ABSTRACT

Recognizing Object is the process of finding instances of real-world objects. Object recognition enables innovative systems like self-driving cars, image retrieval, and autonomous robotics. In this paper we explore how MATLAB addresses the most common challenges encountered while developing object recognition systems. This paper will cover new capabilities for recognizing of objects and image processing.

The Linde-Buzo-Gray (LBG) algorithm is the most commonly used codebook design algorithm due to the fact that it was the earliest proposed method and consistently outperforms other methods in a variety of applications. It is an iterative technique which repeatedly moves codewords to cluster centroids in an effort to find a codebook which will display the lowest error when encoding the training data (i.e. a modified version of the AT-Means clustering algorithm).

There are many impediments in achieving high recognition rate. Some of them are change in expression, illumination, rotation and scaling. To deal with these problems some local features have been introduced by researchers. Local features such as corners and blobs are mostly used for object recognition

Contents

List of Figures

Chapter 1

INTRODUCTION

Object recognition is a process for identifying a specific object in a digital image or video. Object recognition algorithms rely on matching, learning, or pattern recognition algorithms using appearance-based or feature-based techniques.Object recognition is useful in applications such as video stabilization, advanced driver assistance systems (ADAS), and disease identification in bio-imaging. Common techniques include deep learning based approaches such as convolutional neural networks, and feature-based approaches using edges, gradients, histogram of oriented gradients (HOG), Haar wavelets, and linear binary patterns.You can recognize objects using a variety of models, including: Feature extraction and machine learning models, Deep learning models such as CNNs, Bag-of-words models with features such as SURF and MSER, Gradient-based and derivative-based matching approaches, The Viola-Jones algorithm, which can be used to recognize a variety of objects, including faces and upper bodies, Template matching, Image segmentation and blob analysis.

1.1 Aim of the Project

The target for this project consists of image processing of a digital image, i.e removing the noise and any kind of irregularities present in an image using the digital computer. The noise or irregularity may creep into the image either during its formation or during transformation . Image Processing systems are becoming popular due to easy availability of powerful personnel computers, large size memory devices, graphics software etc
In general there are five major steps performed in Object Recognition as :

1. Image Input
2. Pre-Processing
3. Codebook Generation Using Vector Quantization
4. Feature Extraction
5. Image Matching Using SVM
6. Precision And Recall

1.2 Organization of Report

In Chapter 2 survey on the existing system and proposed system is discussed for Obejct Recognition . The chapter 3 deals with the steps and processes of Object Recognition. In chapter 4 results and discussions are provided. In chapter 5 conclusion and future scope of project are given.

This Chapter discussed the aim of the project, Development of Efficient file integrity security system. The chapter also highlighted the problem definition of the project along with the solutions to the functional problems in the existing system. The next Chapter will provide description on literature survey and requirement analysis for the project.

Chapter 2

LITERATURE SURVEY

Object Recognition method using top down recognition and bottom up image segmentation. Object classification techniques using machine learning model. It represents as in depth experimental study on pedestrian classification multiple feature classifier combinations are examined with respect to their performance and efficiency. Object recognition in images by components: The system is structured with four distinct example-based detectors that are trained to separately find the four components of the human body: the head, legs, left arm, and right arm. They have ensured that these components are present in the proper geometric configuration, a second example-based classifier combines the results of the component detectors to classify a pattern as either a person or a non-person .This type of hierarchical architecture, in which learning occurs at multiple stages, an Adaptive Combination of Classifiers (ACC). They have presented results that show that this system performs significantly better than a similar full-body person detector. Object Detection Using Image Processing: Here work is done in Python-OpenCV and can be performed they have prefered Python because we can include it in OpenCV programs and the execution time in Python is lesser and simple. In here this project reports by pointing it use in surveillance and obstacle detection process. Further this program can be used to control the cameras in a UAV and navigate through obstacles effectively.

Techniques for Object Recognition in Images and Multi-Object Detection: Here in this paper, the have discussed various object detection techniques. The template matching technique requires large database of image templates for correct object recognition. Object recognition also find their application in fields such as biometric recognition, medical analysis, surveillance, etc. A method for multiple object detection is also presented.

