**67750: Advanced Practical Machine Learning** 2019/20

Exercise 4 — Manifold Learning and Information Bottleneck

*Dr. Matan Gavish* *TA: Omri Bloch*

# 1 Theoretical Questions - Information Bottleneck (10 points)

In this chepter we are going to play with some basic definitions from information theory and statistics, and establish there relation to the Information Bottleneck paradigm. all the definitions we are going to use are very fundamental; hence, I recommend the English Wikipedia as a primary reference.

## 1.1 Information

1. lets mark entropy of random variable $X$ as $H(X)$, and conditional entropy of $X$ conditioned on $Y$ as $H(X|Y)$. let $H(X,Y)$ be the entropy of the joint distribution of $X, Y$ together. show that $H(X,Y) = H(X) + H(Y|X)$.
2. let $X$ be a RV, $p$ and $q$ two distributions for it, $D_{KL}(p,q)$ the KL-divegence for $p$ and $q$, and $CE(p,q)$ the cross-entropy of p and q. show that $CE(p,q) = D_{KL}(p,q) + H(X)$, where $H(X)$ is the entropy of $X$ calculated using the distribution $p$.
3. The Mutual Information of 2 RV $X$ and $Y$ is defined to be: $I(X,Y) = D_{KL}(p(x,y) \,\|\, p(x) \cdot p(y))$. show that $I(X,Y) = H(X) - H(X|Y)$

## 1.2 Statistics

I this section we are taking a short overview about the fundamental concepts of sufficient statistic and minimal sufficient statistic. I hope there is nothing new here for you.

Let P be a a family of probability distributions, parametrized by $\theta \in \Theta$. Let $S = (x_1 ... x_n)$ be a sample from this distribution, and $T : S \to \Theta$ be a statistic. (a statistic is just a function of the sample).

**Definition 1.** *Sufficient Statistic*

*A statistic T is sufficient for the model ($P_\theta$) if the conditional distribution of the sample X given T(X) under $P_\theta$ is constant with respect to $\theta$.*

1. Show that the mean of the sample is sufficient statistic for a bernuli random variable.
2. Show that the statistic that just takes the first element in the sample is not sufficient in this case.

3. show that the statistic that takes all the sample is a sufficient statistic for the same case.

**Definition 2.** *Minimal Sufficient Statistic*

*A sufficient statistic $T$ is minimal if it is a function of any other sufficient statistic.*

show that all the minimal sufficient statistics are equivalent. (any one of them is a function of any one of them).

use the Data Processing Inequality to show that given a sufficient statistic T and a minimal sufficient statistic M:

$$I(M(X), X) \leq I(T(X), X)$$

### 1.3 Information bottleneck

1. Assume the following sampling process: we choose randomly a category by a random variable Y. based on this category we are sampling a text, $X$, represented as bag-of-words. now we apply some statistic - $T$ - on this sample, trying infer about the original Y. use DPI to show that if T is a sufficient statistic, then $I(Y, X) = I(\theta, T(X))$

2. in the information bottleneck paradigm we are looking for statistic that will give us minimal $I(T(X), X)$ with maximal $I(T(X), Y)$. show that the minimal sufficient statistic is what we are looking for.

In practice, the minimal sufficient statistic may not exist in our hypotheses class. in this case, we will have some trade-off between the difference between the information we want to reserve and the information we want to forget.

## 2 Theoretical Questions -Manifold Learning (15 Points)

### 2.1 PCA

In PCA we diagonalize the empirical covariance matrix of our data, $S = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T$. Assume the data is centered, meaning $\bar{x} = 0$.

1. Show that $S$ is a PSD matrix.

2. Show that the data sits on a $d$-dimensional subspace $V \subset \mathbb{R}^n$ if and only if $S$ is of rank $d$. hint: show that $S$ and $X$ have the same rank.

3. Show that the new coordinates are the result of an isometry on the subspace $V$.

### 2.2 LLE

In this question we will walk through the derivation of the second stage of LLE (finding the $W$ matrix). In class, we saw that the goal of the second stage of LLE is to describe every point $x_i$ as an affine combination of its $k$ nearest neighbors:

$$W_{i,:} = \underset{w}{argmin}||x_i - \sum_{j \in N(i)} w_j x_j||^2 = \underset{w}{argmin}||\sum_{j \in N(i)} w_j z_j||^2$$

Where we define $z_j = x_j - x_i$.

Define $G$ to be the Gram matrix of the $z$ vectors, namely $G_{a,b} = z_a^T z_b$.

1. Show that $|| \sum_{j \in N(i)} w_j z_j||^2 = w^T G w$.

2. The above means that we can find $w$ by minimizing the quadratic form $w^T G w$, under the constraint that $\sum_i w_i = 1$. Formulate the Lagrangian and derive the solution:

   $$w = \frac{\lambda}{2} G^{-1} \mathbf{1}$$

   Note that we do not have to analytically derive $\lambda$ - it is only there to make sure that the constraint holds up (that $w$ sums to 1). This means we can calculate $w = G^{-1}\mathbf{1}$ and then just normalize the vector to sum to one...

