CSCI 5521: Introduction to Machine Learning (Spring 2019)[1]

Homework 3

Questions

1. (**30 points**) Derive the EM algorithm for estimating a mixture of Laplacian distributions (*double exponential distributions*). The probability density function of a Laplacian distribution for class C_i is

$$p(x|C_i) = f(x|\mu_i, b_i) = \frac{1}{2b_i}\exp\Big(-\frac{|x-\mu_i|}{b_i}\Big),\ b_i > 0,$$

where μ_i and b_i are referred to as the location and diversity parameters of the distribution. The mixture density of K Laplacian distributions is

$$P(x) = \sum_{i=1}^{K} P(x|C_i)P(C_i) = \sum_{i=1}^{K} \pi_i \frac{1}{2b_i}\exp\Big(-\frac{|x-\mu_i|}{b_i}\Big),$$

where $\sum_{i=1}^{K} \pi_i = 1$. Given the data $\mathcal{X} = \{x^1, x^2, \ldots, x^t, \ldots, x^N\}$, define the log-likelihood function and the complete log-likelihood function, and derive the EM equations including the expectation for the responsibility $\gamma(z_i^t)$ (z_i^t is the binary indicator of sample t in cluster i) and maximum likelihood learning for $\{\pi_i, \mu_i, b_i\}_{i=1,\ldots,K}$.

Important hint:

   (a) There is no easy way to solve the maximum likelihood learning of μ_i. An approximation is to binarize $\gamma(z_i^t)$ as $b_i^t = 1$ if $i = argmax_j \gamma(z_j^t)$ and otherwise $b_i^t = 0$ (the same as we do in k-means) before estimating μ_i. You should still use $\gamma(z_i^t)$ to estimate π_i and b_i.

   (b) The absolute error $\sum_{t=1}^{N} |\theta - x^t|$ is minimized if θ is the median of the N numbers).

[1]Instructor: Rui Kuang (kuan0009@umn.edu). TA: Jungseok Hong (jungseok@umn.edu) and Ujval Bangalore Umesh (banga038@umn.edu).

2. In this question, we will implement the EM algorithm to estimate a mixture of Gaussian distributions for image compression.

   (a) (**20 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. We will fix the problem in part (d)).

   (b) (**10 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.

   (c) (**10 points**) Try to run your EM implementation on the image "goldy.bmp" with $k = 7$ and report your observation (**Hint:** If your algorithm falls here, don't panic. Continue to the rest part.). Next, use the build-in Matlab k-means function[2] to cluster the pixels with $k = 7$. Plot the compressed image given by kmeans. Explain why kmeans and EM behaved differently on the image.

   (d) (**20 points**) Next, implement an improved version of EM to handle singular covariance matrix. In the likelihood function, we can add the following regularization term, $-\frac{\lambda}{2}\Sigma_{i=1}^{k}\Sigma_{j=1}^{d}(\Sigma_i^{-1})_{jj}$, where $(\Sigma_i^{-1})_{jj}$ is the (j, j)-th entry of matrix Σ_i^{-1} and $\lambda > 0$. This regularization term encourages the diagonal of Σ_i^{-1} to be small such that Σ_i is not singular. After adding this regularization term, the expectation step is unchanged and in the maximization step, the maximum likelihood learning of μ_is and π_is are also unchanged. Derive the maximum likelihood learning of Σ_is using the following result,

   $$\frac{\partial(-\frac{\lambda}{2}\Sigma_{i=1}^{k}\Sigma_{j=1}^{d}(\Sigma_i^{-1})_{jj})}{\partial\Sigma_i^{-1}} = -\frac{\lambda I}{2}$$

   (hint: modify the derivation on slide 32 in parametric.pdf to solve the problem).

   (e) (**10 points**). Implement the new model and test the new model on "goldy.bmp". Explain your observations.

[2] http://www.mathworks.com/help/stats/kmeans.html

Instructions

- Solutions to all questions must be presented in a report which includes results, explanations, all images and plots.
- All programming questions must be written in Matlab, no other programming languages will be accepted. The code must be able to be executed from the Matlab command window on the cselabs machines. Each function must take the inputs in the order specified and print/display the required output to the Matlab command window. 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. **Please follow the rules strictly. If we cannot run your code, you will receive no credit.**
- **Question 2:**
  - hw3_Q2_ce a driver script which 1) uses kmeans to display a compressed image 2) runs your standard EM implementation finishing by outputting your own error message explaining why it fails 3) runs your improved EM implementation to display the compressed image. For loading the "goldy.bmp" image, you can use the following code:

    ```
    [img cmap] = imread('goldy.bmp');
    img_rgb = ind2rgb(img,cmap);
    img_double = im2double(img_rgb);
    ```

  - EMG(*flag*: a binary indicator variable, *image.bmp*: file path to an image, k: scalar value of the number of clusters). Function EMG implements the standard EM algorithm if *flag* is 0, while implements the improved EM algorithm if *flag* is 1. The function must print to the Matlab workspace 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 to convert the image into a 3D matrix in Matlab. 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 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, $p(\mathbf{x^t}|\mathbf{\Phi})$, you may use the *mvnpdf()* function.
- In your EM implementation, you can use the function *kmeans()* to find the initial clusters, however you can only run kmeans for at most 3 iterations. Note, kmeans may error out when a cluster is empty for certain versions of Matlab. To avoid this, use *kmeans(. . . , EmptyAction, singleton).*
- 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.

Submission

- **Things to submit:**
  1. hw3_sol.pdf: A PDF document which contains the report with solutions to all questions.
  2. run_EMG.m: Code for calling EMG() function to answer all the questions in Question 2.
  3. Any other files, except the data, which are necessary for your code.
- **Submit**: Upload hw3_sol.pdf as a separate file and a zip file which is the compression of all other files. All material must be submitted electronically via Canvas.