Introduction to Deep Learning, 2016-2017

Instructor: Iasonas Kokkinos, iasonas.kokkinos[at]ecp.fr

Programming Assignment 1 - 4 Points/20 Deliverable: 6 December

1 Data modelling: PCA and K-means (1.5 point)

You are provided with a set of images corresponding to the digit '7'[1] . Each image comes in the form of a 28×28. Go through the code that has been provided with you and start by filling in the pieces indicating 'your code here'..

We will explore the potential of the two basic data modelling methods introduced in Lecture 1, PCA and K-means for image data.

1.1 PCA

The code provided to you learns an eigenbasis for the digit '7' using 3133 training images. We will use the remaining 3132 images ('Dtest') to assess the quality of the learned model. In particular you are asked to provide the plot of the function:

$$E(D) = \sum_{n \in \text{Testset}} \| I_n - \left(\sum_{d=1}^{D} c_{n,d} \mathbf{e}_k + \mathbf{m} \right) \|_2 \tag{1}$$

as a function of D, for $D = 1, \ldots, 10$. In the equation above, D is the dimensionality of your model (i.e. the number of eigenvectors), I_n is the n-th **testing** image, $c_{n,d}$ is the expansion coefficient of the n-th image on the d-th eigenvector, $\mathbf{e}_k$ is the eigenvector with the k-th largest eigenvalue, $\mathbf{m}$ is the mean digit, estimated from the training set, and $\|\mathbf{x}\|_2 = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$ is the euclidean L_2 norm.

1.2 K-means

- For $k = 2$ evaluate the distortion criterion being optimized by k-means and to verify that it decreases as the k-means algorithm progresses.

- In order to mitigate the local minima problem of k-means, repeat the k-means algorithm 10 times, and keep the solution that yields the smallest distortion at the end. Show the resulting digit clusters.

- Repeat the procedure outlined above for values of $k = 3, 4, 5, 10, 50, 100$. For each such value of k report the distortion cost of the training and testing data. What do you observe? Make a short (two-three lines of text) qualitative comparison with PCA.

[1] Images taken from the MNIST dataset, a well-known handwritten digit recognition dataset. Convolutional models (LeNet) were successfully applied to this task more than twenty years ago in : this demo.

2 Ising model: MCMC & learning. (2.5 Points)

In this exercise, based on `4x4lattice.m`, we consider an Ising model on a $N \times N$ lattice, with $N = 4$. A state is a vector $\mathbf{x} \in \{-1, 1\}^{N^2}$. Given the small size of the lattice, we can exhaustively consider all $2^(N^2)$ states, energies and associated probabilities. We will use this to compare the (exact) expectations of features with those obtained with MCMC.

In particular, the energy that we consider is of the form:

$$E(\mathbf{x}; J) = -J \sum_{(i,j) \in \mathcal{E}} x_i x_j = -\frac{J}{2} \mathbf{x}^T \mathbf{A} \mathbf{x} \tag{2}$$

where J is a scalar and $\mathcal{E}$ is the set of edges of a rectangular grid, $\mathbf{A}$ is the 16×16 incidence matrix of the network. We note that the $\frac{1}{2}$ factor has been introduced to avoid double-counting of edges, while $J > 0$ implies that states with similar neighboring nodes are favorable (i.e. have lower energy).

2.1 Part-1: Brute-force evaluations (.5)

- Evaluate the energy of the network for all states.
- Evaluate the partition function of the Boltzmann-Gibbs distribution.
- Evaluate the probability $P(\mathbf{x}; J)$ that the network is in any state $\mathbf{x}$.
- For all feaures $\phi_k(\mathbf{x}), k = 1, \ldots, K$, compute their expectations under the model:

$$E_k^{P_J} = \langle \phi_k(\mathbf{x}) \rangle_{P(\mathbf{x};J)} = \sum_{\mathbf{x}} P(\mathbf{x}; J) \phi_k(\mathbf{x}) \tag{3}$$

Include the plots generated by your code in your report. You should be getting something like the following plots:

2.2 Part-2: MCMC evaluations (1)

- Use Gibbs sampling to obtain a set of N samples $\mathbf{x}_n, n = 1, \ldots, N$ drawn from $P(\mathbf{x}; J)$.
- Use these samples to obtain Monte Carlo estimates of the quantites computed in subsection 2.1:

$$E_k^{MC}(N) = \sum_{n=1}^{N} \phi_k(\mathbf{x}_n) \simeq \sum_{\mathbf{x}} P(\mathbf{x}; J) \phi_k(\mathbf{x}) \tag{4}$$

  Compare E^{MC} and E^P for different numbers of iterations.

You can find details on MCMC/Gibbs sampling in Chapter 24 of K. Murphy's book, while the matlab code provided to you contains additional guidance.

