Experiment:1.1

Aim: Write a program to convert RGB image to GRAYSCALE and implement various feature extraction techniques like ORB and HOG for image classification.

Software Required: Any Python IDE (e.g.: PyCharm, Jupyter Notebook, GoogleCollab)

Technique used:

Various feature extraction methods are often employed in picture classification problems. These methods seek to extract pertinent data from photos and turn it into meaningful representations that machine learning algorithms may use to classify images. Feature extraction methods include:

1. Histogram of Oriented Gradients (HOG): The Histogram of Oriented Gradients (HOG) is a popular feature descriptor technique in computer vision and image processing. It analyses the distribution of edge orientations within an object to describe its shape and appearance. The HOG method involves computing the gradient magnitude and orientation for each pixel in an image and then dividing the image into small cells.
2. ORB (Oriented FAST and Rotated BRIEF): ORB is a feature extraction and description technique commonly used in computer vision tasks like image matching and object recognition. ORB combines the strengths of two key components: FAST (Features from Accelerated Segment Test) for keypoint detection and BRIEF (Binary Robust Independent Elementary Features) for feature description.

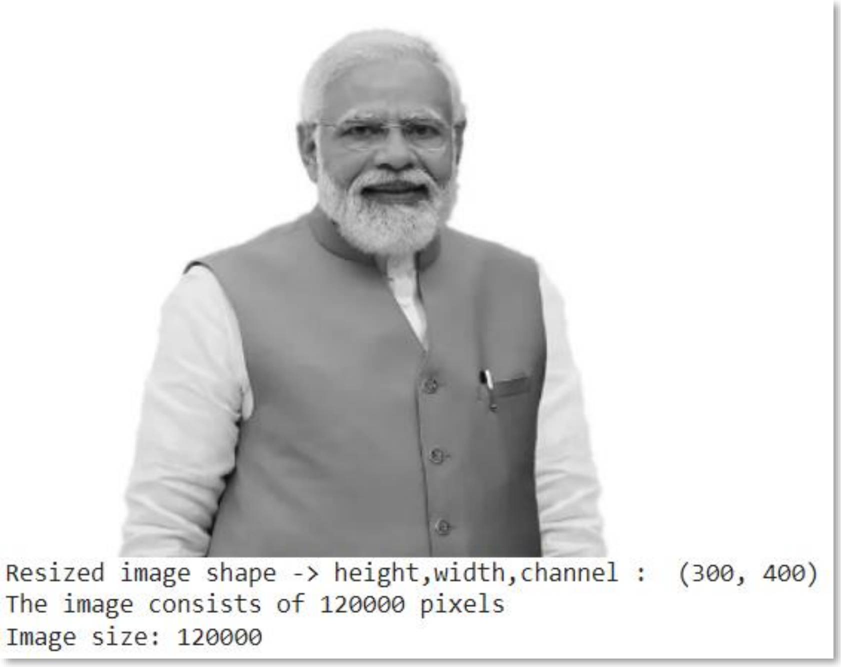
Steps:

1. Import Necessary libraries.
2. Load the image.
3. Resize the image.
4. Convert image from BRG to GRAYSCALE.
5. Printing resized image shape and its size.
6. Calculating pixels and displaying image.

Implementation:

```
# importing libraries
from google.colab.patches import cv2_imshow
import cv2
import matplotlib.pyplot as plt
# reading images from folder using imread
image = cv2.imread('/content/drive/MyDrive/Colab Notebooks/Images/Modi-Ji.webp')
# converting to grayscale
resized_image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
# Resize the image
resized_image = cv2.resize(resized_image, (400, 300))
# Display the image using cv2_imshow
cv2_imshow(resized_image)
# resized_image.shape has (height, width, channels)
print("Resized image shape -> height,width,channel : ",resized_image.shape)
# calculating and printing the total number of pixels
print("The image consists of %i pixels" % (resized_image.shape[0] *
resized_image.shape[1]))
print("Image size:", resized_image.size)
```

Output Screenshots:



```
Resized image shape -> height,width,channel :  (300, 400)
The image consists of 120000 pixels
Image size: 120000
```

HOG Features using skimage Library:

Steps:

1. Import Necessary libraries.
2. Load/Read the image.
3. Resizing the image and printing resized image shape.
4. Creating and visualizing HOG Features.
5. Printing dimension of HOG feature vector.

