CSP 304: Machine Learning Lab (Spring 2023)

Indian Institute of Information Technology (IIIT), Kota
Instructor: Ankit Sharma

Posted in week: April 24- 28, 2023
Due in week: May 8-12, 2023

Homework 3

Topic **Clustering and Expectation Maximization** This homework illustrates the use of expectation maximization (EM) methodology for data reduction or clustering.

Data Images given in the stadium.bmp file attached.

Problems

We will implement the EM algorithm to estimate a mixture of Gaussian distributions for image compression. (Please read the **Instructions** below before you start programming)

A1 **Method Implementation (50 points):** Implement the EM algorithm to estimate a mixture of k Gaussian distributions and run it on the image file stadium.bmp. Cluster the pixels into $k = \{4, 8, 12\}$ clusters and plot the compressed images for each value of k as described below. (Note: your program might fail if Σ is singular; in this case, restart your EM again.)

A2 **Plot Expected Likelihood (50 points):** Run your EM implementation on the stadium.bmp image for $k = \{4, 8, 12\}$ and plot the expected complete log-likelihood function $\mathcal{Q}(\Phi|\Phi_l)$ after each E-step and M-step of the EM algorithm in one curve for each value of k. Use different colors to plot the log-likelihood after the E-step and M-step in the curve. Briefly explain the results.

Instructions

- Solutions to all questions must be presented in a report which includes results, explanations, all images and plots.
- *numpy, scipy, skimage and matplotlib* can be relied on to implement the algorithm (Note that you are only allowed to use *KMeans* function in cluster module of *sklearn* package in this homework). Each function must take the inputs in the order specified and print/display the required output to either terminal or PyCharm console or notebook. For each part, you can submit additional files/functions (as needed) which will be used by the main functions specified below. Put comments in your code so that one can follow the key parts and steps.
- *Function definition:* **EMG**(*image.bmp*: file path to an image, k: scalar value of the number of clusters). Function EMG implements the EM algorithm . The function must print to either terminal or PyCharm console and return in variables the following for a single image and value of k: (1) h: a $n \times k$ matrix where n is the number of pixels, (2) m: a $k \times d$ matrix where $d = 3$ denotes the RGB values, and (3) $\mathcal{Q}$: a column vector of expected complete log-likelihood values. The outputs $\{h, m, \mathcal{Q}\}$ are defined in Section 7.4 of the textbook. The function must also display: (1) a single compressed **color** image for a single value of k and (2) a single plot for the expected complete log-likelihood function value vs iteration number for a single value of k.

- You can use the *imread()* function in *io* module of *skimage* package to convert the image into a 3D matrix in Python. You will then need to convert the 3D matrix into a 2D matrix with $d = 3$ columns, in which each row are the three RGB values of a pixel. You can use the *reshape()* function in *numpy* package to do this.
- To decide the membership of each pixel x^t, you can select

$$\underset{i}{\text{argmax}} \;\; h_i^t$$

- To visualize the compression, use the mean estimated from C_i as the color of pixels in C_i.
- To compute the component densities, $p(\mathbf{x^t}|\Phi)$, you may use the *multivariate_normal()* function in *stats* module in *scipy* package.
- In your EM implementation, you can use the function *KMeans()* in cluster module of *sklearn* package to find the initial clusters, however you can only run *kmeans* for at most 3 iterations.
- Your program must be efficient and finish running for a single image and value of k in at most a couple of minutes. You can set a maximum number of iterations for the EM algorithm however use no less than 100 iterations.

Deliverables

Report — A brief report in PDF format describing the various experimental tasks mentioned above. Description should include the details of your experiments (process & setup), results and discussions/comments on the observations.

Code — Properly commented code file(s) or notebook(s) or any other setup and/or read-me files. All programming questions must be written in MATLAB/Python, no other programming languages will be accepted. For MATLAB you have to provide your own implementation using basic functionalities and no ready made libraries.

If there are several of them, please bind all the code files (NOT the report file) into a single ZIP file and upload as a response to Google classroom.

Notes — Although not mandatory, the report is encouraged to be written in LaTeX preferably via www.overleaf.com using the NIPS format available here: https://neurips.cc/Conferences/2021/PaperInformation/StyleFiles. Also, an example overleaf NIPS template can be readily found here: https://www.overleaf.com/latex/templates/neurips-2021/bfjnthbqvhgs.
