Machine Learning Assignment 2: Neural Networks

Fabian Flores Gobet

19 November 2023

Question 1

To solve this exercise lets first define some notation, namely

$$
\begin{aligned}
z_k &= x_k \cdot W^{(1)} + b^{(1)} \\
\Big[z_k\Big]_m &= b_m^{(1)} + \sum_{n=1}^{2} \Big[x_k\Big]_n W_{nm}^{(1)} \\
f(z_k) &= \Big[f\Big([z_k]_1\Big),\ f\Big([z_k]_2\Big),\ f\Big([z_k]_3\Big)\Big] = \Big[f\Big([z_k]_i\Big)\Big]_i \\
f(Z) &= \Big[f(z_1),\ f(z_2),\ f(z_3),\ f(z_4)\Big]^T = \Big[f(z_i)\Big]_{i,1}
\end{aligned}
$$

x_k	row vector
$(y-t)$	column vector
z_k	row vector
$f(z_k)$	row vector
$f(Z)$	column vector

- $\frac{\partial L}{\partial b^{(2)}}$

$$
\begin{aligned}
\frac{\partial L}{\partial b^{(2)}} &= \frac{\partial}{\partial b^{(2)}}\Big(\frac{1}{2}\sum_{k=1}^{4}(y_k - t_k)^2\Big) \\
= \sum_{k=1}^{4}(y_k - t_k)\frac{\partial y_k}{\partial b^{(2)}} &= \sum_{k=1}^{4}(y_k - t_k) \\
&= (y-t)^T \cdot \mathbf{1}_4
\end{aligned}
$$

where $\mathbf{1}_4$ is a column vector of 1s.

- $\frac{\partial L}{\partial W^{(2)}}$

$$\begin{aligned}
\frac{\partial L}{\partial W^{(2)}} &= \frac{\partial}{\partial W^{(2)}}\Big(\frac{1}{2}\sum_{k=1}^{4}(y_k - t_k)^2\Big) \\
= \sum_{k=1}^{4}(y_k - t_k)\frac{\partial y_k}{\partial W^{(2)}} &= \sum_{k=1}^{4}(y_k - t_k) f(z_k) \\
&= (y-t)^T \cdot f(Z)
\end{aligned}$$

- $\frac{\partial L}{\partial b^{(1)}}$

$$\begin{aligned}
\frac{\partial L}{\partial b^{(1)}} &= \frac{\partial}{\partial b^{(1)}}\Big(\frac{1}{2}\sum_{k=1}^{4}(y_k - t_k)^2\Big) \\
&= \sum_{k=1}^{4}(y_k - t_k)\frac{\partial y_k}{\partial b^{(1)}} \qquad (1)
\end{aligned}$$

Lets solve $\frac{\partial y_k}{\partial b^{(1)}}$ and then return to (1), hence lets consider the partial derivative with respect to component i of $b^{(1)}$, i.e. $\frac{\partial y_k}{\partial b_i^{(1)}}$. Then

$$\begin{aligned}
\frac{\partial y_k}{\partial b_i^{(1)}} &= \frac{\partial}{\partial b_i^{(1)}}\Big(b^2 + \sum_{m=1}^{3}\Big[W_m^{(2)} f([z_k]_m)\Big]\Big) \\
&= \sum_{m=1}^{3}\Big[W_m^{(2)}\frac{\partial}{\partial b_i^{(1)}}\Big(f([z_k]_m)\Big)\Big] \\
&= \sum_{m=1}^{3}\Big[W_m^{(2)} f'([z_k]_m)\frac{\partial [z_k]_m}{\partial b_i^{(1)}}\Big] \\
= \sum_{m=1}^{3}\Big[W_m^{(2)} f'([z_k]_m)\frac{\partial}{\partial b_i^{(1)}}&\Big(b_m^{(1)} + \sum_{n=1}^{2}\Big[x_k\Big]_n W_{nm}^{(1)}\Big)\Big] \\
= \sum_{m=1}^{3}\Big[W_m^{(2)} f'([z_k]_m)\delta_{im}\Big] &= W_i^{(2)} f'([z_k]_i)
\end{aligned}$$

Generalizing for $b^{(1)}$ we have

$$\frac{\partial y_k}{\partial b^{(1)}} \;=\; \Big[W_i^{(2)} f'([z_k]_i)\Big]_i \;=\; f'(z_k) \cdot diag(W^{(2)})$$

where $diag(W^{(2)})$ is a diagonal matrix with the components of $W^{(2)}$.