Implementation:

```
# Importing Libraries
from skimage.io import imread
from skimage.transform import resize
from skimage.feature import hog
import matplotlib.pyplot as plt
```

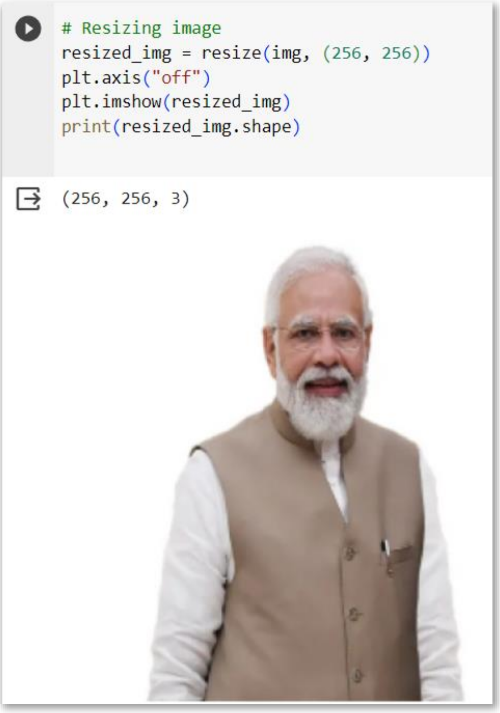
```
# Reading the image
img = imread('/content/drive/MyDrive/Colab Notebooks/Images/Modi-Ji.webp')
plt.axis("off")
plt.imshow(img)
print(img.shape)
```

```
# Resizing image
resized_img = resize(img, (256, 256))
plt.imshow(resized_img)
print(resized_img.shape)
```

```
# Creating and visualizing HOG Features
fd, hog_image = hog(resized_img, orientations=9, pixels_per_cell=(16, 16),
                  cells_per_block=(2, 2), visualize=True, multichannel=True)
plt.imshow(hog_image, cmap="gray")
plt.show()
```

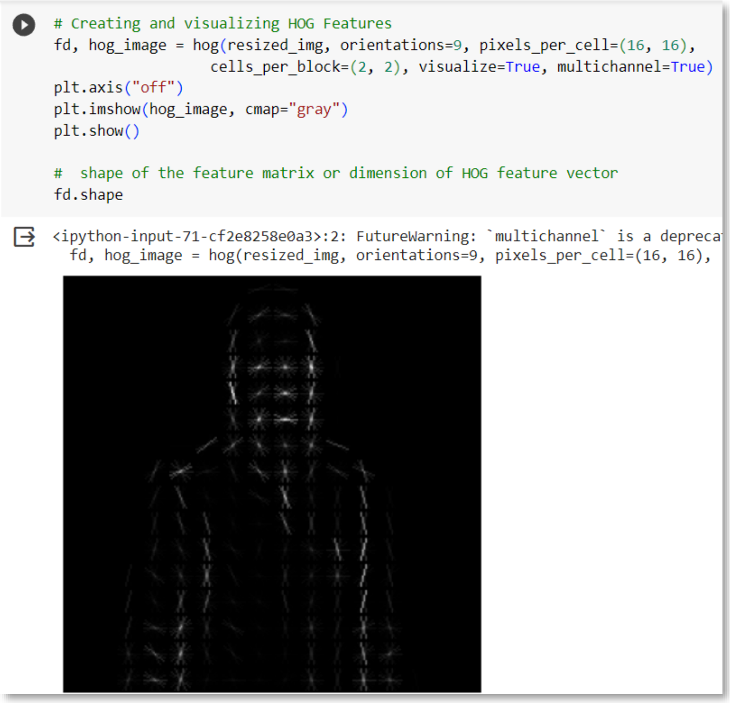
```
#  shape of the feature matrix or dimension of HOG feature vector
fd.shape
```

Output Screenshots:



```
# Resizing image
resized_img = resize(img, (256, 256))
plt.axis("off")
plt.imshow(resized_img)
print(resized_img.shape)
```

```
(256, 256, 3)
```



```
# Creating and visualizing HOG Features
fd, hog_image = hog(resized_img, orientations=9, pixels_per_cell=(16, 16),
                    cells_per_block=(2, 2), visualize=True, multichannel=True)
plt.axis("off")
plt.imshow(hog_image, cmap="gray")
plt.show()

#  shape of the feature matrix or dimension of HOG feature vector
fd.shape
```

```
<ipython-input-71-cf2e8258e0a3>:2: FutureWarning: `multichannel` is a depreca
  fd, hog_image = hog(resized_img, orientations=9, pixels_per_cell=(16, 16),
```

ORB Features:

Steps:

1. Import Necessary libraries.
2. Load/Read the image.
3. converting image to grayscale.
4. Create an ORB feature detector.
5. Detect ORB keypoints and compute descriptors.
6. Drawing the keypoints on the image to visualize them.
7. Display the image with keypoints.

Implementation:

```
# ORB implementation
#import libraries
import cv2
from google.colab.patches import cv2_imshow
```

```
# reading image
image = cv2.imread('/content/drive/MyDrive/Colab Notebooks/Images/Modi-Ji.webp')
# converting image to grayscale
image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Create an ORB feature detector
orb = cv2.ORB_create()
# Detect ORB keypoints and compute descriptors
keypoints, descriptors = orb.detectAndCompute(image, None)

# keypoints: A list of detected keypoints in the image
# descriptors: The descriptors corresponding to the keypoints
# drawing the keypoints on the image to visualize them
image_with_keypoints = cv2.drawKeypoints(image, keypoints, image)

# Display the image with keypoints
cv2_imshow(image_with_keypoints)
```

Output Screenshots: