清華大學

Tsinghua University

Department of Computer Science and Technology

Machine Learning

Homework 3

Sahand Sabour

2020280401

1 Clustering: Mixture of Multinomials (2 points)

1.1 MLE for multinomial (1 point)

The likelihood function for this multinomial distribution is given as

$$P(x|\mu) = \frac{n!}{\prod_i x_i!} \prod_i \mu_i^{x_i}, \quad i = 1, ..., d \tag{1}$$

Taking log from both side of the above equation gives the log-likelihood function

$$\mathcal{L}(\mu) = log(P(x|\mu)) = log(n!) - log(\prod_i x_i!) + log(\prod_i \mu_i^{x_i}) \tag{2}$$

This can be considered a Lagrange problem with the constraint $\sum_i \mu_i = 1$. Hence, the Lagrangian equation can be formulated as

$$\mathcal{L}(\mu) = log(n!) - log(\prod_i x_i!) + log(\prod_i \mu_i^{x_i}) - \lambda(\sum_i \mu_i - 1) \tag{3}$$

where λ is Lagrangian multiplier, giving

$$\mathcal{L}(\mu) = log(n!) - \sum_i log(x_i!) + \sum_i x_i log(\mu_i) - \lambda(\sum_i \mu_i - 1) \tag{4}$$

Taking the derivative of the equation with respect to μ_i and setting it to 0 gives

$$\frac{\partial \mathcal{L}}{\partial \mu_i} = \frac{\sum_i x_i}{\sum_i \mu_i} - \lambda = 0 \tag{5}$$

Hence, we get that

$$\lambda = \frac{\sum_i x_i}{\sum_i \mu_i} = \frac{n}{1} = n \tag{6}$$

Accordingly, we could derive the maximum-likelihood estimator μ_i as

$$\mu_i = \frac{x_i}{\lambda} = \frac{x_i}{n}, \quad i = 1, ..., d \tag{7}$$

1.2 EM for mixture of multinomials (1 point)

Initially, for each document d, a latent topic c_d is generated as follows:

$$p(c_d = k) = \pi_k \quad where \quad k = 1, 2, ..., K \tag{8}$$

Accordingly, given a topic μ_k, the bag of words for d is generated:

$$p(d|c_d = k) = \frac{n_d!}{\prod_w T_{dw}!} \prod_w \mu_{w_k}^{T_{dw}} \quad where \quad n_d = \sum_w T_{dw} \tag{9}$$

Combining the above two equations gives

$$p(d) = \sum_{k=1}^{K} p(d|c_d = k) p(c_d = k)$$
$$p(d) = \frac{n_d!}{\prod_w T_{dw}!} \sum_{k=1}^{K} \pi_k \prod_w \mu_{w_k}^{T_{dw}}$$

We have the log likelihood as

$$logp(\mathcal{D}|\mu, \pi) = \sum_{d=1}^{D} log(\sum_{k=1}^{K} \pi_k \prod_w \mu_{w_k}^{T_{dw}}) \tag{10}$$

Accordingly, we consider the log likelihood as the below Lagrangian optimization

$$L = \sum_{d=1}^{D} log(\sum_{k=1}^{K} \pi_k \prod_w \mu_{w_k}^{T_{dw}}) + \lambda_1(1 - \sum_{k=1}^{K} \sum_w \mu_{wk}) + \lambda_2(1 - \sum_{k=1}^{K} \pi_k) \tag{11}$$

with the following constraints

$$\sum_{k=1}^{K} \sum_w \mu_{wk} = 1 \quad and \quad \sum_{k=1}^{K} \pi_k = 1$$

and solve it with respect to μ_{wk} and π_k.

$$\frac{\partial L}{\partial \mu_{wk}} = \sum_{d=1}^{D} \frac{\pi_k \prod_w \mu_{w_k}^{T_{dw}} T_{dw}}{\sum_{j=1}^{J} \pi_j \prod_w \mu_{w_j}^{T_{dw}}} - \lambda_1 = 0$$
$$\frac{\partial L}{\partial \pi_k} = \sum_{d=1}^{D} \frac{\prod_w \mu_{w_k}^{T_{dw}}}{\sum_{j=1}^{J} \pi_j \prod_w \mu_{w_j}^{T_{dw}}} - \lambda_2 = 0$$

