EE-613: Probabilistic Generative Models, Lab 2, Part 2

Fall 2019

1 Gaussian Mixture Model - Maximum-A-Posteriori adaptation

The goal is to study the GMM algorithm and how it can be used for unsupervised learning. In the lab, we will model the distribution of observed data points using a Gaussion Mixtures Model (GMM) and use this GMM to estimate the hidden variables which have generated those observations. It can also be used to perform model adaptation.

The illustration context is on a simplified setting on how to estimate the Visual Focus of Attention (VFOA) of a person, i.e. to which target a person is looking at, given his/her head pose estimate.

Situation and problem. Assume that scene consists of a room with the Nao robot, a human and 2 paintings. Nao is interacting with the human, commenting about the painting and having a questions & answers session. It is assumed that there are 3 possible VFOA targets for the human: Nao and the two paintings. The situation is illustrated in Fig. 1.

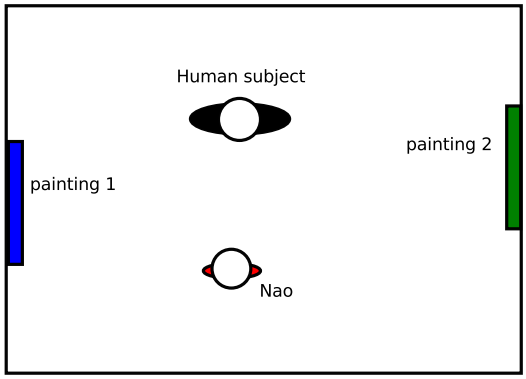

Figure 1: *Task overview.* A robot (Nao) is interacting with a human subject. From the video that looks at the subject, we can extract the person's head orientation $\mathbf{x} = (x_1, x_2)$, with x_1 representing the pan angle, i.e. looking right or left, and x_2 the tilt angle, looking up or down. From this observation, we would like to know whether the person looks at the robot, at the first painting, or at the second painting.

During the interaction with the robot, we assume the following generative process for the head

poses of the person[1]

1. draw randomly a focus $\mathbf{z}$ according to the categorical distribution parameterized by $\boldsymbol{\pi}$:

$$\mathbf{z} \sim p(\mathbf{z} = k|\boldsymbol{\pi}) = \pi_k$$

   We will define that $k = 2$ corresponds to looking at painting 1, $k = 1$ corresponds to looking at painting 2, and $k = 3$ corresponds to looking at nao.

2. given the focus, draw the head pose from a Gaussian distribution:

$$\mathbf{x} \sim p(\mathbf{x}|\mathbf{z} = k, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \Sigma_k)$$

   We will denote $\boldsymbol{\theta}_k = (\boldsymbol{\mu}_k, \Sigma_k)$ and $\boldsymbol{\theta} = \{\boldsymbol{\pi}, \boldsymbol{\theta}_k, k = 1..K\}$.

You are given a jupyter notebook, with functions that you need to complete.

Questions

(Exercise a) You are given a training dataset (the file Training.mat) containing complete observable data $(\mathbf{X}_{train}, \mathbf{Z}_{train})$ from different people. Use this dataset to learn the parameters $\boldsymbol{\theta}$ of the model, by completing the function in section 2 of the notebook:

```
# Learn parameters observable using the input data (X, Z).
# The function returns the prior (pi), the Gaussian mean (mu)
# and its covariance (sigma).
def learn_parameters_observable(K, X, Z):
    # Input:
    # K: number of classes
    # X: D x Nsamples  data matrix, D observation dimension
    # Z: 1 X Nsamples : class labels
    #
    # Output:
    #  pi:      array of K priors
    #  mean:    D x K array containing the K means
    #  sigma:   D x D x K array containing the K covariance matrices
```

Once this is done, go to Section 3 of the notebook, and look at the estimated gaussians. Do the Gaussian locations (means) corresponds to what you would expect in terms of head pose (see Fig. 1)?

Exercise (b) - Supervised learning. You are now given the sequences of head poses $\mathbf{X} = \{\mathbf{x}_i, i = 1, ..., N\}$ of a **new person**. Write down how you can recognize the VFOA for each of the head pose given the model.

Verify that the code for doing the prediction (predict_label in utils.py) corresponds to your formula.

[1]Note that in principle we could have considered the time variable, since the VFOA of a person normally remains the same for short segment of time.

The notebook loads the data, display them, and then make the prediction using the model learned from the training data (previous exercice). When visualizing the data from each person, is their any comment you can make?

Look a the the classification error for TestPerson1.mat and TestPerson2.mat. When cheating (i.e. using the test data to learn the model), can we obtain better results?

Exercice (c) - Unsupervised learning. Assume that we would not be given training data, and that we would like to nevertheless cluster the head pose data of each person, assuming that they are clustered in 3 main clusters (one for each VFOA target). To this end, we want to use a GMM model where each obtained Gaussian will be assumed to correspond to looking towards one VFOA target.

Show that the distribution of head poses $p(\mathbf{x})$ of a person follows a GMM model. Then, complete the function for the E-Step and the M-Step which are called in the GMMEM function that learns the parameters of a GMM using the EM algorithm.

```
# E-step: Compute the responsibilities of a mixture to explain the data
# or in other words, compute the posterior p(Z|X)
#
def GMMGamma(X, pis, means, sigmas):
    #
    # X: Nsamples X D array  (D data dimension). Data
    # pis: mixture priors
    # means: D x K array, with K number of mixtures, D data dimension
    # sigmas: D x D x K array of covariance matrices



# M-step: compute the optimal GMM parameters given the data and the gammas
# (posteriors on the z variable)
#
def GMMMstep(X, gamma):
    #
    # Input:
    # - X:       Nsamples X D array  (D data dimension). Data
    # - gamma:   K x Nsamples
    #
    # Output: restimated model parameters
    # - pis: mixture priors
    # - means:  D x K array, with K number of mixtures, D data dimension
    # - sigmas: D x D x K array of covariance matrices
```

The GMM learning is unsupervised (we don't use any class label). So, to evaluate how well it performs, we need to find which cluster/mixture obtained by the GMM algorithm better corresponds to each VFOA class. This is done using the function:

```
piem1, meanem1, sigmaem1 = utils.reorder(piem1, meanem1, sigmaem1, means_train)
```

that follows the call to the GMMEM function in the notebook.

