Texture Classification by Texton: Statistical versus Binary

Zhenhua Guo¹, Zhongcheng Zhang¹, Xiu Li², Qin Li^{3*}, Jane You⁴

1 Shenzhen Key Laboratory of Broadband Network & Multimedia, Graduate School at Shenzhen, Tsinghua University, Shenzhen, China, **2** Graduate School at Shenzhen, Tsinghua University, Shenzhen, China, **3** College of Software, Shenzhen Institute of Information Technology, Shenzhen, China, **4** Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China

Abstract

Using statistical textons for texture classification has shown great success recently. The maximal response 8 (Statistical_MR8), image patch (Statistical_Joint) and locally invariant fractal (Statistical_Fractal) are typical statistical texton algorithms and state-of-the-art texture classification methods. However, there are two limitations when using these methods. First, it needs a training stage to build a texton library, thus the recognition accuracy will be highly depended on the training samples; second, during feature extraction, local feature is assigned to a texton by searching for the nearest texton in the whole library, which is time consuming when the library size is big and the dimension of feature is high. To address the above two issues, in this paper, three binary texton counterpart methods were proposed, Binary_MR8, Binary_Joint, and Binary_Fractal. These methods do not require any training step but encode local feature into binary representation directly. The experimental results on the CURET, UIUC and KTH-TIPS databases show that binary texton could get sound results with fast feature extraction, especially when the image size is not big and the quality of image is not poor.

Citation: Guo Z, Zhang Z, Li X, Li Q, You J (2014) Texture Classification by Texton: Statistical versus Binary. PLoS ONE 9(2): e88073. doi:10.1371/journal.pone.0088073

Editor: Konradin Metze, University of Campinas, Brazil

Received: July 24, 2013; **Accepted:** January 6, 2014; **Published:** February 10, 2014

Copyright: © 2014 Guo et al. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Funding: The authors sincerely thank MVG and VGG for sharing the source codes of LBP and Statistical_MR8. The work is partially supported by the Natural Science Foundation of China (NSFC) (number 61101150), Shenzhen special fund for the strategic development of emerging industries (grant number JCYJ20120831165730901) and the National High-Tech Research and Development Plan of China (863) (number 2012AA09A408). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: kenneth_lee_qin@qq.com

Introduction

Texture analysis is an active and fundamental research topic in the fields of computer vision and pattern recognition. Generally speaking, there are four basic problems in texture analysis: classifying images based on texture content; segmenting an image into regions of homogeneous texture; synthesizing textures for graphics applications; and establishing shape information from texture cues [1]. Texture classification has been widely studied because of many potential applications, including fabrics inspection [2], remote sensing [3], and medical image analysis [4].

Early texture classification methods focus on the statistical analysis of texture images. The representative ones include the co-occurrence matrix method [5] and the filtering based method [6]. These methods could achieve good classification results if the training and testing samples are captured by similar orientations. To address the rotation invariance issue, some model-based methods were proposed, such as circular autoregressive model [7], multiresolution autoregressive model [8], hidden Markov model [9], and Gaussian Markov random field [10]. Recently, scale and affine invariance receive extensive attention, and some algorithms were developed to address this issue, such as fractal transform [11], and local phase information [12].

In fact, classifying texture images taken under arbitrary viewing and illumination conditions is a difficult task. The method of statistically representing image local features has achieved great success for this problem [13], two paradigms for image representation were proposed, Signature (representative descriptors of an image [13]) and Statistical Texton (“the putative units of pre-

attentive human texture perception” [30] which does not have a specific definition.). In the former paradigm, an image is represented by signatures which are adaptively extracted from each image [14], and Earth mover’s distance [15] is utilized to compare different images; in the latter paradigm, an image is modeled by feature texton histogram over a dictionary of textons [16–22], and histogram dissimilarity (usually chi-square) is used for histogram comparison [23–24].

Statistical texton based methods are simple to implement and could achieve good performance on texture image classification [16–22]. However, these methods suffer two disadvantages. First, it requires an offline step to learn a texton dictionary from training samples, thus the recognition accuracy is related with the training samples; second, to build the histogram, one needs to search the nearest texton from the dictionary for each pixel. This step is time consuming especially when the dimension of feature is high and the size of dictionary is large.

Local binary pattern (LBP) is a simple and efficient operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the results as a binary number [25], and it is not influenced by the above mentioned issues. Inspired by the idea of LBP, in this paper three binary texton methods, Binary_MR8, Binary_Joint, and Binary_Fractal, were proposed. These methods do not require learning and they are fast for feature extraction. They could be regarded as the counterpart of the three state-of-the-art statistical texton methods, Statistical_MR8 [16–17], Statistical_Joint [18–19] and Statistical_Fractal [20]. Our previous work showed that Binary_MR8 could get better accuracy than Statistical_MR8 on CURET database

Figure 1. The Statistical_MR8 filter bank consists of a series of anisotropic filters (an edge and a bar filter at 6 orientations and 3 scales), and 2 rotationally symmetric ones (a Gaussian and a Laplacian of Gaussian) [16–17].

doi:10.1371/journal.pone.0088073.g001

[26]. This paper extended previous work by proposing two new binary texton methods, Binary_Joint and Binary_Fractal, and by doing more comprehensive experiments on three databases, CURET database [27], UIUC database [14] and KTH-TIPS database [28].

