University of Waterloo

SYDE 372: Pattern Recognition

Lab 3: Image Classification

Team Members:

Adrian Chan | 20548981

Maria Cheng | 20567883

Devika Khosla | 20570142

Michael Lim | 20567193

Vyshakh Sanjeev | 20554572

1.0 Introduction

To effectively process images and extract contextually relevant information such as identifying tumours in medical images and depth cues from scenes, accurate texture analysis of an image is critical. Within the research field, a standard set of textured images, known as Brodatz images, are used as a benchmark dataset used to validate newly developed classification tools. Throughout this lab, image classification was performed on a given set of Brodatz images, focussing on feature analysis and labelled classification using a Minimum Intra-Class Distance (MICD) approach. Finally, unlabelled clustering was performed on the same set of textured images as a form of unsupervised image classification, implementing the K-means algorithm.

2.0 Feature Analysis

In this section, classification constraints will be defined. The texture image will be split into sixteen n x n blocks where the features will be extracted. Features will be extracted based on the formula below:

$$\underline{x}_{ij} = \begin{bmatrix} \dfrac{\Sigma_{\alpha=1}^{n}\Sigma_{\beta=1}^{n-1}\left(d_{ij}(\alpha,\beta) - d_{ij}(\alpha,\beta+1)\right)^2}{n(n-1)} \\ \dfrac{\Sigma_{\alpha=1}^{n-1}\Sigma_{\beta=1}^{n}\left(d_{ij}(\alpha,\beta) - d_{ij}(\alpha+1,\beta)\right)^2}{n(n-1)} \end{bmatrix}$$

$d_{ij}(\alpha, \beta)$ is defined as the gray level value of the pixel at (α, β) in the j^{th} block and i^{th} image.

Three feature sets were provided for this lab which include the features $\underline{x}_{ij}$, *i*th *pixel, and j*th *pixel*. An alphabet plot of das shown in Figure 1 to visualize the feature set provided

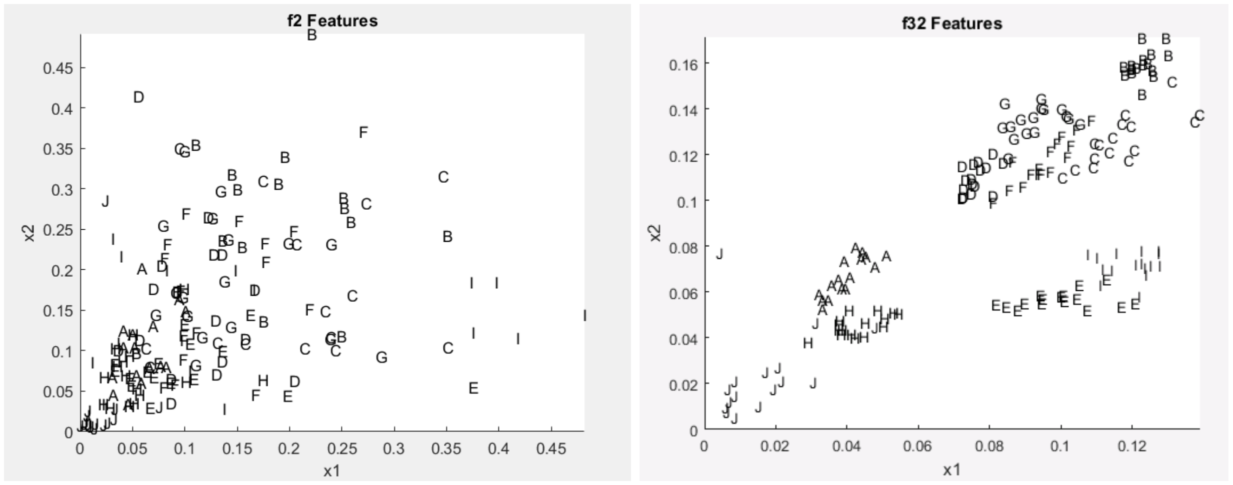

Figure 1. f2 and f32 Alphabet Plot Features.

