Project-5

CIS 666 Artificial Intelligence

Face Recognition using Local Binary Pattern

Fathima Syeda

Student ID: 2790024

Abstract

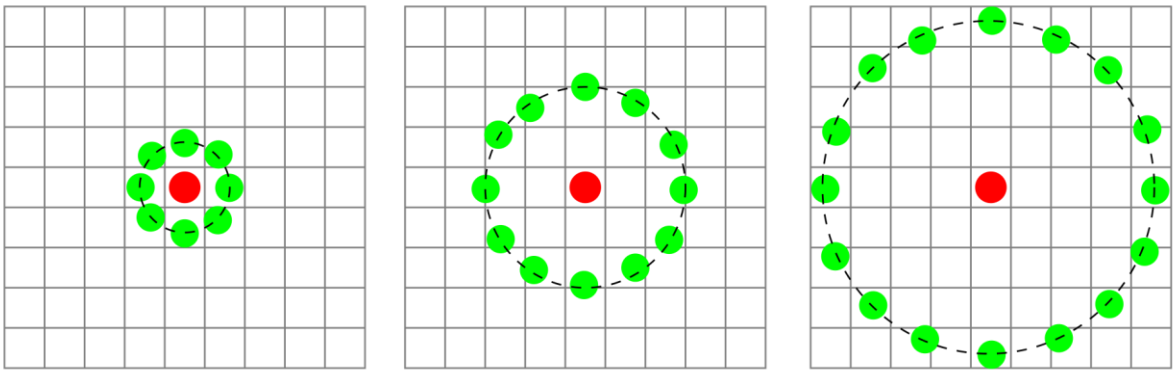
In this project we use the AR face dataset for face recognition using Local Binary Pattern for texture extraction .The AR dataset contains over 4000 colour face images of 126 subjects, including frontal views of faces with different facial expressions, illumination conditions and occlusions. For this project, we will only use a subset of the dataset with large variations in both illumination and expression, which corresponds to 50 male subjects and 50 female subjects. For each subject there are two sections, one for training and the other for testing. Each section contains 7 images per subject. We take each image in the dataset and find the LBP for each of the pixels in the image and then find the histogram of the LBP matrix. The histogram of training images is then used for training the SVM classifier with different kernels viz linear, rbf, poly and also the KNN classifier. The accuracy of each of the classifier is found and compared.

Contents

1. Introduction:

Local Binary Pattern is a type of visual descriptor and a strong feature for texture classification.

To get the LBP feature vector we divide the image into either3x3 or 16x16 cells . Now for each of the compare the pixel to each of its 8 neighbours (on its left-top, left-middle, left-bottom, right-top, etc.). Follow the pixels along a circle, i.e. clockwise or counter- clockwise. Find the difference of the centre with its neighbours to get $Z_p = g_p - g_c$. Then if the difference is negative then the value becomes 0 else it becomes 1 i.e. $S(Z_p) = 1$ if $Z_p >= 0$ and $S(Z_p)=0$ if $Z_p <0$. This way we get a binary pattern which is then converted to a decimal value $LBP(x_c, y_c) = {}^{p-1}\sum_{p=0} S(Z_p) * 2^p$. The histogram of the above is computed and is a 256-dimension feature vector.



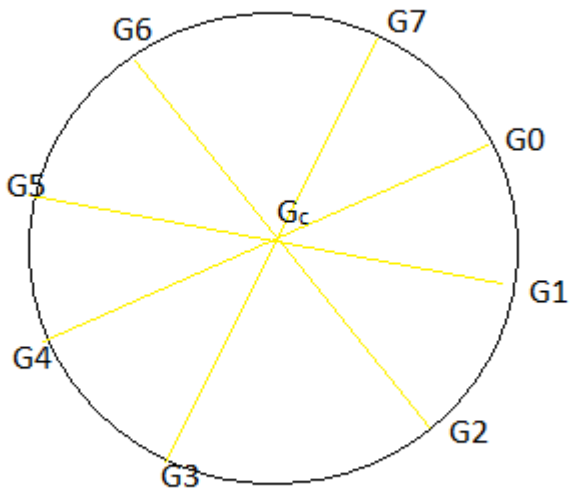
p=8 ,r=1 p=12, r=2 p=16, r=3

The above image shows three neighbourhood examples to get the texture and calculate the LBP.

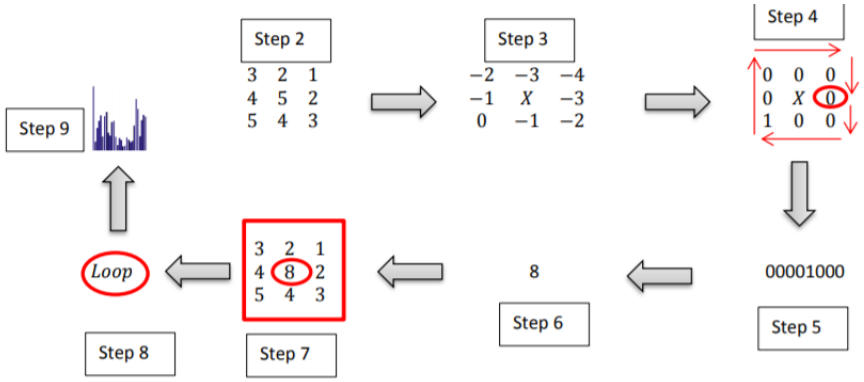
Centre- Symmetric LBP

In Centre symmetric LBP the LBP is calculated by subtracting or comparing the points along the opposite end of the diameter.

$$CS\ LBP = S(g_0 - g_4) * 2^0 + S(g_1 - g_5) * 2^1 + S(g_2 - g_6) * 2^2 + S(g_3 - g_7) * 2^3$$



2. Steps involved in finding Local Binary Pattern



The above image shows the steps involved in finding the LBP for each pixel in the image.

