Face Retrieval Based On Local Binary Pattern and Its Variants: A Comprehensive Study

Phan Khoi, Lam Huu Thien, Vo Hoai Viet
Department of Computer Vision and Robotics, University of Science, VNU-HCM, Viet Nam

***Abstract*—Face retrieval (FR) is one of the specific fields in content-based image retrieval (CBIR). Its aim is to search relevant faces in large database based on the contents of the images rather than the metadata. It has many applications in important areas such as face searching, forensics, and identification… In this paper, we experimentally evaluate Face Retrieval based on Local Binary Pattern (LBP) and its variants: Rotation Invariant Local Binary Pattern (RILBP) and Pyramid of Local Binary Pattern (PLBP). We also use a grid LBP based operator, which divides an image into 6 × 7 sub-regions then concentrates LBP feature vector from each of them into a spatially enhanced feature histogram. These features were firstly tested on three fontal face datasets: The Database of Faces (TDF), Caltech Faces 1999 (CF1999) and the combination of The Database of faces and Caltech Faces 1999 (CF). Good result on these dataset has encouraged us to conduct tests on Labeled Faces in the Wild (LFW), where the images were taken from real-world condition. Mean average precision (MAP) was used for measuring the performance of the system. We carry out the experiments in two main stages indexing and searching with the use of k-fold cross-validation. We further boost the system by using Locality Sensitive Hashing (LSH). Furthermore, we also evaluate the impact of LSH on the searching stage. The experimental results have shown that LSH is effective for face searching as well as LBP is robust feature in fontal face retrieval.**

Keywords—Face Retrieval; LBP; PLBP; Grid LBP; LSH

I. Introduction

The principle of face retrieval problem is the comparison of faces so that those which higher “likelihood” will near the top of result list. In practice, these “likelihood” parameters are presented by a number, which is calculated by a distance function. Next problem is to quantify the faces in the way that a computer can understand. A face can be seen as a combination of smaller part such as eyes, nose, and lips. From that assumption, our approach falls into global features because they can manifest the relative position of these parts in a face or else the global shape. Furthermore, LBP feature can characterized local object appearance rather well. Finally, we choose Euclidean distance to compare LBP feature vector due to the simplification in implement and its low computation cost.

Despite having applications in important areas (face finding, medical, forensics, etc.), retrieving images of faces with high performance in real-world condition is a difficult task due to technical and practical challenges. Beside semantic gap, there are challenges that we should consider when designing a face retrieval system. The first challenge is measuring similarity among images. This was done by compare their feature vector.

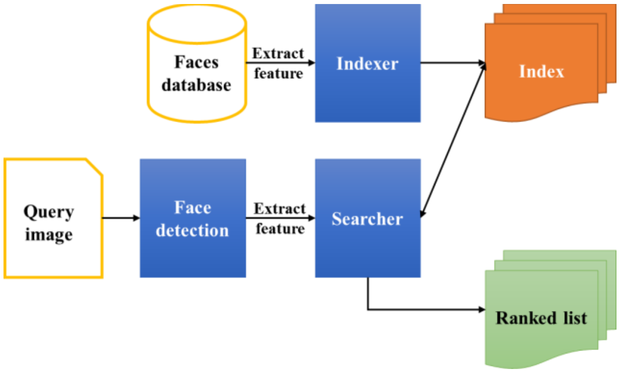

Fig. 1. Flowchart of a face retrieval system

A good method will increase the performance of system while maintaining the computational time. The next challenge is about how good is your system work in real life. It is difficult to design the system that work well in any conditions, due to broad domains (faces of many people in database), high variance of content (images captured in daylight/night, bad weather/good weather, indoor/outdoor…), capture setting (high/low brightness, HDR, etc.), among many others. Finally, understanding of user needs and intentions is the key. This involved in user interface as well as accuracy of the system. For example, face retrieval applications in forensics which required high accuracy and fast search time to find criminal in very large database is different with the applications for finding some famous people look like you which easy to use and do not require a good accuracy.

A face retrieval system consist of indexing, organizing, annotating, and retrieving visual information about faces. Its goal is to retrieve desired face images from the given information (a face image). In this test, we concern the effect of LBP and its extension on indexing and retrieving to find out if good features can increase the system performance. In Figure 1, we introduced a face retrieval system that focus on the important of feature extraction and indexing process. They are the key issues due to the role of index in FR system. Feature extraction decides what feature(s) to be save in the index while the indexing process influent the way of searcher works and the execution time of searching process. In our tests, we used pre-cropped face images so there is no use of face detection module. In application view, this module is needed for extracting face from user provided image. We can easily build it by using existing function from OpenCV library.