E-Step: estimate the responsibilities.

$$\gamma(c_{dk}) = \frac{\pi_k \prod_w \mu_{w_j}^{T_{dw}}}{\sum_{j=1}^{J} \pi_j \prod_w \mu_{w_j}^{T_{dw}}} \tag{12}$$

M-Step: re-estimate the parameters. We have

$$\lambda_1 = \sum_{k=1}^{K} \sum_{d=1}^{D} \gamma(c_{dk}) T_{dw} \quad and \quad \lambda_2 = \sum_{k=1}^{K} \sum_{d=1}^{D} \frac{\gamma(c_{dk})}{\pi_k} \tag{13}$$

Which gives

$$\mu_{wk} = \frac{1}{D_k} \sum_{d=1}^{D} \gamma(c_{dk}) T_{dw} \quad and \quad \pi_k = \frac{D_k}{D} \tag{14}$$

2 PCA (2 points)

2.1 Minimum Error Formulation (2 points)

Assuming that we have a set of complete orthonormal basis $\{\mu_i\}$, where $i \in [1, p]$, we have that $\mu_i^T \mu_j = \partial_{ij}$ and each data point can be represented as $x_n = \sum_i a_{ni} \mu_i$. Accordingly, due to orthonormal property, we can get that

$$a_{ni} = x_n^T \mu_i \tag{15}$$

Inserting this in the data point representation gives

$$x_n = \sum_i (x_n^T \mu_i) \mu_i \tag{16}$$

For this approach, the aim is to formulate PCA as minimizing the mean-squared-error of a low-dimensional approximation of the given basis. Hence, we assume a low-dimensional approximation of the point representation as follows

$$\widetilde{x}_n = \sum_{i}^{d} z_{ni} + \sum_{i=d+1}^{p} b_i \mu_i \quad \text{where } b_i \text{ are constants for all points} \tag{17}$$

Therefore, the best approximation is to minimize the following error

$$\min_{U,z,b} J := \frac{1}{N} \sum_{n=1}^{N} ||x_n - \widetilde{x}_n||^2 \tag{18}$$

Consequently, we have that

$$\begin{aligned} J &= \frac{1}{N} \sum_{n=1}^{N} ||x_n - \widetilde{x}_n||^2 \\ &= \frac{1}{N} \sum_{n=1}^{N} (x_n - \widetilde{x}_n)^T (x_n - \widetilde{x}_n) \\ &= \frac{1}{N} \sum_{n=1}^{N} x_n^T x_n - 2 x_n^T \widetilde{x}_n + \widetilde{x}_n^T \widetilde{x}_n \end{aligned}$$

Inserting equation 10 in the above equation and replacing $\widetilde{x}_n$ gives

$$J = \frac{1}{N}\sum_{n=1}^{N} x_n^T x_n - 2x_n^T(\sum_i^d z_{ni}\mu_i + \sum_{i=d+1}^{p} b_i\mu_i)$$
$$+(\sum_i^d z_{ni}\mu_i^T + \sum_{i=d+1}^{p} b_i\mu_i^T)(\sum_i^d z_{ni}\mu_i + \sum_{i=d+1}^{p} b_i\mu_i)$$

Accordingly, for minimizing this error, we calculate the derivative with respect to z and b and set it to 0.

$$\frac{\partial J}{\partial z_{nj}} = \frac{1}{n}[-2x_n^T\mu_j + \mu_j^T(\sum_i^d z_{ni}\mu_i + \sum_{i=d+1}^{p} b_i\mu_i) + (\sum_i^d z_{ni}\mu_i^T + \sum_{i=d+1}^{p} b_i\mu_i^T)\mu_j] = 0$$
$$\frac{\partial J}{\partial z_{nj}} = \frac{1}{n}[-2x_n^T\mu_j + 2\mu_j^T(\sum_i^d z_{ni}\mu_i + \sum_{i=d+1}^{p} b_i\mu_i)] = 0$$
$$2\mu_j^T(\sum_i^d z_{ni}\mu_i + \sum_{i=d+1}^{p} b_i\mu_i) = 2x_n^T\mu_j$$
$$\sum_i^d z_{ni}\mu_j^T\mu_i + \sum_{i=d+1}^{p} b_i\mu_j^T\mu_i = x_n^T\mu_j$$
$$\sum_i^d z_{ni}\partial ij + \sum_{i=d+1}^{p} b_i\partial ij = z_{ni} + 0 = x_n^T\mu_i$$

