CSCI567 Theory Assignment 4

Yifan Wang

3038184983

wang608@usc.edu

Problem 1 Gaussian Mixture Model and EM (30 points)

1.1 In the lecture we applied EM to learn Gaussian Mixture Models (GMMs) and showed the M-Step without a proof. Now it is time that you prove it.

Consider a GMM with the following PDF of $\mathbf{x}_n$:

$$p(\mathbf{x}_n) = \sum_{k=1}^{K} \omega_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \sum_{k=1}^{K} \frac{\omega_k}{\left(\sqrt{2\pi}\right)^D |\boldsymbol{\Sigma}_k|} \exp\left(-\frac{1}{2}(\mathbf{x}_n - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k)\right)$$

where $K \in \mathbb{N}$ is the number of Gaussian components, $D \in \mathbb{N}$ is dimension of a data point $\mathbf{x}_n$. This GMM has K tuples of model parameters $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \omega_k)$, which standards for the mean vector, covariance matrix, and component weight of the k-th Gaussian component. $|\Sigma|$ denotes the determinant of matrix Σ.

For simplicity, we further assume that all components are isotropic Gaussian, i.e., $\Sigma_k = \sigma_k^2 I$. Find the MLE of *the expected complete log-likelihood*. Equivalently, find the optimal solution to the following optimization problem.

$$\begin{aligned} \arg\max_{\omega_k, \boldsymbol{\mu}_k, \Sigma_k} \quad & \sum_n \sum_k \gamma_{nk} \ln \omega_k + \sum_n \sum_k \gamma_{nk} \ln \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \\ \text{s.t. } & \omega_k \geq 0 \\ & \sum_{k=1}^{K} \omega_k = 1 \end{aligned}$$

where γ_{nk} is the posterior of latent variables computed from the E-Step. **(20 points)**

$$\gamma_{nk}$$

is derived from E step.

$$\log(N(x_i, \mu_k, \Sigma_k)) = -\log(2\pi^{-d/2}) - 0.5\log(det(\sum_k)) - 0.5(x_i - \mu_k)'\Sigma_k^{-1}(x_i - \mu_k)$$

$$\sum_i \gamma_{ik} \log(N(x_i, \mu_k, \Sigma_k))$$

$$= \sum_i \gamma_{ik}(-0.5\log(det(\sum_k)) - 0.5(x_i - \mu_k)'\Sigma_k^{-1}(x_i - \mu_k)$$

$$\frac{\partial \sum_i -0.5\gamma_{ik}(x_i - \mu_k)'\Sigma_k^{-1}(x_i - \mu_k)}{\partial \mu_k}$$

$$= \sum_i -\gamma_{ik}\Sigma_k^{-1}(x_i - \mu_k) = 0$$

$$\mu_k \sum_i \gamma_{ik} = \sum_i \gamma_{ik} x_i$$

$$\mu_k = \frac{\sum_i \gamma_{ik} x_i}{\sum_i \gamma_{ik}}$$

$$-0.5\sum_i \gamma_{ik}\log(det(\Sigma_k)) - 0.5\sum_i \gamma_{ik}(x_i - \mu_k)'\Sigma_k^{-1}(x_i - \mu_k$$

$$= 0.5\sum_i \gamma_{ik}\log(det(\Sigma_k^{-1})) - 0.5\sum_i \gamma_{ik}(x_i - \mu_k)'\Sigma_k^{-1}(x_i - \mu_k)$$

$$\frac{\partial(above)}{\Sigma_k^{-1}} = 0.5\sum_i \gamma_{ik}\Sigma_k - 0.5\sum_i \gamma_{ik}(x_i - \mu_k)(x_i - \mu_k)' = 0$$

$$\Sigma_k = \frac{\sum_i \gamma_{ik}(x_i - \mu_k)(x_i - \mu_k)'}{\sum_i \gamma_{ik}}$$

1.2 The posterior probability of z in GMM can be seen as a *soft* assignment to the clusters; In contrast, K-means assign each data point to one cluster at each iteration (*hard* assignment). Describe how to set the model parameters such that the GMM in the previous question reduces to a K-means; please also derive $p(z_n = k|\mathbf{x}_n)$ in this case.

