ECE 661 Fall 2018
Homework #7

Ye Shi
shi349@purdue.edu

October 30, 2018

Contents

1 Methodology **2**
1.1 Local Binary Pattern (LBP) Extraction . . . 2
1.2 Nearest Neighbour(NN) Classifier . . . 3
1.3 Performance Measures . . . 3

2 Results **3**
2.1 Examples of LBP Histograms . . . 3
2.2 Performance Measures . . . 5
2.3 Observations . . . 6

3 Source Code **6**
3.1 Main . . . 6
3.2 LBP Extraction . . . 7
3.3 5-NN Classifier . . . 8
3.4 Confusion Matrix . . . 9

1 Methodology

1.1 Local Binary Pattern (LBP) Extraction

In this homework, I demonstrate how to generate LBP Histograms with 8 neighbouring points on a unit circle. The number of points and radius of the circle can be changed due to different applications.

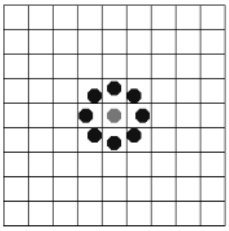


Figure 1: A pixel with 8 neighbouring points on a unit circle.

1. In Fig.1 is a pixel point (gray) with its 8 neighbouring points (black) on a unit circle. The exact positions of the neighbouring points vis-a-vis the pixel under consideration is given by

$$(\Delta u, \Delta v) = \left(R\cos(\frac{2\pi p}{P}), R\sin(\frac{2\pi p}{P}) \right),$$

   where radius $R = 1$, number of neighbouring points $P = 8$ and $p = 0, 1, 2, \ldots, 7$.

2. For the neighbouring points exactly at the central of the pixels, we assign them the pixel value as their value.

   For those neighbouring points not at the central of the pixels, we assign them the value by 2D-linear interpolation given as

$$I(p) = (1 - \Delta k)(1 - \Delta l)I(A) + (1 - \Delta k)\Delta l I(B) + \Delta k(1 - \Delta l)I(C) + \Delta k \Delta l I(D)$$

   where A, B, C, and D stand for 4 corner pixels around p. I is the pixel value and Δk is the distance along x-axis from A to p, Δl is the distance along y-axis from A to p.

3. We start from $p = 0$ to 7, if $I(p) \geq I$, then we assign it 1, otherwise 0. Then we get a 8 bit binary number b from $p = 0$ to 7, for example $b =$'10101010'.

4. In order to make this pattern invariant to in-plane rotations, we need to find the minimum integer value of b. We can simple right or left shift it 1-bit each time and find the minimum integer value in binary representation m, which we also called minimum value pattern, for example from the former step, $m =$'01010101'.

5. After we have acquired m, we encode these texture patterns based on the number of runs of 1 and 0 as follows:

   - If there are exactly 2 runs, the pattern is encoded by the number of 1s in the second run;
   - If the pattern has all 0s, then it is encoded by 0;
   - If the pattern has all 1s, then it is encoded by P, we have $P = 8$ here;
   - If the pattern has more than two runs, we encode it as $P + 1$, we have $P + 1 = 9$ here.

   For example from the former step, the encoded pattern is 9.

6. Once we repeat from Step 1 to 5 for every pixel, we can generate a histogram of the encoded pattern from 0 to 9. This is the feature we will use to classify the images.

1.2 Nearest Neighbour(NN) Classifier

In this homework, I pick 5 nearest neighbour to classify the test images. The fundamental of the classifier is to find the euclidean distance in LBP feature between the test image and all training images. Then we pick out the training image labels of the minimum 5 distances to vote. The mode label will be the predict label of the test image.

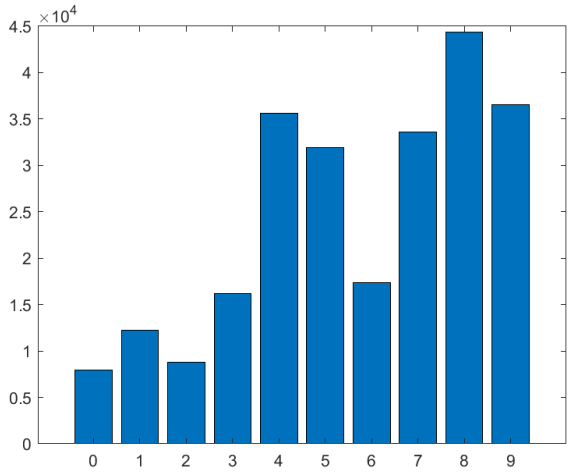
1.3 Performance Measures

I use confusion matrix to evaluate the classification results. The confusion matrix will be 5×5, each row represents the real class labels, each column represents the predicted class labels. If the classification is correct, the should be on the diagonal in the matrix. It is similar to a 2-D histogram.
Additionally, I also check the overall accuracy as