Look at the obtained gaussians. Do they match our expectation regarding the model?

EM optimization and Log-likelihood. The GMMEM code also computes the log-likelihood of the data at the end of each E and M step iteration. It is displayed. Verify that it increases, as it should in theory. Do you observe a different behavior for person 1 and person 2? Also, the

GMMEM algorithm prints the lower bound on the log-likelihood which is actually optimized in the EM algorithm. What do you observe? Do we observe the same behavior for the Q function?

If you use the expected posterior of the latent mixture variable z (the gamma responsibilities) computed using the new GMM parameters (uncomment the relevant line in GMMEM), what relation do you observe between the log-likelihood and the bound? Is it normal? (check the demonstration in the course).

Classification. We then compute the classification accuracy for the test person using the obtained model. Which performance do we obtain? Explain the results.

Initialization. We know that initialization can affect the end-result in EM learning. By default, learning is done in the code by initializing the parameters with the Kmeans algorithm (see the GMMEM code).

Given that we have a problem at hand, we can use domain knowledge. Another possibility could be to start by initializing the means with the values we would expect for a person to look at the different VFOA targets. In the code, this is done by providing the means learned in Exercise a from the training data. Print (or plot) for person 1 and person 2 the mean values obtained after the GMMEM algorithm when initializing with (a) Kmeans and (b) the trained means. Are they similar? Does the initialization affect the classification accuracy in the current case?

Exercise (d) A person may not always look at the 3 VFOA targets. How would you change the model to account for head poses that would not correspond to any of the 3 predefined targets?

Exercice (e) (optional) - MAP adaptation. Code. Unfortunately (see exercise (d)), the observations of a test person are potentially noisy, and different persons may use on average different head poses to look at the same targets (e.g. depending on whether the wear glasses for instance). In addition, during an interaction, people may look more or less at different paintings (e.g. if Nao does not speak about painting 2, people may not look at it), so that we can not necessarily expect a good balance of samples for the different classes.

In this context, to handle these issues, using a prior on each focus is a must for our problem. We would thus like to use a generic model and adapt its parameters for the specific person and interaction. This can be achieved using MAP adaptation. This could potentially lead to an increase of performance compared to using the model learned from other persons.

We remind that MAP adaptation does not affect the E-step, where the distribution over the hidden variables is computed with the current estimates of the parameter values. In the M-step, the MAP maximizes the function $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) + \log(p(\boldsymbol{\theta}|\boldsymbol{\Theta}))$, where $\boldsymbol{\Theta}$ are the parameters defining the prior distribution. For convenience, we use the appropriate conjugate distribution for each factor. For the categorical distribution over $\mathbf{z}$, we thus use a dirichlet $\text{Dir}(\boldsymbol{\pi}|\boldsymbol{\alpha})$. For each of the Gaussian, we use a normal-inverse-wishart prior, as given by Eq. 1 in the first part of the lab: i.e. $p(\boldsymbol{\theta}_k|\boldsymbol{\beta}) = \mathcal{NIW}(\boldsymbol{\mu}_k, \Sigma_k|\boldsymbol{\beta}_k)$.

It can be shown that the MAP parameters of each of the conditional probability distribution at each M-step is then simply given by the MAP for each distribution separately, using as data the set of weighted samples:

$$\mathcal{X}_k = \{(\mathbf{x}_i) \text{ with weight } \gamma(z_{ik}), i = 1, ..., N\} \tag{1}$$

where the weights are estimated in the E-step. The MAP for the parameters $\boldsymbol{\pi}$ of the categorical distribution is given in the course. The MAP expression of the mean and covariance of each Gaussian is given by Eq. 6 and 7 in the first part of the lab, modified to take into account the weighting factor. For instance, for the mean, we have:

$$\boldsymbol{\mu}_{MAP,k} = \frac{\tau_k \mathbf{m}_k + c_k \bar{\mathbf{x}}_k}{\tau_k + c_k} \text{ with } c_k = \sum_{i=1}^{N} \gamma(z_{ik}), \bar{\mathbf{x}}_k = \frac{1}{c_k} \sum_{i=1}^{N} \gamma(z_{ik}) \mathbf{x}_i$$

Write a GMMMAPadapt function similar to the GMMEM function that performs a MAP adaptation in the M-step rather than a ML estimation as in the standard EM.

Exercice (f) (optional) MAP adaptation. Application Apply the GMM MAP learning to the data of person 1 and 2. In particular, initialize the Dirichlet parameters $\boldsymbol{\alpha}$ for the prior on the VFOA (i.e. Dir($\boldsymbol{\pi}|\boldsymbol{\alpha}$)), and the parameters $\boldsymbol{\beta}_k$ for each of the Gaussian using the parameters estimated from the training data and taking into account the analysis performed in Part 1.
Observe and comment how the different components and their weights change during the adaptation for each of the test person.

Comment the classification results obtained, by contrasting them to those obtained in the supervised setting, or in the fully unsupervised case.
Based on the analysis conducted in part 1 of this lab session, try different options for setting the prior parameters (τ and ν). In a real setting, what would influence your choice for setting these parameters ?