(10 points)

GMM is soft labels, while KMeans is hard labels, to make a GMM model KMeans, after we finish the E step and get a new assignment, we set the one with largest probability to 1 and all others to 0.

p(z_n=k|x_n) = 1 if x_n's closest centroid is k else p=0

Problem 2 Hidden Markov Model (30 points)

Recall that an HMM is parameterized as follows:

- initial state distribution $P(X_1 = s) = \pi_s$
- transition distribution $P(X_{t+1} = s' \mid X_t = s) = a_{s,s'}$
- emission distribution $P(O_t = o \mid X_t = s) = b_{s,o}$

2.1 Suppose we observe a sequence of outcomes $o_1, \ldots, o_T$ and wish to predict the next state X_{T+1}

$$P(X_{T+1} = s \mid O_{1:T} = o_{1:T}).$$

Denote the forward message as

$$a_s(T) = P(X_T = s, O_{1:T} = o_{1:T}).$$

Please derive how $P(X_{T+1} = s \mid O_{1:T} = o_{1:T})$ can be represented using $a_s(T)$. **(10 points)**

$$P(X_{T-1} = s|O_{1:T} = o_{1:T})$$

$$= \frac{P(X_{T-1} = s, O_{1:T} = o_{1:T})}{P(O_{1:T} = o_{1:T})}$$

$$= \frac{\sum_{s'} P(X_T = s', O_{1:T} = o_{1:T}) P(X_{T+1} = s|X_T = s', O_{1:T} = o_{1:T})}{\sum_{s'} P(X_T = s', O_{1:T} = o_{1:T})}$$

$$= \frac{\sum_{s'} \alpha_{s'}(T) P(X_{T+1} = s|Z_T = s')}{\sum_{s'} \alpha_{s'}(T)}$$

$$= \frac{\sum_{s'} \alpha_{s'}(T) a_{s',s}}{\sum_{s'} \alpha_{s'}(T)}$$

2.2 Describe how to set the model parameters such that an HMM reduces to the GMM described in Problem 1. In addition, derive the posterior probability $P(X_2 = s \mid O_1 = o_1, O_2 = o_2)$ using the parameters you set to show that the HMM really reduces to GMM. **(20 points)**

α_s becomes the soft assigment γ in GMM $a_{s,s'}$ becomes the x_i (data) in GMM

$$P(X_2|O_1 = o_1, O_2 = o_2) = \frac{P(O_2 = o_2|X_2 = s, O_1 = o_1)P(X_2 = s, O_1 = o_1)}{P(O_1 = o_1, O_2 = o_2)}$$

$$= \frac{P(O_2 = o_2|X_2)P(X_2 = s, O_1 = o_1)}{P(O_1 = o_1, O_2 = o_2)}$$

$$= \frac{b_{s,o_2} \sum_{s'} P(X_2 = s, X_1 = s', O_1 = o_1)}{P(O_1 = o_1, O_2 = o_2)}$$

$$= \frac{b_{s,o_2} \sum_{s'} P(X_2 = s|X_1 = s')P(X_1 = s', O_1 = o_1)}{P(O_1 = o_1)P(O_2 = o_2)}$$

$$= \frac{b_{s,o_2} \sum_{s'} a_{s',s} \alpha_{s'}(1)}{P(O_1 = o_1)P(O_2 = o_2)}$$

Problem 3 Viterbi Algorithm (40 points)

In this problem, we want to fit DNA sequence data with a generative model. In particular, we assume that they are generated by a hidden Markov model (HMM). Let $O_{1:N}$ which denote the sequence $[O_1 O_2 \ldots O_N]$ be random variables corresponding to a DNA sequence of length N, controlled by hidden states $X_{1:N} = [X_1, X_2 \ldots X_N]$. Each O_n takes a value in $\{A, C, G, T\}$ and each X_n takes one of the two possible states $\{s_1, s_2\}$. This HMM has the following parameters:

$$\theta = \{\pi_i, a_{ij}, b_{ik}\} \text{ for } i \in \{1, 2\}, j \in \{1, 2\}, \text{and } k \in \{A, C, G, T\}$$

- Initial state distribution π_i for $i \in \{1, 2\}$:

$$\pi_1 = P(X_1 = s_1) = 0.6; \pi_2 = P(X_1 = s_2) = 0.4$$

- Transition probabilities $a_{ij} = P(X_{n+1} = s_j | X_n = s_i)$ for any $n \in N^+, i = \{1, 2\}$ and $j = \{1, 2\}$:

$$a_{11} = 0.7, a_{12} = 0.3, a_{21} = 0.2, a_{22} = 0.8$$

- Emission probabilities $b_{ik} = P(O_n = k | X_n = s_i)$ for any $n \in N^+, i \in \{1, 2\}$ and $k \in \{A, C, G, T\}$:

$$b_{1A} = 0.4, b_{1C} = 0.1, b_{1G} = 0.4, b_{1T} = 0.1$$

$$b_{2A} = 0.2, b_{2C} = 0.3, b_{2G} = 0.2, b_{2T} = 0.3$$

We observe a sequence $o_{1:4} = [o_1, o_2 \ldots o_4] = [\text{AGCT}]$, please answer the following questions with step-by-step computations:

3.1 Compute probability of the observed sequence, i.e. compute $P(O_{1:4} = o_{1:4}; \theta)$. **(10 points)**

Init:

$$\alpha_i^1 = P(o_1 | \pi_i = S_i) P(\pi_i = S_i)$$

iter:

$$\alpha_i^k = p(x_i | \pi_i = S_i) \sum_j \alpha_{i-1}^j a_{i,k}$$

\alpha	S1	S2
1	0.6*0.4=0.24	0.4*0.3=0.12
2	0.0768	0.0504
3	6.384e-3	0.019
4	4.24e-4	5.13e-3

$$P(O_{1:4} = o_{1:4}) = \sum_s \alpha_s(T) = 5.59e-3$$

3.2 Find out the most likely explanation i.e. compute $s_{1:4}^* = [s_1^*, s_2^* \ldots s_4^*] = \arg\max_{s_{1:4}} P(X_{1:4} = s_{1:4} | O_{1:4} = o_{1:4}; \theta)$. **(20 points)**

In [51]:

```
from hmm import HMM
import numpy as np

def hmm_test(data):
    A = np.array(data['A'])
    B = np.array(data['B'])
    pi = np.array(data['pi'])
    obs_dict = data['observations']
    states_symbols = dict()
    for idx, item in enumerate(data['states']):
        states_symbols[item] = idx
    Oseqquence = np.array(data['Oseqquence'])
    N = len(Oseqquence)
    model = HMM(pi, A, B, obs_dict, states_symbols)
    delta = model.forward(Oseqquence)
    print("Your forward function outputs:", delta)
    gamma = model.backward(Oseqquence)
    print('Your backward function outputs:', gamma)
    prob1 = model.sequence_prob(Oseqquence)
    print('Your sequence_prob function outputs:', prob1)
    prob2 = model.posterior_prob(Oseqquence)
    print('Your posterior_prob function outputs:', prob2)
    prob3 = model.likelihood_prob(Oseqquence)
    print('Your likelihood_prob function outputs:', prob3)
    viterbi_path = model.viterbi(Oseqquence)
    print('Your viterbi function outputs: ', viterbi_path)
    return model
model = hmm_test({"A": [[0.7, 0.3], [0.2, 0.8]],
                  "pi": [0.6, 0.4],
                  "states": ["1", "2"],
                  "B": [[0.4, 0.1, 0.4, 0.1], [0.2, 0.3, 0.3, 0.3]],
                  "observations": {"A": 0, "C": 1, "T": 3, "G": 2},
                  "Oseqquence":  ["A","G","C","T"]})
```

```
Your forward function outputs: [[0.24       0.0768     0.006384   0.00082704]
 [0.12       0.0504     0.019008   0.005136481]]
Your backward function outputs: [[0.015592 0.0346   0.16     1.      ]
 [0.038512 0.0656   0.26     1.      ]]
Your sequence_prob function outputs: 0.00596352
Your posterior_prob function outputs: [[0.62749517 0.44558918 0.17128139 0.138683191
 0.37250483 0.55441082 0.82871861 0.861316811]
Your likelihood_prob function outputs: [[0.38999053 0.14423696 0.07493561]
 [0.23760464 0.30135222 0.09634578]]

 [[0.05569865 0.02704443 0.06374759]
 [0.31680618 0.52724639 0.76497102]]
Your viterbi function outputs:  ['2', '2', '2', '2']
```

the state is `['2', '2', '2', '2']` from above running result.

3.3 Predict most likely observation for the next step i.e. compute $o^* = \arg\max_{o_5} P(O_5 = o_5 | O_{1:4} = o_{1:5}; \theta)$. **(10 points)**

$$P(O_5 = o_5|_{1:4} = o_{1:4}; \theta)$$

$$= \frac{P(O_{1:4} = o_{1:4}, O_5 = o_5|\theta)}{P(O_{1:4} = o_{1:4})}$$

the denominator is independent of $O_5 so that we cen edt oargmax P(O_{1:4} = o_{1:4}, O_5 = o_5|\theta)$$

Based on the following likelihood, 'T' is the prediction

In [134]:

```
cand = ['A','T','G','C']
for i in cand:
    prob3 = model.likelihood_prob(["A","G","C","T"]+[i])
    print(i, 'likelihood')
    print(prob3[:,:,-1])
    print()
```

```
A likelihood
[[0.15295841 0.0327768 ]
 [0.2714216  0.54284319]]

T likelihood
[[0.03944158 0.05071061]
 [0.06998829 0.81985951]]

G likelihood
[[0.11877405 0.03817737]
 [0.21076214 0.63226643]]

C likelihood
[[0.03944158 0.05071061]
 [0.06998829 0.81985951]]
```

In [ ]: