Project Report

Saeed Razavi

IML - Dr. Amini - Spring 2023

Contents

1 Abstract

In the previous phase we have learned how to find parameters of a GMM distribution using EM Algorithm. In this phase we want to apply this algorithm to infer "clean" image from corrupted images (in this case, MNIST dataset). This process is known as "Regularization Inverse Problem". We will solve this problem in multiple steps.

2 Implementation

2.1 Step1: Data Preparation

2.1.1 Simulation Question 1

This function takes an input image and divides it into overlapping patches of size m x m. It returns an array of flattened patches:

```
def patchify(image:np.ndarray, m : int ):

    r,c=image.shape
    vec=[]
    for i in range(0,r-m+1):
        for j in range(0,c-m+1):
            vec.append(image[i:i+m,j:j+m].flatten())

    return np.array(vec)
```

Using patchify function along with the code below, we extract raw features of an image:

```
    for i,image in enumerate(X_train):

        if(i==0):
           datapoints=patchify(image,m=16)
        else :
           datapoints=np.concatenate((datapoints, patchify(image,m=16)), axis=0)
```

2.2 Step2: Denoising Algorithm

2.2.1 Simulation Question 2

for GMM training, we use GMM function from sklearn library: (in this question, k=11 as a number of clusters gives us best answer)

```
    gm = GaussianMixture(n_components=11, random_state=0).fit(datapoints)
```

2.2.2 Theory Question 1

First let's define 2 lemmas:

Lemma 1: For MVN distribution we know:

$$\begin{bmatrix} \boldsymbol{y_1} \\ \boldsymbol{y_2} \end{bmatrix} \sim \mathcal{N}\Big(\begin{bmatrix} \boldsymbol{\mu_1} \\ \boldsymbol{\mu_2} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma_{11}} & \boldsymbol{\Sigma_{12}} \\ \boldsymbol{\Sigma_{21}} & \boldsymbol{\Sigma_{22}} \end{bmatrix} \Big)$$

Then We have:

$$p(\boldsymbol{y_1}|\boldsymbol{y_2}) = \mathcal{N}(\boldsymbol{y_1}|\boldsymbol{\mu_{1|2}}, \boldsymbol{\Sigma_{1|2}})$$

So that:

$$\boldsymbol{\mu_{1|2}} = \boldsymbol{\mu_1} + \boldsymbol{\Sigma_{12}\Sigma_{22}^{-1}(y_2 - \mu_2)} \quad , \quad \boldsymbol{\Sigma_{1|2}} = \boldsymbol{\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}} \tag{1}$$

We know Gaussian prior is a conjugate prior for Gaussian likelihood. So for the problem we have: (note that here we suppose $\boldsymbol{y_1 = y, y_2 = z}$)

$$p(\boldsymbol{y}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{Wz + b, \Sigma_{y|z}}) \implies$$

$$\boldsymbol{\mu_{y|z} = Wz + b = \mu_y + \Sigma_{yz}\Sigma_z^{-1}(z - \mu_z)} \implies \begin{cases} \boldsymbol{W = \Sigma_{yz}\Sigma_z^{-1}} \implies \boldsymbol{\Sigma_{yz} = W\Sigma_z} \\ \boldsymbol{b = \mu_y - \Sigma_{yz}\Sigma_z^{-1}\mu_z = \mu_y - W\mu_z} \end{cases}$$
$$\implies \boldsymbol{\mu_y = b + W\mu_z}$$

$$\boldsymbol{\Sigma_{y|z} = \Sigma_y - \Sigma_{yz}\Sigma_z^{-1}\Sigma_{zy} = \Sigma_y - W\Sigma_z\Sigma_z^{-1}(W\Sigma_z)^T = \Sigma_y - W\Sigma_z W^T}$$
$$\implies \boldsymbol{\Sigma_y = \Sigma_{y|z} + W\Sigma_z W^T}$$

So for joint distribution we have:

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{z} \end{bmatrix} \sim \mathcal{N}\Big(\begin{bmatrix} \boldsymbol{\mu_y} \\ \boldsymbol{\mu_z} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma_y} & \boldsymbol{\Sigma_{yz}} \\ \boldsymbol{\Sigma_{zy}} & \boldsymbol{\Sigma_z} \end{bmatrix} \Big)$$