Segmentation and object recognition using edge detection techniques: It focuses mainly on the Image segmentation using edge operators. The interaction between image segmentation and object recognition in the framework of the Sobel, Prewitt, Roberts, Canny, LoG, Expectation-Maximization (EM) algorithm, OSTU Algorithm and Genetic Algorithm are studied. MATLAB 7.9. is used for experimentation image. Expectation-Maximization algorithm and OTSU algorithm exhibited stable segmentation effect.

An automatic algorithm for object recognition and detection based on ASIFT keypoints: It presents an object recognition and detection algorithm. Their targets are achieved by combining ASIFT and a region merging segmentation algorithm based on a similarity measure. The merging process is started by using keypoints and presented similarity measure (Euclidean distance). The regions will be merged based on the merging role Here final result is that the more keypoints are obtained, and the more accurate they are, the results will be better and more acceptable.

In the previous chapter we have given the aim of the project for ameliorating the existing system with their respective processes. In this chapter literature survey on the existing file integrity security system is highlighted in reference to the performance and approach of the current system.

2.1 Existing System

The Linde-Buzo-Gray (LBG) algorithm is the most commonly used codebook design algorithm due to the fact that it was the earliest proposed method and consistently outperforms other methods in a variety of applications. It is an iterative technique which repeatedly moves codewords to cluster centroids in an effort to find a codebook which will display the lowest error when encoding the training data (i.e. a modified version of the AT-Means clustering algorithm).

There are many impediments in achieving high recognition rate. Some of them are change in expression, illumination, rotation and scaling. To deal with these problems some local features have been introduced by researchers. Local features such as corners and blobs are mostly used for object recognition.

PROBLEMS IN EXISTING SYSTEM:

- Poor memory utilization.
- Reduced efficiency.
- Degraded performance.
- Only cluster analysis is used with vector quantization.

Chapter 3

METHODOLOGY

In the previous chapter we studied the existing processes and algorithms used in image processing. Object detection is an important, yet challenging vision task. It is a critical part in many applications such as image search, image auto-annotation and scene understanding; however it is still an open problem due to the complexity of object classes and images.

3.1 Proposed System

To overcome the drawback of existing system we are trying to develop a process from which image processing is done with proper approximation. Getting accuracy in retrieving and recognizing image is the target we are being trying to achieve.

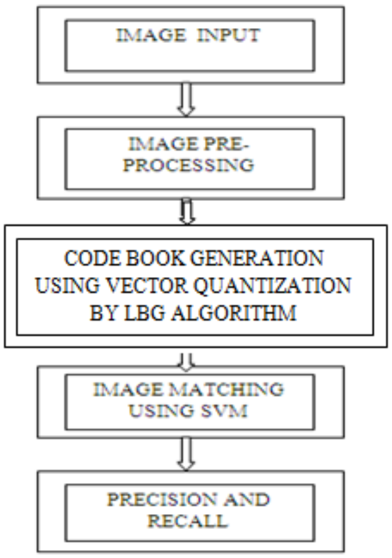


Fig 3.1: Processing For Object Recognition

3.1.1 Image Input

Getting a input from a user and Retrieving this image for recognizing the object.

3.1.2 Image Pre-processing

An input image is pre-processed to normalize contrast and brightness effects. A very common preprocessing step is to subtract the mean of image intensities and divide by the standard deviation. Sometimes, gamma correction produces slightly bet-ter results. While dealing with color images, a color space transformation (e.g. RGB to LAB color space) may help get better results.

Pre-processing is a common name for operations with images at the lowest level of abstraction both input and output are intensity images. These iconic images are of the same kind as the original data captured by the sensor, with an intensity im-age usually represented by a matrix of image function values (brightnesses).

The aim of pre-processing is an improvement of the image data that suppresses unwilling distortions or enhances some image features important for further processing, although geometric transformations of images (e.g. rotation, scaling, translation) are classified among pre-processing methods here since similar techniques are used. Often an input image is pre-processed to normalize contrast and brightness ef-fects.