By looking at the texture images themselves, the image that is most likely to be confused with other images is the cloth. This is because the subsections of cloth all appear the same where they have little to no detail and similar pixel values.

Also, the grass, pigskin, cork, and paper images all look similar, so they are likely to be confused with each other. Like the cloth, they have little detail and no distinct pattern. In the plot of the f32 features, the clusters C, D, F, and G are very close together, so they will likely be more difficult to classify.

The face is likely to be the most distinct because each section of the image has different details. Furthermore, there are clear and notable features from this image allowing the image classifier to be more accurate when detecting the face.

3.0 Labelled Classification

In this section, we implement an MICD classifier for the three feature matrices that are provided, f2, f8, and f32. The number indicated the size of the $n \times n$ block, where f2 is composed of 2×2 blocks. The discriminant function for an MICD classifier in the two class case can be expressed as,

$$x \in A \text{ iff } (x_A - \mu_A)^T \Sigma^{-1} (x_A - \mu_A) < (x_B - \mu_B)^T \Sigma^{-1} (x_B - \mu_B)$$

where x is the feature vector, μ is the sample mean, and Σ is the sample covariance. Since MICD utilizes a distance metric to classify the points, it can be extended to cases with multiple classes.

Here the classifier is applied to the test data f2t, f8t, and f32t. As evident Figure 1 in the assignment, the textures in the smaller 8×8 block is harder to discriminate than those in the large 32×32 block. Thus, we would expect to see better performances with the larger blocks than the smaller blocks. Below, in Tables 1-3, are the confusion matrices for the three test data sets.

Table 1. Confusion Matrix for $n = 2$.

f2t										
	Classified									
Truth	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	1	0	0	2	3	0	1	4	5	0
2	0	7	4	0	2	1	2	0	0	0
3	0	3	0	3	0	0	1	1	7	1
4	0	3	0	1	4	0	1	1	6	0
5	1	0	0	2	4	0	0	5	4	0
6	0	2	3	2	0	2	1	2	3	1
7	0	5	4	0	2	0	1	0	4	0
8	0	0	0	2	0	0	0	9	3	2
9	2	1	2	4	2	0	2	1	2	0
10	0	0	0	0	1	0	0	1	0	14

Table 2. Confusion Matrix for $n = 8$.

f8t										
	Classified									
Truth	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	5	0	0	4	0	0	0	5	0	0
2	0	3	5	0	0	2	3	0	0	0
3	0	2	1	2	0	1	2	0	8	0
4	1	0	0	3	0	1	3	0	2	0
5	0	0	0	2	4	0	0	3	5	0
6	0	0	4	3	0	2	3	0	3	0
7	0	0	0	1	0	1	4	0	0	0
8	0	0	0	0	1	0	0	2	0	4
9	0	0	3	4	5	0	0	0	4	0
10	0	0	0	0	2	0	0	0	1	15

Table 3. Confusion Matrix for $n = 32$.

f32										
	Classified									
Truth	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	5	0	0	0	0	0	0	0	0	1
2	0	3	0	0	0	0	0	0	0	0
3	0	2	1	0	0	0	0	0	0	0
4	0	0	0	3	0	0	0	0	0	0
5	0	0	0	0	4	0	0	0	5	0
6	0	0	4	3	0	2	3	0	0	0
7	0	0	0	0	0	1	4	0	0	0
8	0	0	0	0	0	0	0	2	0	4
9	0	0	0	0	5	0	0	0	4	0
10	0	0	0	0	0	0	0	2	0	15

The diagonals of the confusion matrices indicate the number correctly sampled points. The sum of the off-diagonals along every row indicates the number of wrongly classified points. As a measure of performance of the classifier, we can use these values to calculate the misclassification rate, which is shown below in Table 4.

