CS280 Fall 2023 Assignment 1
Part A

Basics & MLP

October 31, 2023

Name: zhao yu

Student ID:2023232115

1. Gradient descent for fitting GMM (10 points).

Consider the Gaussian mixture model

$$p(\mathbf{x}|\theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

where $\pi_j \geq 0, \sum_{j=1}^{K} \pi_j = 1$. (Assume $\mathbf{x}, \boldsymbol{\mu}_k \in \mathbb{R}^d, \boldsymbol{\Sigma}_k \in \mathbb{R}^{d\times d}$)

Define the log likelihood as

$$l(\theta) = \sum_{n=1}^{N} \log p(\mathbf{x}_n|\theta)$$

Denote the posterior responsibility that cluster k has for datapoint n as follows:

$$r_{nk} := p(z_n = k|\mathbf{x}_n, \theta) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{k'} \pi_{k'} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_{k'}, \boldsymbol{\Sigma}_{k'})}$$

(a) Show that the gradient of the log-likelihood wrt $\boldsymbol{\mu}_k$ is

$$\frac{d}{d\boldsymbol{\mu}_k} l(\theta) = \sum_n r_{nk} \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k)$$

Show:

the log-likelihood $l(\theta)$ in terms of the Gaussian mixture model:

$$l(\theta) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)$$

take the derivative with respect to $\boldsymbol{\mu}_k$:

$$\frac{d}{d\boldsymbol{\mu}_k} l(\theta) = \sum_{n=1}^{N} \frac{1}{\sum_{k'} \pi_{k'} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_{k'}, \boldsymbol{\Sigma}_{k'})} \cdot \frac{d}{d\boldsymbol{\mu}_k} \left(\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)$$

the derivative inside the summation:

$$\frac{d}{d\boldsymbol{\mu}_k} \left(\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right) = \pi_k \frac{d}{d\boldsymbol{\mu}_k} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

the derivative of the multivariate Gaussian density function with respect to $\boldsymbol{\mu}_k$ is:

$$\frac{d}{d\boldsymbol{\mu}_k} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = -\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k) \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

substitute this back into up expression:

$$\frac{d}{d\boldsymbol{\mu}_k} l(\theta) = \sum_{n=1}^{N} \frac{\pi_k \cdot -\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k) \cdot \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{k'} \pi_{k'} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_{k'}, \boldsymbol{\Sigma}_{k'})}$$

Finally, simplify the expression using the responsibility r_{nk}:

$$\frac{d}{d\boldsymbol{\mu}_k} l(\theta) = \sum_{n=1}^{N} r_{nk} \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k)$$

(b) Derive the gradient of the log-likelihood wrt π_k without considering any constraint on π_k. (bonus 2 points: with constraint $\sum_k \pi_k = 1$.)

Given the definition of $l(\theta)$ and r_{nk} from the previous context, we have:

$$l(\theta) = \sum_{n=1}^{N} \log p(\mathbf{x}_n|\theta)$$

To derive $\frac{d}{d\pi_k} l(\theta)$, take the derivative inside the summation:

$$\sum_{n=1}^{N} \frac{d}{d\pi_k} \log p(\mathbf{x}_n|\theta) = \sum_{n=1}^{N} \frac{d}{d\pi_k} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)$$

Simplifying, we have:

$$\sum_{n=1}^{N} \frac{1}{\sum_{k=1}^{K} \pi_k}$$

So, the derivative of the log-likelihood with respect to π_k is max $\frac{1}{\pi_i}, i = 1 \to K$.

Bonus:

Considering, with constraint $\sum_k \pi_k = 1$,

We define the Lagrangian function:

$$L(\theta, \lambda) = l(\theta) + \lambda \left(1 - \sum_k \pi_k \right)$$

we take partial derivatives with respect to π_k and λ, and set them equal to zero to find the optimal solution:

Taking the partial derivative with respect to π_k:

$$\frac{\partial L}{\partial \pi_k} = \frac{1}{\pi_k} - \lambda = 0$$

Solving for π_k:

$$\pi_k = \frac{1}{\lambda}$$

Taking the partial derivative with respect to λ:

$$\frac{\partial L}{\partial \lambda} = 1 - \sum_k \pi_k = 1 - K \cdot \frac{1}{\lambda} = 0$$

Solving for λ:

$$\lambda = K$$

Substituting $\lambda = K$ back into $\pi_k = \frac{1}{\lambda}$, we get:

$$\pi_k = \frac{1}{K}$$

This means that under the constraint $\sum_k \pi_k = 1$, the derivative of the log-likelihood $l(\theta)$ with respect to π_k remains $\frac{1}{\pi_k}$, but now π_k follows a uniform distribution, where each π_k is equal to $\frac{1}{K}$.

2. Sotfmax & Computation Graph (10 points).

Recall that the softmax function takes in a vector $(z_1, \ldots, z_D)$ and returns a vector $(y_1, \ldots, y_D)$. We can express it in the following form:

$$r = \sum_j e^{z_j} \qquad y = \frac{e^{z_j}}{r}$$

(a) Consider $D = 2$, i.e. just two inputs and outputs to the softmax. Draw the computation graph relating z_1, z_2, r, y_1, and y_2.

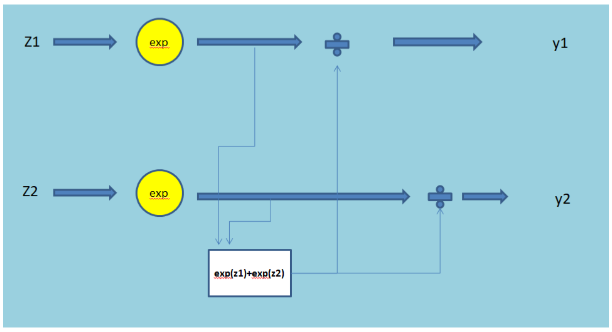

Figure 1: An image of a computation

(b) Determine the backprop updates for computing the $\bar{z}_j$ when given the $\bar{y}_i$. You need to justify your answer. (You may give your answer either for $D = 2$ or for the more general case.)

$$\bar{r} = -\sum_i \bar{y}_i \frac{e^{\bar{z}_i}}{r^2}$$

$$\bar{z}_j = \bar{y}_j \frac{e^{\bar{z}_j}}{r} + \bar{r} e^{\bar{z}_j}$$

(c) Write a function to implement the vector-Jacobian product (VJP) for the softmax function based on your answer from part (b). For efficiency, it should operate on a mini-batch. The inputs are:

- a matrix Z of size $N \times D$ giving a batch of input vectors. N is the batch size and D is the number of dimensions. Each row gives one input vector $z = (z_1, \ldots, z_D)$.
- A matrix $\mathbf{Y_{bar}}$ giving the output error signals. It is also $N \times D$

The output should be the error signal $\mathbf{Z_{bar}}$. Do not use a for loop.

function:

```
import numpy as np

def softmax_vjp(Z, Y_bar):
    R = np.sum(np.exp(Z), axis = 1, keepdims=True)
    R_bar = -np.sum(Y_bar * np.exp(Z), axis=1, keepdims=True)/R**2
    Z_bar = Y_bar * (np.exp(Z)/R) + R_bar * np.exp(Z)
    return Z_bar
```