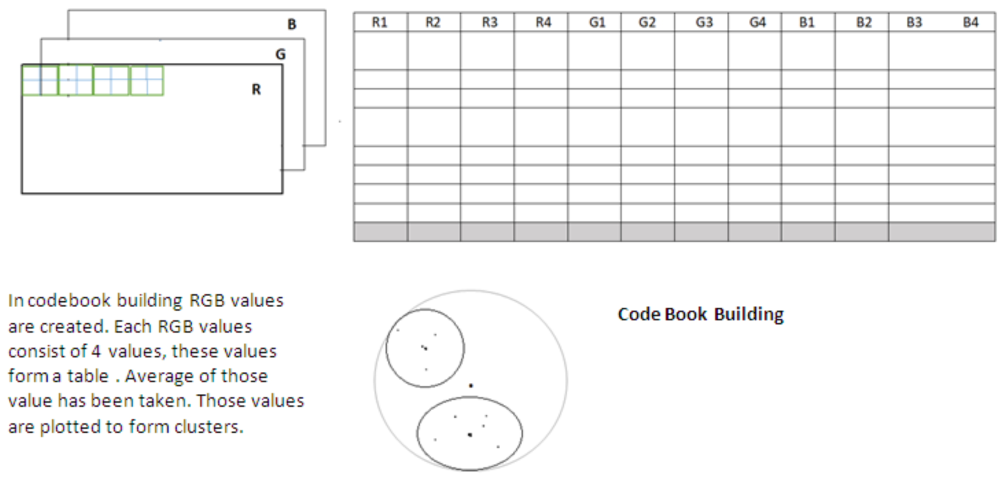
A very common preprocessing step is to subtract the mean of image intensities and divide by the standard deviation. Sometimes, gamma correction produces slightly better results. While dealing with color images, a color space transformation (e.g. RGB to LAB color space) may help get better results.

3.1.3 Codebook Generation Using Vector Quantization

In Vector Quantization by grouping input sequences together and encoding them as a single block, Here are obtained efficient lossy as well as lossless compression algorithms. Some of the quantization techniques that operate on blocks of data. These blocks are looked as vectors. This kind of quantization technique is called vector quantization

In vector quantization we need to build a representative set for the input sequences. If there is an input sequence, we can express it as one of the elements in the representative set. In vector quantization, we first group the input into blocks or vectors. All the operations in vector quantization will be applied to whole vectors. At both the encoder and decoder sides, there is a set of vectors called the codebook.

The vector in the codebook is called the code vector or code word. Normally, the size of the code vector is the same as the input vector. There is a search engine in the encoder to find the code vector that can best match the input vector. The input vector is compared with each code vector in the codebook. The best match code vector is the quantized value of that input vector.



In codebook building RGB values are created. Each RGB values consist of 4 values, these values form a table . Average of those value has been taken. Those values are plotted to form clusters.

Figure 3.1: Code Book Generation Using Vector Quantization

3.1.4 Feature Extraction

Feature extraction starts from an initial set of measured data and builds derived val-ues (features) intended to be informative and non-redundant, facilitating the subse-quent learning and generalization steps, and in some cases leading to better human interpretations. Feature extraction is related to dimensionality reduction.

When the input data to an algorithm is too large to be processed and it is suspected to be redundant (e.g. the same measurement in both feet and meters, or the repetitive-ness of images presented as pixels), then it can be transformed into a reduced set of features (also named a feature vector). Determining a subset of the initial features is called feature selection. The selected features are expected to contain the relevant information from the input data, so that the desired task can be performed by using this reduced representation instead of the complete initial data.

3.1.5 Matching using SVM

Matching an image from the database withheld by us. Using the code book built by us to match the image so as to extract the proper co-ordinates and plot them accordingly. Matching the image by accuracy so as to get a accurate image with good quality and no noise present that affects the image quality.

SVMs are helpful in text and hypertext categorization as their application can significantly reduce the need for labeled training instances in both the standard inductive and transductive settings.

Classification of images can also be performed using SVMs. Experimental results show that SVMs achieve significantly higher search accuracy than traditional query refinement schemes after just three to four rounds of relevance feedback. This is also true of image recognition systems, including those using a modified version SVM.

The full SVM classifier (SVM Model) is more than four times the compact SVM classifier (Compact SVM Model). Full SVM classifiers (i.e., Classification SVM classifiers) hold the training data. For efficiency, you might not want to predict new labels using a large classifier. Train an SVM classifier. It is good practice to standardize the predictors and specify the order of the classes.

