Reducing the Complexity of Multi-Dimensional LBP Texture Features using Genetic Optimisation

Niraj Doshi and Gerald Schaefer
Department of Computer Science
Loughborough University, U.K.
n.doshi@lboro.ac.uk, gerald.schaefer@ieee.org

Abstract— *Texture analysis and classification have received significant research interest and have been shown to be essential in many computer vision systems and applications. Local binary patterns (LBP) form a simple yet powerful texture descriptor characterising local neighbourhood properties, which, due to its effectiveness and robustness, is widely employed. LBP information can be gathered at multiple scales to improve the performance of the descriptor. While in conventional LBP this information is recorded, in form of a histogram, separately for each of the scales, it was shown that a multi-dimensional (MD) feature representation removes some ambiguity and leads to better texture classification. However, the generated MD-LBP histograms result in relatively large feature descriptors which limit their practical use. In this paper, we show that a feature selection stage based on a genetic algorithm can be successfully applied to reduce the dimensionality of MD-LBP features while maintaining effective texture classification.*

Keywords: Texture, local binary patterns, LBP, MD-LBP, feature selection, genetic algorithm.

1. Introduction

Texture analysis and classification form an important part of many computer vision tasks including content-based image retrieval, face analysis, medical image analysis, multimedia content classification and annotation. While various powerful texture descriptors have been developed, changes in orientation, scale, illumination and other confounding imaging factors still present challenges.

Local binary patterns (LBP), first introduced in [1], represents a relatively simple yet powerful texture descriptor describing the relationship of a pixel to its immediate neighbourhood. Later work [2] extended that concept to varying neighbourhoods, to make the method greyscale and rotation invariant and to emphasise "uniform" texture patterns.

LBP descriptors are obtained at pixel locations and summarised into histograms which then serve as texture descriptors. For multi-scale LBP, the information is obtained at multiple radii and the resulting histograms are concatenated into a feature vector. Although this gives improved texture recognition, it was shown in [3] that it also leads to a loss of information and added ambiguity. A multi-dimensional LBP (MD-LBP) histogram was thus proposed to preserve the relationships between scales and was shown to lead to further improved texture classification performance.

Although MD-LBP allows for better texture classification, it results in relatively large feature descriptors which limit its usefulness. For example, while the original uniform rotation invariant LBP descriptor calculated at three scales leads to a feature length of 30, an MD-LBP histogram with the same parameters has a total of 1000 bins. In this paper, we show that through application of a feature selection stage based on a genetic algorithm (GA) the dimensionality of MD-LBP features can be drastically reduced while maintaining good texture classification performance. We demonstrate this based on experiments on the Outex dataset.

2. LBP texture features

Local binary patterns (LBP) are simple yet effective texture descriptors. The original LBP variant [1] operates on a per-pixel basis, and describes the 8-neighbourhood pattern of a pixel in binary form. If

$$B = \begin{pmatrix} g_8 & g_1 & g_2 \\ g_7 & g_{(0,0)} & g_3 \\ g_6 & g_5 & g_4 \end{pmatrix} \quad (1)$$

is the 3×3 grayscale block of a pixel at location $(0, 0)$ and its 8-neighbourhood, then the neighbouring pixels are set to 0 and 1 by thresholding them with the centre pixel value. The value of the central pixel is subtracted from each neighbour

$$LBP_1 = \begin{pmatrix} g_8 - g_c & g_1 - g_c & g_2 - g_c \\ g_7 - g_c & & g_3 - g_c \\ g_6 - g_c & g_5 - g_c & g_4 - g_c \end{pmatrix} \quad (2)$$

where $g_c = g_{(0,0)}$ for convenience, and the binary code is then generated by applying the thresholding function

$$s(x) = \begin{cases} 1 & \text{for } x \geq 0 \\ 0 & \text{for } x < 0 \end{cases} \quad (3)$$

at each location which results in

$$LBP_2 = \begin{pmatrix} s(g_8 - g_c) & s(g_1 - g_c) & s(g_2 - g_c) \\ s(g_7 - g_c) & & s(g_3 - g_c) \\ s(g_6 - g_c) & s(g_5 - g_c) & s(g_4 - g_c) \end{pmatrix} \quad (4)$$

Finally, the LBP pattern is obtained by

$$LBP = \sum_{p=1}^{8} s(g_p - g_c)2^{p-1} \tag{5}$$

The 256 possible patterns resulting from the above procedure are used to build a histogram, which serves as a texture descriptor for the image.