The rest of the paper is organized as follows. Section 2 introduces three statistical texton methods. Section 3 shows the three proposed binary texton methods and dissimilarity metric. Section 4 reports the experimental results on three texture databases. Section 5 gives the conclusion and provides a suggestion for future work.

Review of Statistical Texton Methods

1. Review of Statistical_MR8 [16–17]

The Statistical_MR8 filter bank consists of 38 filters, which are shown in Fig.1. To achieve rotation invariance, the filters are

implemented at multiple orientations and on multiple scales. On each scale only the maximal response among the different orientations is kept. The final response at each pixel is an 8-dimension feature vector (3 scales for the edge and bar filters, plus 2 isotropic filters).

During dictionary learning, a selection of n images is chosen for each class of texture, and the filter responses to all these images are aggregated, then c texton cluster centres are computed using the standard K-Means algorithm [29]. The learnt textons for each texture are then collected into a single dictionary ($n \times c$). For a given image, after getting 8-dimension feature vector for one pixel, the feature is searched in the dictionary to find the closest one and label the pixel with that texton. Finally, an appearance frequency of all textons in the whole dictionary is built as the histogram feature for the image.

Figure 2. Illustration of computation for Statistical_Fractal [20].
doi:10.1371/journal.pone.0088073.g002

2. Review of Statistical_Fractal [20]

To address the scale and affine issue, fractal feature was proposed based on Statistical_MR8 filter banks [20]. Given an image point (x, y) , after getting 8-dimensional filter response by Statistical_MR8 filter banks, $f(x, y) = [f_1(x, y), f_2(x, y), \dots, f_8(x, y)]^T$, fractal is computed by an assumption: given a suitable measure μ , the “size” of local point sets in textured images follows a local power law.

$$\begin{aligned} \mu(B_i(x, y, r)) &\propto r^{D_i(x, y)} \Rightarrow \log \mu(B_i(x, y, r)) \\ &= D_i(x, y) \log r + L_i(x, y) \end{aligned} \quad (1)$$

where $\mu(B_i(x, y, r))$ is the sum of all pixel filter responses of the i^{th} dimension ($i = 1, 2, \dots, 8$) that lies within a closed disk B of radius r centered at an image point (x, y) , $\mu(B_i(x, y, r)) = \sum_{(x-x')^2 + (y-y')^2 \leq r^2} f_i(x', y')$. $D_i(x, y)$ (slope of $\log \mu$ versus $\log r$) and $L_i(x, y)$ (intercept of $\log \mu$ versus $\log r$) are computed by least square estimation. The former is the local fractal dimension and is invariant to scales changes, while the latter is the local fractal length and is rotation invariant only [20]. Fig. 2 illustrates an

example to compute D and L . Two new 8-dimension features, $D(x, y)$ and $L(x, y)$ for each pixel is computed, after that the same dictionary learning and feature extraction procedure as Statistical_MR8 is used. In the following figure, Statistical_Fractal_D and Statistical_Fractal_L represent the texton histogram of D and L , respectively.

3. Review of Statistical_Joint [18–19]

Instead of applying filters on gray level images, Statistical_Joint is proposed to use multi-dimension intensity value (gray value) for each texton [18–19]. For a point x , a $r \times r$ rectangle around x is selected and the intensity of the rectangle is used to represent the texton feature for the point. Fig. 3 shows an example for $r = 3$.

Proposed Binary Texton Methods and Dissimilarity Metric

Fig. 4 shows an overview of the statistical texton and binary texton methods. As illustrated in Fig. 4, binary texton is extracted from intensity value or filter response directly, it does not need any training stage and is fast to build the feature map.

Figure 3. Illustration of texton feature for Statistical_Joint [18–19]. A 3*3 image patch is converted to a 1*9 texton through recording intensity values row by row.
doi:10.1371/journal.pone.0088073.g003

Figure 4. Major framework of texton feature extraction. a) statistical texton; b) binary texton.
doi:10.1371/journal.pone.0088073.g004

1. Binary_MR8

As shown in Fig.5, some local regions may have multiple dominant orientations. The magnitude of the filter response at each angle could be treated as a confidence measurement in the feature occurring at that orientation [17]. It is intuitive to define a binary texton for multiple orientations as:

$$P(x,y) = \sum_{i=1}^N s(F_i(x,y))2^i, \quad (2)$$

$$s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}, i = 1, 2, \dots, N$$

$$F_i(x,y) = g_i * I(x,y) \quad (3)$$

where g_i is the filter as shown in Fig. 1, $*$ is the convolution operator, and I is the input image.

However, the feature length of Eq. (2) is too long, it has 38 bits and is not rotation invariant. Thus, the 38 bits is divided into 8 rows based on the filters shown in Fig. 1, for each row, a rotation invariant sub-texton is defined through one scale of filter(s). For the last two rows, as there is only 1-bit string, it is in nature rotation invariant. For the first six rows, the output is 6-bits binary string, a rotation invariant sub-texton designed based on the idea of “rotation invariant uniform” [25] is defined:

$$ST^j(x,y) = \begin{cases} \sum_{i=1}^6 s(F_{\theta_i}^j(x,y)) & \text{if } U(SP^j) \leq 2 \\ 7 & \text{otherwise} \end{cases}, \theta_i = \frac{i-1}{6} \pi \quad (4)$$

$$U(SP^j) = \left| s(F_{\theta_6}^j) - s(F_{\theta_1}^j) \right| + \sum_{i=2}^6 \left| s(F_{\theta_i}^j) - s(F_{\theta_{i-1}}^j) \right| \quad (5)$$

where j is the index of row, and $F_{\theta_i}^j(x,y)$ is the convolution output by the filter in the j^{th} row and i^{th} column as shown in Fig. 1. By definition, exactly 7 “rotation invariant uniform” binary patterns (“000000”, “000001”, “000011”, “000111”, “001111”, “011111”, “111111”) can occur in a circularly symmetric neighbor set of 6 binary bits, while the remaining (non-uniform) are grouped into a “miscellaneous” label (7). Thus, totally, there are 8 distinct output values for 6-bit binary string.