The SVM algorithm uses structural risk minimization to find the hyper plane that optimally separates two classes of objects. This is equivalent to minimizing a bound on generalization error. The optimal hyper plane is computed as a decision surface of the form

We use support vector machines (SVM) to classify the data vectors resulting from the Haar wavelet representation of the components. SVMs were proposed by Vapnik and have yielded excellent results in various data classification tasks, including people recognition and text classification .Traditional training techniques for classifiers like multi layer perceptrons use empirical risk minimization and lack a solid mathematical justification. The SVM algorithm uses structural risk minimization to find the hyper plane that optimally separates two classes of objects. This is equivalent to minimizing a bound on generalization error. The optimal hyper plane is computed as a decision surface of the form:

$$f(x) = sgn(g(x)),$$

(3.1)

$$g(x) = ((_{(}i = 1)^{n} yiaiK(x, xi) + b)$$

(3.2)

In, K is one of many possible kernel functions, yi,-1,a is the class label of the data point xi, and xini=1 is a subset of the training data set. The xi are called support vectors and are the points from the data set that define the separating hyper plane. Finally, the coefficients aiand b are determined by solving a large-scale quadratic programming problem. One of the appealing characteristic of SVMs is that there are just two tunable parameters, Cpos and Cneg, which are penalty terms for positive and negative pattern misclassification, respectively. The kernel function K that is used in the component classifiers is a quadratic polynomial and is K(x,xi)=(x.xi+1)2. The binary class of a data point is the sign of the raw output g(x) of the SVM classifier.

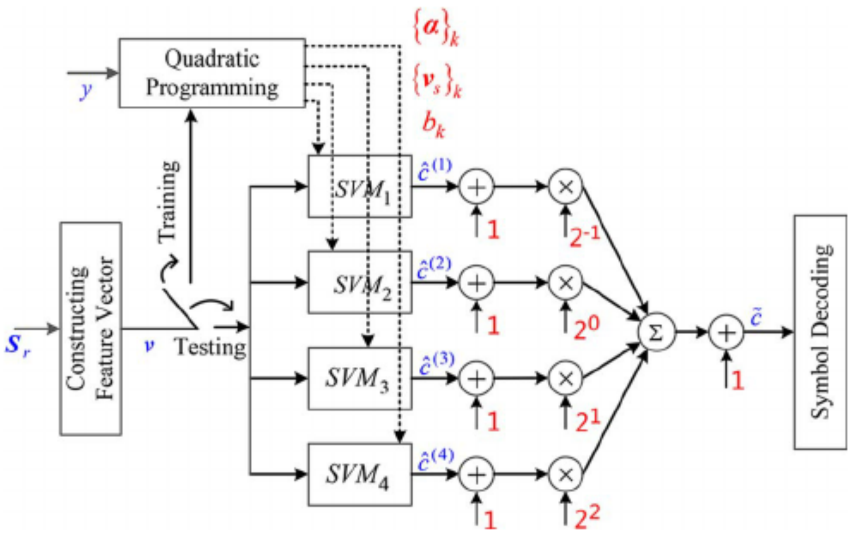


Figure 3.2: Code Book Generation Using Vector Quantization

3.1.6 Performance Measure

True Positive test result is one that detects the condition when the condition is pre-sent.

True Negative test result is one that does not detect the condition when the condition is absent.

False Positive test result is one that detects the condition when the condition is ab-sent.

False Negative test result is the one that does not detect the condition when the con-dition is present

Accuracy: For accuracy of a measurement system is the level of proximity measurements of the quantity to the quantitys actual value.

Accuracy = True Positive + True Negative Total Number of Elements

Precision: For information retrieval, precision is the fraction of retrieving actual value.

Precision = True Positive/True Positive+ False Positive

Recall: Recall in value retrieval is the fraction of the documents that are appropriate to the query that are successfully redeemed.