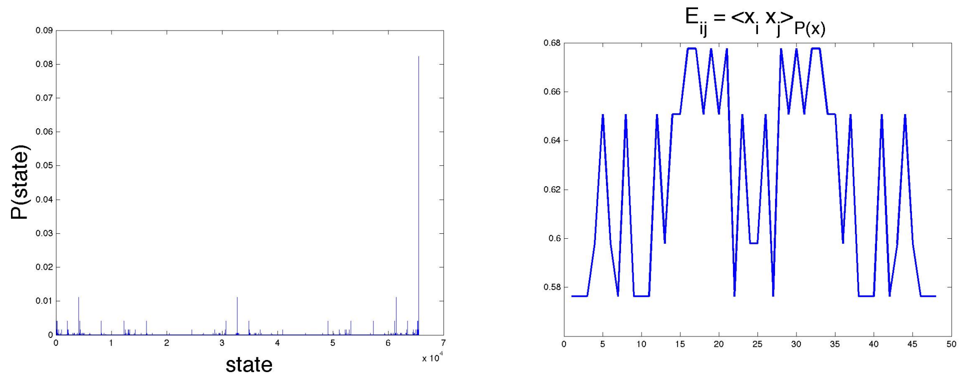

Figure 1: Figures to-be-obtained from subsection 2.1

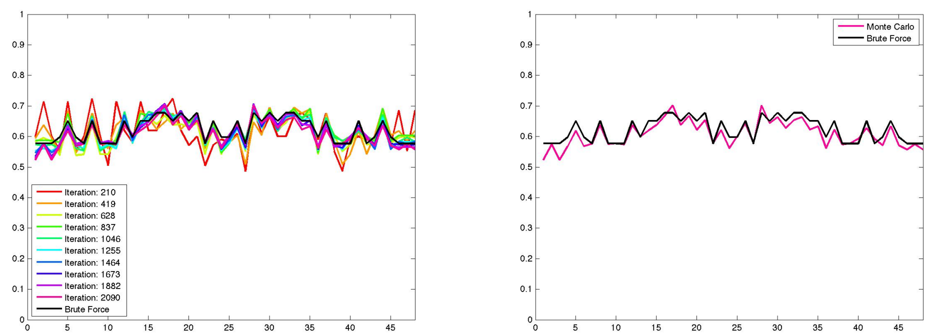

Figure 2: Figures to-be-obtained from subsection 2.2

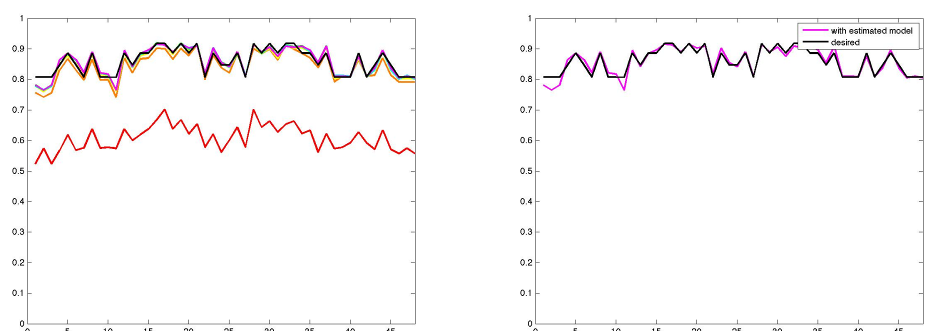

Figure 3: Figures to-be-obtained from subsection 2.3

2.3 Part-3: Parameter Estimation (1)

In Part-2 of this exercise we do not know the value of J in 2, but we have access to E^{P_J} (we load them from a .mat file). Our goal is to estimate the value of J that is most likely to be responsible for these values of E^{P_J}, i.e. the maximum likelihood estimate of J.

For this, we observe that the Ising model described above is a special case of the general exponential family:

$$E(\mathbf{x};\mathbf{w}) = \sum_k w_k \phi_k(\mathbf{x}) \tag{5}$$

if we identify $w_1 = -J$, $\phi_1(\mathbf{x}) = \sum_k x_i(k)x_j(k)$, with $k \leftrightarrow (i(k), j(k))$ mapping grid edges to 'feature' functions.

Use the gradient of the likelihood of a Boltzmann-Gibbs distribution with energy 2 with respect to J to find the right value of J, starting from $J = .5$. Report the value of J that you find.

Matlab figure hints

- Once a figure is generated you can save it into a jpeg,eps,png,... file using print. Type `>> help print` for details. A quick example:
```
>> input = [0:.01:1]; plot(input,sin(input));
>> title('Sinusoidal'); xlabel('input'); ylabel('output');
>> print -depsc sinusoid
```
- You can superimpose plots using `hold on`.
- For your convenience you can merge multiple plots in a single figure using `subplot`.
  Type `help subplot` for details.