The filtering output at each position is a 8-dimensional vector, and there are 1,048,576 (8^8) kinds of patterns in total. Such a dimension is too large to build histogram and it will bring a computation issue. To reduce the feature size, we empirically divide the 38 filters into 2 groups as shown in Fig. 6. Thus for each image, only two 4-dimensional histograms need to be built and then the 2 histograms are concatenated. The final histogram size is reduced to 2,048 (8^8). Fig. 7 shows an

Figure 5. The top row shows 3 texture images. The central image patch (highlighted by red rectangle) is matched with an edge filter at all orientations. The magnitude (absolute value) of the filter response versus the orientation is plotted in the bottom row.
doi:10.1371/journal.pone.0088073.g005

example to illustrate the difference between Statistical_MR8 and Binary_MR8.

2. Binary_Fractal

In Statistical_Fractal, only selective fractal values are left. As shown in Section 3.1, the texture image is complex and may contain multiple orientations at a local region. Similar to Binary_MR8, all fractal values are binarized and kept for feature extraction in Binary_Fractal.

After getting 38 filtered images, $F_i (i = 1, 2, \dots, 38)$, by Statistical_MR8 filter banks, 38 images of local fractal dimensions (D_i) and 38 images of local fractal lengths (L_i) are gotten by Eq. (1) respectively. Unlike Binary_MR8, the local fractal values are all positive values, thus “0” which is used in Binary_MR8 could not

be used as the threshold to get binary results. Instead, median value of each fractal image is selected as the threshold to get binary output:

$$BD_i(x,y) = \begin{cases} 1, & \text{if } D_i(x,y) \geq \text{median}(D_i) \\ 0, & \text{otherwise} \end{cases}, i = 1, 2, \dots, 38 \quad (6)$$

$$BL_i(x,y) = \begin{cases} 1, & \text{if } L_i(x,y) \geq \text{median}(L_i) \\ 0, & \text{otherwise} \end{cases}, i = 1, 2, \dots, 38 \quad (7)$$

Figure 6. The Statistical_MR8 filter banks are divided equally into two groups.
doi:10.1371/journal.pone.0088073.g006

Figure 7. An illustration example of Statistical_MR8 and Binary_MR8.

doi:10.1371/journal.pone.0088073.g007

Where $median(D_i)$ and $median(L_i)$ are median value of the whole image of D_i and L_i respectively.

Then, similar to Binary_MR8, for each position, the 38 binary bits of $BD_i(x,y)$ and $BL_i(x,y)$ are divided into 8 rows respectively and for each row a rotation invariant sub-texton is defined. Finally, as discussed in Section 3.1, the 8 rows are divided into 2 groups to reduce feature size and a feature histogram with size 2,048 ($8 \times 8 \times 2 \times 2$) is built for Binary_Fractal_D and Binary_Fractal_L respectively.

3. Binary_Joint

Although Statistical_Joint could get good results on texture classification, it is too slow for some real time applications. Taking 7×7 patch based Statistical_Joint as an example, it is time consuming to search 49-dimension statistical textons especially for a large size image [19]. Inspired by the idea of LBP, a binary counterpart of local patch is proposed.

As 7×7 patch based Statistical_Joint could get good results in most of applications [18–19], the Binary_Joint is defined on a 7×7

patch only. First, taking a position (x, y) in the original image as the central point, a 7×7 rectangle is selected from the original image. 49 binary bits are computed:

$$B_{m,n} = \begin{cases} 1, & \text{if } G_{m,n} \geq G_{0,0}, m,n \in [-3,3] \text{ and } (m,n) \neq (0,0) \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

$$B_{0,0} = \begin{cases} 1, & \text{if } G_{0,0} \geq G_I \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

where $G_{m,n}$ is the gray value at coordinate (m, n) , and G_I is the average intensity of the whole image.

As 49 binary bits are too long for feature extraction, the 7×7 binary rectangle is divided into 6 blocks as shown in Fig. 8.

Then for each block, a sub-histogram is built. For blocks 1, 2, and 3, the 7 bits of the central column form a sub-texton:

Figure 8. A 7*7 patch is divided into 6 blocks.
doi:10.1371/journal.pone.0088073.g008

$$ST^c = \sum_{i=1}^7 B_i^c 2^{i-1} \quad (10)$$

Where B_i^c is the i^{th} bit of the central column. There are 128 possible values for 7 bits. To reduce the feature size, the idea of “uniform” from LBP [25] is used to reduce the number of sub-textons. A sub-texton is labeled to “non-uniform”, if U' , the number of bitwise 0/1 changes, is bigger than 2. U' is defined as:

$$U' = |B_1^c - B_7^c| + \sum_{i=2}^7 |B_i^c - B_{i-1}^c| \quad (11)$$

Thus, the total number of possible sub-textons of the central column is 45. Table 1 lists the 45 kinds of sub-textons with their labels.

