CS498: Advanced Machine Learning **Spring 2018**

Machine Problem 8 - Mean Field Inference on Boltzman Machine

Professor *David A. Forsyth* *Auto-graded assignment*

Introduction

Mean-Field Approximation is a useful method for inference, that originated in statistical physics. The Ising model in its two-dimensional form is difficult to solve exactly, and therefore the mean-field approximation methodology was utilized. You can find the similarity of the ising model with the probabilistic models you learned in this chapter.

This assignment will incorporate Mean field inference for denoising binary images. The MNIST dataset consists of 60,000 images of handwritten digits, curated by Yann LeCun, Corinna Cortes, and Chris Burges. You can find the dataset at http://yann.lecun.com/exdb/mnist/ together with a collection of statistics on recognition, etc.

You will be using an autograder for this assignment. Therefore, we advise you to follow the steps exactly.

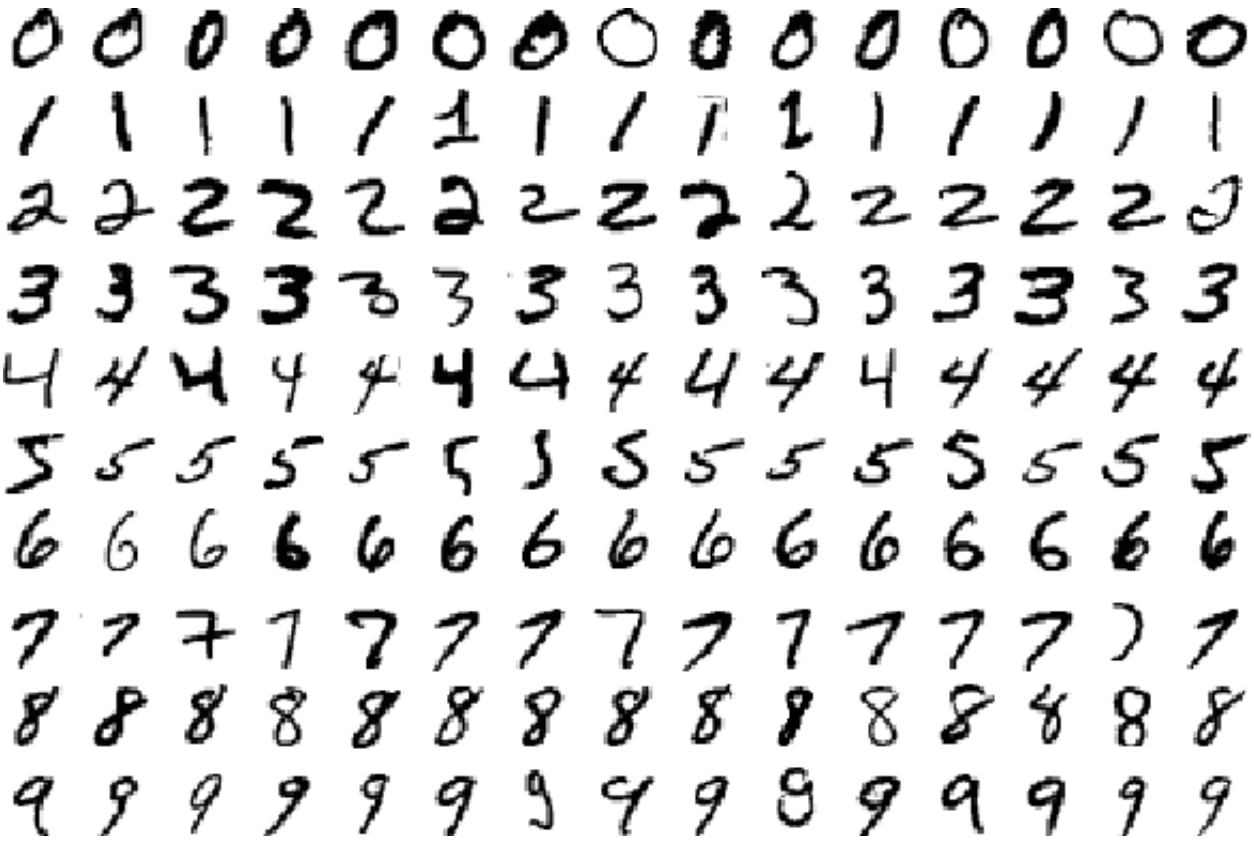

Figure 1: A handful of images in the MNIST dataset

Setting up the machine

1. **Obtaining the dataset**

   Obtain the MNIST training set, and binarize the first 20 images by mapping any value below .5 to -1 and any value above to 1.