### 2.3 Diffusion Maps

In diffusion maps, we create the kernel matrix $K$ by running the heat kernel on the data. We then normalize each row such that it sums to 1:

$$A = D^{-1} K$$

We can treat this matrix as a Markov matrix, where $A_{i,j}$ defines the probability of moving from point $x_i$ to point $x_j$ in a random walk on the graph:

$$A_{i,j} = \mathbb{P}(X_t = x_j | X_{t-1} = x_i)$$

1. Show, using induction, that the Markov matrix $A$ can reflect the probabilities after $t$ time steps, by taking the $t$th power of $A$:

   $$A^t_{i,j} = \mathbb{P}(X_t = x_j | X_0 = x_i)$$

2. Show that the all-ones vector, $\mathbf{1}$, is an eigenvector of $A$ with an eigenvalue of 1.

3. Show, without using Peron-Frobenius, that all eigenvectors of $A$ are smaller or equal to 1 in absolute value:

   $$\forall i : |\lambda_i| \leq 1$$

# 3 Practical Exercise (75 Points)

Please submit a single tar file named "ex4_<YOUR_ID>". This file should contain your code, along with an "Answers.pdf" file in which you should write your answers and provide all figures/data to support your answers (write your ID in there as well, just in case). Your code will be checked manually so please write readable, well documented code.

If you have any constructive remarks regarding the exercise (e.g. question X wasn't clear enough, we lacked the theoretical background to complete question Y...) we'll be happy to read them in your Answers file.

## 3.1 Exercise Requirements

1. Implement the MDS algorithm (10 points)
2. Generate scree plots to learn more about MDS (10 points)
3. Implement the LLE algorithm (17.5 points)
4. Implement the Diffusion Map algorithm (17.5 points)
5. Explore different data sets using the implemented algorithms (20 points)

## 3.2 Datasets

In this exercise we will use the algorithms we learned in class to reduce the dimensionality of several data sets, while trying to preserve their innate structure.

### 3.2.1 MNIST Digits

We will attempt to visualize the MNIST dataset it in two dimensions while still maintaining the clusters of the original labels.

We will use a lower resolution MNIST, which can be accessed using the scikit-learn Python package. An example for loading the data can be found in the complimentary code (the digits_example function).

### 3.2.2 Swiss Roll

The Swiss Roll is an artificially generated data set of two dimensional data (a rectangle) embedded in three dimensions as a spiral. If we are able to successfully approximate the geodesic distances in the original plane, we should be able to embed the data in two dimensions as the original rectangle.

This data set can also be generated easily using scikit-learn, and an example of using and plotting the Swiss roll can be found in the complimentary code (the swiss_roll_example function).

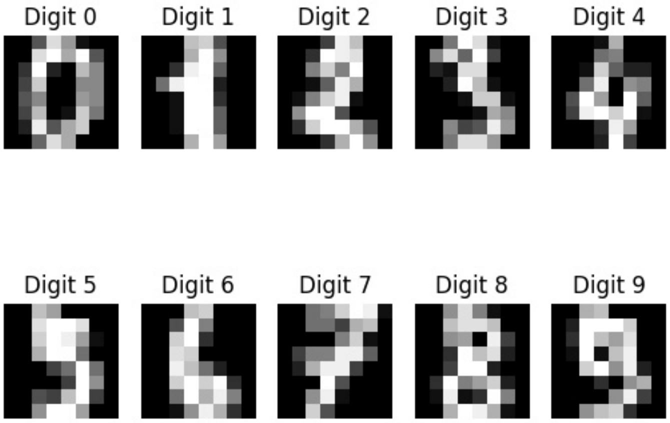

Figure 1: An example of the low resolution MNIST data set

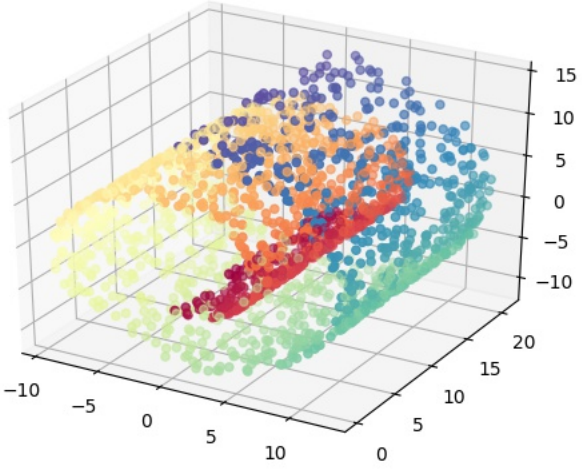

Figure 2: The Swiss Roll data set - a rectangle embedded non linearly in 3D

#### 3.2.3 Faces

The data set consists of simple 64x64 pictures of a face from different angles and lighting. Since the changes in lighting and angle are gradual, we should be able to detect this local structure and the highly non linear but low dimensional degrees of freedom in the data can be embedded well in low dimension.