2.1 Circular LBP

In the above procedure, the 8-neighbourhood of each pixel is utilised. Clearly, four of these neighbours are at a different distance ($\sqrt{2}$) than the other four. To compensate this, a circular neighbourhood can be defined [2] where locations that do not fall exactly at the centre of a pixel are obtained through interpolation.

A neighbourhood is defined by R and P where R defines the distance of the neighbours to the centre, while P gives the number of samples at that distance that are employed as neighbours. If g_c is at $(0, 0)$, then the co-ordinates of the neighbouring pixels g_p, $p = 1, 2, \ldots, P$, are given by $(-R\sin(2\pi p/P))$, $(R\cos(2\pi p/P))$.

2.2 Rotation invariant LBP

It has been shown [2] that rotation invariance is relatively simple to address in LBP. If a texture is rotated, essentially the patterns (that is the 0s and 1s around the centre pixel) rotate with respect to the centre.

Rotation invariant LBP codes, $LBP^{ri}_{P,R}$, can be generated through shift operations on the bit sequence so as to arrive at a sequence with a maximal number of leading 0s, i.e.

$$LBP^{ri}_{P,R} = \min\{ROR(LBP_{P,R}, i) \mid i = 0, 1, \ldots, P-1\} \tag{6}$$

where $ROR(x, i)$ performs a circular bit-wise right shift by i bits.

2.3 Uniform LBP

Certain binary patterns are fundamental properties of texture and sometimes their frequency exceeds 90%. These patterns are called uniform [2], leading to $LBP^{u}_{P,R}$, and are defined by a uniformity measure which corresponds to the number of spatial transitions (i.e., changes from 0 to 1 and vice versa). Patterns with a uniformity measure of 2 are given by

$$LBP^{u2}_{P,R} = \begin{cases} \sum_{p=0}^{P-1} s(g_p - g_c) & \text{if } U(LBP_{P,R}) \leq 2 \\ P + 1 & \text{otherwise} \end{cases} \tag{7}$$

where

$$\begin{aligned} U(LBP_{P,R}) = &|s(g_p - g_c) - s(g_0 - g_c)| \\ &+ \sum_{p=1}^{P-1} |s(g_p - g_c) - s(g_{p-1} - g_c)| \end{aligned} \tag{8}$$

Clearly, rotation invariance, leading to $LBP^{riu2}_{P,R}$, can be achieved in the same way as above. For eight neighbours there are nine rotation invariant uniform LBP codes, two without any 0-1 changes (i.e., one with all 0s and one with all 1s) and the remaining seven with $1, \ldots, 7$ ones in sequence. It has been shown [2] that focussing on these uniform patterns while aggregating all other (i.e., non-uniform) patterns into one group leads to improved texture descriptors. While LBP generates 256 patterns for an 8-neighbourhood, and LBP^{ri} generates 36 patterns, LBP^{riu2} results in 10 pattern classes for the same neighbourhood.

2.4 Multi-scale LBP

By defining several radii around a pixel, multiple concentric neighbourhood LBP codes can be extracted. While in principle any radius is feasible, attention is often restricted to the sets $r = \{1, 3\}$ and $r = \{1, 3, 5\}$. Also, while in general any number of neighbours could be defined, we found in our experiments that choosing 8 neighbours at all distances does not compromise accuracy while also corresponding to those directions (horizontal, vertical, and plus/minus 45 degrees) to which the human visual system is most sensitive to.