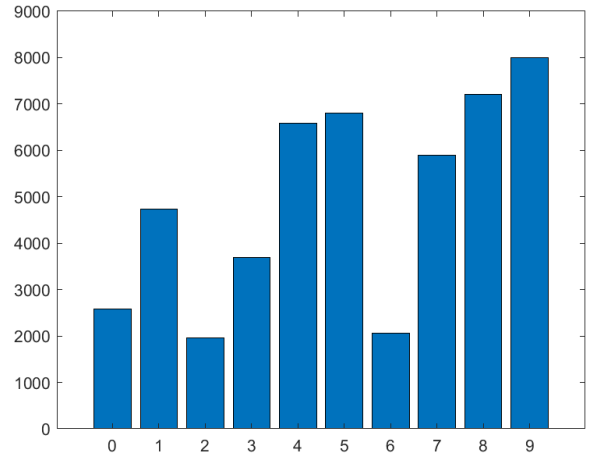
$$r = \frac{\#\text{ of correct predictions}}{\#\text{ of all test images}}.$$

2 Results

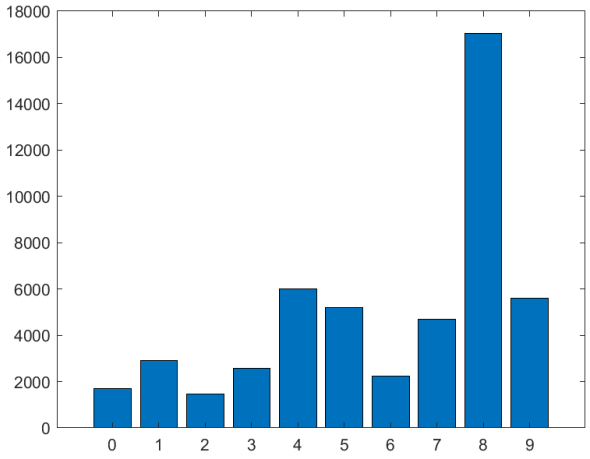
2.1 Examples of LBP Histograms



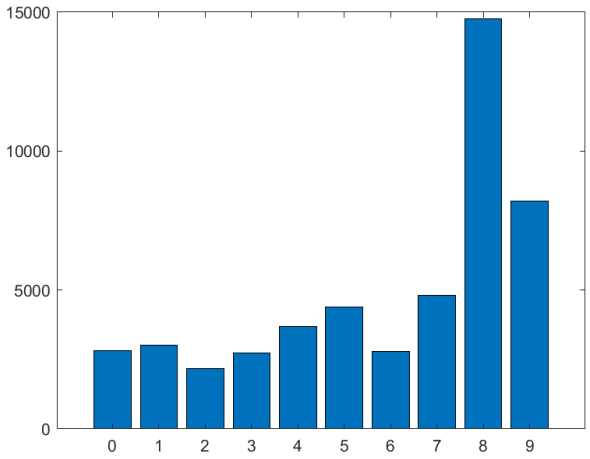
(a) Histogram of 'training\beach\1.jpg'



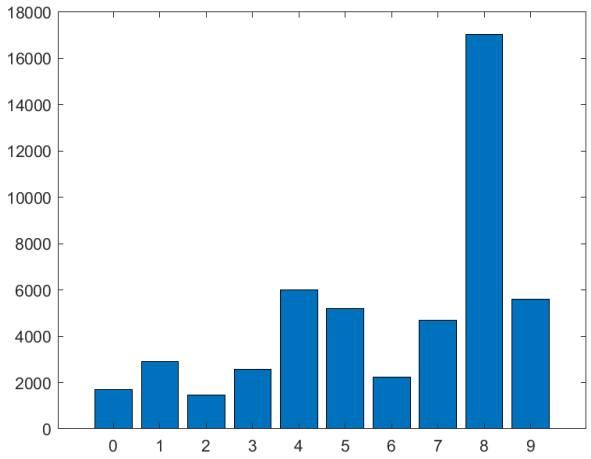
(b) Histogram of 'training\building\01.jpg'