For the left and right columns, the number of 1-bits is compared with the central column. There are three possible values.

$$ST^l = \begin{cases} 0, & \text{if } \sum_{j=1}^7 B_j^l < \sum_{j=1}^7 B_j^c \\ 1, & \text{if } \sum_{j=1}^7 B_j^l = \sum_{j=1}^7 B_j^c \\ 2, & \text{if } \sum_{j=1}^7 B_j^l > \sum_{j=1}^7 B_j^c \end{cases} \quad (12)$$

$$ST^r = \begin{cases} 0, & \text{if } \sum_{j=1}^7 B_j^r < \sum_{j=1}^7 B_j^c \\ 1, & \text{if } \sum_{j=1}^7 B_j^r = \sum_{j=1}^7 B_j^c \\ 2, & \text{if } \sum_{j=1}^7 B_j^r > \sum_{j=1}^7 B_j^c \end{cases} \quad (13)$$

Where B_j^l and B_j^r is the j^{th} bit of the right and left columns, respectively. A sub-histogram based on $[ST^r, ST^c, ST^l]$ is built

Table 1. 45 kinds of sub-textons with 7 bits and their labels.

"0000001" (Label: 0)	"0000010" (Label: 1)	"0000100" (Label: 2)	"0001000" (Label: 3)	"0010000" (Label: 4)	"0100000" (Label: 5)	"1000000" (Label: 6)
"0000011" (Label: 7)	"0000110" (Label: 8)	"0001100" (Label: 9)	"0011000" (Label: 10)	"0110000" (Label: 11)	"1100000" (Label: 12)	"1000001" (Label: 13)
"0000111" (Label: 14)	"0001110" (Label: 15)	"0011100" (Label: 16)	"0111000" (Label: 17)	"1110000" (Label: 18)	"1100001" (Label: 19)	"1000011" (Label: 20)
"0001111" (Label: 21)	"0011110" (Label: 22)	"0111100" (Label: 23)	"1111000" (Label: 24)	"1110001" (Label: 25)	"1100011" (Label: 26)	"1000111" (Label: 27)
"0011111" (Label: 28)	"0111110" (Label: 29)	"1111100" (Label: 30)	"1111001" (Label: 31)	"1110011" (Label: 32)	"1100111" (Label: 33)	"1001111" (Label: 34)
"0111111" (Label: 35)	"1111110" (Label: 36)	"1111101" (Label: 37)	"1111011" (Label: 38)	"1110111" (Label: 39)	"1101111" (Label: 40)	"1011111" (Label: 41)
"0000000" (Label: 41)	"1111111" (Label: 43)	"non-uniform" (Label: 44)				

doi:10.1371/journal.pone.0088073.t001

Figure 9. An illustration example of Statistical_Joint and Binary_Joint.
doi:10.1371/journal.pone.0088073.g009

for the whole image. Thus, for blocks 1, 2, and 3, the feature size of a sub-histogram is 405 ($45 \times 3 \times 3$).

Similarly, for blocks 4, 5, and 6, a sub-texton is based on the central row, and the number of 1-bit is compared with the up and down rows. A sub-histogram with size of 405 is extracted for each block. Finally, six sub-histograms are concatenated and a feature histogram with 2,430 (405×6) bins is extracted for each image. Fig. 9 shows an example to illustrate the difference between Statistical_Joint and Binary_Joint.

4. Dissimilarity Metric

The dissimilarity of sample and model histograms is a test of goodness-of-fit, which could be measured with a nonparametric statistic test. There are many metrics to evaluate the goodness between two histograms, such as histogram intersection, log-likelihood ratio, and chi-square statistic [25]. In this study, a test sample T is assigned to the class of model L that minimizes the chi-square distance:

$$D(T, L) = \sum_{n=1}^N \frac{(T_n - L_n)^2}{T_n + L_n} \quad (14)$$

where N is the number of bins, and T_n and L_n are the values of the sample and model image at the n^{th} bin, respectively. In this paper, the nearest neighborhood classifier with chi-square distance is used to measure the dissimilarity between two histograms, because it is

equivalent to the optimal Bayesian classification [23], and good performance for texture classification can be achieved [24].

Experimental Results and Discussion

To evaluate the effectiveness of the proposed methods, we carried out a series of experiments on three large and comprehensive texture databases: the Columbia-Utrecht Reflection and Texture (CURET) database, which contains 61 classes of real-world textures, each imaged under different combinations of illumination and viewing angle [27], University of Illinois at Urbana-Champaign (UIUC) database [14], which includes 25 classes and 40 images per class collected under significant viewpoint variations, and Kungliga Tekniska högskolan (Swedish)-Textures under varying Illumination, Pose and Scale (KTH-TIPS) database [28], which include contains 10 classes and 81 images per class imaged under different scales, different poses and different illumination conditions.

