Assignment 1: Implementation of Local Binary Pattern for Image Recognition

Stefanela Stevanović
IBB 22/23, FRI, UL
ss51676@student.uni-lj.si

I. INTRODUCTION

The goal of this assignment is to implement and evaluate Local Binary Pattern (LBP) algorithm for the human ear recognition. In this paper different modifications of LBP were implemented: using different levels of local region overlaps, different radii and code lengths and different input image sizes. Accuracy of these LBP modifications for the recognition task was tested by computing rank-1 recognition rate for each type of feature extractor. Euclidean distance was used as a main distance measure for feature vector comparison in this assignment.

II. RELATED WORK

LBP is a local texture descriptor first described in 1994. Given a pixel in the image, an LBP code is computed by

$$\mathrm{LBP}_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c)2^p, \; s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

comparing it with it's neighbors

where g_c is the gray value of the central pixel, g_p is the value of it's neighbors, P is the total number of involved neighbors, and R is the radius of the neighborhood. [1]

III. METHODOLOGY

The first step in this assignment was to do the plain pixel-wise comparison of the images, in order to obtain a baseline against which the improvement of recognition with the LBP implementation could be observed. Since the images are two-dimensional matrices of pixels, they were transformed into one-dimensional feature vectors that were compared with each other. For each vector the closest vector to the currently observed one was found using Euclidean distance measure and it was checked whether they belong to the same class. This was repeated for all the vectors and the percentage of correct predictions gave a rank-1 recognition rate. Rank-1 recognition rate was computed in this way for all types of feature extractors used in this assignment.

The basic LBP feature vector (R=1, L=8) was obtained by calculating the LBP code of individual central-pixels. This was done by comparing the values of each of 8 neighbors with the values of the central pixel starting from the left-top corner and going in the clockwise direction. For each neighbor of the central pixel (threshold) a new binary number was set: 0 for the values lower than the threshold and 1 for the values higher than the threshold. These values were concatenated into an 8-bit binary number, that when converted to decimal number represented the LBP value of the central pixel. At the end of this procedure a new image was obtained that better represents the characteristics of the original image.

For the LBP version without local region overlaps, the input image was viewed as a set of many 3x3 non-overlapping subregions, where LBP code was computed for each central pixel. After that, LBP was implemented using different levels of local region overlaps, different input image sizes, different radii and code lengths in order to observe how those parameters affect the recognition rate. At the end, the results from the different LBP modifications were compared with the Skimage implementation of LBP.

IV. EXPERIMENTS

The dataset used in the experiments consists of 100 classes, where each class contains 10 images of ears belonging to one person (total of 1000 images in the dataset). The code was written in the Python programming language (version 3.9.7) in the Spyder environment. All the images from the dataset were opened in a fixed image size (128x128) and converted to grayscale using Python Pillow Library. The experiments listed below were conducted and rank-1 recognition rates for different types of feature extractors were computed using the Euclidean distance measure for feature vector comparison.

The conducted experiments with the used parameters are listed here:

1. Comparison of the results obtained by plain pixel-wise image comparison and LBP implementation with different levels of local region overlaps. The parameters used for LBP are R=1, L=8 and step size 1, 2 and 3. The step size parameter tells with what step the LBP algorithm takes the central pixels for which the LBP code is calculated, so a smaller step size – a larger local region overlaps. Step size 3 means that the LBP is calculated for every third pixel in the original image, which combined with parameter R=1 means no local region overlaps.
2. LBP with different radii, R and different code lengths, L. The following sets of parameters were used:
(R = 1, L = 4, step = 1); (R = 1, L = 8, step = 1); (R = 2, L = 4, step = 1) and (R = 2, L = 8, step = 1).
3. LBP with parameters R = 1, L = 8, step = 1 on different input image sizes: 32x32, 64x64, 128x128 and 256x256.
4. Experiment with combined LBP parameters that showed high recognition rates.
5. Skimage implementation of LBP and comparison with previously obtained results.

V. RESULTS AND DISCUSSION

In this part of assignment the results obtained in the experiments described in chapter III are presented and discussed.

A. Results

Table 1 shows the recognition rates for feature extractors used in the experiment 1, that is pixel-by-pixel feature vector and LBP with different level of local region overlaps.

TABLE 1
Rank-1 recognition rate for pixel-by-pixel feature extractor and LBP with different levels of local region overlaps

Feature extractor	Recognition rate [%]
Pixel-by-pixel	13.2
LBP (R = 1, L = 8, step = 3)	18.0
LBP (R = 1, L = 8, step = 2)	20.3
LBP (R = 1, L = 8, step = 1)	21.6

Table 2 shows the recognition rates for LBP with different radii and code lengths. Since LBP with step 1 gave the best results, this parameter was used in all the following experiments. On the Fig.1. LBP output images are shown for different sets of parameters.

TABLE 2
Rank-1 recognition rate for LBP with different radii and code lengths

LBP parameters	Recognition rate [%]
R = 1, L = 4, step = 1	12.6
R = 1, L = 8, step = 1	21.6
R = 2, L = 4, step = 1	10.6
R = 2, L = 8, step = 1	27.5

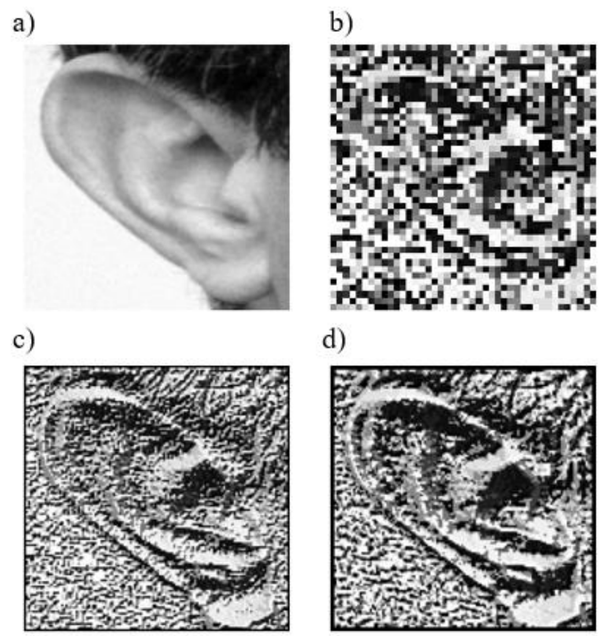

Fig.1. Image 128x128 at different levels of LBP implementation: a) original input image; b) R = 1, L =8, step=3; c) R = 1,L = 8, step =1; d) R = 2, L = 8, step =1.

In table 3 recognition rate results are shown for different input image sizes (experiment 3).

TABLE 3
Rank-1 recognition rate for LBP with different input image sizes

Image size	Recognition rate [%]
32x32	35.0
64x64	28.7
128x128	21.6
256x256	20.0

In the experiment 4, combinations of parameters that give best results were evaluated and showed in the table 4.

TABLE 4
Optimization of LBP parameters

Image size	LBP parameters	Recognition rate [%]
32x32	R = 1, L = 8, step = 1	35.0
	R = 2, L = 8, step = 1	36.0
64x64	R = 1, L = 8, step = 1	28.7
	R = 2, L = 8, step = 1	34.7

In the table 5 rank-1 recognition rate was compared between the skimage and developed LBP for the same set of parameters.

TABLE 5
Comparison of skimage LBP implementation with developed LBP

Parameters	LBP implementation	Recognition rate [%]
R = 1, L = 8 Image size: 128x128	my approach	21.6
	skimage	20.3
R = 2, L = 8 Image size: 128x128	my approach	27.5
	skimage	24.6
R = 1, L = 8 Image size: 64x64	my approach	28.7
	skimage	26.5
R=2, L = 8 Image size: 64x64	my approach	34.7
	skimage	30.9

B. Discussion

Experiment 1 showed that there was a significant improvement in recognition accuracy even with the basic implementation of LBP (18.0%), compared to plain pixel-wise comparison (13.2%). Table 1 shows that the greater the local region overlap in the LBP, the greater is the recognition rate, with the highest recognition rate for the LBP algorithm with step size 1 (21.6%). This stems from the fact that the smaller step in the LBP algorithm means that the LBP code is calculated for more pixels in the image, thus the new LBP image better represents the characteristics of the original image.

Table 2 shows that a higher recognition rate occurs when parameters R = 2 and L = 8 are used (27.5%), than with shorter code length L or R = 1. By considering more neighbors of the central pixel, the local environment of the central pixel is more precisely described, and with greater radius, the larger area around central pixel is considered. Figure 1 shows how adding the improvements such as local region overlaps and greater radius improves the texture of the LBP output image.

Table 3 shows that reducing the image resolution leads to an increase in the rank-1 recognition rate, where the LBP algorithm correctly classifies 35% of 32x32 images. However, it should be noted that reducing the image size leads to the loss of information of the original image. The highest recognition rate of 36% was observed with the combination of parameters R = 2, L = 8 and image size od 32x32 (table 4), but the image size of 64x64 showed a greater increase in the recognition rate (from 28.7 to 34.7%) with the change of the radius from 1 to 2. Therefore, 64x64 image size would be preferred as it leaves more room for improvement in terms of further increasing the radius and also preserves more information from the original image compared to 32x32 image size.

When comparing the results obtained with the skimage implementation of LBP with my approach (table 5), it was shown that my approach gives a slightly higher recognition rate for different sets of parameters. These recognition rates are on average 1,07 - 1,12 times higher compared to skimage recognition rates.

VI. CONCLUSION

Although the recognition rate results for LBP are not ideal, they are shown to be significantly better than the plain pixel-wise image comparison. After all, these results were expected as LBP is just a texture descriptor and for such a demanding task as ear recognition, CNN should be implemented. But there is still a lot of room for improvement of this LBP algorithm. Some of the ideas are to consider more neighbors around the central pixel (e.g. R = 2, L = 16), use the greater radius or to use multi-block LBP to capture large scale structure. In this case LBP histogram can be calculated for every block and concatenated in the final histogram.

REFERENCES

[1] Guo, Zhenhua, Lei Zhang, and David Zhang. "A completed modeling of local binary pattern operator for texture classification." *IEEE transactions on image processing* 19.6 (2010): 1657-166