The rest of the paper is organized as follows. Firstly, we present a face retrieval system designed to work with global features. This system focuses mainly on the indexing process because it decides how the searching process occurred. In addition, time in FR is an important factor so the way index was built wills greatly influent searching time. Secondly, we empirically evaluate LBP and its variants for face retrieval on several dataset that focus on fontal face. Furthermore, we also test our approach on the Labeled Faces in the Wild (LFW) dataset has some changes, for example, difference in facial detail or facial expression… Results from these dataset help understanding the effect of dataset properties on the system. LFW puts the face retrieval system in real-world condition by using image collected around the web. Testing on this dataset will determine the practical applicability of tested features. Thirdly, we boost the system by using LSH. We also conducted tests to prove its ability to the traditional nearest neighbor method. The tests include indexing time, searching time, and searching time on different size of dataset.

The rest of this paper is structured as follows. In section II, we present related works. The original LBP and its variants are introduced in section III. We discuss the indexing process of FR in section IV. Experimental result and discussion are showed in section V. Finally, section VI draws conclusions of our work and indicates future studies.

II. Related Works

A. Content-based image retrieval

The origin of the term “Content-based image retrieval” was mentioned in [1] that Kato (1992) used it to describe his tests in automatic retrieval of images from a database using color and shape feature. Since then, the term has been widely used to define the process of retrieving images from a large collection using features that can be automatically extracted from the images. These features can be primitive or semantic but the extraction process must be predominantly automatic.

The vital difference between CBIR and classical information retrieval (IR) is the image databases differ from text databases. A digital image can be seen as an array of pixel intensities which we cannot directly understand while text (collection of words stored as ASCII strings) is logically organized by the author. When the search process occur on the databases, IR searcher use metadata or full-text in the index and CBIR searcher use a distance function to measure the similarity of the query image’s feature(s) among images’ feature(s) in the databases. Therefore, searching images by keywords from manually assigned is not CBIR.

There are keys issues that was mention in [1] as the important needs when designing a CBIR system:

- Understanding of user’s needs and intention.
- Choosing feature(s), which balance in computing cost and retrieval performance.
- Providing compact storage for bulk images as well as efficient ways to access, update, and delete them.
- Designing a good method that reflects human similarity judgments to match query and stored images.

CBIR uses many methods from the field of image processing and computed vision since it involved these fields in feature extraction, preprocessing images (resizing, adjusting contrast, cropping…)… We can think of CBIR as a combination of databases with these methods in a way that created a working system for searching images based on theirs content.

B. Face retrieval

We can simply regard a face retrieval system as a CBIR system that works with raw pixel values from face images. In 1987, a face retrieval application, FRAME – Face Retrieval and Matching Equipment, was introduced to help the police in their investigation [2]. In the early form of FR presented in [3], the indexing stage is not fully automatic due to the lack of good face component detection. So there is a module that allows user to add addition information and modify the result of face’s components detection. There also some limitations, for example, the way of features extraction was complicated and computation cost, the system was still depend on supplement information as text to increase retrieval performance… We can stage that this is not truly a CBIR system but this work has pointed out the important parts of a face retrieval: feature extraction, way of storing and accessing images in the database, a distance function to measure image similarity, how we use an image as a query to search on the database.

Later in 1994, MIT’s Media Laboratory presented a famous FR system that its core is the eigenfaces database [4 – 5]. The special thing about this system is it was trained with large face images database to compute 20 features (called eigenfeatures). These features can characterize any human face in higher level of information which more robust and brought better precision than raw pixel values. The drawbacks of this method are the system needs training before further process, and it cannot completely automated for broad domains of images [6].

C. Local Binary Pattern

LBP was first introduce in [7] as a powerful feature for texture classification. It was later use for solving face recognition [8 – 10] due to its ability to represent face image [10 – 12]. It also used for facial expressions recognition [13], gender classification [14], human detection [15], etc. It also combined with block-based method for image retrieval [16]. Many works were done to extend its invariant against rotation [17] and scale [18].

III. LBP and Its Variants

In this section, we will brief introduction about the original LBP and its extensions: Rotation Invariant Local Binary Pattern, Pyramid of LBP and Grid LBP. The content of each sub-section included the idea behind, the process of calculating histogram for each version.

A. Local Binary Pattern

A LBP operator considered each pixel has a code (called Local Binary Patterns codes or LBP codes) which was calculated by thresholding its 3×3 neighborhood with the center value (Figure 2). As a consequence, this make LBP invariant against gray-scale. It also presented local primitives such as curve edges, spots, corner and so on [11].

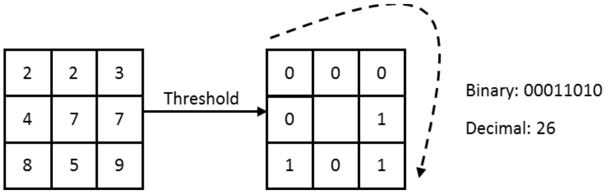

Fig. 2. A basic LBP operator

In Figure 3, the $3x3$ neighborhood was expanded to capture dominant feature with large-scale structures. The neighborhood can be denoted by a pair (P, R) where P is the sampling points on a circle of radius of R. Therefore, there are 2^p different output values.