2.5 Multi-dimensional LBP

Multi-scale LBP is performed by concatenating the LBP features from each radii into a one-dimensional feature vector. In [3], it was shown that this results in a loss of information between different scales and added ambiguity. This is illustrated in Fig. 1 where an "image" consisting of the two samples on the top will lead to exactly the same LBP descriptor as the two samples on the bottom. Both resulting LBP histograms will have one entry each for bins (00001111) and (00111111) for both radii.

A multi-dimensional histogram is hence used to preserve relations between different radii. At each pixel, LBP codes at different scales are extracted, while the combination of these codes identifies the histogram bin that

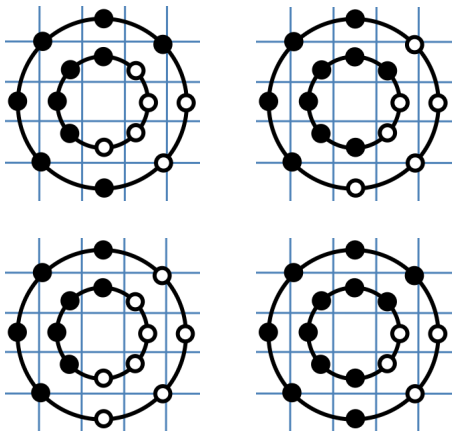

Fig. 1: Multi-scale LBP example.

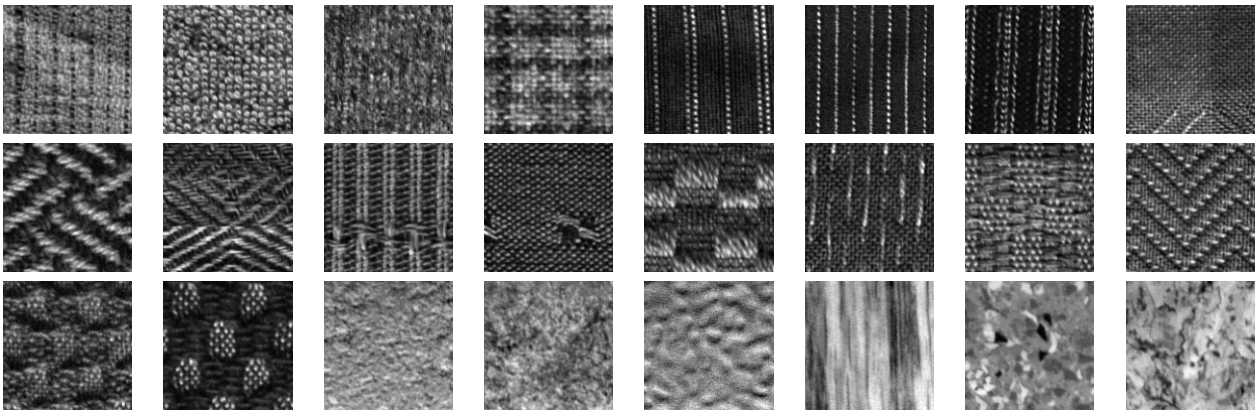

Fig. 2: Sample images of the 24 texture classes.

is incremented. For the example in Fig. 1, this gives a histogram with one entry for bin $(00001111, 00111111)$ and one entry for bin $(00111111, 00001111)$ for the top "image", while for the example at the bottom a histogram with one entry for $(00001111, 00001111)$ and one entry for $(00111111, 00111111)$ is obtained; i.e. two distinct histograms and hence two distinguishable texture descriptors are generated. MD-LBP has been shown to allow for improved texture classification in comparison to the original LBP variants [3].

3. Feature selection for MD-LBP

MD-LBP retains the relationships between scales, but at the cost of relatively large feature descriptors resulting in higher memory requirements and reduced processing speed. In this paper, we address this problem by applying a feature selection technique to reduce the length of MD-LBP descriptors.

In particular, we employ a feature selection algorithm that employs a genetic algorithm (GA) for selecting an optimal set of MD-LBP histogram bins, based on the technique presented in [4].