Recall =True Positive/True Positive+ False Negative

Chapter 4

IMPLEMENTATION

4.0.1 Code

```
function [centroid]=Function_EV_Rotation(s,CB);
block_len=12;
mx=size(s);
if mod(mx(1),2) == 0
else mx(1)=mx(1)-1;
end
if mod(mx(2),2) == 0
else mx(2)=mx(2)-1;
end
E=[1 1 1 1 1 1 1 1 1 1 1 1];
vect_count=0;
k=1;
i=1;
for t=1:mx(1)/2;
    j=1;
    for p=1:mx(2)/2;
        v(k,1:3)=s(i,j,1:3);
        v(k,4:6)=s(i,j+1,1:3);
        v(k,7:9)=s(i+1,j,1:3);
        v(k,10:12)=s(i+1,j+1,1:3);
          j=j+2;
        k=k+1;
              vect_count = vect_count + 1;
    end
    i=i+2;
end
v1(1,1:vect_count)=[1:vect_count];
for j=1:block_len;
    sum1=0;
    for k=1:vect_count;
        sum1 = sum1 + double(v(k,j));
    end
    centroid(1,j)=sum1/vect_count;
end

centroid(1,1:block_len);
c_pt=[k];
d1=0;
for k = 1:c_pt(1); %Calculation of mean square error
    for j=1:block_len;
        d1=d1+(double(v(k,j)) - centroid(1,j))*(double(v(k,j)) - cen-troid(1,j))
            ;
    end
end

mse(1)=d1/(c_pt(1)*block_len);

netmse=mse(1);
cb_size=1;
cb=centroid;

%clear k;
%clear s;
%clear v;
pack;
ff=1;
while cb_size<=CB/2
    pack;
    cu=1;
    c=1;
    vc=1;
```

```

for out=1:cb_size

    c0(c,:)=cb(out,:) + E;
    c0(c+1,:)=cb(out,:) - E;
    c0;
    m=0;
    q=0;
    vect_count1=c_pt(vc);
    for k = 1:vect_count1; %Creation of clusters
        d1=0;
        d2=0;
        for j=1:block_len;
            d1=d1+(double(v(v1(out,k),j)) - c0(c,j))*(double(v(v1(out,k),j))
                - c0(c,j));
            d2=d2+(double(v(v1(out,k),j)) - c0(c+1,j))*(double(v(v1(out,k),j
                )) - c0(c+1,j));
        end

        if d1<d2
            m=m+1;
            %for j=1:block_len;
                clu(cu,m)=v1(out,k);
            %end
        else
            q=q+1;
            %for j=1:block_len;
                clu(cu+1,q)=v1(out,k);
            %end
        end
    end

    clu_pt(c)=[m];
    clu_pt(c+1)=[q];
    pack;
    for j=1:block_len; %Calculates centroid for the cluster 1
        sum1=0;
```

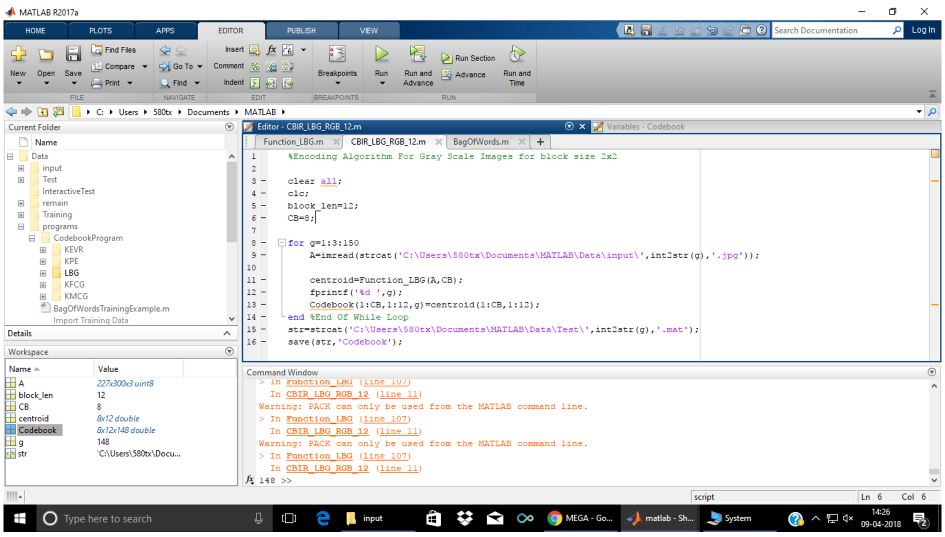
```
            for k=1:clu_pt(c);
                sum1 = sum1 + double(v(clu(cu,k),j));
            end
            if clu_pt(c)== 0
               centroid(cu,j)=−1;
            else
                centroid(cu,j)=sum1/clu_pt(c);
            end
        end

        for j=1:block_len; %Calculates centroid for the cluster 2
            sum1=0;
            for k=1:clu_pt(c+1);
                sum1 = sum1 + double(v(clu(cu+1,k),j));
            end
            if clu_pt(c+1)== 0
                centroid(cu+1,j)=0;
            else
                centroid(cu+1,j)=sum1/clu_pt(c+1);
            end
        end
    cu=cu+2;
    c=c+2;
    vc=vc+1;
    end %End Of Outer For Loop
    cb=centroid;
    c_pt=clu_pt;
    v1=clu;
    netmse=0;
    count=0;

    ff=ff+1;

    cb_size=cb_size*2;
    cb_size;
end %End Of While Loop
```