Oajala's study [7] has proved that some bins in LBP histogram contain more information than others. These bins is called uniform patterns. A LBP code must contain at most two bitwise transitions from zero to one or vice versa when the binary string is considered circular (Figure 5). It was shown that $(8, 1)$ neighborhood accounting nearly 90% of all patterns and about 70% for in the $(16, 2)$ neighborhood.

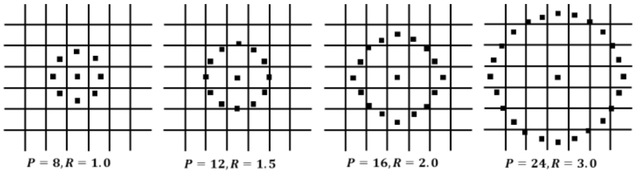

Fig. 3. Extended LBP operator compare to the basic one

A histogram of an image labeled by LBP operator can be detail as follows:

$$H_i = \sum_{x,y} I(f_l(x, y) = i), \ \ i = 0, \dots, n-1 \tag{1}$$

here n is the number of total different output generated by the LBP operator as follows:

$$I(A) = \begin{cases} 1 \ \ if\ A\ is\ true \\ 0\ if\ A\ is\ false \end{cases} \tag{2}$$

This LBP histogram is statistical since each local micro-patterns in Figure 3 have their own LBP code and creating a histogram over the whole image will presented their distribution. Therefore, it can describe image characteristics.

B. Rotation Invariant Local Binary Pattern

In Figure 4, we can see that an edge pattern has LBP code of 26 but if we rotate this pattern 90° clockwise its LBP code is 7. Therefore, rotation has distinct these pattern into different bins.

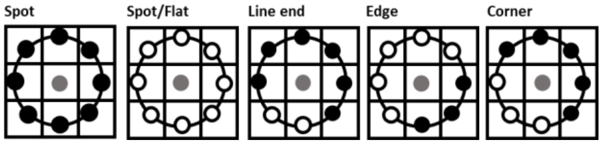

Fig. 4. Examples of texture primitives presented by LBP (white circles mean zeros and the black circles mean ones)

In generality, if we compare an image with its rotated one, computer will consider these two images different base on their LBP histogram. To extended LBP ability against rotation, Ojala et al [17] has provided a simple yet effective operator call LBP_8^{ri36}. As in Figure 6, this operator use (8,1) neighborhood to generate 36 different values that represent 36 unique rotation invariant LBP patterns. We can obtain these codes easily by rotating LBP code:

$$LBP_{8,1}^{ri} = \min\{ROR(LBP_8, i) | i = 0,1, \dots, 7\} \tag{3}$$

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

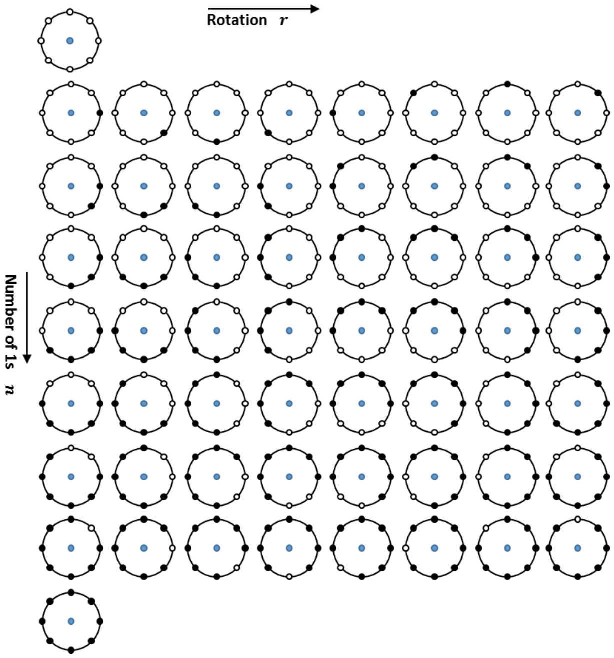

Fig. 5. All uniform pattern of $LBP_{8,1}$ operator

Also in [17] mention this operator is not good at discrimination due to the distraction of some patterns to the analysis and crude quantization of the angular space at 45° intervals. To overcome this problem, 36 patterns were reduced to 9 patterns which were numbered in Figure 6. The process of calculating these patterns is denoted by:

$$LBP_{8,1}^{rui2} = \begin{cases} \sum_{i=1}^{8} s(g_i - \ g_0)\ if\ U(LBP_{8,1}) \leq 2 \\ \quad 9 \qquad\qquad\qquad otherwise \end{cases} \tag{4}$$

where:

- $s(x) = \begin{cases} 1, \ x \geq 0 \\ 0, \ x \leq 0 \end{cases}$, g_0 is the center pixel and g_i is its 3×3 neighborhood.
- $U(c)$ is a function that calculate number of spatial transitions in LBP$_{8,1}$ binary code which was denoted by:

$$U\left(LBP_{P,R}\right) = \left|s(g_{P-1} - g_c) - s(g_0 - g_c)\right| + \sum_{p=1}^{P-1}\left|s\left(g_p - g_c\right) - g_c\right| \tag{5}$$

There is a LBP_{16}^{riu2} operator which is consider 5 x 5 square contain 16 pixel neighborhood but we don't go into detail about it since we use the original LBP operator in our test.

