SIMC National Junior College Report

Liu Linxi, Kang Yu Kiat Zavier, Evan Huang Haochan

May 22, 2024

1 Classifying a few noiseless patterns

1.1 Task 1(a)

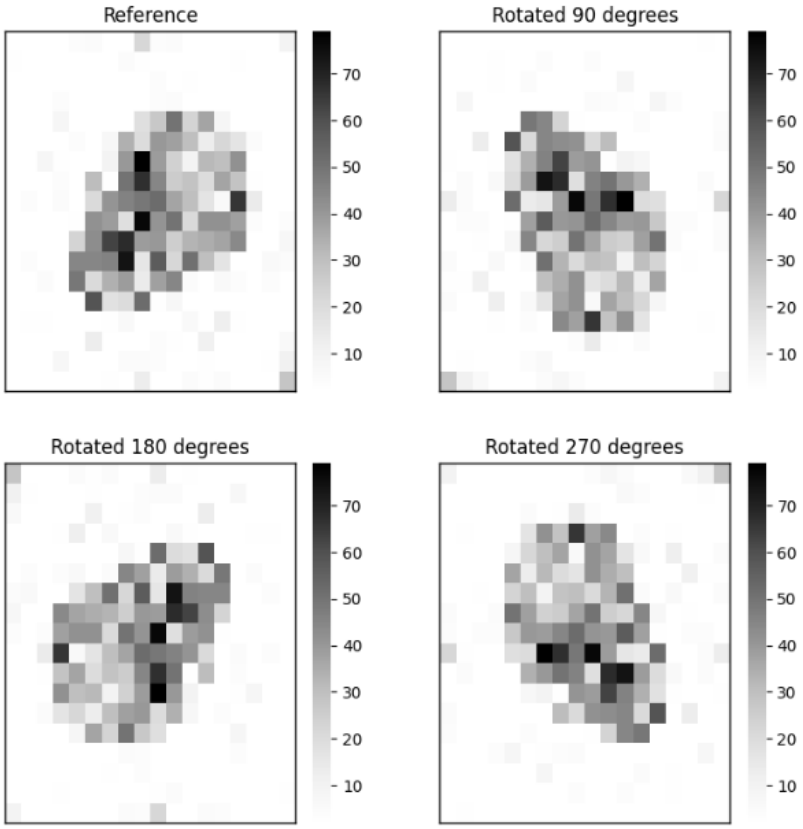

Figure 1: Four orientation of the image

1.2 Task 1(b)

In the order of 0, 90, 180, 270 degrees : [6, 10, 6, 3]

1.3 Task 1(c)

Suppose we have a N x P x P dimensional matrix, where P x P is the number of pixels in the pattern. Taking the first pattern as the reference pattern, we generate four matrices representing the four orientations of the reference pattern using the rot90 method of NumPy. Taking the matrices generated, we used Seaborn's heatmap function and the colormap "Greys" to plot each image. We then hashed each of the matrix data and stored it in an ordered list. Iterating through the N patterns in the task1 dataset, we compare their hashes and increment a corresponding element in the ordered list we use to store the counts. This method only works with the following assumptions; The data is noiseless, which is given; The data is not symmetrical, hence the pattern varies from its different orientations. If the pattern were symmetrical, this task would be unsolvable.

2 Flattening 2D patterns into 1D representation

2.1 Task 2(a)

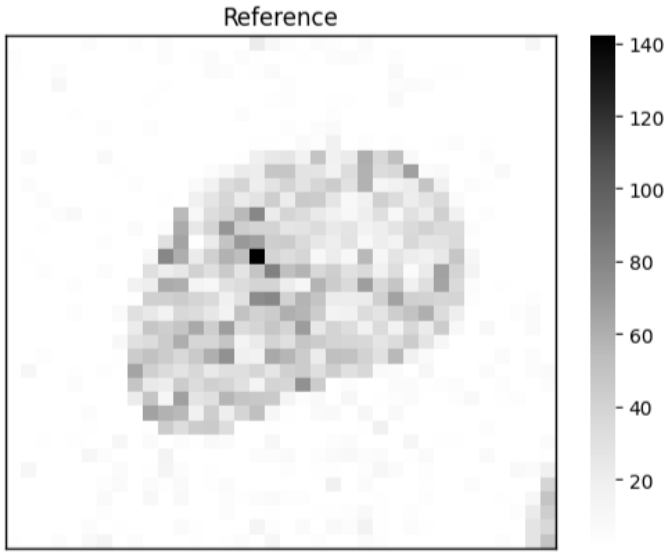

Figure 2: 2D master image

2.2 Task 2(b)

In the order of 0, 90, 180, 270 degrees : [22, 32, 21, 25]