Except Binary_Fractal, the image sample was normalized to have an average intensity of 0 and a standard deviation of 1 [16–20] for different methods. This is to remove global intensity and contrast [16–20]. For Binary_Fractal, the image sample was normalized to have an average intensity of 128 and a standard deviation of 20 as this setting could get better accuracy. 7×7 local patch is used for Statistical_Fractal and Binary_Joint. Although large size patch could get better recognition accuracy, it is more time consuming [19] and the main focus of this work is to investigate the effect of binary representation. For comparison, the

Figure 10. Textures from the Columbia-Utrecht database. In this work, all images are converted to monochrome, so colour is not used to discriminate between different textures.
doi:10.1371/journal.pone.0088073.g010

Table 2. Classification rate (% , mean $\pm$ standard deviation) for CURET database.

Method	Feature Size	L			
		46	23	12	6
$LBP^{riu2}_{8,1} + LBP^{riu2}_{16,3} + LBP^{riu2}_{24,5}$	54	95.84 $\pm$ 0.82	91.96 $\pm$ 1.39	86.41 $\pm$ 2.05	78.09 $\pm$ 3.33
Statistical_MR8	2440	97.79 $\pm$ 0.67	95.03 $\pm$ 1.27	90.48 $\pm$ 1.98	82.90 $\pm$ 3.44
Binary_MR8	2048	98.23 $\pm$ 0.58	95.67 $\pm$ 1.06	91.11 $\pm$ 2.08	83.27 $\pm$ 3.75
Statistical_Joint	2440	97.66 $\pm$ 0.67	94.58 $\pm$ 1.34	89.40 $\pm$ 2.39	81.06 $\pm$ 3.73
Binary_Joint	2430	96.99 $\pm$ 0.65	93.77 $\pm$ 1.18	88.49 $\pm$ 2.15	80.08 $\pm$ 3.53
Statistical_Fractal_D	2440	96.12 $\pm$ 0.37*	92.50 $\pm$ 0.51*	86.70 $\pm$ 0.72*	78.05 $\pm$ 0.97*
Statistical_Fractal_L	2440	97.50 $\pm$ 0.30*	94.69 $\pm$ 0.45*	89.74 $\pm$ 0.66*	81.67 $\pm$ 0.96*
Binary_Fractal_D	2048	96.20 $\pm$ 0.91	92.69 $\pm$ 1.39	87.28 $\pm$ 2.30	78.98 $\pm$ 3.31
Binary_Fractal_L	2048	97.80 $\pm$ 0.65	95.14 $\pm$ 1.24	90.57 $\pm$ 1.98	82.86 $\pm$ 3.45

*Classification rates are originally reported in [20]. Other classification rates are computed by us.
doi:10.1371/journal.pone.0088073.t002

Table 3. Classification rate (% , mean $\pm$ standard deviation) for Statistical_MR8 and Statistical_Joint on CURET database with different training conditions.

Method	L			
	46	23	12	6
Statistical_MR8($C=61$)	97.79 $\pm$ 0.67	95.03 $\pm$ 1.27	90.48 $\pm$ 1.98	82.90 $\pm$ 3.44
Statistical_MR8($C=20$)	96.65 $\pm$ 0.79	93.45 $\pm$ 1.41	88.40 $\pm$ 2.12	80.35 $\pm$ 3.51
Statistical_MR8($C=10$)	95.82 $\pm$ 0.87	92.32 $\pm$ 1.33	86.92 $\pm$ 2.21	78.65 $\pm$ 3.50
Statistical_Joint($C=61$)	97.66 $\pm$ 0.67	94.58 $\pm$ 1.34	89.40 $\pm$ 2.39	81.06 $\pm$ 3.73
Statistical_Joint($C=20$)	97.21 $\pm$ 0.73	93.83 $\pm$ 1.44	88.45 $\pm$ 2.42	79.83 $\pm$ 3.81
Statistical_Joint($C=10$)	96.87 $\pm$ 0.74	93.41 $\pm$ 1.48	88.00 $\pm$ 2.31	79.34 $\pm$ 3.86

Classification rates are computed by us.
doi:10.1371/journal.pone.0088073.t003

typical binary feature, LBP, is compared with Statistical_MR8, Statistical_Fractal, Statistical_Joint and the proposed methods. In the following experiments, for LBP, each texture sample was normalized to have an average 128 and a standard of 20 [25]. To get better results, multiscale scheme is used for the LBP method [25]. The Chi-square dissimilarity defined in Section 3.4 and the nearest neighborhood classifier were used for all methods here.

1. Experimental Results on CURET Database

The CURET database contains 61 textures, as shown in Fig. 10, and there are 205 images of each texture acquired at different viewpoints and illumination orientations. There are 118 images which have been shot from a viewing angle of less than 60°. Of these 118 images, we selected 92 images, from which a sufficiently large region could be cropped (200*200) across all texture classes [17]. To get statistically significant experimental results [19], L training images are randomly chosen from each class while the remaining 92- L images are used as the test set. The first 23 images of each class are used to learn the library and 40 textons are clustered from each of the texture classes. The average accuracy and standard deviation over 1000 random splits are listed in Table 2.

Two findings could be found in Table 2. First, the proposed methods are much better than simple LBP method. However, the feature length is a little long. Fortunately, it is not a big issue for nowadays computer.

Second, binary operators show their superiority over statistical operators on CURET database. For example, Binary_MR8 and Binary_Fractal could get better results than Statistical_MR8 and

Statistical_Fractal. While, Binary_Joint could get competitive results with Statistical_Joint.