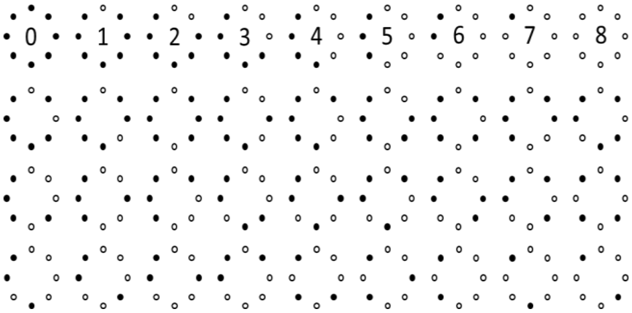

Fig. 6. Observed 36 $LBP_{8,1}^{ri}$ patterns. Black and white dot indicate to bit value of zeros and ones

C. Pyramid of LBP

LBP was extended its feature against texture resolution by cascading the hierarchical spatial pyramids information of LBP [18]. Detail of this process was showed in Figure 7. Firstly, edge contours was extracted from input image by using Canny edge detection which using 3×3 Sobel filter without Gaussian smoothing. Then the image was partition into l level. In each level, instead of calculating gradients, we use $LBP_{8,1}^{ri}$ operator to get histogram from sub-region in that level. Finally, these histograms was concentrated by order of level to form a PLBP feature vector.

D. Grid LBP

A face image is a combination of eyes, nose, lip… each component can be seen as a micro-pattern which effectively presented by the LBP histogram. This can be done by equally dividing face images into small regions $R_0, R_1, \dots, R_m$. The LBP histograms obtained from these regions are concatenated into a spatially enhanced, single feature histogram, which was defined as follows:

$$H_{i,j} = \sum_{x,y} I\{f_l(x,y) = i\} I\left\{(x,y) \in R_j\right\} \tag{6}$$

where $i = 0, \dots, n-1$, $j = 0, \dots, m-1$, f_l is an image labeled by using LBP operator.

The relative position of each sub-regions' histogram indicated their location on the face image. This represented global shape of face images as well as the local texture. Therefore, it is perceptive that LBP feature can represent face images [10 – 12]. Following the setting in [10], we choose $LBP_{8,1}^{u2}$ operator and divide face images into 6x7 sub-regions (Figure 8). Thus the final histogram will have the length of 2478 (59x42).

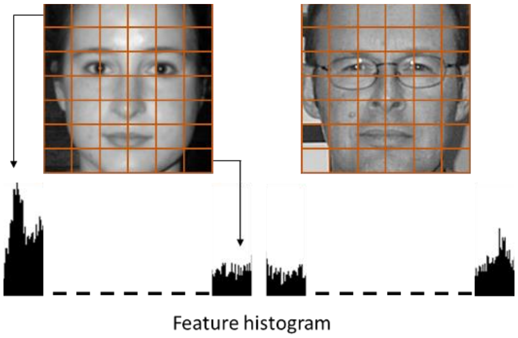

Fig. 8. Illustration of applying the grid 6x7 on images

IV. INDEXING

To boost the searching process in trade-off of low computation and space, we use locality sensitive hashing. The idea behind LSH [19] is using feature vectors to create random projections then combined them into a hash string. A LSH function map a feature vector $v \in \mathbb{R}^d$ to a collection of integer $h_1 \dots h_k$ that each integer was calculate as follows:

$$h_i(v) = \left\lfloor \frac{a_i + b_i}{w} \right\rfloor, i \in \{1,2,3 \dots, k\} \tag{8}$$

where $a_i \in \mathbb{R}^d$ is a vector which created by random independent values from a normal distribution, the constant $w \in \mathbb{R}^+$ affects the granularity of the results and a random, the uniformly distributed number $b_i \in [0, w)$.

There are different ways of using LSH depend on types of problem. A more detail can be found in [20]. In retrieval problem, these parameters: a_i, b_i, w and k were initialized once with unchanged value before indexing. In searching stage, due to the use of L_2 distance (Euclidean distance), we assume two images can be used for nearest neighbors if there are m which $m < k$ matched hashes of hash tuples.