2.3 Task 2(c)

Suppose we have a N x P dimensional matrix, where P is the number of pixels in the pattern. It is then assumed that $\sqrt{P}$ x $\sqrt{P}$ are the dimensions of the pattern. Taking the first pattern as the reference pattern, we first reshape it to a $\sqrt{P}$ x $\sqrt{P}$ matrix. Similar to the first task, we generated the four matrices representing the four orientations with the same method. Iterating through the N patterns in the task2 dataset, we hash their reshaped matrix and compare it

with the hashes stored in the list, incrementing the count of the matching orientation. This method follows the method used in Task 1, and shares the same assumptions. Additionally, it is assumed that all patterns were originally square matrices and that they were all flattened in row-major (C-style) order.

3 Scaling up to thousands of patterns

3.1 Task 3(a)

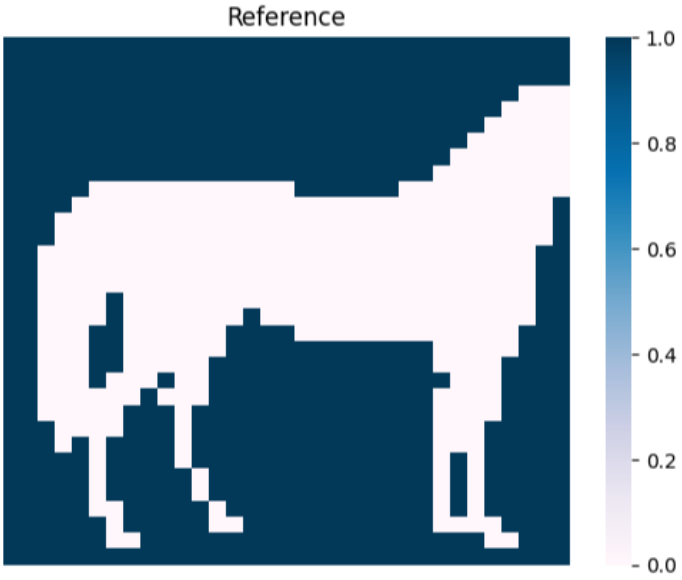

Figure 3: 2D master image of a horse

3.2 Task 3(b)

In the order of 0, 90, 180, 270 degrees : [254, 236, 250, 260]

3.3 Task 3(c)

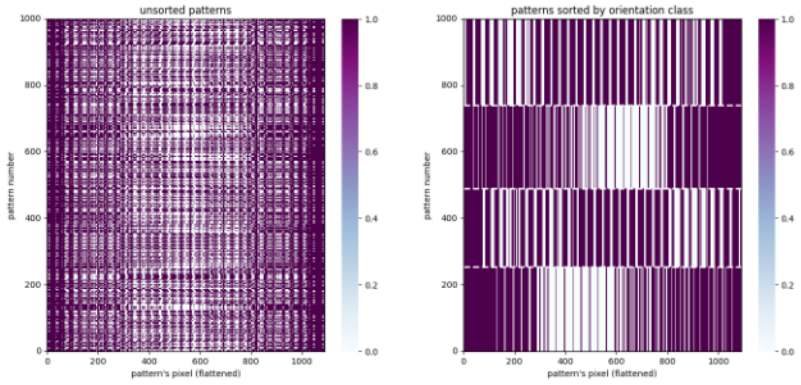

Figure 4: Sorted and unsorted design matrix

3.4 Task 3(d)

Suppose we have a N x P dimensional matrix, where P is the number of pixels in the pattern. We then assume the same conditions as in task 2 and follow the same procedure. Additionally, while iterating through the patterns in the task3 dataset, we append its label to the pattern in a copy of the dataset. We then sort the new dataset before removing the column of labels. The sorted dataset is used to plot the design matrix for task 3(c). The method for this question follows the method used in Task 2, and shares the same assumptions.