Returning to (1) comes

$$\begin{aligned}(1) \;&=\; \sum_{k=1}^{4} (y_k - t_k) \frac{\partial y_k}{\partial b^{(1)}} = \sum_{k=1}^{4} (y_k - t_k) f'(z_k) \cdot diag(W^{(2)}) \\ &=\; (y-t)^T \cdot f'(Z) \cdot diag(W^{(2)})\end{aligned}$$

- $\frac{\partial L}{\partial W^{(1)}}$

  Just like in (1), lets first calculate $\frac{\partial y_k}{\partial W_{ij}^{(1)}}$, generalize and form a conclusion about $\frac{\partial L}{\partial W^{(1)}}$. Hence

$$\begin{aligned}\frac{\partial y_k}{\partial W_{ij}^{(1)}} \;&=\; \frac{\partial}{\partial W_{ij}^{(1)}} \Big(b^2 + \sum_{m=1}^{3} \Big[W_m^{(2)} f([z_k]_m) \Big] \Big) \\ &=\; \sum_{m=1}^{3} \Big[W_m^{(2)} f'([z_k]_m) \frac{\partial}{\partial W_{ij}^{(1)}} \Big(b_m^{(1)} + \sum_{n=1}^{2} \Big[x_k \Big]_n W_{nm}^{(1)} \Big) \Big] \\ &=\; \sum_{m=1}^{3} \Big[W_m^{(2)} f'([z_k]_m) \Big(\sum_{n=1}^{2} \Big[x_k \Big]_n \frac{\partial W_{nm}^{(1)}}{\partial W_{ij}^{(1)}} \Big) \Big] \\ &=\; W_j^{(2)} f'([z_k]_j) \Big[x_k \Big]_i\end{aligned}$$

Generalizing for $W^{(1)}$ we have

$$\frac{\partial y_k}{\partial W^{(1)}} \;=\; \Big[W_j^{(2)} f'([z_k]_j) [x_k]_i \Big]_{ij} \;=\; x_k^T \cdot f'(z_k) \cdot diag(W^{(2)}) \;=\; T(x_k).$$

Notice both x_k and $f'(z_k)$ are row vectors of size 2 and 3, respectively.
Thus $x_k^T \cdot f'(z_k)$ produces a 2x3 matrix ($(\cdot)$ is the matrix multiplication operator).

Lets consider $T = \Big[T(x_1),\ T(x_2),\ T(x_3),\ T(x_4)\Big]^T$, then finally

$$\frac{\partial L}{\partial W^{(1)}} \ = \ \sum_{k=1}^{4}(y_k - t_k)\frac{\partial y_k}{\partial W^{(1)}} =$$

$$= \ \sum_{k=1}^{4}(y_k - t_k)T(x_k) \ = \ (y-t)^T \cdot T$$

- Calculating the gradients

  We are now in conditions of calculating the gradients. As such, I did a very simple python implementation with numpy where i deployed all the calculated formulas

```
import numpy as np
x = np.array([[0.6, -1.0], [0.8, -1.0], [-0.4, 0.9], [0.2, 0.0]])
t = np.array([-.8, -0.1, 0.9, 0.7]).reshape((4,1))
w1 = np.array([[-.8,-0.7,0.6],[-1.0,0.5,-1.0]]).reshape((2,3))
b1 = np.array([-0.2, -1.0, -0.7]).reshape((1,3))
w2 = np.array([0.1, -1.0, 0.5]).reshape((1,3))
b2 = np.array(-0.7)
z = x@w1+b1
zr = np.maximum(0,z)
y = w2@np.transpose(zr) + b2
y = np.transpose(y)
db2 = np.transpose(y-t)@np.ones_like(y-t)
dw2 = np.transpose(y-t)@zr
dzr = (z > 0).astype(int)
diagw2 = np.diag(w2.reshape(-1))
db1 = np.transpose(y-t)@dzr@diagw2
def T(k):
  xk = x[k].reshape((2,1))
  dzrk = dzr[k].reshape((1,3))
  first = xk@dzrk
  return first@diagw2
T = [T(i) for i in range(4)]
T_yt = [a*b for a,b in zip(y-t,T)]
dw1 = np.sum(T_yt, axis=0)
```