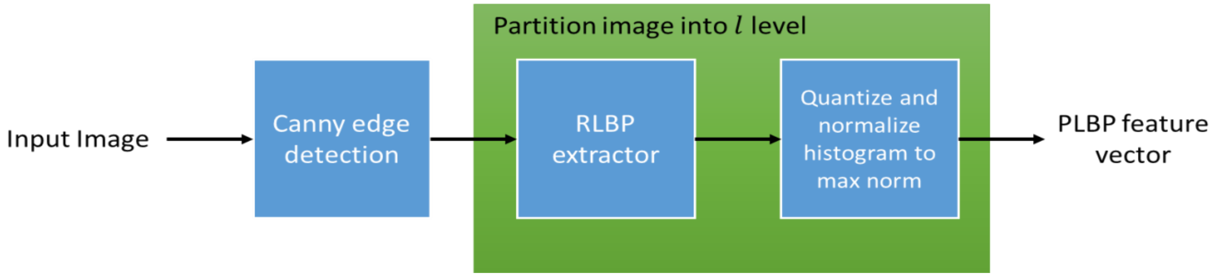
Input Image
Canny edge detection
Partition image into l level
RLBP extractor
Quantize and normalize histogram to max norm
PLBP feature vector

Fig. 7. The process of extracting PLBP feature vector

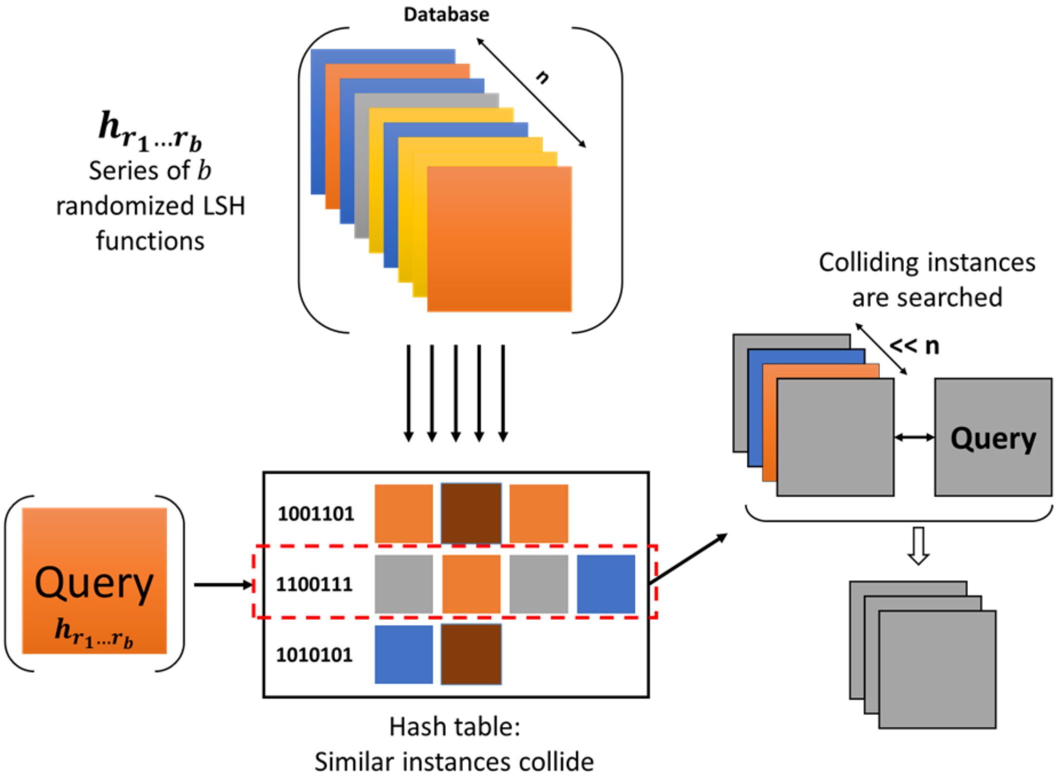
Database
n
$h_{r_1 \ldots r_b}$
Series of b randomized LSH functions
Colliding instances are searched
<< n
Query
Query
$h_{r_1 \ldots r_b}$
1001101
1100111
1010101
Hash table:
Similar instances collide

Fig. 9. Illustration of LSH works in the retrieval system

For implementation, a hash bundle was randomly generated before using in indexing and searching. To draw normally distributed random numbers for a_i, the Box-Muller [21] transform was used as follows:

$$z_0 = \sqrt{-2\ln(k_1)}\cos(2\pi k_2)$$
$$z_1 = \sqrt{-2\ln(k_1)}\sin(2\pi k_2) \quad (9)$$

where z_0, z_1 are variables for a_i and $k_1, k_2 \in (0,1]$ are uniformly distributed random numbers.

In the indexing process, after feature vectors obtained from feature extractor, each will be projected into hashes. Then information represent for an image (path, feature vector, hashes) is packed. To index that image, indexer considers hashes information from package as a string so that it can index this string in an inverted index. The searching process on existing index was done by generating hashes from query image's feature vector and using it as a string query. Result of this process is a list of items (each item considered as a package) that later sorted based on the similarity of query image's feature vector to each item's feature vector.