4 Noisy Patterns

4.1 Task 4(a)

The average sum of all rows in task 4: 1251.047

4.2 Task 4(b)

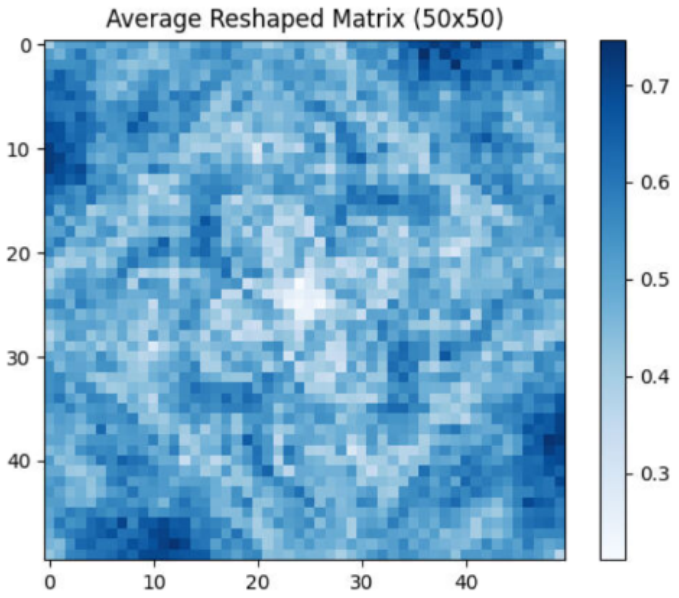

Figure 5: 2D master image

4.3 Task 4(c)

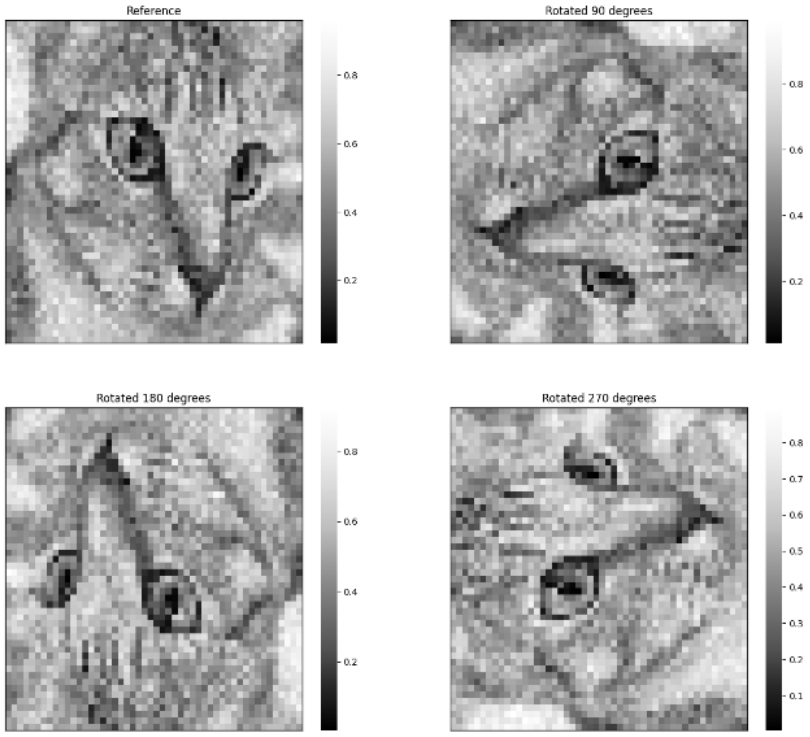

Figure 6: Four orientation of the image

4.4 Task 4(d)

In task 4(a), we took the total pixel values of each row in the original matrix and find its average. In task 4(b), by assuming that they were all in the same orientation, we took the total pixel values of each column in the original matrix and find its average. We then reshape the resultant 1 by 2500 matrix into a 50 by 50 matrix. Lastly, we employed k means to sort each row from the original matrix into its own clusters, we made sure that each cluster also have equal number of labels. For example, the cluster containing reference image should have around 250 rows with the same label.

