Optimization and Machine Learning, Fall 2023
Homework 5
(Due Thursday, Jan 11 at 11:59pm (CST))

1. [10 points] [Deep Learning Model]

   (a) Consider a sequential 2D convolution block consist of 10 layers. Suppose the input size is $4 \times 64 \times 64 \times$ (channel, width, height) and we use 3×3 (width, height) Conv2D with 4 channels input and 4 channels output to convolve with it. Set stride $= 1$ and pad $= 1$. What is the output size? Let the bias for each kernel be a scalar, how many parameters do we have in the ? [5 points]

   (b) The convolution layer is followed by a max pooling layer with 2×2 (width, height) filter and stride $= 2$. What is the output size of the pooling layer? How many parameters do we have in the pooling layer? [5 points]

   (a) The output channel is 4, so the output size is $4 \times 64 \times 64$
   parameters: $10 \times 4 \times (4 \times 3 \times 3 + 1) = 1480$

   (b) $64 \div 2 = 32 \Rightarrow$the output has $4 \times 32 \times 32$
   and the pooling layer has no parameters.

2. [10 points] Use the k-means++ algorithm and Euclidean distance to cluster the 8 data points into $K = 3$ clusters. The coordinates of the data points are:

$$x^{(1)} = (2, 8),\ x^{(2)} = (2, 5),\ x^{(3)} = (1, 2),\ x^{(4)} = (5, 8),$$
$$x^{(5)} = (7, 3),\ x^{(6)} = (6, 4),\ x^{(7)} = (8, 4),\ x^{(8)} = (4, 7).$$

Suppose that initially the first cluster centers is $x^{(1)}$.
To ensure consistent results, please use random numbers in the order shown in the table below. When selecting a center, arrange it in ascending order of sequence number. For example, when the normalized weights of 5 nodes are 0.2, 0.1, 0.3, 0.3, and 0.1, if the random number is 0.3, the selected node is the third one. Note that you don't necessarily need to use all of them.

0.6	0.2	0.5	0.9	0.3

(a) Perform the k-means++ algoptionorithm to initialize other centers and report the coordinates of the resulting centroids. [3 points]

(b) Calculate the loss function

$$Q(r, c) = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{K} r_{ij} ||x^{(i)} - c_j||^2, \tag{1}$$

where $r_{ij} = 1$ if $x^{(i)}$ belongs to the j-th cluster and 0 otherwise. [2 points]

(c) How many more iterations are needed to converge? [3 points] Calculate the loss after it converged. [2 points]

(a)

$$c_1 = x_1$$

$\Rightarrow$

x_i	x_2	x_3	x_4	x_5	x_6	x_7	x_8
$d^2(c_1, x_i)$	9	37	9	50	32	52	5

$\Rightarrow S = 194 \Rightarrow$

$d^2(c_1, x_i)$	x_2	x_3	x_4	x_5	x_6	x_7	x_8
p	0.046	0.191	0.046	0.258	0.165	0.268	0.026

$$\text{random is } 0.6 \Rightarrow c_2 = x^{(6)}$$

$\Rightarrow$

x_i	x_2	x_3	x_4	x_5	x_7	x_8
$d^2(c_2, x_i)$	17	29	17	2	4	25

$\Rightarrow$

x_i	x_2	x_3	x_4	x_5	x_7	x_8
$d^2_{\min}(x_i)$	9	29	9	2	4	5

$$\text{random is } 0.2 \Rightarrow c_3 = x^{(3)}$$
$$\Rightarrow c = \{x^{(1)}, x^{(6)}, x^{(3)}\}$$

(b) from (a):

$$\begin{cases} x^{(1)}: & x^{(2)}, x^{(4)}, x^{(8)} \\ x^{(6)}: & x^{(5)}, x^{(7)} \\ x^{(3)}: & \end{cases}$$

$$\Rightarrow Q(r, c) = \frac{1}{8}(9 + 9 + 5 + 2 + 4) = \frac{29}{8}$$

(c)

$$means: \begin{cases} c_1: & \left(\frac{13}{4}, 7\right) \\ c_2: & (1, 2) \\ c_3: & \left(7, \frac{11}{3}\right) \end{cases}$$

$$\Rightarrow \begin{cases} x^{(1)}: x^{(2)}, x^{(4)}, x^{(8)} \\ x^{(6)}: x^{(5)}, x^{(7)} \\ x^{(3)}: \end{cases}$$

$$\Rightarrow Q(r, c) = 1.93$$

3. [10 points] Name 2 deep generation networks. [2 points] Briefly describe the training procedure of a GAN model.(What's the objective function? How to update the parameters in each stage?) [8 points]

(a) CNN, RNN

(b) **Objective Functions**

1. Discriminator (D) Objective: Distinguish between real data x and fake data generated by the Generator $G(z)$.

2. Generator (G) Objective: Generate data that mimics real data as closely as possible.

Training Stages

1. Training Discriminator (D):

- Train D to output high probability $D(x)$ for real data x.
- Train D to output low probability $D(G(z))$ for fake data $G(z)$ from Generator.
- Update D's parameters using gradient descent.

2. Training Generator (G):

- G generates data $G(z)$, passed to D.
- Goal of G is to make D classify $G(z)$ as real.
- Update G's parameters using gradient ascent based on D's output.

Loss Functions

Discriminator's Loss: Binary cross-entropy, aiming to accurately classify real and fake data.

Generator's Loss: Inversely related to the confidence of D in mistaking $G(z)$ as real.

Training involves alternating updates between G and D until equilibrium is reached.