If we have M features of which we want to select N, then the resulting combinatorial optimisation problem has $\frac{M!}{2(M-N)!}$ possible solutions. Clearly, and especially for larger values of M, an exhaustive search is not feasible, and we consequently use a GA to search for a good feature set. The GA fitness function $\Phi = V - P$ is based on the principle of maximum relevance and minimum redundancy, where V, the relevance of a set with N features, is defined as

$$V = \frac{1}{N} \sum_{i=1}^{N} I(yh_i; y), \tag{9}$$

where $I(yh_i; y)$ is the mutual information of features and labels, and P is the redundancy between those features given by

$$P = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} I(yh_i; yh_j), \tag{10}$$

where $I(yh_i; yh_j)$ is the mutual information between output features.

In the GA, each individual corresponds to a feature set of size N. The GA starts by randomly initialising a populating of N_{pop} (200 times the desired feature length in our implementation) individuals. Then, for each generated feature set V and P are calculated and the fitness Φ is recorded.

During each iteration, parents are randomly selected based on an asymmetric distribution function defined as

$$\text{Parent}_j = \text{round}(N_{pop}(e^{a\vartheta_j} - 1)/(e^a - 1)), \tag{11}$$

where $j = (1, 2)$ represents the parent number, a is set to 6 and ϑ_j is a random number with uniform distribution.

From the two parents, an offspring is generated by randomly selecting features from the two parents and ensuring that no duplicate features are selected. Selection and crossover is repeated until a full new population has been generated.

As stopping criterion, the uniformity of the population is used, expressed as the difference between the average and maximum of Φ. The algorithm terminates if this falls below a threshold (0.002) or if the maximum number of iterations (80) is reached. The individual of highest fitness then gives the selected features.

4. Experimental results

In our experiments, we performed texture classification on two databases from the Outex test suite [5]. Fig. 2 shows a sample image for each of the 24 classes used in both test suites; as can be seen the dataset is not simple as several texture classes are rather similar. As classifier, we employ standard support vector machines (SVMs) [6]. Since we have

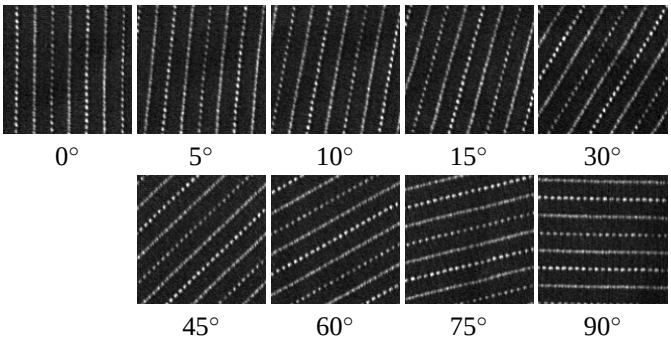

Fig. 3: Sample texture from the Outex_TC_10 dataset under different rotations.

more than two classes, we employ a one-against-one multi-class SVM [7] where for each SVM, we use a linear kernel and optimise the cost parameter $C \in [-1.1; 3.1]$ using a cross validation approach [8].

Our first experiment, performed on the Outex TC10 database, is designed to investigate a useful range of feature lengths to be used for our approach. The TC10 dataset is built from 24 texture classes captured at 9 rotation angles under the same illumination (see Fig. 3 for an example), with 20 samples of each class. The classifier is trained on 20 samples (at angle 0°) in each texture class, that is on 480 (24×20) images. Testing is performed on the other 8 angles, i.e. on 3840 ($24 \times 20 \times 8$) images.

We used $MD\text{-}LBP^{riu2}_{R=1,3}$ which gives $10 \times 10 = 100$ features and perform feature selection to extract 5 to 99 features. The results of these (and a curve fitting to a polynomial) are given in Fig. 4. From there, we observe that accuracy increases sharply up to about 30 features which we can hence consider as an indicator for minimum feature length. In the following, we thus perform GA feature selection based on 20, 30 and 40% of the total features for all experiments.

Table 1 gives classification results on the TC10 dataset.

