Seyed-Vahid Sanei-Mehri, HW2

October 6, 2018

1 Problem 1

Solution: We know that for $f(x) = \frac{h(x)}{g(x)}$, $f'(x) = \frac{g'(x)h(x)-h'(x)g(x)}{h^2(x)}$. Let say $g(i) = e^{z_i}$, and $h(i) = \sum_{k=1}^{n} e^{z_k}$. For the two following cases, we have:

1) $i = j$:

$$\frac{\partial \frac{e^{z_i}}{\sum_{k=1}^{n} e^{z_k}}}{\partial z_j} = \frac{e^{z_j} \sum_{k=1}^{n} e^{z_k} - e^{z_j} e^{z_i}}{\left(\sum_{k=1}^{n} e^{z_k}\right)^2}$$

2) $i \neq j$:

$$\frac{\partial \frac{e^{z_j}}{\sum_{k=1}^{n} e^{z_k}}}{\partial z_j} = \frac{0 - e^{z_j} e^{z_i}}{\left(\sum_{k=1}^{n} e^{z_k}\right)^2}$$

By definition $p_i = \frac{e^{z_i}}{\sum_{k=1}^{n} e^{z_k}}$. Therefore, we derive:

$$\frac{\partial p_j}{\partial z_i} = \begin{cases} p_i(1-p_j), & i = j \\ -p_i p_j, & i \neq j \end{cases}$$

With Kronecker delta,

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

We have:

$$\frac{\partial p_j}{\partial z_i} = p_i(\delta_{ij} - p_j)$$

2 Problem 2

Solution:

- Part (a)

$$\frac{\partial J}{\partial z_j} = -\sum_{i=1}^{n} y_i \frac{\partial \log(p_i)}{\partial z_j}$$

We the chain rule we have:

$$\frac{\partial J}{\partial z_j} = -\sum_{i=1}^{n} y_i \frac{\partial \log(p_i)}{\partial p_i} \times \frac{\partial p_i}{\partial z_j}$$

We know that log is natural logarithm, then $\frac{\log(p_i)}{\partial p_i} = \frac{1}{p_i \ln(e)} = \frac{1}{p_i}$. Also using the derivative of softmax function we have:

$$\frac{\partial J}{\partial z_j} = -y_j(1-p_j) - \sum_{i \neq j}^{n} \frac{y_i}{p_i}(-p_i \cdot p_j)$$

$$= -y_j + y_j \cdot p_j + \sum_{i \neq j}^{n} \frac{y_i}{p_i}(p_i \cdot p_j)$$

$$= -y_j + p_j \times \left(y_j + \sum_{i \neq j}^{n} y_i \right)$$

$$= -y_j + p_j \times \left(\sum_{i=1}^{n} y_i \right)$$

By definition, we know that $\sum_{i=1}^{n} y_i = 1$. Therefore,

$$\frac{\partial J}{\partial z_j} = p_j - y_j$$

- Part (b) code is attached. Code for problem 3 & 4 is also attached.

3 Problem 3

Solution: iteration = 0
z = [[0.0, 0.0], [0.0, 0.0]]
dz = [[0.25, -0.25], [-0.25, 0.25]]
p = [[0.5, 0.5], [0.5, 0.5]]
dW = [[0.25, -0.25], [-0.25, 0.25]]
db = [[0.25, -0.25], [-0.25, 0.25]]
iteration = 1
z = [[-0.5, 0.5], [0.5, -0.5]]
dz = [[0.36552928931500245, -0.36552928931500245], [-0.36552928931500245, 0.36552928931500245]]
p = [[0.2689414213699951, 0.7310585786300049], [0.7310585786300049, 0.2689414213699951]]
dW = [[0.36552928931500245, -0.36552928931500245], [-0.36552928931500245, 0.36552928931500245]]
db = [[0.36552928931500245, -0.36552928931500245], [-0.36552928931500245, 0.36552928931500245]]

4 Problem 4

Solution:

- (a) Single layer, 10 neurons, softmax + cross entropy objective function.
  iter 0, J=3.365241, accu=0.14
  iter 1000, J=2.327228, accu=0.17

iter 2000, J=2.709532, accu=0.17
iter 3000, J=2.212323, accu=0.25
iter 4000, J=2.030988, accu=0.38
iter 5000, J=1.940232, accu=0.39
iter 6000, J=1.639684, accu=0.44
iter 7000, J=1.473641, accu=0.51
iter 8000, J=1.523897, accu=0.50
iter 9000, J=1.383028, accu=0.53

- (b) Three layers, [(50, ReLU), (50, ReLU), (10, Linear)], softmax + cross entropy objective function.
iter 0, J=2.321385, accu=0.13
iter 1000, J=2.137152, accu=0.28
iter 2000, J=2.041333, accu=0.41
iter 3000, J=1.867324, accu=0.47
iter 4000, J=1.612603, accu=0.53
iter 5000, J=1.491312, accu=0.57
iter 6000, J=1.416254, accu=0.61
iter 7000, J=1.295941, accu=0.65
iter 8000, J=1.215856, accu=0.68
iter 9000, J=1.103257, accu=0.70

- (c) for [(70, ReLU), (70, ReLU), (10, ReLU)] we get:
iter 0, J=2.431493, accu=0.14
iter 1000, J=0.665606, accu=0.83
iter 2000, J=0.643690, accu=0.87
iter 3000, J=0.392387, accu=0.88
iter 4000, J=0.399857, accu=0.89
iter 5000, J=0.476496, accu=0.89
iter 6000, J=0.312687, accu=0.90
iter 7000, J=0.348203, accu=0.90
iter 8000, J=0.414610, accu=0.90
iter 9000, J=0.279489, accu=0.90