The relationship between parameters will be:

$$\boldsymbol{\Sigma_{yz} = W\Sigma_z} \ , \ \boldsymbol{\mu_y = b + W\mu_z} \ , \ \boldsymbol{\Sigma_y = \Sigma_{y|z} + W\Sigma_z W^T} \tag{2}$$

$$\implies \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{z} \end{bmatrix} \sim \mathcal{N}\Big(\begin{bmatrix} \boldsymbol{b + W\mu_z} \\ \boldsymbol{\mu_z} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma_{y|z} + W\Sigma_z W^T} & \boldsymbol{W\Sigma_z} \\ \boldsymbol{\Sigma_z W^T} & \boldsymbol{\Sigma_z} \end{bmatrix} \Big)$$

So we have:

$$\begin{bmatrix} \boldsymbol{z} \\ \boldsymbol{y} \end{bmatrix} \sim \mathcal{N}\Big(\begin{bmatrix} \boldsymbol{\mu_z} \\ \boldsymbol{b + W\mu_z} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma_z} & \boldsymbol{\Sigma_z W^T} \\ \boldsymbol{W\Sigma_z} & \boldsymbol{\Sigma_{y|z} + W\Sigma_z W^T} \end{bmatrix} \Big) \qquad \blacksquare$$

Lemma 2: For MVN distribution we know:

$$\begin{bmatrix} \boldsymbol{y_1} \\ \boldsymbol{y_2} \end{bmatrix} \sim \mathcal{N}\Big(\begin{bmatrix} \boldsymbol{\mu_1} \\ \boldsymbol{\mu_2} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma_{11}} & \boldsymbol{\Sigma_{12}} \\ \boldsymbol{\Sigma_{21}} & \boldsymbol{\Sigma_{22}} \end{bmatrix} \Big) \quad , \quad \boldsymbol{\Lambda = \Sigma^{-1}} = \begin{bmatrix} \boldsymbol{\Lambda_{11}} & \boldsymbol{\Lambda_{12}} \\ \boldsymbol{\Lambda_{21}} & \boldsymbol{\Lambda_{22}} \end{bmatrix}$$

Then we have:

$$p(\boldsymbol{y_1}|\boldsymbol{y_2}) = \mathcal{N}(\boldsymbol{y_1}|\boldsymbol{\mu_{1|2}}, \boldsymbol{\Sigma_{1|2}})$$

So that:

$$\boldsymbol{\mu_{1|2}} = \boldsymbol{\Sigma_{1|2}(\Lambda_{11}\mu_1 - \Lambda_{12}(y_2 - \mu_2))} \quad , \quad \boldsymbol{\Sigma_{1|2}} = \boldsymbol{\Lambda_{11}^{-1}} \tag{3}$$

Considering the result of previous part we have: (note that here we suppose $(\boldsymbol{y_1 = z, y_2 = y})$

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma_z} & \boldsymbol{\Sigma_z W^T} \\ \boldsymbol{W\Sigma_z} & \boldsymbol{\Sigma_y + W\Sigma_z W^T} \end{bmatrix}$$

For matrix calculations we have: $\boldsymbol{\Lambda}$ With the help of complement schur, we have:

$$\boldsymbol{\Lambda = \Sigma^{-1}} = \begin{bmatrix} \boldsymbol{\Sigma_z^{-1} + W^T\Sigma_y^{-1}W} & \boldsymbol{-W^T\Sigma_y^{-1}} \\ \boldsymbol{-\Sigma_y^{-1}W} & \boldsymbol{\Sigma_y^{-1}} \end{bmatrix}$$

