Assignment #1: Basic Recognition

Nikolay Vasilev
IBB 22/23, FRI, UL
nv7834@student.uni-lj.si

I. Introduction

For the first assignment I will implement and evaluate the algorithm Local Binary Patterns (LBP) for recognition. In order to see, how good the implementation works, I will use a plain pixel-wise comparison of images and compare my results also with OpenCV's implementation.

II. Related Work

To find out more about the LBP algorithm and to be able to implement it, I have found multiple scientific papers, like the paper Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns [1], which allowed me to understand how to calculate the default and uniform LBP. Going deeper I have found out, that there are also new types of LBP proposed, which give bigger results. One of them is the Robust Local Binary Pattern (RLBP) [2], which outperforms other descriptors. Anyway in my implementation I will focus on the basic LBP algorithm and on the ways to get better classification results.

III. Methodology

To do the experiment, I had to develop a program, which would read samples and compare them, in order to calculate rank-1 classification accuracy. I created a python program, starting with function for calculating rank-1 using pixel-wise image comparison. I am calculating the rank-1 for 3 different distances - Euclidean, Manhattan and Cosines distance. I also developed functions that calculate default and uniform LBP (custom and OpenCV) for different radius (R) and code length (L), and also some help functions that load images, grayscale them, resize them and generates a 2D array from them. Some other functions convert 2D arrays (vectors) to 1D and generate the histogram values of pixels and LBP values to give me the possibility to get more information from the experiment.

IV. Experiments

To do my experiment I used 1000 images, which are classified in 100 different classes - 10 images per class. After that I have calculated the Euclidean, Manhattan and Cosines distance between each image with all others. The closest images (smallest distane), if from the same class are counted and in the end the counts for each of the distances are divided by 1000 (all images) to get the rank-1 for each of them.

V. Results and Discussion

Here are presented the results for rank-1 classification accuracy of 2 different size of images - 64x64 and 128x128. For each size images I have calculated rank-1 for pixel-by-pixel comparison image comparison, default and uniform LBP with and without histogram values for $R = 1, L = 8$ and $R = 2, L = 16$. Of course I have also calculated the default and uniform LBP from OpenCV (LBP Scikit, LBPS) with and without histogram values, to be able to report the difference between my implementation of the LBP algorithm and the already implemented LBP algorithm in OpenCV.

A. Results

First we will start by examining the results from the images with size 64x64. As you can see on the table I the results from our LBP is a lot lower than the LBP from OpenCV for all 3 distances, but the histogram values are not so different than the one from OpenCV. Interesting thing is that my uniform LBP has better values than the uniform LBP from the OpenCV. The most visible difference here is that, when I use bigger radius and code length I get lower values from my LBP algorithm, but from the OpenCV algorithm they get better. Histogram values from my LBP uniform algorithm also here are not so different from the uniform LBP OpenCV algorithm, which is good. The bigger images 128x128 as we can see

TABLE I
Rank-1 classification accuracy for images with size 64x64

Type	Euclidean	Manhattan	Cosines
Pixel-by-pixel	0.060	0.064	0.008
Pixel-by-pixel histogram	0.029	0.037	0.032
LBP (R = 1, L = 8)	0.081	0.089	0.089
LBP (R = 2, L = 16)	0.072	0.087	0.080
LBP histogram (R = 1, L = 8)	0.038	0.034	0.036
LBP histogram (R = 2, L = 16)	0.043	0.041	0.047
LBP uniform (R = 1, L = 8)	0.065	0.097	0.083
LBP uniform (R = 2, L = 16)	0.065	0.083	0.078
LBP uniform histogram (R = 1, L = 8)	0.021	0.021	0.021
LBP uniform histogram (R = 2, L = 16)	0.013	0.017	0.015
LBPS (R = 1, L = 8)	0.245	0.305	0.176
LBPS (R = 2, L = 16)	0.308	0.316	0.232
LBPS histogram (R = 1, L = 8)	0.052	0.050	0.054
LBPS histogram (R = 2, L = 16)	0.083	0.106	0.093
LBPS uniform (R = 1, L = 8)	0.023	0.043	0.009
LBPS uniform (R = 2, L = 16)	0.051	0.121	0.059
LBPS uniform histogram (R = 1, L = 8)	0.022	0.026	0.024
LBPS uniform histogram (R = 2, L = 16)	0.018	0.022	0.018

from the table II have similar values for pixel-by-pixel and pixel-by-pixel histogram as the 64x64. As we can see here the LBP values and the LBP uniform values are higher than the values from the 64x64 images, but they are still lower than the OpenCV values. Interesting thing here is that for this size images the default and uniform LBP from OpenCV has decreased its values, but my the values from the LBP algorithm has increased its values. Here my LBP uniform values are a lot better than the OpenCV uniform LBP values and the LBP values are closer to the OpenCV LBP. The other values here have similar dependencies as on the table I, but for better values for my algorithm and worse for the OpenCV algorithm. From both tables we can see that the rank-1 using the Manhattan distance gives us the highest values for most of the tests and that histogram values are not always appropriate to be used because they give very low rank-1 most of the time, but are very close the the OpenCV. From the experiment it becomes clear that the uniform LBP works better on smaller images for my as well as for the OpenCV LBP algorithm.

TABLE II
RANK-1 CLASSIFICATION ACCURACY FOR IMAGES WITH SIZE 128X128

Type	**Euclidean**	**Manhattan**	**Cosines**
Pixel-by-pixel	0.060	0.068	0.010
Pixel-by-pixel histogram	0.028	0.037	0.031
LBP (R = 1, L = 8)	0.079	0.088	0.083
LBP (R = 2, L = 16)	0.076	0.094	0.093
LBP histogram (R = 1, L = 8)	0.038	0.033	0.035
LBP histogram (R = 2, L = 16)	0.039	0.032	0.035
LBP uniform (R = 1, L = 8)	0.064	0.086	0.077
LBP uniform (R = 2, L = 16)	0.070	0.091	0.078
LBP uniform histogram (R = 1, L = 8)	0.030	0.027	0.029
LBP uniform histogram (R = 2, L = 16)	0.010	0.011	0.010
LBPS (R = 1, L = 8)	0.172	0.232	0.043
LBPS (R = 2, L = 16)	0.266	0.288	0.163
LBPS histogram (R = 1, L = 8)	0.038	0.040	0.043
LBPS histogram (R = 2, L = 16)	0.054	0.061	0.053
LBPS uniform (R = 1, L = 8)	0.015	0.019	0.014
LBPS uniform (R = 2, L = 16)	0.020	0.036	0.013
LBPS uniform histogram (R = 1, L = 8)	0.029	0.025	0.031
LBPS uniform histogram (R = 2, L = 16)	0.033	0.033	0.031

B. Discussion

As we can see from the results my LBP algorithm works better on bigger photos and with radius and length not too big for the photo, because in my algorithm I don't use different levels of local region and overlaps, which is used in the OpenCV and that is the main reason, why my LBP algorithm most of the times gives worse results than the OpenCV LBP algorithm, especially for the 64x64 photos.

VI. CONCLUSION

In this assignment, I analysed the algorithm LBP for recognition using a python program, which calculated the rank-1 recognition rate for Euclidean, Manhattan and Cosines distance between all images. With that experiment I found out that my algorithm is not fully ready yet, since it is missing some important functionalities as overlapping, which makes its rank-1 smaller than the OpenCV one, but its values increase, when working on bigger images, because of that.
The final results tell us that even when using the LBP OpenCV we don't get very big rank-1 values, that is why we have to look ways to improve that algorithm.

REFERENCES

[1] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[2] J. Chen, V. Kellokumpu, G. Zhao, and M. Pietikäinen, "Rlbp: Robust local binary pattern." in *BMVC*, 2013.