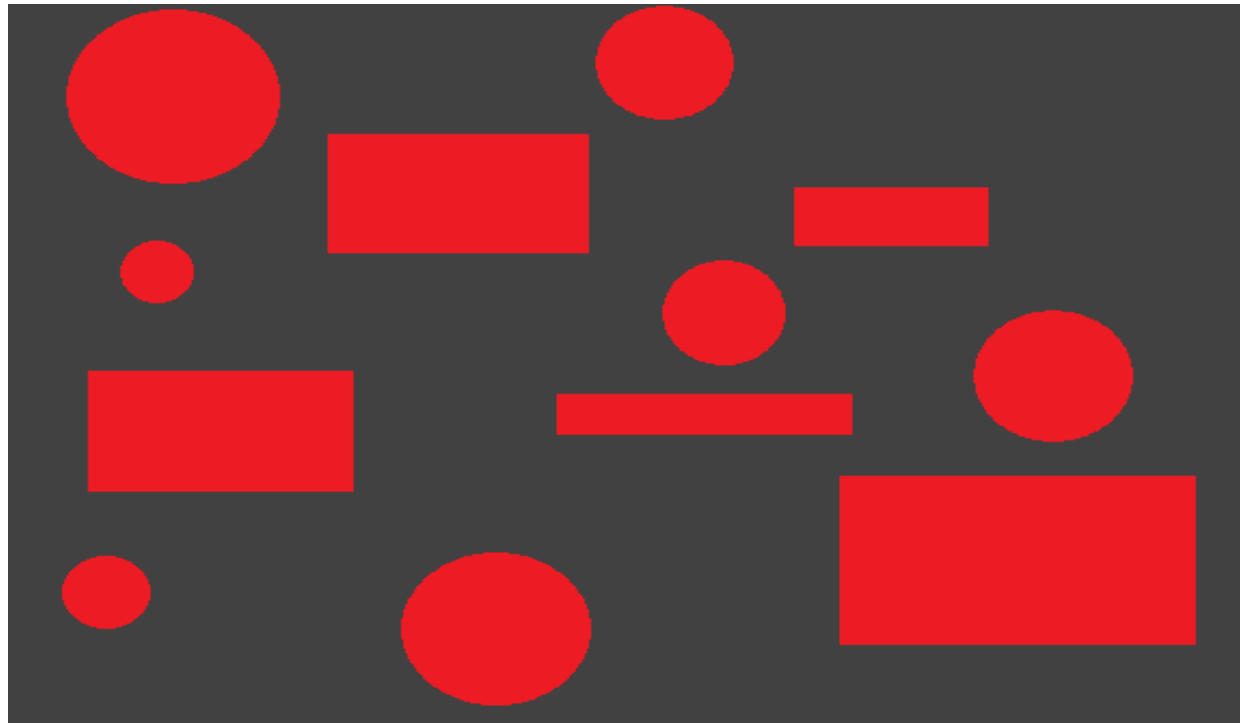


Object Classification and Image Segmentation Using DE

Presented By:
Divyanshu Pabia

Problem: Objects Classification System

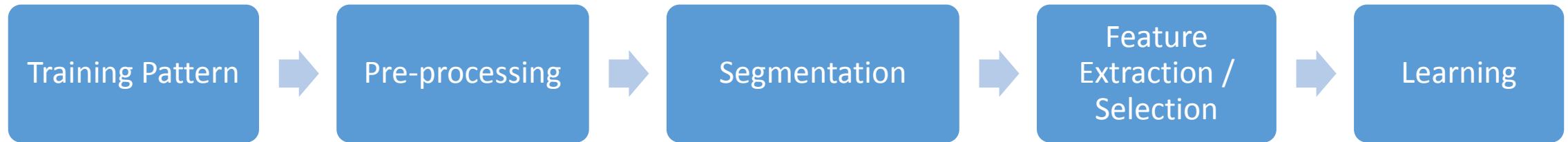
- Automated objects classification into two classes using supervised method
- Classes
 - Circle
 - Rectangle



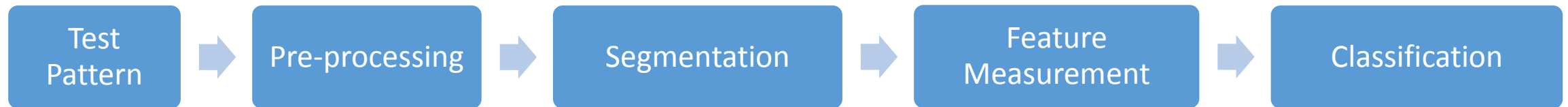
Input Image

Automated object classification (Supervised method)