Giving $z_{ni} = x_n^T\mu_i$ for $i \in [1, d]$. Similarly, we the derivative with respect to b

$$\frac{\partial J}{\partial b_j} = \frac{1}{n}\sum[-2x_n^T\mu_j + \mu_j^T(\sum_i^d z_{ni}\mu_i + \sum_{i=d+1}^{p} b_j\mu_i) + (\sum_i^d z_{ni}\mu_i^T + \sum_{i=d+1}^{p} b_j\mu_i^T)\mu_j] = 0$$
$$\frac{\partial J}{\partial b_j} = \frac{1}{n}\sum[-2x_n^T\mu_j + 2\mu_j^T(\sum_i^d z_{ni}\mu_i + \sum_{i=d+1}^{p} b_j\mu_i)] = 0$$
$$\sum(\sum_i^d z_{ni}\mu_j^T\mu_i + \sum_{i=d+1}^{p} b_j\mu_j^T\mu_i) = \sum x_n^T\mu_j$$
$$\sum b_j = Nb_j = \sum x_n^T\mu_j \quad \text{giving} \quad b_j = \sum \frac{1}{n}x_n^T\mu_j = \bar{x}^T u_j$$

Which in turn gives $b_i = \bar{x}^T u_i$ for $i \in [d+1, p]$. Accordingly, from equation 9, we can get the displacement lines in the orthogonal subspace as follows

$$x_n - \widetilde{x}_n = \sum_{i=d+1}^{p} \{(x_n - \bar{x})^T \mu_i\} \mu_i \tag{19}$$

Which produces the following optimization problem for error J

$$\underset{\mu_j}{min} J \quad \text{where} \quad \mu_i^T \mu_i = 1 \tag{20}$$

Assuming d=1 (1-dimensional subspace) and p=2 (2-dimensional space), the optimization problem becomes

$$\underset{\mu_2}{min} J = \mu_2^T S \mu_2 \quad \text{where} \quad \mu_2^T \mu_2 = 1 \tag{21}$$

Which gives $S\mu_2 = \lambda_2 \mu_2$, meaning that μ_2 should be chosen as the eigenvector that corresponds to the smaller eigenvalue. Accordingly, the principal subspace is chosen by the eigenvector of the larger eigenvalue.

3 Deep Generative Models: Class-conditioned VAE (5 Points)

In the MNIST dataset, there are 10 possible labels for the samples (0-9). Binarizing the labels with the one-hot encoding method, gives a sequence of 10 digits with one 1 and nine 0s. Hence, there could be 10 locations for the 1; the probability of a label l to be one of the 10 labels L would be p(l = L) = $\frac{1}{10}$ = 0.1. According to the lecture notes, the variational lower bound for the normal case of VAE was obtained as follows:

$$L(\theta, x) = E_{q(z|x)}[logp(z, x; \theta) - logq(z|x)] = E_{q(z|x)}[logp(x|z; \theta)] - KL(q(z|x)||p(z; \theta))$$

However, it can be noticed that the output of this equation is only dependent on the latent variable z and therefore, does not produce any specific results, which is not practical for our case. Hence, we should modify the lower bound to include the label l of the sample we would like to generate likewise.

$$L(\theta, x, l) = E_{q(z|x,l)}[logp(x, l|z; \theta)] - KL(q(z|x, l)||p(z; \theta))$$

Since $z \sim \mathcal{N}(0, 1)$ for Gaussian, the KL-divergence is as follows:

$$-KL(q(z|x, l)||p(z; \theta)) = \frac{1}{2}(1 + log\sigma^2 - \mu^2 - \sigma^2) \tag{22}$$

Consequently, the expected log-likelihood would be

$$E_{q(z|x,l)}[logp(x,l|z;\theta)] = E_{q(z|x,l)}[-\sum_j \frac{1}{2}log\sigma_j^2 + \frac{(x_{ij}-\mu_{xi})^2}{\sigma^2}] \tag{23}$$