5 Likelihood to succeed with noisy patterns

5.1 Task 5(a)

$$\begin{aligned}\mathcal{L}\left(r = 90^\circ \mid K_{(4)}, \mu_{(4)}\right) &\equiv P(k_1 = 1 \mid \beta\lambda) \cdot P(k_2 = 0 \mid \lambda) \cdot P(k_3 = 0 \mid \beta\lambda) \cdot P(k_4 = 0 \mid \beta\lambda) \\ &= (\beta\lambda)\,(1-\lambda)\,(1-\beta\lambda)^2\end{aligned}$$

5.2 Task 5(b)

$$\begin{aligned}\frac{\mathcal{L}\left(r=0^{\circ} \mid K_{(4)}, \mu_{(4)}\right)}{\mathcal{L}\left(r=90^{\circ} \mid K_{(4)}, \mu_{(4)}\right)} &= \frac{\lambda\left(1-\beta\lambda\right)^{3}}{(\beta\lambda)\left(1-\lambda\right)\left(1-\beta\lambda\right)^{2}} \\ &= \frac{1-\beta\lambda}{\beta(1-\lambda)}\end{aligned}$$

5.3 Task 5(c)

$$\begin{aligned}\frac{1-\beta\lambda}{\beta\left(1-\lambda\right)} = 1 \Rightarrow & 1-\beta\lambda = \beta\left(1-\lambda\right) \\ \Rightarrow & 1-\beta\lambda = \beta-\beta\lambda \\ \Rightarrow & \beta = 1\end{aligned}$$

5.4 Task 5(d)

$$\begin{aligned}\frac{\mathcal{L}(aligned)}{\mathcal{L}(misaligned)} &\equiv \frac{\mathcal{L}\left(r=0^{\circ} \mid K_{(16)}, \mu_{(16)}\right)}{\mathcal{L}\left(r=90^{\circ} \mid K_{(16)}, \mu_{(16)}\right)} \\ &= \frac{\lambda\left(1-\beta\lambda\right)^{15}}{(\beta\lambda)\left(1-\lambda\right)\left(1-\beta\lambda\right)^{14}} \\ &= \frac{1-\beta\lambda}{\beta(1-\lambda)}\end{aligned}$$

5.5 Task 5(e)

$$\begin{aligned}\frac{\mathcal{L}(aligned)}{\mathcal{L}(misaligned)} &\equiv \frac{\mathcal{L}\left(r=0^{\circ} \mid K_{(16)}, \mu_{(16)}\right)}{\mathcal{L}\left(r=90^{\circ} \mid K_{(16)}, \mu_{(16)}\right)} \\ &= \frac{\lambda^{3}\left(1-\beta\lambda\right)^{13}}{(\beta\lambda)^{3}\left(1-\lambda\right)^{3}\left(1-\beta\lambda\right)^{10}} \\ &= \frac{(1-\beta\lambda)^{3}}{\beta^{3}\left(1-\lambda\right)^{3}} \\ &= (\frac{1-\beta\lambda}{\beta(1-\lambda)})^{3}\end{aligned}$$

5.6 Task 5(f)

$$\begin{aligned}\frac{1}{N} ln(\mathcal{L}(aligned \mid \overrightarrow{\mu}, \overrightarrow{k})) &= \frac{1}{N} \sum_{i=1}^{Npixels} (k_i ln(\lambda) + (1-k_i) ln(1-\beta\lambda)) \\ &= \frac{1}{N}(M ln(\lambda) + (N-M) ln(1-\beta\lambda)) \\ &= \frac{M}{N} ln(\lambda) + (1-\frac{M}{N}) ln(1-\beta\lambda)\end{aligned}$$