By having the matrix: $\boldsymbol{\Lambda}$ And help of the lemma 2, and the equation $\boldsymbol{\mu_y = W\mu_z + b}$ we have:

$$\boldsymbol{\Sigma_{z|y} = \Lambda_{11}^{-1} = (\Sigma_z^{-1} + W^T\Sigma_y^{-1}W)^{-1} \implies \Sigma_{z|y}^{-1} = \Sigma_z^{-1} + W^T\Sigma_y^{-1}W}$$

$$\begin{aligned}\boldsymbol{\mu_{z|y}} &= \boldsymbol{\Sigma_{z|y}(\Lambda_z\mu_z - \Lambda_{zy}(y - \mu_y)) = \Sigma_{z|y}\big((\Sigma_z^{-1} + W^T\Sigma_y^{-1}W)\mu_z + W^T\Sigma_y^{-1}(y - \mu_y)\big)} \\ &= \boldsymbol{\Sigma_{z|y}\big((\Sigma_z^{-1} + W^T\Sigma_y^{-1}W)\mu_z + W^T\Sigma_y^{-1}(y - W\mu_z - b)\big)} \\ &= \boldsymbol{\Sigma_{z|y}\big(\Sigma_z^{-1}\mu_z + W^T\Sigma_y^{-1}W\mu_z - W^T\Sigma_y^{-1}W\mu_z + W^T\Sigma_y^{-1}(y - b)\big)} \\ &= \boldsymbol{\Sigma_{z|y}\big(\Sigma_z^{-1}\mu_z + W^T\Sigma_y^{-1}(y - b)\big)} \qquad \blacksquare\end{aligned}$$

2.2.3 Theory Question 2

We can say that:

$$p_{\boldsymbol{Z}}(\boldsymbol{z}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu_{z,k}}, \boldsymbol{\Sigma_{z,k}}) \quad , \quad p_{\boldsymbol{Y|Z}} = \mathcal{N}(\boldsymbol{y}|\boldsymbol{Wz + b}, \boldsymbol{\Sigma_{y|z}})$$

$$p_{\boldsymbol{Z|Y}}(\boldsymbol{z}|\boldsymbol{y}) = \frac{p_{\boldsymbol{Y|Z}}(\boldsymbol{y}|\boldsymbol{z})p_{\boldsymbol{Z}}(\boldsymbol{z})}{p_{\boldsymbol{Y}}(\boldsymbol{y})} = \frac{\sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu_{z,k}}, \boldsymbol{\Sigma_{z,k}})\mathcal{N}(\boldsymbol{y}|\boldsymbol{Wz + b}, \boldsymbol{\Sigma_{y|z}})}{p_{\boldsymbol{Y}}(\boldsymbol{y})}$$

Also based on the theory question 1, we can conclude that:

$$\frac{\mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu_z}, \boldsymbol{\Sigma_z})\mathcal{N}(\boldsymbol{y}|\boldsymbol{Wz + b}, \boldsymbol{\Sigma_{y|z}})}{p_{\boldsymbol{Y}}(\boldsymbol{y})} = \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu_{z|y}}, \boldsymbol{\Sigma_{z|y}}) \implies$$

$$p_{\boldsymbol{Z|Y}}(\boldsymbol{z}|\boldsymbol{y}) = \frac{\sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu_{z,k}}, \boldsymbol{\Sigma_{z,k}})\mathcal{N}(\boldsymbol{y}|\boldsymbol{Wz + b}, \boldsymbol{\Sigma_{y|z}})}{p_{\boldsymbol{Y}}(\boldsymbol{y})} = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu_{z,k|y}}, \boldsymbol{\Sigma_{z,k|y}})$$

So according to the relationship above, posterior distribution is also GMM, and for its parameters we can say that:

$$\boldsymbol{\Sigma_{z,k|y}^{-1} = \Sigma_{z,k}^{-1} + W^T\Sigma_y^{-1}W} \tag{4}$$

$$\boldsymbol{\mu_{z,k|y} = \Sigma_{z,k|y}\big(W^T\Sigma_y^{-1}(y - b) + \Sigma_{z,k}^{-1}\mu_{z,k}\big)} \tag{5}$$

Not that the π-parametes for posterior distribution and prior distribution are equal.

2.2.4 Theory Question 3

We could say that based on theory question 2, if the prior distribution of Z be GMM, and likelihood distribution be Gaussian, then the posterior distribution will be GMM with equation (4) and (5) parameters. So for clustering problem with K classes, GMM distribution for random variable Z is suggested.