  As such, the results are

$$\frac{\partial L}{\partial b^{(2)}} = -2.732 \qquad \frac{\partial L}{\partial W^{(2)}} = \begin{bmatrix}0.1168 & 0 & 0.1536\end{bmatrix}$$

$$\frac{\partial L}{\partial b^{(1)}} = \begin{bmatrix}0.0268 & 0 & 0.134\end{bmatrix} \qquad \frac{\partial L}{\partial W^{(1)}} = \begin{bmatrix}0.0122 & 0 & 0.061 \\ -0.0268 & 0 & -0.134\end{bmatrix}$$

Question 2

Before differentiating in order to y_i, let's first modify E to another form so that this process becomes simpler. Assuming log() denotes the natural logarithm, we have

$$
\begin{aligned}
E &= -\sum_k t_k \log\Big(\frac{\exp(y_k)}{\sum_i \exp(y_i)}\Big) \\
&= -\sum_k \Big[t_k \log\Big(\exp(y_k)\Big) - t_k \log\Big(\sum_i \Big[\exp(y_i)\Big]\Big)\Big] \\
&= \log\Big(\sum_i \Big[\exp(y_i)\Big]\Big)\sum_k \Big[t_k\Big] - \sum_k \Big[t_k y_k\Big]
\end{aligned}
$$

Then

$$
\begin{aligned}
\frac{\partial E}{\partial y_i} &= \frac{\partial}{\partial y_i}\Big(\log\Big(\sum_i \Big[\exp(y_i)\Big]\Big)\sum_k \Big[t_k\Big] - \sum_k \Big[t_k y_k\Big]\Big) \\
&= \frac{\partial}{\partial y_i}\Big(\log\Big(\sum_i \Big[\exp(y_i)\Big]\Big)\Big)\sum_k \Big[t_k\Big] - \frac{\partial}{\partial y_i}\Big(\sum_k \Big[t_k y_k\Big]\Big) \\
&= \frac{\exp(y_i)}{\sum_i \Big[\exp(y_i)\Big]}\sum_k \Big[t_k\Big] - t_i
\end{aligned}
$$

If t is a one-hot vector, then only one of its components is 1, whereas the rest are 0. Let's assume that the component of index k is 1, i.e. $t_k = \delta_{ik}$ (where δ is the Kronecker delta). As a result we have that

$$
\frac{\partial E}{\partial y_i} = \frac{\exp(y_i)}{\sum_i \Big[\exp(y_i)\Big]} t_k - t_i
$$

Question 3

For this exercise, because the minimum is found at $x = 0$, lets suppose that $x_1 \neq 0$. We know that the derivative $f(x)$ is $f'(x) = 10x$. Then

$$\begin{aligned}
x_1 &= x_1 \\
x_2 &= x_1(1 - 10\eta) \\
x_3 &= x_1(1 - 10\eta)^2 \\
&\cdots \\
x_{n+1} &= x_1(1 - 10\eta)^n
\end{aligned}$$

In order for the gradient descent to converge for any initial point $x_1 \neq 0$, it has to satisfy $f(x_{n+1}) < f(x_n)$. Hence we have

$$\begin{aligned}
f(x_{n+1}) &< f(x_n) \\
\equiv\ 5(x_1(1 - 10\eta)^n)^2 &< 5(x_1(1 - 10\eta)^{n-1})^2 \\
\equiv\ (1 - 10\eta)^{2n} &< (1 - 10\eta)^{2(n-1)} \\
\equiv\ (1 - 10\eta)^{2(n-1)}((1 - 10\eta)^2 - 1) &< 0 \\
\equiv\ (1 - 10\eta)^{2(n-1)}100\eta(\eta - \frac{1}{5}) &< 0
\end{aligned}$$

The term $(1 - 10\eta)^{2(n-1)}$ is always positive because it has an even power. Therefore, we have to look at the term $\eta(\eta - \frac{1}{5})$ and determine when its less than 0. Hence

$$\begin{aligned}
100\eta(\eta - \frac{1}{5}) &< 0 \\
\eta &\in (0, \frac{1}{5})
\end{aligned}$$

In conclusion, for values of $\eta \in (0, \frac{1}{5})$, for the given polynomial $f(x)$ we can ensure that gradient descent always converges. In the trivial case of $x_1 = 0$ the minimum has been achieved.