Listing 4.1: CENTROID GENERATION

```

Encoding Algorithm For Gray Scale Images for block size 2x2

clear all;
clc;
block_len=12;
CB=8;

for g=1:1:150
    A=imread(strcat('C:\Users\vinayak-pc\Documents\matlab\Data\input\',int2str(g
        ),'.jpg'));
    centroid=Function_LBG(A,CB);
    fprintf('%d ',g);
    Codebook(1:CB,1:12,g)=centroid(1:CB,1:12);
end %End Of While Loop
str=strcat('C:\Users\vinayak-pc\Documents\matlab\Data\',int2str(g),'.mat');
save(str,'g');
```

Listing 4.2: CODEBOOK GENERATION

4.0.2 Screenshots

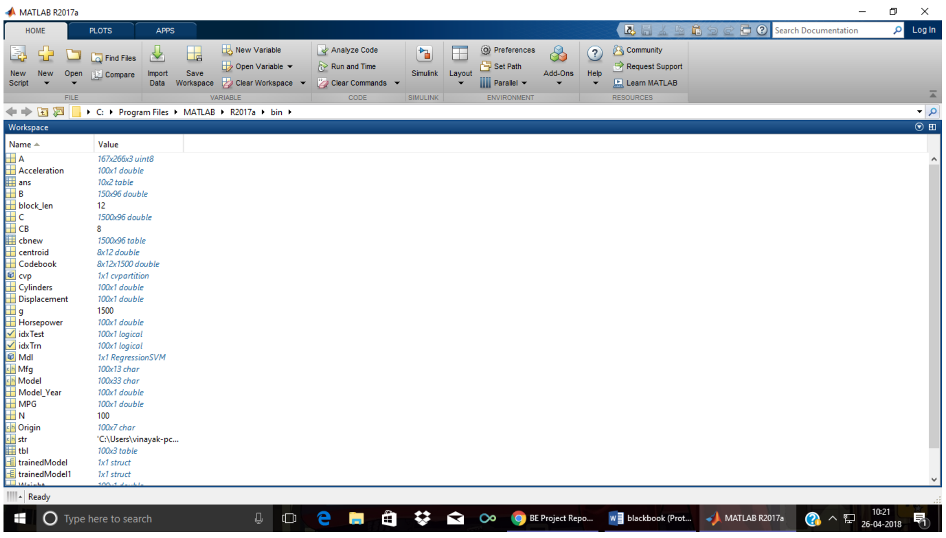


```
%Encoding Algorithm For Gray Scale Images for block size 2x2

clear all;
clc;
block_len=12;
CB=8;

for g=1:3:150
    A=imread(strcat('C:\Users\580tx\Documents\MATLAB\Data\input\',int2str(g),'.jpg'));

    centroid=Function_LBG(A,CB);
    fprintf('%d ',g);
    Codebook(1:CB,1:12,g)=centroid(1:CB,1:12);
end %End Of While Loop
str=strcat('C:\Users\580tx\Documents\MATLAB\Data\Test\',int2str(g),'.mat');
save(str,'Codebook');
```

```
> In Function_LBG (line 107)
  In CBIR_LBG_RGB_12 (line 11)
Warning: PACK can only be used from the MATLAB command line.
> In Function_LBG (line 107)
  In CBIR_LBG_RGB_12 (line 11)
Warning: PACK can only be used from the MATLAB command line.
> In Function_LBG (line 107)
  In CBIR_LBG_RGB_12 (line 11)
148 >>
```