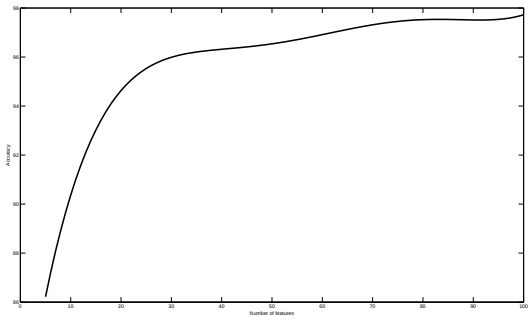

Fig. 4: feature length vs. classification accuracy for Outex TC10 dataset and $MD\text{-}LBP^{riu2}_{R=1,3}$

Table 1: Texture classification results on Outex TC10 dataset.

	no. of features	accuracy
$LBP^{riu2}_{R=1,3}$	20	95.81
$MD\text{-}LBP^{riu2}_{R=1,3}$	100	97.58
$GA\text{-}MD\text{-}LBP^{riu2}_{R=1,3}$	20	94.92
	30	95.32
	40	96.30
$LBP^{riu2}_{R=1,3,5}$	30	94.61
$MD\text{-}LBP^{riu2}_{R=1,3,5}$	1000	95.34
$GA\text{-}MD\text{-}LBP^{riu2}_{R=1,3,5}$	200	94.71
	300	95.21
	400	95.26

Table 2: Texture classification results on Outex TC12 dataset.

	no. of features	accuracy
$LBP^{riu2}_{R=1,3}$	20	85.35
$MD\text{-}LBP^{riu2}_{R=1,3}$	100	90.76
$GA\text{-}MD\text{-}LBP^{riu2}_{R=1,3}$	20	86.10
	30	88.43
	40	88.68
$LBP^{riu2}_{R=1,3,5}$	30	86.18
$MD\text{-}LBP^{riu2}_{R=1,3,5}$	1000	91.96
$GA\text{-}MD\text{-}LBP^{riu2}_{R=1,3,5}$	200	91.70
	300	91.76
	400	91.73

We can see that for $MD\text{-}LBP^{riu2}_{R=1,3}$ and selecting only 40% of the features, we obtain better classification performance compared to standard multi-scale LBP, although we don't quite match the performance of MD-LBP. For $MD\text{-}LBP^{riu2}_{R=1,3,5}$ and again selecting 40% of features, we again outperform conventional LBP and obtain results within a small margin (0.07%) of MD-LBP despite discarding 60% of its features.

The Outex TC12 database allows to evaluate texture classification across different illuminations. A classifier is trained on the same image set as TC10, but is tested on a total of 8640 images of the same textures under different rotations and captured under different light sources. Fig. 5 shows some samples from this dataset.

Classification results on this dataset are given in Table 2.

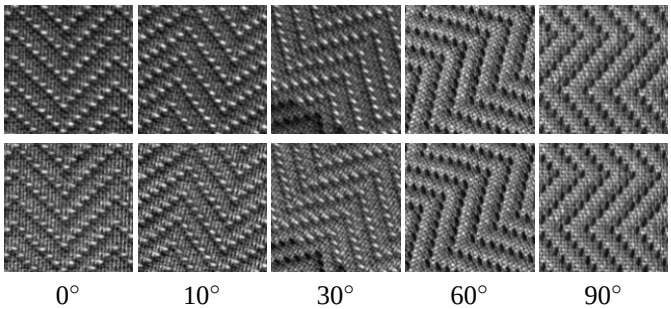

Fig. 5: Sample texture from the Outex_TC_12 dataset under different rotations and different illumination (top row "horizon", bottom row "tl84").

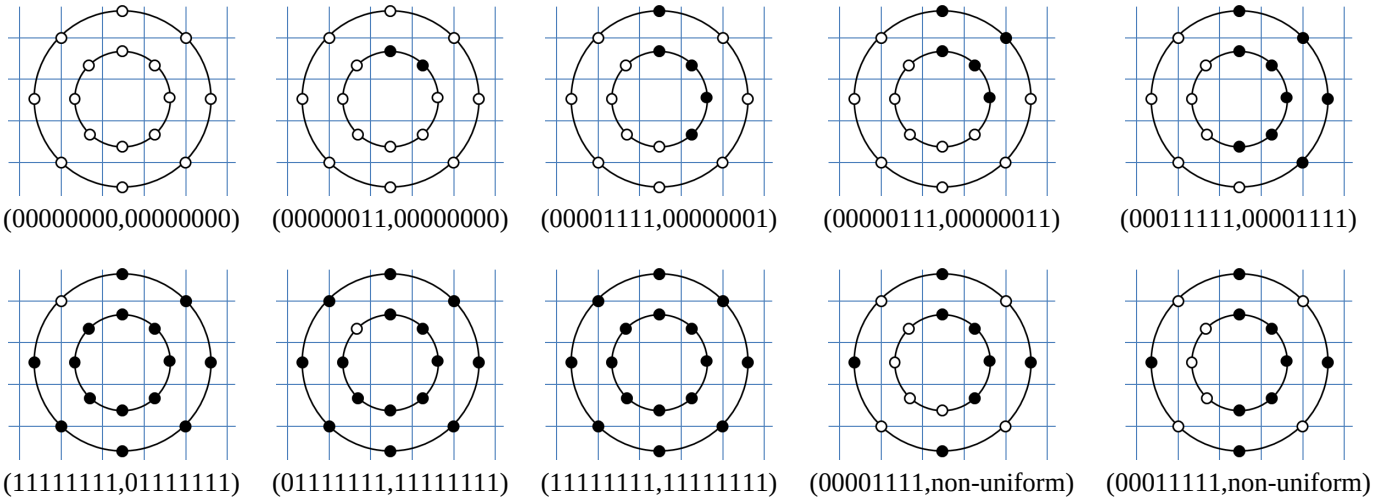

Fig. 6: Discriminative $MD\text{-}LBP^{riu2}_{R=1,3}$ patterns.

From there it is apparent that the results are similar to those obtained for TC10. That is, good texture classification based on a reduced MD-LBP feature set is possible, in particular for $MD\text{-}LBP^{riu2}_{R=1,3,5}$ where even when using only 20% of the features the results are almost the same as for full MD-LBP histograms.

Overall, it is clear that through application of the employed GA-based feature selection method, we are able to significantly reduce the dimensionality of MD-LBP features while maintaining good classification performance, in particular when calculating and integrating texture descriptors at three different radii.

We also inspected the features that were selected. For $MD\text{-}LBP^{riu2}_{R=1,3}$, a set of 10 features were selected in all experiments for both databases. These textures, which are depicted in Fig. 6 should hence give an indication of the most discriminative MD-LBP histogram bins.

5. Conclusions

Texture analysis and classification play an important role in many computer vision applications. Local binary patterns (LBP) is known as a powerful texture descriptor, especially when calculated at different scales. While this information from different LBP radii can be integrated into a single – MD-LBP – texture histogram, this also leads to relatively large feature vectors. In this paper, we have shown that through application of a feature selection algorithm, formulated as an optimisation problem and implemented using a genetic algorithm, a significant reduction of feature dimensionality is possible while maintaining good classification accuracy.

References

[1] T. Ojala, M. PietikÃd'inen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51 – 59, 1996.

[2] 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, pp. 971–987, 2002.

[3] G. Schaefer and N. Doshi, "Multi-dimensional local binary pattern descriptors for improved texture analysis," in *21st Int. Conference on Pattern Recognition*, 2012, pp. 2500–2503.

[4] O. Ludwig and U. Nunes, "Novel maximum-margin training algorithms for supervised neural networks," *IEEE Transactions on Neural Networks*, vol. 21, no. 6, pp. 972–984, 2010.

[5] T. Ojala, T. Maenpaa, M. Pietikainen, J. Viertola, J. Kyllonen, and S. Huovinen, "Outex - new framework for empirical evaluation of texture analysis algorithms." in *16th Int. Conference on Pattern Recognition*, 2002, pp. 1:701 – 706.

[6] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.

[7] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.

[8] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011.