Furthermore, as there is a training step in statistical operators, the performance will be related with the training set. Table 3 shows the classification rate of Statistical_MR8 and Statistical_Joint under different training samples. Here C is the number of classes of training samples to learn the texton dictionary. As shown in Table 3, when the training set is reduced, the performance will be degraded.

Compared with statistical texton, binary texton is sensitive to noise. A small disturbance may output the same statistical textons, but the binary textons will be different. Fig. 11 shows an example, in a 5*5 local patch, a pixel value is changed from 0.51 to 0.49, while other pixels are the same. As the difference is very small, the statistical texton may be the same, however, as illustrated in Section 3.3, the binary texton will be significantly different.

To further illustrate noise effect, we created three texture databases with noise:

$$\hat{I}(x,y) = I(x,y) + t * n(x,y) \quad (15)$$

where $I(x,y)$ is the gray value of original image pixel and $\hat{I}(x,y)$ is the transformed value. $n(x,y)$ is random noise with 0 mean and 1 standard. t ($t=1,2,3$) is the parameter to control the degree of noise. Table 4 shows the classification rate for different datasets. As shown in Table 4, statistical texton method could get sound results even when $t=3$, while Binary_Joint fails to over 30% at the same

Figure 11. Two 5*5 local patches. These two patches may generate the same statistical textons, but different binary textons.
doi:10.1371/journal.pone.0088073.g011

Table 4. Classification rate (% , mean $\pm$ standard deviation) for noise sets.

Method	L			
	46	23	12	6
Statistical_Joint($t=1$)	96.87 $\pm$ 0.77	93.20 $\pm$ 1.43	87.52 $\pm$ 2.47	78.71 $\pm$ 3.77
Statistical_Joint($t=2$)	91.74 $\pm$ 1.00	87.28 $\pm$ 1.49	81.11 $\pm$ 2.39	72.27 $\pm$ 3.61
Statistical_Joint($t=3$)	71.99 $\pm$ 1.07	68.03 $\pm$ 1.25	62.95 $\pm$ 1.68	56.34 $\pm$ 2.59
Binary_Joint($t=1$)	89.78 $\pm$ 1.03	85.02 $\pm$ 1.44	78.87 $\pm$ 2.25	70.38 $\pm$ 3.46
Binary_Joint($t=2$)	50.71 $\pm$ 0.89	48.04 $\pm$ 0.91	44.98 $\pm$ 1.28	41.14 $\pm$ 1.86
Binary_Joint($t=3$)	25.04 $\pm$ 0.57	23.71 $\pm$ 0.59	22.41 $\pm$ 0.68	20.87 $\pm$ 0.86

Classification rates are computed by us.
doi:10.1371/journal.pone.0088073.t004

Figure 12. Samples of the 25 textures in UIUC database.

doi:10.1371/journal.pone.0088073.g012

Table 5. Classification rate (% , mean±standard deviation) for UIUC database.

Method	Feature Size	L			
		20	15	10	5
$LBP_{8,1}^{riu2} + LBP_{16,3}^{riu2} + LBP_{24,5}^{riu2}$	54	76.83±1.91	72.87±1.88	66.68±1.91	55.26±2.11
Statistical_MR8	2440	92.94±1.06 [^]	91.16±1.11 [^]	88.29±1.32 [^]	81.12±1.74 [^]
Binary_MR8	2048	89.33±1.46	86.53±1.48	81.95±1.72	71.96±2.09
Statistical_Joint	2440	83.54±1.76	80.17±1.81	74.72±1.76	64.26±2.18
Binary_Joint	2430	67.93±2.26	63.04±2.02	56.19±1.97	44.76±1.93
Statistical_Fractal_D	2440	95.40±0.92*	94.09±0.98*	91.64±1.18*	85.35±1.69*
Statistical_Fractal_L	2440	94.96±0.91*	93.66±0.96*	91.25±1.13*	84.96±1.66*
Binary_Fractal_D	2048	85.03±1.63	81.85±1.62	76.56±1.72	65.60±2.13
Binary_Fractal_L	2048	89.17±1.19	86.58±1.32	82.14±1.59	72.25±2.00

*Classification rates are originally reported in [20].

Classification rates are originally reported in [19]. Other classification rates are computed by us.

doi:10.1371/journal.pone.0088073.t005

Table 6. Classification rate (% , mean±standard deviation) for different resolution.

Method	L			
	20	15	10	5
Statistical_Joint (1/2 Image)	81.47±1.66	78.16±1.72	72.64±1.74	62.05±2.02
Statistical_Joint (1/4 Image)	79.32±1.73	75.76±1.64	70.69±1.71	61.14±1.94
Binary_Joint (1/2 Image)	70.40±2.11	65.75±2.04	59.08±1.98	47.99±2.19
Binary_Joint (1/4 Image)	71.74±1.96	67.72±1.97	61.43±1.98	50.44±2.11

Classification rates are computed by us.

doi:10.1371/journal.pone.0088073.t006

condition. So a preprocessing technique is necessary for binary operators if the image contains severe noise.

2. Experimental Results on UIUC Database

The UIUC texture database [14] includes 25 classes and 40 images in each class. The resolution of each image is 640*480. The database contains materials imaged under significant viewpoint variations as shown in Fig. 12. Similar to Section 4.1, L training images are randomly chosen from each class while the remaining 40- L images are used as the test set. The first 10 images of each class are used to learn the library and 100 textons are clustered from each of the texture classes. The average accuracy and standard deviation over 1000 random splits are listed in Table 5.