Table 4. Misclassification Rate for each image and each n.

f2										
Image	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Misclassification Rate (%)	93.75	56.25	100.00	93.75	75.00	87.50	93.75	43.75	87.50	12.50

f8										
Image	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Misclassification Rate (%)	56.25	62.50	93.75	43.75	62.50	81.25	12.50	31.25	75.00	18.75

f32										
Image	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Misclassification Rate (%)	6.25	0.00	12.50	0.00	31.25	62.50	6.25	25.00	31.25	12.50

From the results above, we see that the misclassification rates are significantly smaller for the larger blocks (32×32) compared to the smaller blocks, consistent with what was expected. This is because the larger blocks capture more details of the textures compared to the smaller blocks, making it easier to differentiate between the textures. With the smaller blocks, the images all look similar, making it difficult to build a reliable classifier. Thus, the classifier trained on larger blocks outperforms those trained on smaller blocks.

4.0 Image Classification and Segmentation

In this section, we are provided an image, *multim*, as well as a second matrix, *multf8*, with the corresponding feature vectors for each pixel. The image *multim* is shown below in Figure 2. Using these feature vectors and the MICD classifier from the previous section for 8×8 blocks, we classify every pixel in *multim*. The array of classified pixels is shown in Figure 3 below, with the color indicating the class each pixel has been assigned to.

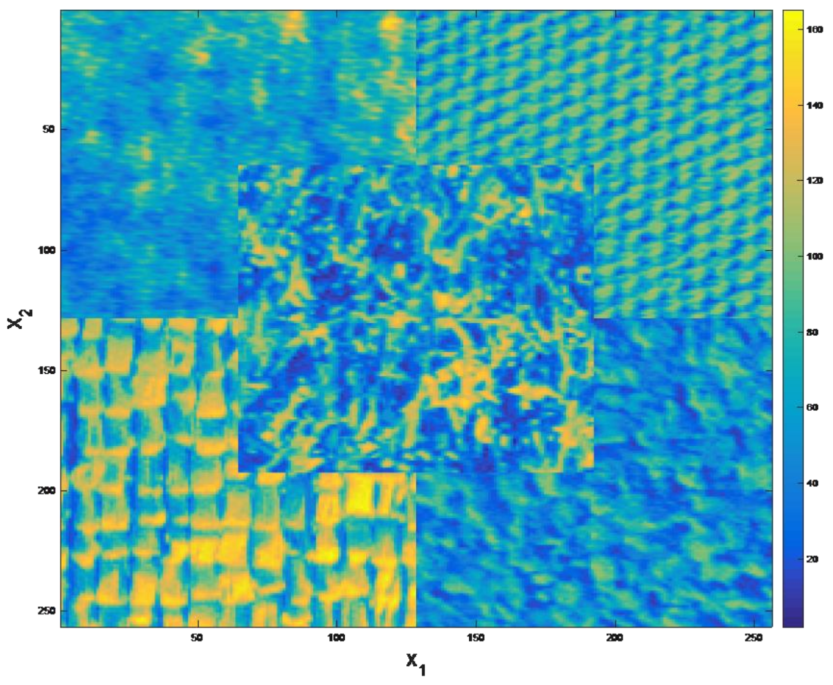

Figure 2. Image composed of multiple different textures.