This data is given to you in this exercise in the file "faces.pickle". An example of loading the data can be found in the complimentary code (the faces_example function).

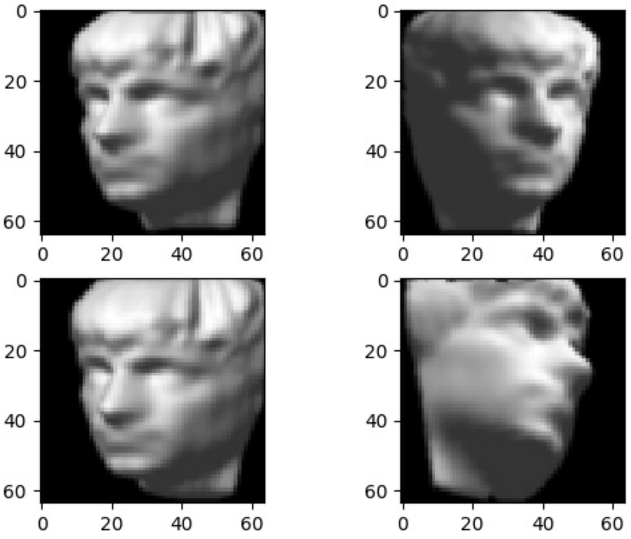

Figure 3: The Faces data set

### 3.3 Review Of The Algorithms

Just to make sure we aren't lost, here is a quick review of the steps of the algorithms:

#### 3.3.1 MDS

Given a data matrix:

1. Compute the squared euclidean distance matrix $\Delta_{i,j} = ||x_i - x_j||^2$
2. From the distance matrix, form the matrix $S = -\frac{1}{2}H\Delta H$. ($H = I - \frac{1}{n}11^T$)
3. Diagonalize $S$ to form $S = U\Lambda U^T$
4. Return the $n \times d$ matrix of columns $\sqrt{\lambda_i}u_i$ for $i = 1...d$ (the eigenvectors corresponding to the largest eigenvalues).

#### 3.3.2 LLE

Given a data matrix:

1. Compute the KNN graph of the data matrix (you may use the naive $O(n^2p^2)$ approach of calculating pairwise distances).
2. Compute $W$ using the inverse of the Gram matrix as explained in the theoretical question. Use the pinv function to avoid cases when $G$ is singular.
   Note - the rows of $W$ can be calculated independently from one another.
3. Decompose $M^TM = (I-W)^T(I-W)$ into its eigenvectors and return the ones corresponding to the $2 \ldots (d+1)$ lowest eigenvalues (the lowest one has eigenvalue 0 and is discarded).

### 3.3.3 Diffusion Map

Given a data matrix:

1. Create a kernel similarity matrix $K$, using the heat kernel.
2. Normalize the rows of $K$ to form the Markov Transition Matrix $A$.
3. Decompose $A$ into its eigenvectors and select only the ones corresponding to the $2 \ldots (d+1)$ highest eigenvalues.
4. Return those eigenvectors, where the $i^{th}$ eigenvector is multiplied by $\lambda_i^t$.

## 3.4 Implementing The Algorithms

Implement MDS, LLE and Diffusion Maps as instructed in “Manifold_Learning.py”.

### 3.4.1 Scree Plot

As we discussed in class, a common way for deciding which dimension we should reduce our data to when using PCA or MDS, is looking at the eigenvalues of the eigendecomposition of the matrix in the algorithm (covariance matrix for PCA, centered distance matrix for MDS). When we see a point where the eigenvalues become low compared to previous eigenvalues, that’s where we decide to stop.

Create a random 2-dimensional data set and embed it in a higher dimension using a random rotation matrix, then add Gaussian noise to your new data set. Test varying degrees of noise and see how the eigenvalues of MDS change as the noise increases.

A reminder - you can obtain a random rotation matrix by creating a random Gaussian matrix, then performing a QR decomposition. The Q matrix you get from the QR decomposition will be your rotation matrix.

### 3.4.2 MNIST

Compare the results of the three algorithms on the MNIST data set. Discuss your results.

### 3.4.3 Swiss Roll

Compare the results of the three algorithms on the Swiss Roll data set.

Discuss the reason for the poor results of MDS.

Between LLE and DM, which algorithm required less parameter tweaking to reach good results?

### 3.4.4 Faces

Compare the results of the three algorithms on the Faces data set.

Were you able to extract the lower dimensional degrees of freedom of the data?

Show plots and discuss the embeddings. You may use the supplied "plot_with_images" function to show the original pictures superimposed on a scatter plot of the lower dimensional data extracted from the algorithm.

### 3.4.5 Parameter Tweaking

Discuss how the k-neighbors parameter of LLE effects the results of the algorithm. What is the problem with choosing a k which is too large or too small?

Show plots to support your claims.

### 3.4.6 Main Function

Finally, please write a basic main function which demonstrates your implementation.