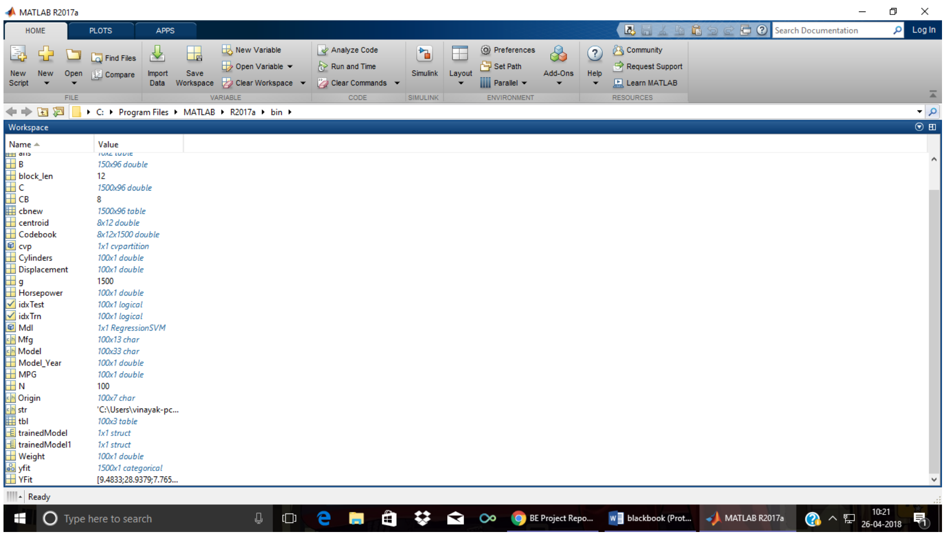
Name	Value
A	227x300x3 uint8
block_len	12
CB	8
centroid	8x12 double
Codebook	8x12x148 double
g	148
str	'C:\Users\580tx\Docu...

Figure 4.1: EXECUTION OF CODES



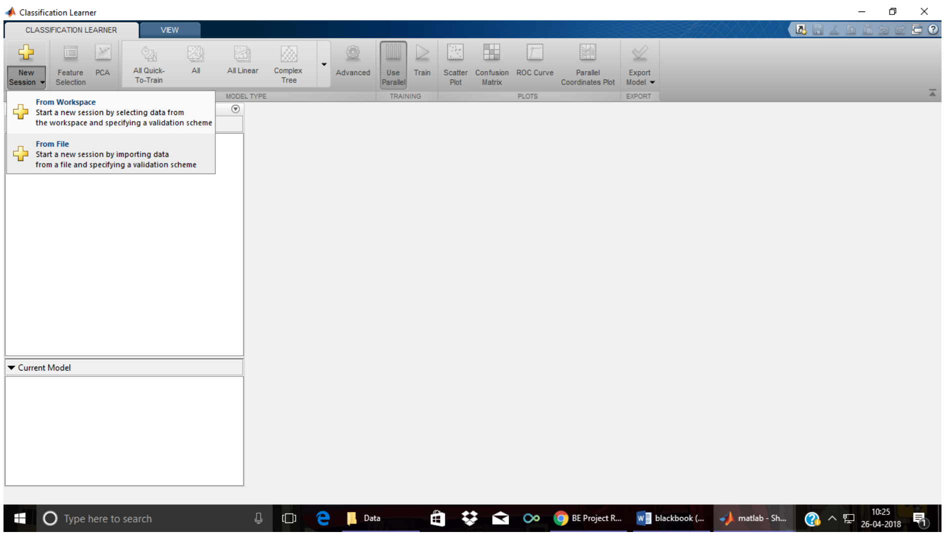
Name	Value
A	167x266x3 uint8
Acceleration	100x1 double
ans	10x2 table
B	150x96 double
block_len	12
C	1500x96 double
CB	8
cbnew	1500x96 table
centroid	8x12 double
Codebook	8x12x1500 double
cvp	1x1 cvpartition
Cylinders	100x1 double
Displacement	100x1 double
g	1500
Horsepower	100x1 double
idxTest	100x1 logical
idxTrn	100x1 logical
Mdl	1x1 RegressionSVM
Mfg	100x13 char
Model	100x33 char
Model_Year	100x1 double
MPG	100x1 double
N	100
Origin	100x7 char
str	'C:\Users\vinayak-pc...
tbl	100x3 table
trainedModel	1x1 struct
trainedModel1	1x1 struct