Several findings could be found in Table 3. First, similar to what we found in Table 2, the proposed methods are much better than simple binary operators, LBP.

Figure 13. Samples of the 10 textures in KTH-TIPS database.

doi:10.1371/journal.pone.0088073.g013

Table 7. Classification rate (% , mean±standard deviation) for KTH-TIPS database.

Method	Feature Size	L			
		20	15	10	5
$LBP_{8,1}^{riu2} + LBP_{16,3}^{riu2} + LBP_{24,5}^{riu2}$	54	95.17±2.22	88.90±3.30	79.18±4.00	67.29±4.97
Statistical_MR8	2500	96.29±1.60	90.46±2.91	81.95±3.67	71.25±5.31
Binary_MR8	2048	96.21±1.28	91.37±2.03	83.72±3.14	73.08±4.91
Statistical_Joint	2500	98.02±1.21	94.40±2.25	88.15±2.73	79.12±4.45
Binary_Joint	2430	95.84±1.82	89.88±2.57	81.05±3.10	70.78±3.99
Statistical_Fractal_D	2500	93.23±1.79	88.66±2.61	82.05±3.40	74.47±4.15
Statistical_Fractal_L	2500	90.49±2.20	83.88±2.99	74.54±3.68	63.32±4.93
Binary_Fractal_D	2048	95.59±1.88	89.57±3.36	81.09±3.74	70.10±4.66
Binary_Fractal_L	2048	94.57±1.89	88.16±3.05	79.07±3.60	68.20±4.71

Classification rates are computed by us.
doi:10.1371/journal.pone.0088073.t007

Second, the proposed scheme fails to get better results than statistical texton methods. This is possibly because of the high resolution. As the image resolution is much higher, statistical textons could describe subtle differences. For example, in UIUC database, there is 4.71% possibility that two different statistical textons have an identical binary texton. While in CURET database, the possibility is only 2.22%. To further illustrate the resolution effect for statistical and binary operators, we manually down-sampled every image of UIUC database to 1/2 and 1/4 of

its original size and applied Statistical_Joint and Binary_Joint methods on these two databases. As shown in Table 6, because statistical textons could describe subtle differences, the classification rate is lower when the image is down-sampled, on the contrary binary texton may get better result. Thus, statistical texton methods are more favorable for high resolution images.

3. Experimental Results on KTH-TIPS Database

The KTH-TIPS texture database [28] contains 10 kinds of materials. Images were taken at 9 different scales spanning two octaves. At each scale, 9 images were taken in a combination of three poses and three illumination conditions. Thus, for each class, there are 81 image samples. A 200*200 patch is manually cropped from each sample. However, for some images, due to large camera-target distances, some samples are smaller than 200*200 pixels. Fig. 13 shows image samples of KTH-TIPS database. Similar to Section 4.1, L training images are randomly chosen from each class while the remaining 40- L images are used as the test set. The first 20 images of each class are used to learn the library and 250 textons are clustered from each of the texture classes. The average accuracy and standard deviation over 1000 random splits are listed in Table 7.

Similar findings could be found in Table 7. First, the proposed methods are much better than simple LBP method. Second, as the image resolution is not high, binary texton methods could still get good performance. Binary_MR8 and Binary_Fractal could get

Table 8. Average time (seconds) cost on feature extraction for different databases.

Method	Database		
	CURET	UIUC	KTH-TIPS
Statistical_MR8	2.3	19.1	1.7
Binary_MR8	1.1	9.0	1.1
Statistical_Joint	26.1	191.1	18.6
Binary_Joint	0.5	4.0	0.5
Statistical_Fractal	5.3	25.3	1.9
Binary_Fractal	1.6	16.1	1.6

doi:10.1371/journal.pone.0088073.t008

better results than Statistical_MR8 and Statistical_Fractal, respectively. But, Binary_Joint is a little worse than Statistical_Joint.

4. Time Cost

The proposed methods and statistical textons are implemented using Matlab R2008a on a windows XP, T6400 CPU (2.13 GHz) and 2 GB Ram PC. As the feature length are similar for each method and the classifier is the same, we only list the average feature extraction time on different databases. As shown in Table 8, Statistical_Joint method is the most time-consuming method while binary methods are much faster than statistical methods.

Conclusion

In this paper, we proposed three binary texton methods and reported their experimental results with their statistical counterpart on three large public texture databases. We empirically found that statistical method could get good results for most cases. However, it may be time consuming for feature extraction,

especially when the image size is big. Furthermore, it requires a training step to build the texton dictionary which may limit the accuracy when the training sample is not enough. For good quality images with small image size, binary texton methods could get better results than statistical ones. And it does not require training step and is fast for feature extraction. As different schemes have different advantages, future work should investigate how to utilize these properties and improve the classification rate further.

Acknowledgments

The authors sincerely thank MVG and VGG for sharing the source codes of LBP and Statistical_MR8.

Author Contributions

Conceived and designed the experiments: ZG ZZ QL. Performed the experiments: ZG ZZ QL. Analyzed the data: ZG ZZ XL QL JY. Contributed reagents/materials/analysis tools: XL JY. Wrote the paper: ZG ZZ XL QL JY.

