EE412 Foundation of Big Data Analytics, Fall 2019
HW4

Name: 오승준

Student ID: 20160381

Discussion Group (People with whom you discussed ideas used in your answers):
손채연 (Activation function, Implement fully-connected network, and proper learning rate)

On-line or hardcopy documents used as part of your answers:
Problem1-(b) - https://tutorials.pytorch.kr/beginner/pytorch_with_examples.html

Answer to Problem 1

(a) Compute the gradients using the chain rule.

Let, Input: x and Output: o. Then,

$$\mathrm{W1} = \begin{bmatrix} w_{11}^1 & w_{21}^1 \\ w_{12}^1 & w_{22}^1 \end{bmatrix}, \mathrm{W2} = \begin{bmatrix} w_{11}^2 & w_{21}^2 \\ w_{12}^2 & w_{22}^2 \end{bmatrix}$$

$$\mathrm{p} = \mathrm{W1} * \mathrm{x}, \mathrm{q} = \sigma(p), r = W2 * q, o = \sigma(r)$$

$$\mathrm{Loss} = \frac{1}{2}\sum_i (o_i - y_i)^2, \mathrm{g(o)} = o - y$$

$$\textit{which means,} \qquad g(o_1) = o_1 - y_i, g(o_2) = o_2 - y_2$$

- Compute the gradient of the loss with respect to w_{ij}^2

$$\mathrm{g}(r_i) = J_r(o_i)g(o_i) = o_i(1 - o_i) * g(o_i) = (\mathrm{o_i}(1 - o_i))(o_i - y_i)$$

$$\text{Since } \mathrm{r_i} = \mathrm{w_{1i}^2 q_1} + \mathrm{w_{2i}^2} q_2,$$

$$\therefore \mathrm{g}(\mathrm{w_{ij}^2}) = \mathrm{J}_{\mathrm{w_{ij}^2}}(r_j)g(r_j) = q_i * g(r_j) = \mathrm{q_i}(o_j(1 - o_j))(o_j - y_j)$$

- Compute the gradient of the loss with respect to w_{ij}^1

$$\text{Similarly, } \mathrm{g(q_i)} = \mathrm{J_{q_i}}(r_1)g(r_1) + J_{q_i}(r_2)g(r_2) = (w_{i1}^2 g(r_1) + w_{i2}^2 g(r_2))$$

$$\mathrm{g(p_i)} = \mathrm{J_{p_i}}(q_i)g(q_i) = (\mathrm{q_i}(1 - q_i))g(q_i)$$

$$\therefore \mathrm{g}(\mathrm{w_{ij}^1}) = \mathrm{J}_{\mathrm{w_{ij}^1}}(p_j)g(p_j) = p_i * g(p_j) = p_i\left(q_j(1 - q_j)\right)g(q_j)$$

$$= p_i\left(q_j(1 - q_j)\right)\{\mathrm{w_{j1}^2}(o_1(1 - o_1))(o_1 - y_1) + w_{j2}^2(o_2(1 - o_2))(o_2 - y_2)\}$$

(b) Implement a fully-connected network to distinguish digits using Python

10 times output

1	2	3	4	5	6	7	8	9	10
0.97 0.797 1000 0.03	0.868 0.749 1000 0.03	0.958 0.796 1000 0.03	0.96 0.826 1000 0.03	0.768 0.665 1000 0.03	0.965 0.81 1000 0.03	0.954 0.798 1000 0.03	0.871 0.742 1000 0.03	0.865 0.724 1000 0.03	0.97 0.81 1000 0.03

Sometimes, test accuracy is below 0.7. It might because of initialization of W1 and W2.

Answer to Problem 2

(a) Solve the following problems.

- Exercise 4.4.1 and 4.4.2

  a. h(x) = 2x + 1 mod 32

Stream	3	1	4	1	5	9	2	6	5
h(x)	7	3	9	3	11	19	5	13	11
Tail length	0	0	0	0	0	0	0	0	0

Estimate of the number of distinct elements = $2^0 = 1$

  b. h(x) = 3x + 7 mod 32

Stream	3	1	4	1	5	9	2	6	5
h(x)	16	10	19	10	22	2	13	25	22
Tail length	4	1	0	1	1	1	0	0	1

Estimate of the number of distinct elements = $2^4 = 16$

  c. h(x) = 4x mod 32

Stream	3	1	4	1	5	9	2	6	5
h(x)	12	4	16	4	20	4	8	24	20
Tail length	2	2	4	2	2	2	3	3	2

Estimate of the number of distinct elements = $2^4 = 16$

Problem:

- If ‘a’ is even number and ‘b’ is odd number, result of hash is always odd number, so its tail length is always 0. Therefore, we cannot estimate the number of distinct elements properly. We have to avoid such situation.

- Exercise 4.5.3

Stream: 3, 1, 4, 1, 3, 4, 2, 1, 2

i	1	2	3	4	5	6	7	8	9
X_i.element	3	1	4	1	3	4	2	1	2
X_i.value	2	3	2	2	1	1	2	1	1

(b) Implement the DGIM algorithm.

Sample command line

```
python hw4_2_p2.py stream.txt 0 1 2 4 8 10 20 38 50 100 500 1000 52486
3000000
```

Output

```
0
0.5
0.5
0.5
2
2
5
11
11
23
95
191
9471
560383
```