V. Experimental Results

A. Datasets

The database of faces (TDF) [22] includes images of 40 distinct subjects. Each subject has ten different images (92×112 pixels), which each image was taken in dark background. This dataset features the varying in lighting, facial expression (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses) in its images.

Caltech Faces 1999 (CF1999) [23] differ from previous datasets by adding backgrounds and facial expressions to its images. However, there is no difference in facial detail. This database has 447 images (896×592 pixels) of 27 unique people. We later crop and resize images in this database so that only frontal face in the image.

Combination of CF1999 and TDF (CF) was created for: testing the sustainability of LBP features against the increasing of the number of images and the diversity of data properties.

Labeled Faces in the Wild (LFW) [24] contain 13,233 copped images (64×64 pixels) which were collected around the web. All the images were taken in real-world condition. As we observed, these images also difference in face poses, ages, blur / not blur… That make this database far more challenge than any database that we used in the test.

B. Mean Average Precision (MAP)

To measure the retrieval effectiveness based on queries result we use MAP, which was denoted as follows:

$$MAP = \frac{1}{Q}\sum_{q=1}^{Q}\frac{1}{X}\sum_{1}^{R}\frac{i}{r_i} \quad (7)$$

where Q is the number of queries, i is the number of correct items among first R recommendations, and r_i is the position (start from 1) of ith item in the recommendation list which has R items ($1 \le r_i \le R$). If a person has $x = 5$ correct images but we request a list of $R = 10$ items then $X = x$ else $X = R$. Equation (7) means that the more correct items appear in recommendation list and the higher their positions in this list, the higher MAP return.

C. *Evaluation Framework*

We prepared images of each dataset for further stages by conduct a preprocessing stage that included cropping, resizing (using Lanczos algorithm) and converting to gray scale. After that these images was divided into three equal parts, two (as training set) for indexing and one (as testing set) for searching. The evaluation of a dataset was ended when all parts was used for querying which means after we done querying with first part, the process (indexing and searching) was repeated with second part (the rest two part was used for indexing) and then with final part.

The features we used in our tests: original local binary pattern ($LBP_{8,1}$ or LBP), uniform local binary pattern (ULBP or $LBP_{8,1}^{u2}$**)** **,** pyramid of local binary pattern (PLBP), pyramid of uniform local binary pattern (PULBP uniform or $PLBP^{u2}$), rotation invariant local binary pattern (RILBP or $LBP_{8,1}^{ri}$), uniform rotation invariant local binary pattern (URILBP or $LBP_{8,1}^{riu2}$) and grid $LBP(6 \times 7)$.

D. *Results and disscussion*

We have done some tests to show the computation trade-off with/without using LSH. The result of LWF has more weight than the rest in Table I because we usually deal with large image database in real-world application. The 63.3% is lower search time of LSH method is significant compare with small increasing of indexing time as showed in Table II.

TABLE I. AVERAGE EXECUTION TIME OF EACH QUERY IN EACH DATASET (MILLISECONDS / QUERY)

Dataset / ***Method***	***TDF***	***CF1999***	***CF***	***LFW***
Without LSH	11.93	13.48	22.15	443.48
With LSH	37.40	44.43	80.68	162.72

The results in Figure 10 have showed the effectiveness of LSH with traditional method (nearest neighbor). We timed 500 queries on different dataset size from 500 to 12000 images. Each query will return a list of 50 items (recommenced faces). Image used in query and dataset are randomly took from LWF dataset. It is clearly that traditional method is good if the size of dataset is under 4000. Beyond that point, the execution time rapidly grown while the time of LSH slightly increase and remain under 180 milliseconds/query even if the size is 12000 images.

TABLE II. AVERAGE INDEXING TIME OF EACH IMAGE IN EACH DATASET (MILLISECONDS / IMAGE)

Dataset / ***Method***	***TDF***	***CF1999***	***CF***	***LFW***
Without LSH	15.10	8.90	8.3	3.89
With LSH	17.53	9.47	9.19	4.23

We conducted the tests with different number of returned items from a query. This will show how probability of finding proper faces changes when you continue to look further at the list. In the later part, we will mainly use MAP results of the list of 10 items. There also an illustration of the result of a query from our system in Figures 11 and 12. Fig.11 shows some results from these queries that use $\boldsymbol{LBP_{8,1}^{u2}}$ $(\boldsymbol{6 \times 7})$ and $\boldsymbol{LBP}$. Figure 12 shows some results from these queries that use on $\boldsymbol{PLBP}$ and $\boldsymbol{RILBP}$. These results show the power of $\boldsymbol{LBP_{8,1}^{u2}}$ $(\boldsymbol{6 \times 7})$ and $\boldsymbol{PLBP}$ compare to $\boldsymbol{LBP}$ and $\boldsymbol{RILBP}$, respectively.