- TRAINING PHASE



- CLASSIFICATION PHASE



Training Phase

- **Image Acquisition**

- To read an image into a matrix

`img = imread('input.bmp');`

- To view an image in MatLab

`imshow(img);`

- Convert the image into grayscale image

`img_gray = rgb2gray(img);`

- To view an image in MatLab

`imshow(img_gray);`

- To find the size of an image

`[M, N] = size(img_gray);`

- To save the image

`imwrite(img_gray, 'output.bmp');`

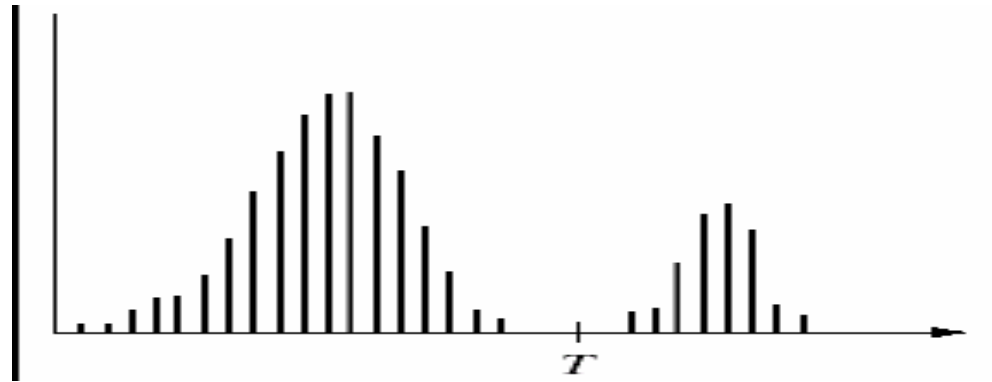
Training Phase

- **Image Segmentation**

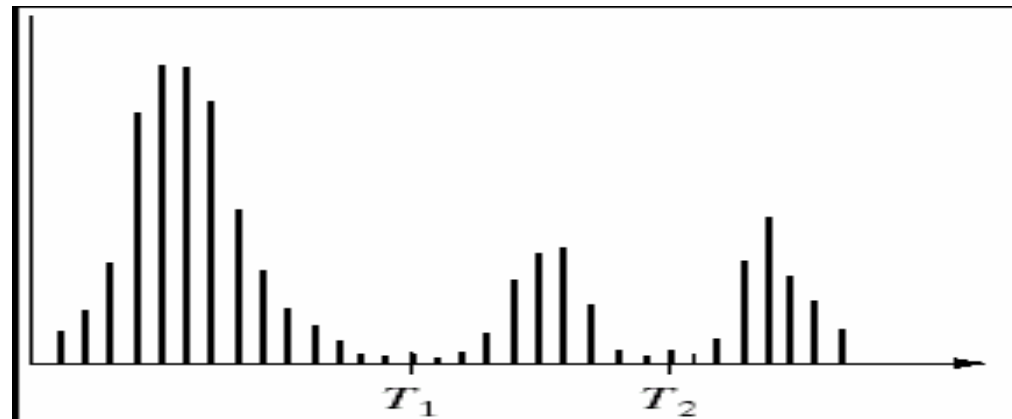
- Segmentation is to subdivide an image into its constituent regions or objects
- Segmentation algorithms generally are based on one of 2 basis properties of intensity values
 - **Discontinuity**: to partition an image based on abrupt changes in intensity (such as edges)
 - **Similarity**: to partition an image into regions that are similar according to a set of predefined criteria

Image Segmentation: Thresholding

- **Single Thresholding:** image with a background and an object



- **Multilevel Thresholding:** image with a background and two or more objects



Problem Formulation

- To classify the pixels of the image into n classes $\{D_1, D_2, \dots, D_n\}$, $n - 1$ thresholds, $(t_1, t_2, \dots, t_{n-1})$, are to be generated

$$f(x, y) = \begin{cases} 0, & f(x, y) \leq t_1 \\ \frac{t_1 + t_2}{2}, & t_1 < f(x, y) \leq t_2 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ \frac{t_{n-2} + t_{n-1}}{2}, & t_{n-2} < f(x, y) \leq t_{n-1} \\ L - 1, & f(x, y) > t_{n-1} \end{cases}$$

- The probability of occurrence of class D_j for each plane is

$$w_j = \sum_{i=t_{j-1}+1}^{t_j} p_i.$$

Problem Formulation

- The inter-class variance can be generally defined as

$$\sigma^2 = \sum_{j=1}^n w_j (\mu_j - \mu)^2.$$

- The objective function for the multilevel thresholding is to maximize the following fitness function

$$\phi = \max_{1 < t_1 < \dots < t_{n-1} < L} \{ \sigma^2(t) \}.$$

TRAINING PHASE

- **Image Segmentation**
 - DE-based Segmentation

Training phase

- **DE-based Segmentation**

- Read Image

```
img=imread('input.bmp');
```

- Show Image

```
imshow(img);
```

- Convert into grayscale

```
img_gray=rgb2gray(img);
```

- Show image

```
imshow(img_gray);
```

- Calculate Threshold

```
[T,variance1,no_of_evaluation] = run1(img_gray);
```

- Convert into binary image

```
img_seg = im2bw(img_gray,T(1)/255);
```

- Show the Segmented Image

```
imshow(img_seg);
```

Feature extraction

- Find the Connected Components

```
cc = bwconncomp(img_seg);
```

- Calculate the Properties

```
stats = regionprops(cc, 'Area','ConvexArea', 'Eccentricity', 'EulerNumber',  
'MajorAxisLength', 'MinorAxisLength', 'Perimeter', 'Solidity', 'Orientation');
```

- Prepare the Feature Vector

```
F=zeros(cc.NumObjects,9);  
for i=1:cc.NumObjects  
    F(i,1)=stats(i,1).Area;  
    F(i,2)=stats(i,1).ConvexArea;  
    F(i,3)=stats(i,1).Eccentricity;  
    F(i,4)=stats(i,1).EulerNumber;  
    F(i,5)=stats(i,1).MajorAxisLength;  
    F(i,6)=stats(i,1).MinorAxisLength;  
    F(i,7)=stats(i,1).Perimeter;  
    F(i,8)=stats(i,1).Solidity;  
    F(i,9)=stats(i,1).Orientation;  
end
```

Recognition using svm

- **Phase 1: Training**

- Import the Training Data if Required

```
load training_data  
xdata = data(:,1:9);  
group = data(:,10:10);
```

- Train the Model

```
svmStruct = svmtrain(xdata,group);
```

- **Phase 2: Testing**

```
output = svmclassify(svmStruct,F);
```