(c) Histogram of 'training\car\01.jpg'



(d) Histogram of 'training\mountain\01.jpg'



(e) Histogram of 'training\tree\01.jpg'

Figure 2: LBP Histogram Examples

2.2 Performance Measures

Table.1 is the confusion matrix of 3-NN classifier, the overall accuracy is 68%.

		prediction				
		beach	building	car	mountain	tree
true class	beach	5	0	0	0	0
	building	0	4	0	1	0
	car	0	1	3	0	1
	mountain	2	1	0	2	0
	tree	0	2	0	0	3

Table 1: Confusion Matrix of 3-NN Classifier

Table.2 is the confusion matrix of the 5-NN classifier. The overall accuracy is 64%.

		prediction				
		beach	building	car	mountain	tree
true class	beach	4	0	1	0	0
	building	0	4	0	1	0
	car	1	1	3	0	0
	mountain	2	2	0	1	0
	tree	0	1	0	0	4

Table 2: Confusion Matrix of 5-NN Classifier

Table.3 is the confusion matrix of 7-NN classifier, the overall accuracy is 68%.

		prediction				
		beach	building	car	mountain	tree
true class	beach	5	0	0	0	0
	building	0	3	0	2	0
	car	0	2	3	0	0
	mountain	0	1	0	4	0
	tree	0	1	0	2	2

Table 3: Confusion Matrix of 7-NN Classifier

2.3 Observations

1. The all above NN classifiers' performances are acceptable, but not good, since it is over 50%, which means it does not flip a coin to classify at least. To change the number of the nearest neighbours, the NN classifier accuracy could raise. It is interesting that some classes are misidentified due to over fitting in 5-NN and 7-NN.

2. After I investigate the confusion matrices, I found the building test images are easily misidentified to mountains, because both of them contain many noise pattern like sky and clouds. And car class is easily misidentified to beech and building due to the texture of glass and reflection of sunlight.

3. The LBP feature actually measures the dominate texture in images, since we do not segment the object in images. As an result, NN classifier would identify images as the class with images have the dominate texture. For an instance, if most part of an image of a building is sky, the euclidean distance is closer to images with more sky texture.

4. In this homework, we only encode the pattern to 9. From Fig.2a, 2b and 2d, we can see a lot of texture are encode to 9. In this case, if we introduce 10 into the encoder when runs> 4, we would have more robust LBP features.

5. It is obvious that the resolution of the images are different. This also cause the LBP features is not robust as expected. In this case, changing R would also help to improve the LBP feature.

3 Source Code

The code is written in Matlab 2017b and Windows 10.

3.1 Main

```
%% ECE 661 2018 Fall Homework 7
% Ye Shi
% shi349@purdue.edu

close all;clear; clc;
grayread = @(x) rgb2gray(imread(x));
label = {'beach','building','car','mountain','tree'};

%% Import training images
Training = struct('data',[],'label',[],'LBP',[]);
num = 0;
for l = 1:5
    for i = 1:20
        num = num+1;
        if strcmp(label{l},'beach')
            Training(num).data = grayread([pwd,'\imagesDatabaseHW7\training\',label
                {l},...
                '\',num2str(i),'.jpg']);
        else
            Training(num).data = grayread([pwd,'\imagesDatabaseHW7\training\',label
                {l},...
                '\',num2str(i,'%02d'),'.jpg']);
        end
        Training(num).label=l;
    end
end

%% find LBP of the trainning set
parfor i = 1:num
    Training(i).LBP = findLBP(Training(i).data);
end
TrainingLBP  = reshape(extractfield(Training,'LBP'),10,num)';
TrainingLabel  = extractfield(Training,'label');

%% Import Test images
testing = struct('data',[],'label',[],'LBP',[],'predict',[]);
```

```
files = dir('imagesDatabaseHW7\testing');
num = 0;
for i = 3:numel(files)
    num = num+1;
    testing(num).data = grayread([files(i).folder,'\',files(i).name]);
    for l = 1:5
        len = length(label{l});
        if strcmp(label{l},files(i).name(1:len))
            testing(num).label = l;
        end
    end
end

%% find LBP of the test set and predict
parfor i = 1:num
    testing(i).LBP = findLBP(testing(i).data);
end
%% 5NN classify

% normalize the training data
% TrainingLBP = normalize(TrainingLBP,2,'range');
% parfor i = 1:num
%     testing(i).predict = LBP5NN(normalize(testing(i).LBP,2,'range'),TrainingLBP,
    TrainingLabel);
% end
num = 25;
parfor i = 1:num
    testing(i).predict = LBP5NN(testing(i).LBP,TrainingLBP,TrainingLabel);
end

%% Examples of LBP histograms
num = 100
for i = 1:num
    if mod(i,20) == 1
        f=figure;
        bar(Training(i).LBP);
        xticklabels({'0','1','2','3','4','5','6','7','8','9'});
        saveas(f, ['LBP',num2str(i),'.png']);
    end
end
close all

%% Confusion Matrix
[CM,rate] = findConfusion(testing)

%% Save Result
save('final.mat')
```