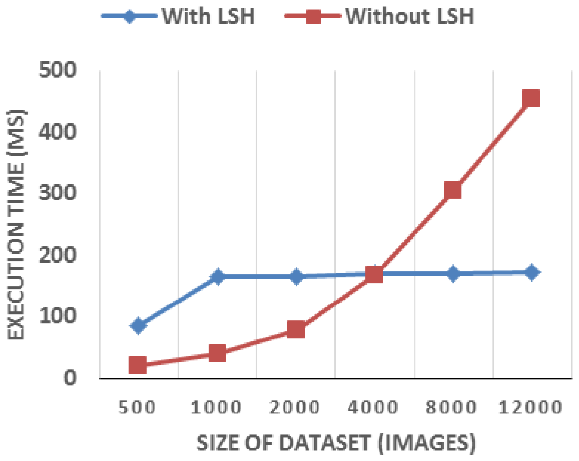

Fig. 10. Average execution time of each query in different dataset size (milliseconds)

Overall, operator that involved parting image into smaller region is the best method. It was showed that $\boldsymbol{PLBP}$, $\boldsymbol{PLBP^{u2}}$ and $\boldsymbol{LBP_{8,1}^{u2}}$ $(\boldsymbol{6 \times 7})$ yield better result than the others. The MAP in LFW dataset is very low compare to the results of other datasets but it showed that in general, the uniform version is better and image partition greatly improved retrieval performance.

TABLE III. MAP OF "CF1999" DATASET

Feature	**Number of retuned items**				
	10	***20***	***30***	***50***	***70***
$\boldsymbol{LBP_{8,1}}$	0.2824	0.2499	0.2693	0.2898	0.2987
$\boldsymbol{LBP_{8,1}^{u2}}$	0.2756	0.2436	0.2628	0.2831	0.2924
$\boldsymbol{LBP_{8,1}^{ri}}$	0.1421	0.1305	0.1463	0.1670	0.1781
$\boldsymbol{LBP_{8,1}^{riu2}}$	0.1208	0.1121	0.1258	0.1433	0.1532
$\boldsymbol{PLBP}$	0.3375	0.2914	0.3191	0.3429	0.3544
$\boldsymbol{PLBP^{u2}}$	0.3755	0.3255	0.3529	0.3759	0.3864
$\boldsymbol{LBP_{8,1}^{u2}}$ $(\boldsymbol{6 \times 7})$	0.8540	0.7869	0.8008	0.8078	0.8111

The special feature of CF1999 is its images were taken different facial expression and backgrounds. Table III showed that $\boldsymbol{LBP_{8,1}^{ri}/LBP_{8,1}^{riu2}}$ operator that indicate image information into small number of bins (36 bins of $\boldsymbol{LBP_{8,1}^{ri}}$ and 9 bins of $\boldsymbol{LBP_{8,1}^{riu2}}$) has the lowest MAP score by 0.1421/0.1208. $\boldsymbol{LBP_{8,1}/LBP_{8,1}^{u2}}$ performs two times better than $\boldsymbol{LBP_{8,1}^{ri}/}$ $\boldsymbol{LBP_{8,1}^{riu2}}$based on MAP score but its MAP still very low. In this dataset, uniform version of $\boldsymbol{PLBP}$ operator is the only operator that perform better that its original version.

$LBP_{8,1}^{u2}$ (6×7) has proven its effectiveness in this case with the highest MAP score of 0.8540, which is over two times than the second high score (0.3755 of $PLBP^{u2}$).

TABLE IV. MAP OF "TDF" DATASET

Feature	Number of retuned items				
	10	*20*	*30*	*50*	*70*
$LBP_{8,1}$	0.4121	0.4427	0.4539	0.4651	0.4702
$LBP_{8,1}^{u2}$	0.4100	0.4406	0.4516	0.4628	0.4680
$LBP_{8,1}^{ri}$	0.1868	0.2180	0.2348	0.2514	0.2583
$LBP_{8,1}^{riu2}$	0.1686	0.1993	0.2134	0.2280	0.2348
$PLBP$	0.4411	0.4735	0.4845	0.4956	0.5005
$PLBP^{u2}$	0.4740	0.5047	0.5169	0.5277	0.5325
$LBP_{8,1}^{u2}$ (6×7)	0.6490	0.6738	0.6825	0.6891	0.6923

TDF expanded CF1999 by adding different facial details and some side movement of face into its content but its image background is constant (dark background). In Table IV, $LBP_{8,1}/LBP_{8,1}^{u2}$ performed nearly 250% better by 0.4411/0.4740 compare to its MAP in CF1999. There is also a slight increase in MAP of $LBP_{8,1}^{ri}/LBP_{8,1}^{riu2}$ but its score is still very low by 0.1868/0.1686. Same as CF1999 dataset, $PLBP^{u2}$ has higher MAP than $PLBP$ and they still hold the second and third high score. While others operator showed their improvement in MAP, $LBP_{8,1}^{u2}$ (6×7) decreased significantly from 0.8540 in CF1999 to 0.6490 in TDF.