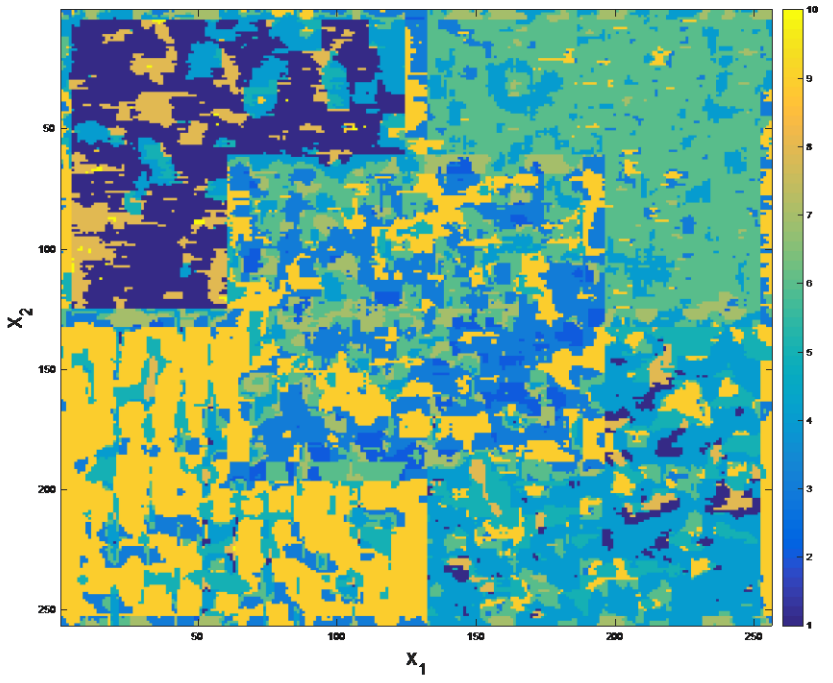

Figure 3. Image Segmentation with MICD classifier for $n = 8$.

From Figure X, we can see that there are five distinct features, one in the center and ones on each of the four corners. Figure Y can also be seen to have identified five segments. The edges evidently are clearer than the centers of each and this is possibly due to the higher contrast at the edges. Furthermore, from Table 4, we see that the misclassification rate for $n = 8$ is large. This explains the large noise within the segments.

5.0 Unlabelled Clustering

In this section, we are to treat the data points in f32 matrix as unlabelled points in feature space. Ten initial prototypes are randomly selected, and the Euclidean distance is calculated for every other data point to those initial prototypes. The data point is then assigned to a cluster number, depending on the minimum Euclidean distance.

$$x \in C_i \; if \; |x - z_i| < |x - z_j|, j \neq i$$

The prototypes are then recalculated based on the cluster's mean. This is an iterative process until the cluster assignment (and therefore prototype mean) does not change anymore, or if a threshold/criteria is reached. Figure 4 below is an example of the result from running the algorithm.

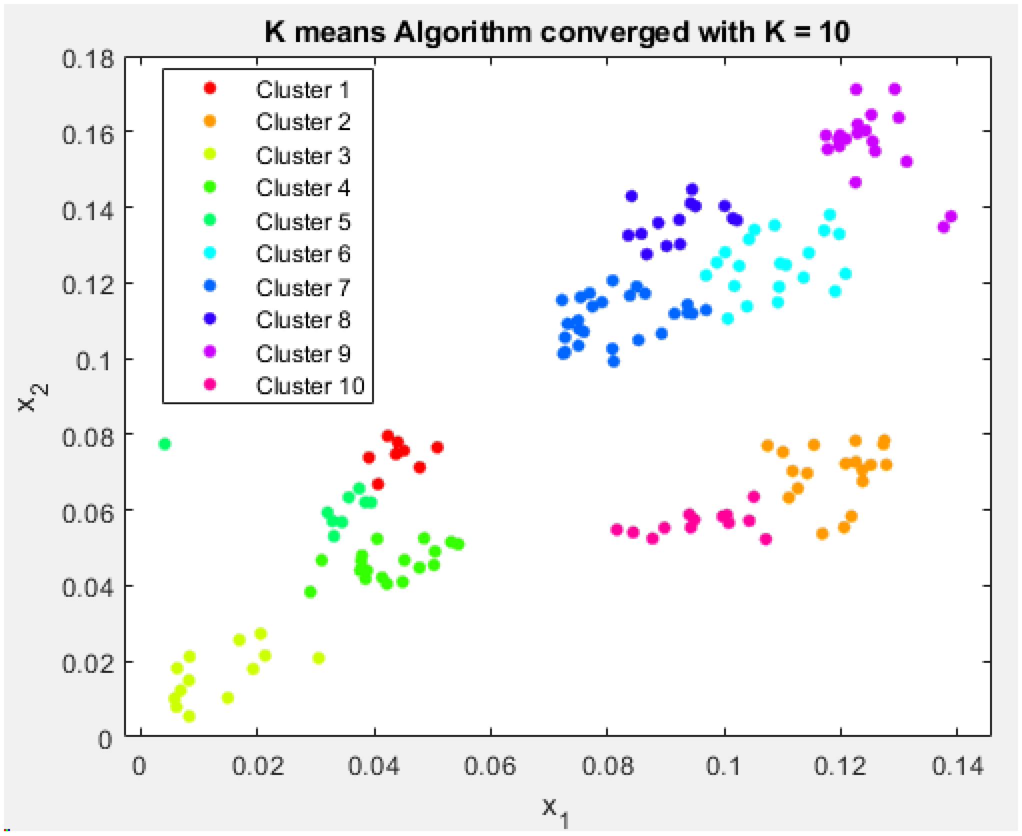

Figure 4. 10 clusters assigned by K-means clustering algorithm.