6 Scaling up with sparse data format

6.1 Task 6(a)

The average sum of all rows in task 6: 23.745372701609828

6.2 Task 6(b)

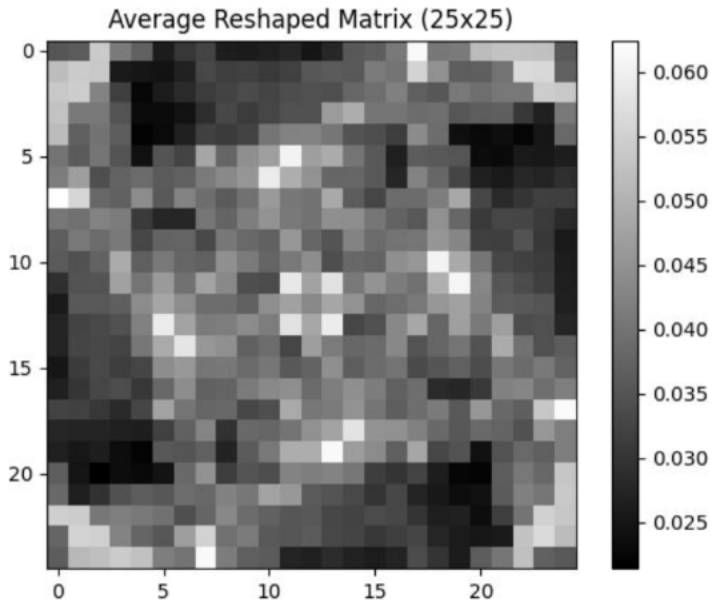

Figure 7: 2D master image

6.3 Task 6(c)

Figure 8: Four orientation of the image

6.4 Task 6(d)

Before we embarked on the task, we had to create a 65535 by 625 zero matrix. We then took the column's and row's indices from files task6a and task6b and generate the original matrix. In task 6(a), we took the total pixel values of each row in the original matrix and find its average. In task 6(b), by assuming that they were all in the same orientation, we took the total pixel values of each column in the original matrix and find its average. We then reshape the resultant 1 by 625 matrix into a 25 by 25 matrix. Lastly, we employed k means to sort each row from the original matrix into its own clusters, we made sure that each cluster also have equal number of labels. For example, the cluster containing reference image should have around $\left(\frac{65535}{4}\right)$ rows with the same label.

7 Sharing data amongst orientations

7.1 Task 7(a)

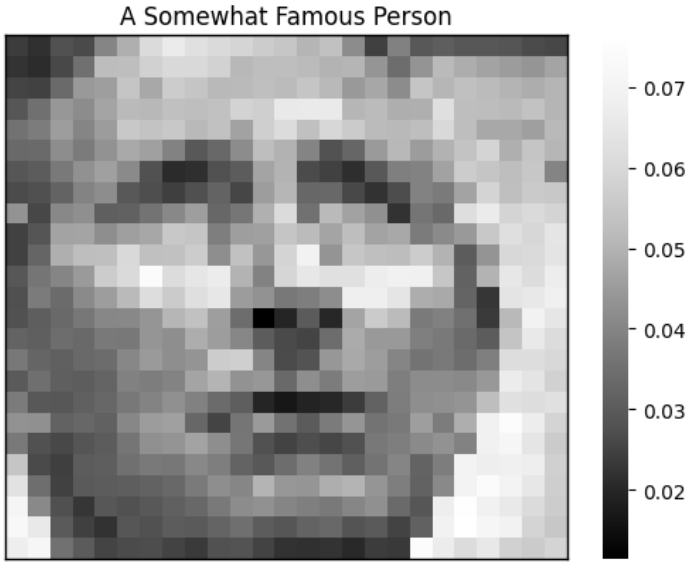

Figure 9: A somewhat famous person, who we assumed to be Putin (current president of Russia)

7.2 Task 7(b)

Similar to the previous task, we first reconstructed the flattened matrix using the row and column indices, given that it was a 100000 x 625 matrix. When plotting out the reshaped average matrix, we noticed the symmetric features in the matrix, which led us to assume that there were four orientations. Initially, we attempted to reuse the same procedure as in Task 6. However, we soon realised that the algorithm was unable to produce any meaningful clusters due to the significant amount of noise. The majority of the time, it generates a large cluster of roughly 70% of the samples and a few much smaller ones. On the rare occasion that it produces a cluster of around 30000, a face was faintly visible. In our following attempts, we tried varying the parameters, including changing the algorithm to Elkan, as well as different clustering algorithms. The ones we tried include Spectral Clustering, DBSCAN, OPTICS and Gaussian Mixture. None gave a clear, suitable clustering and some failed to even produce a result. For example, DBSCAN marked too many samples as noisy and we gave up on Spectral Clustering after it took more than 20 minutes. We next attempted manual clustering. We observed that at various segments of the reshaped average matrix, there appeared to be some features that could be used to identify the orientation. We tried to sum certain groups of pixels for each pattern and cluster the patterns according to that, which led to too specific clusters and did not produce results either. Eventually, we chose to use dimensionality reduction, specifically Principal Component Analysis. Since the main issue with the clustering was due to the large amount of noise, we chose to minimize the effects of the noise through PCA. As the pixels with noise would be considered insignificant dimensions, we used PCA to reduce its impact. The result was each cluster mean being an specific orientation of a clear picture

of a person. As mentioned in the question, the master image may be too noisy if only one cluster is used to render it. To overcome this, we oriented the other clusters to align with the upright image. Applying Equation (1), with M_i being the mean matrix of a cluster i and n_i being the number of samples in it, we computed the overall matrix M_{new}. Our final result is shown in Figure 9.

$$M_{new} = \frac{n_1 M_1 + n_2 M_2 + ... + n_k M_k}{n_{total}} \tag{1}$$