References

1. Tuceryan M, Jain AK (1993) Texture analysis, Handbook of pattern recognition and computer vision. C.H. . Chen, L.F. . Pau, and P.S.P. . Wang eds. Chapter 2, pp. 235–276.
2. Cohen FS, Fan Z, Attali S (1991) Automated Inspection of Textile Fabrics Using Textural Models. IEEE Transactions on Pattern Analysis and Machine Intelligence 13(8), pp. 803–808.
3. Anys H, He DC (1995) Evaluation of textural and multipolarization radar features for crop classification. IEEE Transactions on Geoscience and Remote Sensing 33(5), pp. 1170–1181.
4. Ji Q, Engel J, Craine E (2000) Texture analysis for classification of cervix lesions. IEEE Transactions on Medical Imaging 19(11), pp. 1144–1149.
5. Haralik RM, Shanmugam K, Dinstein I (1973) Texture features for image classification. IEEE Transactions on Systems, Man, and Cybernetics 3(6), pp. 610–621.
6. Randen T, Husy JH (1999) Filtering for texture classification: a comparative study. IEEE Transactions on Pattern Analysis and Machine Intelligence 21(4), pp. 291–310.
7. Kashyap RL, Khotanzad A (1986) A model-based method for rotation invariant texture classification. IEEE Trans. on Pattern Analysis and Machine Intelligence 8(4), 472–481.
8. Mao J, Jain AK (1992) Texture classification and segmentation using multiresolution simultaneous autoregressive models. Pattern Recognition 25(2), 173–188.
9. Wu WR, Wei SC (1996) Rotation and gray-scale transform-invariant texture classification using spiral resampling, subband decomposition, and hidden Markov model. IEEE Transactions on Image Processing 5(10), 1423–1434.
10. Deng H, Clausi DA (2004) Gaussian MRF rotation-invariant features for image classification. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(7), 951–955.
11. Xu Y, Ji H, Fermuller C (2009) Viewpoint invariant texture description using fractal analysis. International Journal of Computer Vision 83(1), pp. 85–100.
12. Mellor M, Hong BW, Brady M (2008) Locally rotation, contrast, and scale invariant descriptor for texture analysis. IEEE Trans. on Pattern Analysis and Machine Intelligence 30(1), pp. 52–61.
13. Zhang J, Marszalek M, Lazebnik S, Schmid C (2007) Local features and kernels for classification of texture and object categories: a comprehensive study. International Journal of Computer Vision 73(2), pp. 213–238.
14. Lazebnik S, Schmid C, Ponce J (2005) A sparse texture representation using local affine regions. IEEE Trans. on Pattern Analysis and Machine Intelligence 27(8), pp. 1265–1278.
15. Rubner Y, Tomasi C, Guibas L (2000) The Earth mover's distance as a metric for image retrieval. International Journal of Computer Vision 40(2), pp. 99–121.
16. Varma M, Zisserman A (2002) Classifying Images of Materials: Achieving Viewpoint and Illumination Independence. European Conference on Computer Vision, pp. 255–271.
17. Varma M, Zisserman A (2005) A statistical approach to texture classification from single images. International Journal of Computer Vision 62(1–2), pp. 61–81.
18. Varma M, Zisserman A (2003) Texture classification, are filter banks necessary? Conference of Computer Vision and Pattern Recognition, pp. 691–698.
19. Varma M, Zisserman A (2009) A statistical approach to material classification using image patch exemplars. IEEE Trans. on Pattern Analysis and Machine Intelligence 31(11), pp. 2032–2047.
20. Varma M, Garg R (2007) Locally invariant fractal features for statistical texture classification. International Conference on Computer Vision, pp. 1–8.
21. Cula OG, Dana KJ (2001) Compact Representation of Bidirectional Texture Functions. Conference of Computer Vision and Pattern Recognition, pp. 1041–1047.
22. Leung T, Malik J (2001) Recognizing Surfaces Using Three-Dimensional Textons. International Conference of Computer Vision 43(1), pp. 29–44.
23. Varma M, Zisserman A (2004) Unifying statistical texture classification frameworks. Image and Vision Computing 22(14), pp. 1175–1183.
24. Rubner Y, Puzicha J, Tomasi C, Buhmann JM (2001) Empirical evaluation of dissimilarity measures for color and texture. Computer Vision and Image Understanding 84(1), pp. 25–43.
25. Ojala T, Pietikäinen M, Mäenpää TT (2002) Multiresolution gray-scale and rotation invariant texture classification with local binary pattern. IEEE Trans. Pattern Anal. Mach. Intell. 24(7), pp. 971–987.
26. Guo Z, Zhang L, Zhang D (2009) Rotation invariant texture classification using binary filter response pattern (BFRP). Conference of Computer Analysis of Images and Patterns, pp. 1130–1137.
27. Dana KJ, Ginneken Bvan, Nayar SK, Koenderink JJ (1999) Reflectance and texture of real world surfaces. ACM Trans. Graph. 18(1), pp. 1–34.
28. Hayman E, Caputo B, Fritz M, Eklundh JO (2004) On the significance of real-world conditions for material classification. European Conference on Computer Vision, pp. 253–266.
29. Duda RO, Hart PE, Stork DG (2001) Pattern Classification. John Wiley and Sons, 2nd edition.
30. Julesz B (1981) Textons, the Element of Texture Perception, and their Interactions. Nature 290, pp. 91–97.