When repeated a second time, the cluster assignments changed, seen in Figure 5.

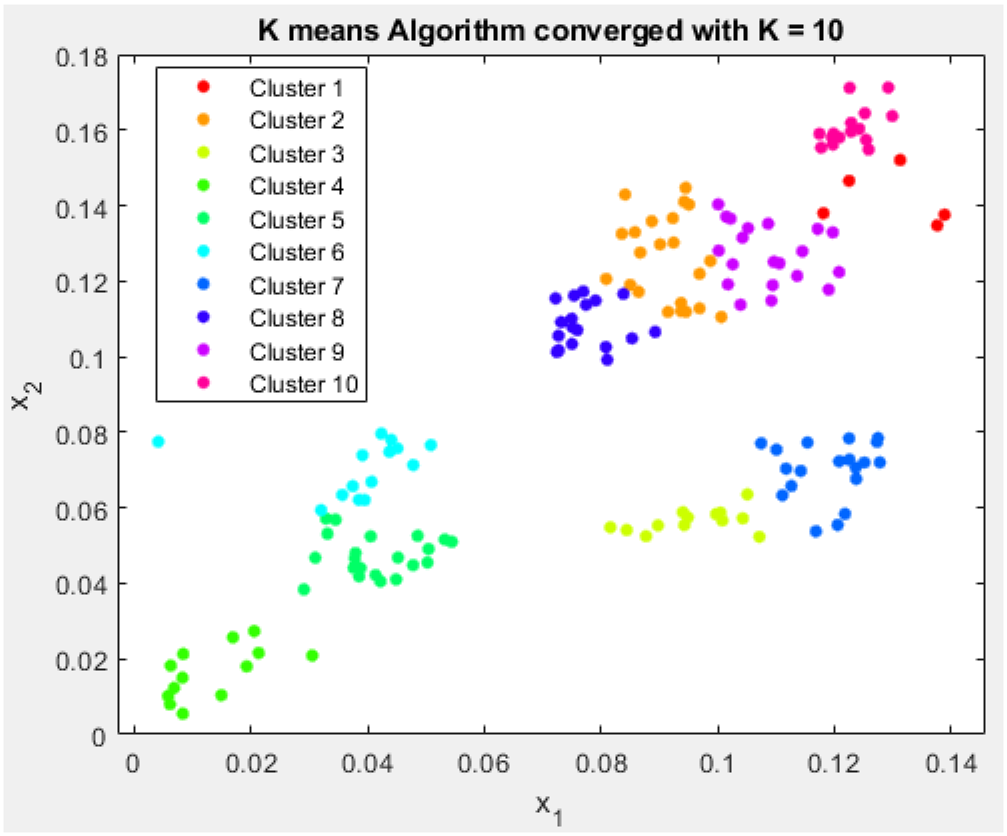

Figure 5. Second repeat of K-means clustering algorithm where K = 10.

This is due to the random initialization of points in the first step. Most of the clusters stay the same between the two figures; however, for the second repeat, the bottom left points are counted as three clusters, whereas for the original, four clusters are identified. This is an indication that the algorithm is quite sensitive to the initialization of points, and it is hard to repeat the same clustering results between attempts. Another observation is the clustering that occurs depends on the number of clusters that were assumed. By just looking at the data, a different number of clusters could be assumed (e.g., 3 large ones). However, we know beforehand that 10 classes exist. Because some of the clusters in the data overlap, this leads to results that are more sensitive to the initialization.

Most of the unlabeled clusters appear distinct from one another; however, there are some differences in cluster groupings depending on which plot is selected for comparison. The initialization of points creates differences between the groupings. MICD misclassification rates appear slightly high for the classes corresponding to 5 and above. Qualitatively, clustering results have better differentiation of points between classes, but the results are not always consistent.