2.2.5 Simulation Question 3

This function calculates the mean and covariance matrix of a patch given the current estimates of the mean (μ_k), covariance matrix (Σ_k), noise variance ($variance$), and the mixing matrix (W). It uses Bayesian inference and returns the updated mean and covariance.

```
def cal_mean(mu_k,sig_k,W,variance,patch):

h,w=W.shape
#-------------------
sigma_y=(variance**2)*np.identity(h)
#-------------------
sig_MAP_inv=inv(sig_k)+W.T@inv(sigma_y)@W
epsilon=1e-2
sig_MAP=inv(sig_MAP_inv)+epsilon*np.eye(h)
#-------------------
mean_MAP=sig_MAP@(W.T@inv(sigma_y)@patch+inv(sig_k)@mu_k)

return mean_MAP,sig_MAP
```

This function reconstructs the full image from the predicted patches. It takes the predicted patches, the patch size (m), and the length of the original image (default is 28 for MNIST images). It averages the overlapping patches to create the final reconstructed image.

```
def reconstruct(predicted_patches,m, length = 28 ):

num= length - m +1
count_matrix = np.zeros((28, 28))
reconstructed_matrix= np.zeros((28, 28))
#---------------------------------
for i,patch in enumerate(predicted_patches):

    patch_temp= patch.reshape((m, m))
    count_temp= np.ones((m,m))
    #-------------------
    col = i // num
    row = i % num
    #-------------------
    reconstructed_matrix[col:col+m,row:row+m]+=patch_temp
    count_matrix[col:col+m,row:row+m]+=count_temp


reconstructed_matrix = np.array(reconstructed_matrix /
    count_matrix,dtype='float64')

return reconstructed_matrix
```

Using two above functions along with following denoise function retrieve noisy image for us.

```
    def denoise(corrupted_image,m,sigma=25,num_k=11):

  patch_arr=patchify(corrupted_image,m)
  retrived_patch=[]
  for i,patch in enumerate(patch_arr) :

      mean_list,sigma_list,prob_list=[],[],[]

      for k in range(num_k):

        mean_k,sigma_k=cal_mean(gm.means_[k],gm.covariances_[k],W,sigma,patch)
        mean_list.append(mean_k)
        sigma_list.append(sigma_k)
        #--------------------
        var = multivariate_normal(mean=mean_k, cov= sigma_k ,allow_singular=True )
        prob_list.append(np.float64(gm.weights_[k]*var.pdf(mean_k)))

      j=np.argmax(prob_list)
      # sample = np.random.multivariate_normal(mean_list[j], sigma_list[j], 1)[0]
          # sampling
      #-------------
      sample=np.array(copy.deepcopy(mean_list[j]),dtype="float64")
      sample[sample>255]=255
      sample[sample<1]=0
      #-------------
      retrived_patch.append(sample)

  return retrived_patch
```

2.2.6 Simulation Question 4

Now we reconstruct noisy images using following commands:

```
    plt.figure(figsize=(20, 20))

for number in range(0,5):

    plt.subplot(5,3,(number)*3+1)
    original_image=plt.imread(f"/content/MNIST/original/10{number}.png")
    original_image=np.array(255*original_image,dtype="uint8")

    plt.imshow(original_image,cmap='gray')
    if(number==0):
      plt.title("original image")

    noisy_image=plt.imread(f"/content/MNIST/corrupted/10{number}.png")
```

```
noisy_image=np.array(255*noisy_image,dtype="uint8")
plt.subplot(5,3,(number)*3+2)
plt.imshow(noisy_image,cmap='gray')
if(number==0):
  plt.title("noisy image")

retrived_patch=denoise(noisy_image,16)
retrived_image = reconstruct(retrived_patch,16)
plt.subplot(5,3,(number)*3+3)
plt.imshow(np.array(retrived_image,dtype='uint8'),cmap='gray')

if(number==0):
   plt.title("denoised image")
```

Below, you can see the result of our GMM model :