   Hint: The original dataset at http://yann.lecun.com/exdb/mnist/ is in compressed format. You can utilize other people's code at the web for converting this data to a usable matrix format. For instance, we found some online code that can take care of this for you, but **we do not guarantee that these online code-pieces would be matching the original dataset. You should know that only the original dataset is the reference for this assignment.**

   - https://github.com/datapythonista/mnist is a python package for downloading the dataset and loading it into some numpy arrays.
   - https://gist.github.com/brendano/39760 is some piece of code that could be used for processing the MNIST data on R.

2. **Adding pre-determined noise to the dataset**

   For each image, create a noisy version by flipping some of the pixels. The exact location of the pixels you need to flip for each image is given to you in the supplementary file `NoiseCoordinates.csv`. All values in this supplementary file are 0-based. In other words, the images are indexed as Image $0, 1, \cdots, 499$. Also, the top left pixel of an image is indexed as being in row 0 and column 0.

Description	Noisy bit 0	Noisy bit 1	Noisy bit 2	Noisy bit 3	Noisy bit 4	
Image 0 Row	21	3	10	27	19	
Image 0 Column	17	17	22	9	20	
Image 1 Row	26	9	4	0	20	
Image 1 Column	17	3	20	22	6	
Image 2 Row	13	24	24	11	0	
Image 2 Column	27	1	21	3	13	
Image 3 Row	10	16	25	15	27	$\ddots$
Image 3 Column	3	22	6	12	23	
Image 4 Row	6	14	23	16	7	
Image 4 Column	9	22	17	12	14	
Image 5 Row	9	13	14	20	26	
Image 5 Column	27	5	20	25	20	
	$\cdots$					

Table 1: A sample of the noise location matrix as given in `NoiseCoordinates.csv` supplementary file. The first flipping noise in image 0 happens at the pixel in row number 21 and column number 17. **Please notice that all values in this matrix are 0-based**. For instance, the top left pixel of an image is indexed as being in row 0 and column 0.

3. **Building a Boltzman Machine for denoising the images and using Mean-Field Inference**

Now denoise each image using a Boltzmann machine model and mean field inference. For this:

- Use $\theta_{ij} = 0.2$ for the H_i, H_j terms and $\theta_{ij} = 2$ for the H_i, X_j terms.
- The order in which you have to update for each image is given in the supplementary file `UpdateOrderCoordinates.csv`. Please see table 2 as a sample of this matrix.

Description	Pixel 0	Pixel 1	Pixel 2	Pixel 3	Pixel 4	Pixel 5	Pixel 6	
Image 0 Row	26	15	13	1	22	27	9	
Image 0 Column	4	15	5	4	17	17	27	
Image 1 Row	0	27	3	25	20	22	5	
Image 1 Column	18	13	25	26	22	4	0	
Image 2 Row	10	7	8	22	23	1	3	...
Image 2 Column	22	17	26	0	21	19	12	
Image 3 Row	1	17	18	20	7	14	9	
Image 3 Column	19	22	17	3	20	11	13	
	...							

Table 2: A sample of the update order coordinates matrix as given in `UpdateOrderCoordinates.csv` supplementary file. **Please notice that all values in this matrix are 0-based**, which means the top left pixel of an image is indexed as being in row 0 and column 0. As an example, in image 0, the following order of updating the distribution parameters should happen: First the pixel indexed at (26,4) row and column respectively, then the pixel at (15,15), then the pixel at (13,5), then the pixel at (1,4), and so on.

- The initial parameters of the model can be found in the supplementary file named as

  `InitalParametersModel.csv`. This file is the initial matrix Q stored as comma-separated values, with a dimension of 28×28. Each entry of the matrix falls in the $[0, 1]$ interval, and the $Q_{r,c}$ entry denotes the $q_{r,c}[H_{r,c} = 1]$ initial probability. Here, a slight change of notation happened with respect to what you have learned in the course; what you have been denoting as $q_i[H_i = 1]$ through the course, is now named $q_{r,c}[H_{r,c} = 1]$ since we have a hidden state for each pixel at the r row and c column index.

  Please note that the same initial parameters are used for all the Boltzman machines built for each image.

- You should run the Mean-Field inference for exactly 10 iterations, where in each iteration the Q model distribution is updated with the same order shown and discussed in table 2.

4. **Turning in the energy function values computed initially and after each iteration** You have to compute the variational free energy as given in the following.

$$\mathrm{E}_Q = \mathbb{E}_Q[\log Q] - \mathbb{E}_Q[\log P(H, X)]$$

where

$$\log P(H, X) = \sum_{i \in H} \sum_{j \in \mathcal{N}(i) \cap H} \theta_{ij} H_i H_j + \sum_{i \in H} \sum_{j \in \mathcal{N}(i) \cap X} \theta_{ij} H_i X_j + K$$

Remember that we made a model assumption about the distribution

$$Q[H] = \prod_{r=1}^{28}\prod_{c=1}^{28} q_{r,c}[H_{r,c}]$$

Using this independence assumption, you can easily compute the entropy term

$$\mathbb{E}_Q[\log Q] = \sum_{r=1}^{28}\sum_{c=1}^{28} \mathbb{E}_{q_{r,c}}[\log q_{r,c}]$$

$$= \sum_{r=1}^{28}\sum_{c=1}^{28} \Big[q_{r,c}[H_{r,c}=1]\log(q_{r,c}[H_{r,c}=1]+\epsilon) + q_{r,c}[H_{r,c}=-1]\log(q_{r,c}[H_{r,c}=-1]+\epsilon) \Big]$$

In order to avoid any computational complications in cases where you may need to compute $0 \times \log 0$, we have added a very tiny value of $\epsilon = 10^{-10}$ insider the log.

Furthermore, we can simplify the log-likelihood term as well, thanks to the independence assumption made when defining the Q distribution

$$\mathbb{E}_Q[\log P(H,X)] = K + \mathbb{E}_Q\Big[\sum_{i\in H}\sum_{j\in\mathcal{N}(i)\cap H} \theta_{ij} H_i H_j + \sum_{i\in H}\sum_{j\in\mathcal{N}(i)\cap X} \theta_{ij} H_i X_j \Big]$$

$$= K + \sum_{i\in H}\sum_{j\in\mathcal{N}(i)\cap H} \theta_{ij}\mathbb{E}_Q[H_iH_j] + \sum_{i\in H}\sum_{j\in\mathcal{N}(i)\cap X} \theta_{ij}\mathbb{E}_Q[H_i]X_j$$

$$= K + \sum_{i\in H}\sum_{j\in\mathcal{N}(i)\cap H} \theta_{ij}\mathbb{E}_{q_i}[H_i]\mathbb{E}_{q_j}[H_j] + \sum_{i\in H}\sum_{j\in\mathcal{N}(i)\cap X} \theta_{ij}\mathbb{E}_{q_i}[H_i]X_j$$

where

$$\mathbb{E}_{q_k}[H_k] = (q_k[H_k=1]) \times (1) + (1 - q_k[H_k=-1]) \times (-1) = 2q_k[H_k=1] - 1$$

Since the K value is intractable to compute and has no effect on the optimization process of mean field approximation, we will ignore it by setting $K = 0$.

Compute the E_Q energy, and evaluate it both initially and after each iteration of updating the Q matrix. You will have to turn in a comma-separated values file which includes the variational free energy values as defined below.

$$\mathrm{E} = \begin{array}{c} \\ \textit{Image 10} \\ \textit{Image 11} \end{array} \begin{array}{c} \begin{array}{cc} \textit{Initially} & \textit{After one iteration} \end{array} \\ \begin{bmatrix} \mathrm{E}_{Q_{10}^{(0)}} & \mathrm{E}_{Q_{10}^{(1)}} \\ \mathrm{E}_{Q_{11}^{(0)}} & \mathrm{E}_{Q_{11}^{(1)}} \end{bmatrix} \end{array}$$

where $\mathrm{E}_{Q_m^{(n)}}$ denotes the variational free energy of the Boltzman Machine used for the image indexed m after the n^{th} iteration of mean field inference. Please note that the first column has the energy of the initial Q matrix given in the supplementary file with respect to each image. Do not include any headers or row names in the CSV file.

You can find a sample of the energy matrix we computed for the first ten images of the dataset in the supplementary file `EnergySamples.csv`.

5. **Displaying the reconstructed images**

   Run the mean-field approximation method for exactly 10 iterations (i.e. going over and updating the hidden distribution of each pixel for exactly 100 times) on the images indexed $10, 11, \cdots, 19$. After

that, based on the approximated Q distribution, compute the MAP image, and store it in a binary $\{0, 1\}$ matrix format.

You should turn in a prediction matrix with a shape of 28 rows, and 280 columns as defined in the following.

$$\hat{\mathbf{X}} = \left[\begin{array}{c|c|c|c} \hat{\mathbf{X}}_{10} \in \{0,1\}^{28\times 28} & \hat{\mathbf{X}}_{11} \in \{0,1\}^{28\times 28} & \cdots & \hat{\mathbf{X}}_{19} \in \{0,1\}^{28\times 28} \end{array}\right]$$

As you can see $\hat{\mathbf{X}} \in \{0, 1\}^{28\times 280}$. Turn in this matrix as a binary CSV file with exactly 28 rows and 280 columns.

In order to give you a hint of what the sample results look like, you can see figure 2, where we have provided you with the result of the method on the first ten images of the dataset. Please note that you have to report your results on the second ten images of the dataset as mentioned above. Those images can be seen in figure 3

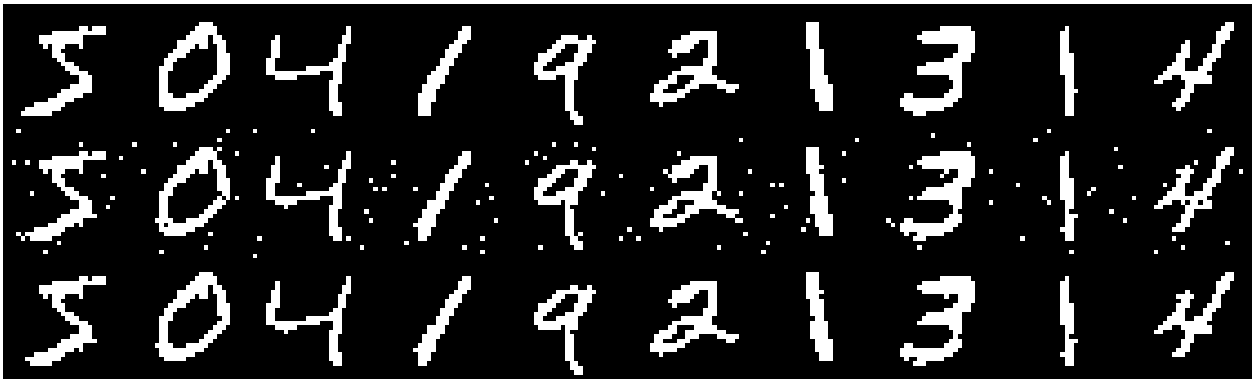

Figure 2: A sample figure for the first ten digits in the training set of the MNIST dataset. The top row shows the original images, the middle row shows the noisy version, and the bottom row shows the reconstructed images. The leftmost digit is the image indexed 0 in the dataset. The second leftmost digit is the image indexed 1 in the dataset, and the rightmost image is indexed 9 in the dataset.

Figure 3: The ten images of the dataset that you have to report the results for. The top row shows the original images, and the bottom row shows the noisy images by adding the predetermined noise. The leftmost digit is the image indexed 10 in the dataset. The second leftmost digit is the image indexed 11 in the dataset, and the rightmost image is indexed 19 in the dataset.

6. **Construction of an ROC curve**

Assume that θ_{ij} for the H_i, H_j terms takes a constant value c. We will investigate the effect of different values of c on the performance of the denoising algorithm.

Think of your algorithm as a noise detection device that accepts an image, and for each pixel, predicts 1 (i.e. positive) whenever the pixel is flipped because of noise or -1 (i.e. negative) otherwise. You can evaluate this in the same way we evaluate a binary classifier, because you know the right value of each pixel. A receiver operating curve is a curve plotting the true positive rate against the false positive rate for a predictor, for different values of some useful parameter. We will use c as our parameter.

Compute the FPR and TPR statistics of this device for $c \in \{10, 1.2, 0.8, 0.7, 0.6, 0.2\}$ using the images indexed at $10, 11, \cdots, 19$ (again, this is a zero-based indexing which means that the image indexed at 10 is the eleventh actual image of the dataset). The first column must contain the FPR values, and the second column must contain the TPR values. First row must correspond to $c = 1.2$, and the last row must correspond to $c = 0.2$. Do not include any header or row names in your generated CSV file.