3.2 LBP Extraction

```
%% ECE 661 2018 Fall Homework 7
% Ye Shi
% shi349@purdue.edu

function H = findLBP(I)
% to find LBP of gray scale image I
[h,w] = size(I);
H = zeros(1,10); % histogram
for i=2:h-1
    for j = 2:w-1
        % find pattern
        n = [i+ cos(2*pi*(0:7)/8);j+sin(2*pi*(0:7)/8)];% 8 neighbours
        p = zeros(1,8); % pattern
        for k = 1:8
            if mod(n(1,k),1)==0 && mod(n(2,k),1)==0 % check for integers
                p(k) = I(n(1,k),n(2,k))>=I(i,j);
            else
                p(k) = findNIValue(n(1,k),n(2,k),I)>=I(i,j);
            end
        end
        % find min binary value
        MSB = 2^8;
        for k = 1:8
```

```
24                  bv = circshift(p,k,2);
25                  msb = bin2dec(num2str(bv));
26                  if MSB >= msb
27                      MSB = msb;
28                      minbv = bv;
29                  end
30              end
31              % find runs
32              %          r = xor(minbv(1), minbv(8));
33              r=0;
34              for k = 1:7
35                  if xor(minbv(k),minbv(k+1))
36                      r = r+1;
37                      if minbv(k+1) == 1
38                          tag = k+1;
39                      end
40                  end
41              end
42
43              % encode
44              if r<2
45                  e = sum(minbv);
46              elseif r == 2
47                  e = sum(minbv(tag:end));
48              else
49                  e = 9;
50              end
51
52              % debug
53  %             r
54  %             tag
55  %             [i,j]
56  %             p
57  %             minbv
58  %             e
59  %
60              H(e+1) = H(e+1)+1;
61          end
62  end
63  % nor
64
65  end
66
67  function i = findNIValue(x,y,I)
68  % find the non Int coordinate pixel value by d4 interpolation
69  xt = ceil(x);
70  xb = floor(x);
71  yt = ceil(y);
72  yb = floor(y);
73
74  A = I(xb,yb);
75  B = I(xb,yt);
76  C = I(xt,yb);
77  D = I(xt,yt);
78
79  dl = y-yb;
80  dk = x-xb;
81  i = (1-dk)*(1-dl)*A + (1-dk)*dl*B+ dk*(1-dl)*C +dk*dl*D;
82
83  end
```

3.3 5-NN Classifier

```
1  %% ECE 661 2018 Fall Homework 7
2  % Ye Shi
3  % shi349@purdue.edu
4
5  function L = LBP5NN(testing, training, traininglabels)
6  % 5-NN for LBP
7  dist = vecnorm(testing - training,2,2);
8  [~,I] = sort(dist,'ascend');
9  L = mode(traininglabels(I(1:5)));
10 end
```

3.4 Confusion Matrix

```
%% ECE 661 2018 Fall Homework 7
% Ye Shi
% shi349@purdue.edu

function [CM, rate] = findConfusion(testing)
turelabel = extractfield(testing,'label');
pred = extractfield(testing,'predict');
CM = zeros(5);
for i = 1:length(turelabel)
    CM(turelabel(i),pred(i)) = CM(turelabel(i),pred(i))+1;
end
rate = sum(diag(CM))/length(turelabel);
end
```