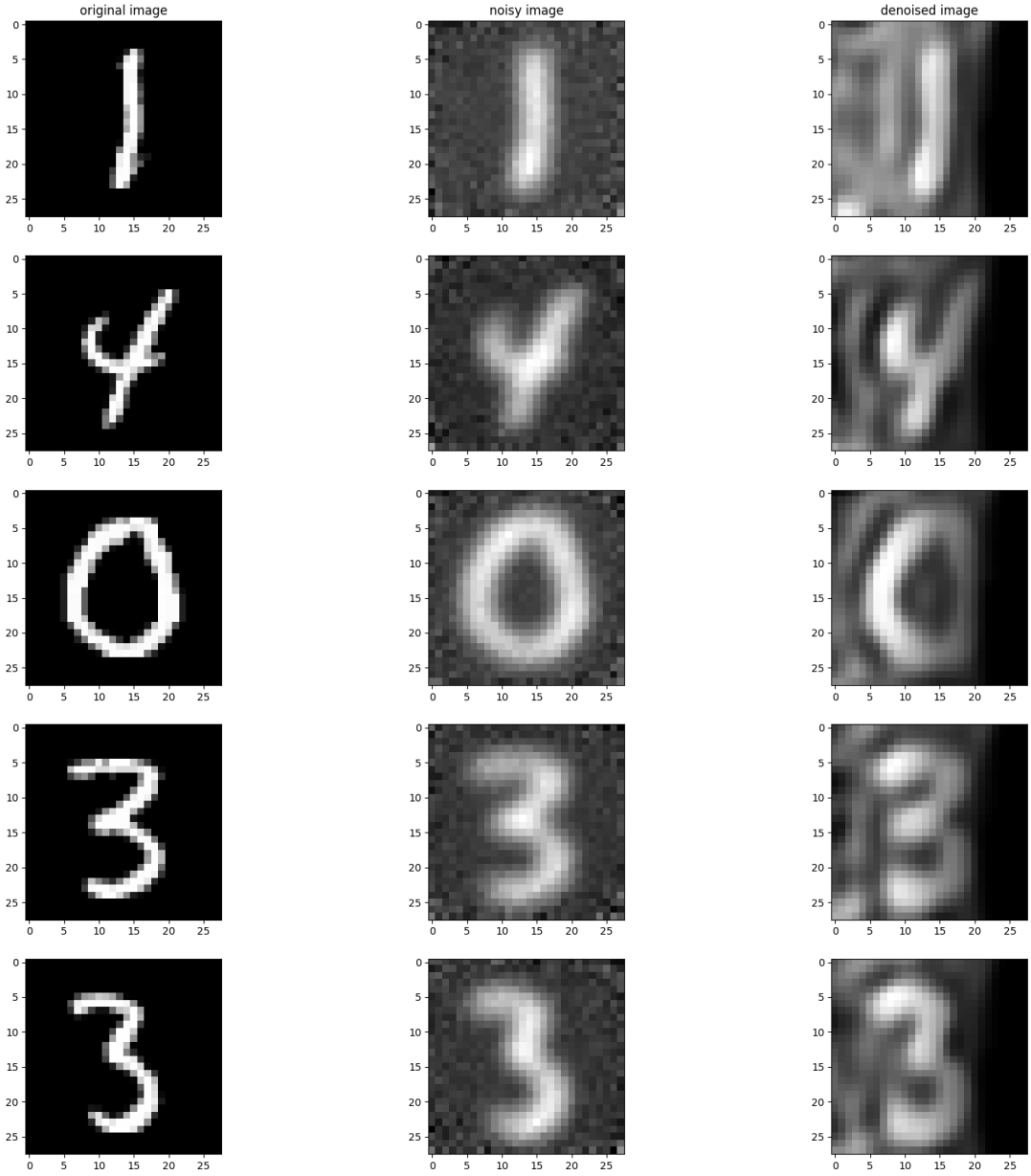

Figure 1: Denoised Images using our GMM model

Above given results are given from these hyperparameteres:

- $standard\ deviation = \sigma = 20$
- $size\ of\ gaussian\ clusters = 11$
- $m = 16$
- $epsilon\ for\ handle\ singularity = 0.01$