1. Read the input images and we see that each of the images is of dimensions 120x165 resize them to 64x64.
2. Pad the input image with 1 row and 1 column on each side of the image, in the end the image would be 66 × 66.
3. For each pixel of the input image, we need to get its 3×3 neighbours. You can get this by using a sliding window of 3x3 over the image.
4. Subtract the centre pixel of the 3x3 matrix from its 8 neighbours to get the $\mathbf{Z_p = g_p - g_c}$
5. Now to get S($\mathbf{Z_p}$) we sign 1 to the values that are greater than or equal 0 and 0 for the values that are less than 0. i.e. $\mathbf{S(Z_p) = 1 \text{ if } Z_p >= 0 \text{ and } S(Z_p)=0 \text{ if } Z_p < 0}$
6. Concatenate the 8-bit binary code clockwise or counter-clockwise to get the Local Binary Pattern of that image $\mathbf{LBP(x_c, y_c) = {}^{p-1}\sum_{p=0} S(Z_p) * 2^p}$
7. Now we save this LBP of each pixel of the image in a new array . Thus, we get am LBP array of 64x64 for each image.
8. Repeat steps 2 – 7 for all pixels of the input image (if it is size 64 × 64, that means you need to do that 4096 times).
9. Once you finish applying the previous steps for all pixels, calculate the histogram of the LBP image that you get, which means you need to get only one histogram for each image.
10. Once we get the Histogram of the LBP of all the images, we must train it using a SVM classifier with different kernels=linear ,ply, rbf.

3. Program Implementation

The program consists of 4 functions to implement the single layer perceptron algorithm to correctly identify handwritten images into 10 digits from 0-9. The following are the functions:

- sliding_window(image, orientation, step Size, window Size)
- getLBP (Training Images):
- SVM_Classification(TrainingData,TrainLabels,TestData,TestLabels,k='linear')

- KNNCLAssifier(TrainingData,TrainingLabels,TestData,TestingLabels):

3.1 sliding_window(image, orientation, step Size, window Size)

This function returns blocks from the given image gradient and its orientation based on the given window size and decides the overlap of these blocks using the given step size.

3.2 getLBP (Training Images)

This function gives the histogram of the Local Binary Pattern of each of the images in the Training images set using the steps mentioned in the previous section. It makes use of the sliding_window() to get the 3x3 matrices for finding the LBP of each of the pixels in the images.

3.3 SVM_Classification(TrainingData,TrainLabels,TestData,TestLabels,k='linear')

This function takes the training data and the testing data along with their corresponding labels. The training and the testing data are the HOG descriptors of all the images. It also takes in an optional parameter k for the kernel of the SVC classifier. The data is trained and tested using in-built functions. Then the accuracy score is found and printed.

accuracy $= \frac{true\ positive + true\ negative}{true\ positive + true\ negative + false\ positive + false\ negative}$

3.4 KNNCLAssifier(TrainingData,TrainingLabels,TestData,TestingLabels)

This function takes the training data and its corresponding labels along with the testing data and labels. It trains the data on the built-in KNN classifier with 5 neighbouring nodes from sklearn. It then tests the testing data and finally returns the accuracy of the classifier.

4. Obtained Results

The dataset used in this project is the AR face data dataset. It contains over 4000 colour face images of 126 subjects, including frontal views of faces with different facial expressions, illumination conditions and occlusions.

For this project, we will only use a subset of the dataset with large variations in both illumination and expression, which corresponds to 50 male subjects and 50 female subjects. For each subject there are two sections, one for training and the other for testing. Each section contains 7 images per subject.

1) When the training data samples are trained on an SVM classifier with **kernel=linear** the results are as:

```
For SVM classified with kernel=linear
Accuracy Score is :1.0
```

2) When the training data samples are trained on an SVM classifier with **kernel=rbf** the results are as:

```
For SVM classified with kernel=rbf
```

```
Accuracy Score is :0.2985714285714286
```

3) When the training data samples are trained on an SVM classifier with **kernel=polynomial** the results are as:

```
For SVM classified with kernel=poly

Accuracy Score is :0.30857142857142855
```

4) When the training data samples are trained on an **KNN classifier** with 5 neighbouring nodes :

```
For KNN classifier with 5 neighbouring nodes is:
Accuracy Score is :0.6857142857142857
```

```
loading the dataset
Finding LBP  for training data
Finding LBP  for testing data
----------------------------
Training the data with linear SVM
For SVM classified with kernel=linear
Accuracy Score is :1.0
----------------------------
Training the data with rbf SVM
For SVM classified with kernel=rbf
Accuracy Score is :0.2985714285714286
----------------------------
Training the data with polynomial SVM
For SVM classified with kernel=poly
Accuracy Score is :0.30857142857142855
----------------------------
Training the data with KNN classifier
For KNN classifier with 5 neighbouring nodes is:
Accuracy Score is :0.6857142857142857
```

5. Conclusion

- The linear SVM kernel gives the highest accuracy of 100% on the given AR face data
- The KNN classifier gives the next best accuracy after Linear SVM with 68% accuracy
- SVM classifier with kernels rbf, polynomial does not give a high accuracy i.e. on30%
- Linear SVM is the best suited with the best accuracy for face recognition when compared to other classifiers like KNN or ANN.

6. References

- Class Notes
- Online Class Notes