f1 = w_1 @ xs
    f2 = (b_1 + f1.T).T
    f3 = relu(f2)
    f4 = w_2 @ f3
    f5 = (b_2 + f4.T).T
    f6 = sigmoid(f5)
    f7 = loss(xs, f6)
    f8 = np.sum(f7, axis=0)
    f9 = np.sum(f8)
    f10 = f9 / N
```

```
    t10 = 1
    t9 = 1 / N
    t8 = np.full((N), 1 / N)
    t7 = np.full((D, N), 1 / N)
    t6 = d_loss(xs, f6) * t7
    t5 = (f6*(1-f6)) * t6
    t4 = t5
    tb2 = np.sum(t5, axis=1)
    t3 = w_2.T @ t4
    tw2 = (f3 @ t4.T).T
    t2 = d_relu(f2) * t3
    t1 = t2
    tb1 = np.sum(t2, axis=1)
    tw1 = t1 @ xs.T
    return f6, tw1, tw2, tb1, tb2, f10

def adam(weight, grad, k, m_prev, v_prev, b1=0.9, b2=0.999, l=0.0000000001, rate=0.05):
    """ Performs ADAM optimisaton.

    :param weight: the list of weights to optimise
    :param grad: the gradient for the error function at the weights
    :param k: the round of optimisation
    :param m_prev, v_prev: the ADAM parameters at the previous round
    :return: the next round of optimisation, the ADAM parameters for the round """
    m = b1 * m_prev + (1 - b1)*grad
    v = b2 * v_prev + (1 - b2)*(grad ** 2)
    m_hat = m / (1 - (b1 ** k))
    v_hat = v / (1 - (b2 ** k))
    weight -= (rate / (np.sqrt(v_hat) + l))*m_hat
    return (k+1, m, v)

w1 = create_weight((30, 784))
w2 = create_weight((784, 30))
b1 = create_weight((30,))
b2 = create_weight((784,))

epochs = 100

past_epochs = []
past_loss_values = []

previous_runnings = [[1,0,0] for i in range(4)]
weights = [w1,w2,b1,b2]

for i in range(epochs):
    res, grad_w1, grad_w2, grad_b1, grad_b2, loss = run(features.T, w1, w2, b1, b2)
    grads = [grad_w1, grad_w2, grad_b1, grad_b2]
    for (j, (weight, previous_running, grad)) in enumerate(zip(weights, previous_runnings, grads)):
        k, m, v = previous_running
        previous_runnings[j] = adam(weight, grad, k, m, v)
    past_epochs.append(i)
    past_loss_values.append(loss)
```

MNIST Error Plotting

The following code was used to plot the 2d error slices

```
S = 10
w1, w2, b1, b2 = pickle.load(open("weights.p", "rb"))
u1 = np.random.normal(0, 1, w1.shape)
u2 = np.random.normal(0, 1, w1.shape)
values = np.zeros((S, S))
for i in range(0, S):
    z1 = i / S
    for j in range(0, S):
        z2 = j / S
        _, l = run(features.T, w1 + z1 * u1 + z2 * u2, w2, b1, b2, no_training=True)
        values[i][j] = l
    print("At {} of {}.".format(i + 1, S))
fig, ax = plt.subplots()
plt.plot(values)
plt.show()
```

MNIST VAE

The following code was used in Question 4.8.

```
import numpy as np
import gzip
import struct
import pickle
from matplotlib import pyplot as plt
import torch
import torchvision
from torch import nn
from torch import optim
from torchvision import transforms

# Obtain MNIST data

def get_features(filename):
    with gzip.open(filename, 'rb') as f:
        zero, data_type, dims = struct.unpack('>HBB', f.read(4))
        shape = tuple(struct.unpack('>I', f.read(4))[0] for d in range(dims))
        return np.fromstring(f.read(), dtype=np.uint8).reshape(shape)

def plot_image(image):
    plt.imshow(image.reshape((28,28)), cmap="gray")

features = get_features("train_features.gz").reshape((100,600,784)) / 255
features = torch.from_numpy(features).float()

# Define VAE structure

class VAE(nn.Module):
  def __init__(self):
    super(VAE, self).__init__()
```

```
    self.relu = nn.ReLU()
    self.sigmoid = nn.Sigmoid()
    self.dropout = nn.Dropout2d()

    #encoder
    self.e_fc1 = nn.Linear(784,400)
    self.e_fc2 = nn.Linear(400,200)
    self.e_fc3 = nn.Linear(200,40)

    #decoder

    self.d_fc1 = nn.Linear(40,400)
    self.d_fc2 = nn.Linear(400,784)


  def forward(self, input, generating=False):
    # encoding
    mean, var = None, None
    if not generating:
      res = self.relu(self.e_fc1(input))
      res = self.relu(self.e_fc2(res))
      mean = self.relu(self.e_fc3(res))
      samples = torch.randn_like(mean)
      input = mean + samples

    # decoding
    res = self.relu(self.d_fc1(input))
    res = self.sigmoid(self.d_fc2(res))
    return (res,mean)

# Initialise VAE and train

net = VAE()

epochs = 50
optimizer = optim.Adam(net.parameters())
losses = []

def loss_function(recon_x, x, mu):
    RL = ((x - recon_x) ** 2).sum()
    KLD = 0.5 * torch.sum(mu.pow(2))

    return (RL + KLD) / 600

for i in range(epochs):
  for j in range(100):
    data = features[j,:,:]
    optimizer.zero_grad()
    generated, mean = net(data)
    loss = loss_function(generated, data, mean)
    loss.backward()
    optimizer.step()
    losses.append(loss.item())
```