Figure 4.2: WORKSPACE



Name	Value
B	150x96 double
block_len	12
C	1500x96 double
CB	8
cbnew	1500x96 table
centroid	8x12 double
Codebook	8x12x1500 double
cvp	1x1 cvpartition
Cylinders	100x1 double
Displacement	100x1 double
g	1500
Horsepower	100x1 double
idxTest	100x1 logical
idxTrn	100x1 logical
Mdl	1x1 RegressionSVM
Mfg	100x13 char
Model	100x33 char
Model_Year	100x1 double
MPG	100x1 double
N	100
Origin	100x7 char
str	'C:\Users\vinayak-pc...
tbl	100x3 table
trainedModel	1x1 struct
trainedModel1	1x1 struct
Weight	100x1 double
yfit	1500x1 categorical
YFit	[9.4833;28.9379;7.765...

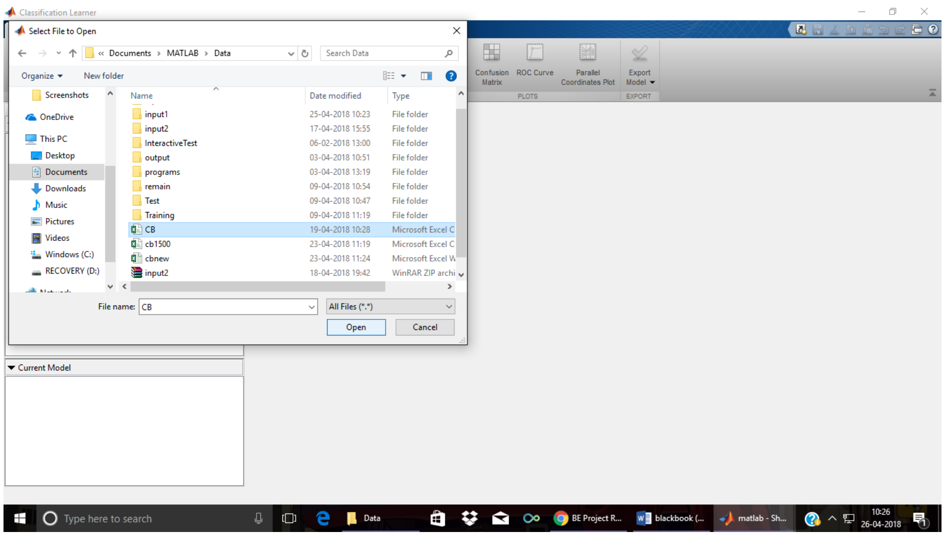
Figure 4.3: WORKSPACE



From Workspace
Start a new session by selecting data from the workspace and specifying a validation scheme

From File
Start a new session by importing data from a file and specifying a validation scheme

Figure 4.4: CLASSIFICATION LEARNER



Name	Date modified	Type
input1	25-04-2018 10:23	File folder
input2	17-04-2018 15:55	File folder
InteractiveTest	06-02-2018 13:00	File folder
output	03-04-2018 10:51	File folder
programs	03-04-2018 13:19	File folder
remain	09-04-2018 10:54	File folder
Test	09-04-2018 10:47	File folder
Training	09-04-2018 11:19	File folder
CB	19-04-2018 10:28	Microsoft Excel C
cb1500	23-04-2018 11:19	Microsoft Excel C
cbnew	23-04-2018 11:24	Microsoft Excel W
input2	18-04-2018 19:42	WinRAR ZIP archi

File name: CB
All Files (*.*)

Figure 4.5: CLASSIFICATION USING CODEBOOK