Approximating the above equation with Monte Carlo methods gives

$$E_{q(z|x,l)}[logp(x,l|z;\theta)] \approx \frac{1}{L}\sum_k logp(x,l|z^{(k)}) \quad \text{where} \quad z^{(k)} \sim q(z|x,l) \tag{24}$$

where $z^{(k)}$ is a random variable, which cannot be used for back-propagation. Hence, by utilizing re-parameterization techniques, we have $z^{(k)} = \mu(x,l) + \sigma(x,l).\epsilon^{(k)} = g(x,l,\epsilon^{(k)})$, where g is a deep neural network. The lower bound becomes

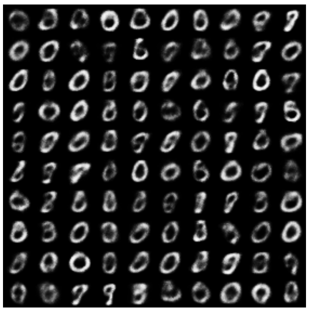
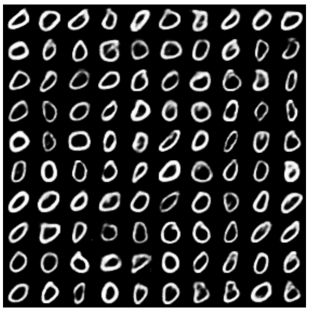
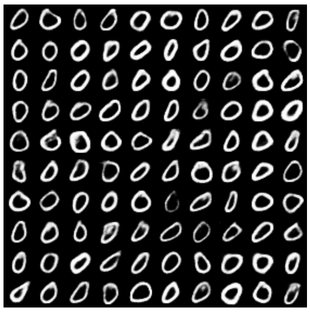
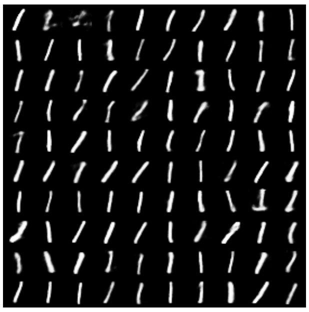
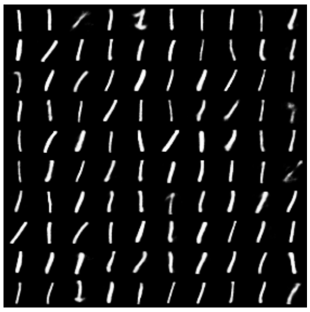
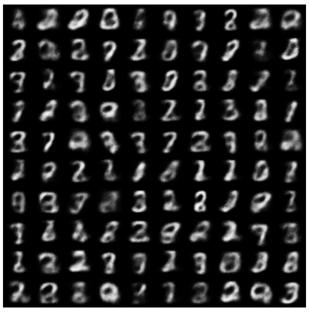
$$L(\theta,x,l) = E_{p(\epsilon)}[log\frac{p(g(x,l,\epsilon),x;\theta)}{q(g(x,l,\epsilon)|x;\theta)}] - KL(q(z|x,l)||p(z;\theta))$$
$$L(\theta,x,l) = \frac{1}{L}\sum_k logp(x,l|z^{(k)}) + \frac{1}{2}\sum_{i=1}^{j}[1 + log\sigma^2 - \mu^2 - \sigma^2]$$

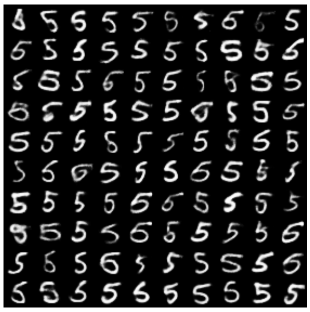
The model was trained on the MNIST dataset (one-hot form) and the obtained lower bound values for some epochs are provided in the table below:

Epoch	1	10	25	50	100
Lower Bound	-167.45	-97.954	-92.546	-90.108	-88.361

Table 1: Table of lower bound based on given epoch

In addition, the obtained digit generation results are provided in the next page.

Digit	Epoch 1	Epoch 50	Epoch 100
0			
1			
2			
3			
4			

Digit	Epoch 1	Epoch 50	Epoch 100
5			
6			
7			
8			
9