For comparison, a plot of the MICD boundaries is shown in Figure 6.

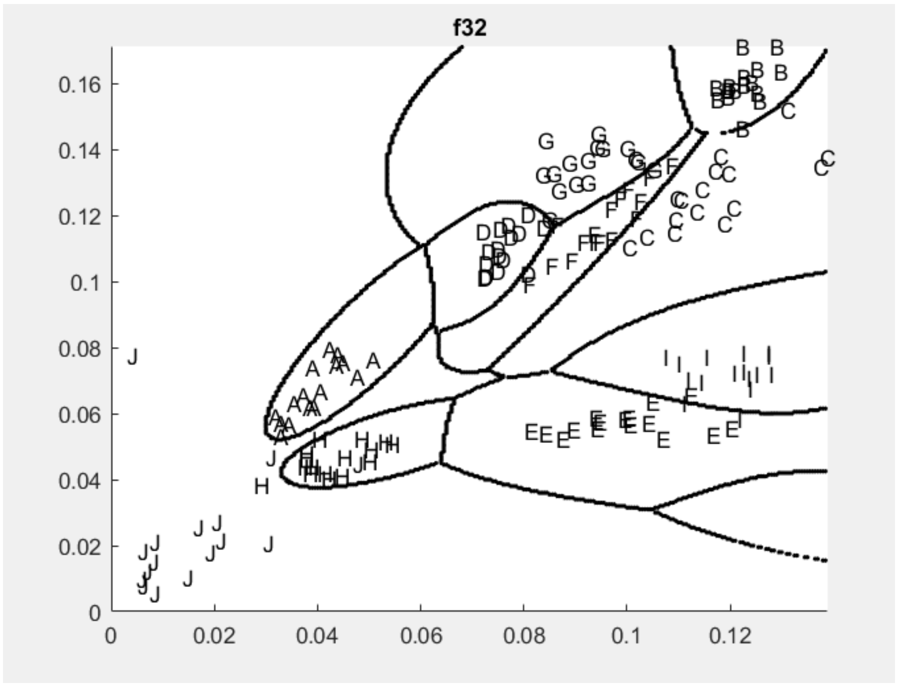

Figure 6. MICD boundaries on clusters

The plots reveal that MICD is slightly better at classifying features C, F, and G. This makes sense because the K-means algorithm is based on Euclidean distance, which causes the clusters to be more circular. However, the clusters corresponding to C and F are more oval shaped. As a result, the K-means misclassifies some of the points in F as belonging to the same cluster as D or G or C.

Multf8 contains pixel values that represent the calculated features for the 8x8 window. K-means should be able to differentiate between similarities between classes, given appropriate initialization values. Similar features would be clustered by their means, and as such, the noise and variability within each textural image segmentation should be reduced. As the two plots of the K-means clustering show, it was able to differentiate between classes A, B, E, H, I, and J fairly well with good initial prototypes. The only issue is the intersecting regions of the stitched images, in which a feature value would be calculated from the 8x8 window, that may correlate to two separate textures. There will also likely be some misclassification for features that lie on the boundaries between classes, and for classes that are elongated or very close together, such as with classes D, F, G, and H. Like the MICD classification it will likely be noisy. Nevertheless, the majority of the *multim* (figure 2) could probably classified by K-means algorithm, given that the number of initial classes are known and ideal.

6.0 Conclusion

In conclusion, image classification on 10 textural images was implemented using MICD and unlabelled clustering (K-means). MICD classification accuracy was determined for 3 different image block sizes (2, 8, and 32). Misclassification decreases with a larger image block size, as more detail can be captured, making it easier to differentiate between the textures. Image classification and segmentation was performed with a window size of 8 against a test image composed of multiple textures, with large noise within each textural image. Lastly, K-means algorithm was used to cluster the images with window size of 32, into 10 clusters. Initialization of prototypes plays a significant role in determining end-result clusters, and as such, results are difficult to duplicate. K-means may be useful in helping differentiate the textural images in the multi-class test image.