TABLE V. MAP OF "CF" DATASET

Feature	Number of retuned items				
	10	*20*	*30*	*50*	*70*
$LBP_{8,1}$	0.3371	0.3366	0.3517	0.3664	0.3730
$LBP_{8,1}^{u2}$	0.3322	0.3319	0.3471	0.3620	0.3687
$LBP_{8,1}^{ri}$	0.1379	0.1439	0.1554	0.1682	0.1751
$LBP_{8,1}^{riu2}$	0.1282	0.1339	0.1448	0.1560	0.1625
$PLBP$	0.3731	0.3617	0.3760	0.3908	0.3974
$PLBP^{u2}$	0.4146	0.4021	0.4184	0.4328	0.4394
$LBP_{8,1}^{u2}$ (6×7)	0.7549	0.7326	0.7424	0.7492	0.7520

CF further tests LBP and its extensions by put all the properties of CF1999 and TDF together in a single dataset. From the result in Table V, it can be seen that except $LBP_{8,1}^{u2}$ (6×7) and$LBP_{8,1}^{ri}/LBP_{8,1}^{riu2}$, the entire MAP score of others operator is higher than CF1999's scores and lower than TDF's scores. $LBP_{8,1}^{ri}/LBP_{8,1}^{riu2}$ continue to show its poorly perform in this dataset by the lowest MAP of 0.1379/0.1282. In the other side, $LBP_{8,1}^{u2}$ (6×7) survived the test and showed the better score than TDF dataset.

We put LBP and its extension in to real-world test with LFW dataset. This dataset included previous datasets properties and real life challenges: different light setup; blur, noise, many faces are partly covered, same person but different ages. All the operator and method perform very poorly. Table VI showed that none the MAP scores in this dataset is over 0.002. Except some result that show the MAP scores from the list of 50 – 70 items of $LBP_{8,1}/LBP_{8,1}^{u2}$. This dataset presented the trend that the uniform version has higher MAP than the original.

TABLE VI. MAP OF "LFW" DATASET

Feature	Number of retuned items				
	10	*20*	*30*	*50*	*70*
$LBP_{8,1}$	0.0056	0.0042	0.0038	0.0035	0.0034
$LBP_{8,1}^{u2}$	0.0057	0.0042	0.0036	0.0033	0.0031
$LBP_{8,1}^{ri}$	0.0033	0.0024	0.0021	0.0019	0.0019
$LBP_{8,1}^{riu2}$	0.0036	0.0027	0.0024	0.0021	0.0020
$PLBP$	0.0047	0.0035	0.0030	0.0027	0.0026
$PLBP^{u2}$	0.0059	0.0043	0.0038	0.0033	0.0031
$LBP_{8,1}^{u2}$ (6×7)	0.0189	0.0128	0.0106	0.0089	0.0081

In the terms of user, we likely look for the correct items in first 10–20 results. So we statistically count number of queries that there is no correct item in their first 10 returned items. Then we divided them to total of queries in each dataset. From Table VII, we can find out that which feature is better when using in real life application. $LBP_{8,1}^{u2}$ (6×7) continue to show its impressive ability in practice, its error on first three dataset always below 1% and despite high error rate in LFW dataset, it has the lowest error rate among the tested features. There also two potential features that can be used in practice although their MAP in previous test was low: $PLBP$ and $PLBP^{u2}$. Their error rate below 10% expect in LFW dataset. $PLBP^{u2}$ has a competitive error rate with $LBP_{8,1}^{u2}$ (6×7) by 89.51% to 82.51%. $LBP_{8,1}^{ri}$ and its uniform version have the highest error rate, over 20% in most cases.

TABLE VII. ZERO MAP AT FIRST 10 RETURNED ITEMS IN PERCENTAGE (%)

Feature	Dataset			
	CF1999	*TDF*	*CF*	*LFW*
$LBP_{8,1}$	13.03%	5.00%	8.64%	90.95%
$LBP_{8,1}^{u2}$	12.58%	5.00%	8.64%	90.80%
$LBP_{8,1}^{ri}$	21.57%	16.25%	23.79%	93.28%
$LBP_{8,1}^{riu2}$	22.92%	16.75%	25.80%	93.41%
$PLBP$	8.76%	5.50%	9.70%	91.97%
$PLBP^{u2}$	8.09%	4.00%	7.46%	89.51%
$LBP_{8,1}^{u2}$ (6×7)	0.67%	0.50%	0.59%	82.41%

In general, we can not use LBP and its extensions in practice but there are two feature that showed their promise potential $LBP_{8,1}^{u2}$ (6×7) grid and $PLBP^{u2}$. The biggest challenges these features encountered so far are: (1) many photo was taken in different backgrounds; (2) skewness make the matching between face images even harder since this break the relation of texture position; (3) blur, noise cause the loss of texture information; (4) different facial expression and detail.

VI. Conclusion

In this paper, we present our result of LBP and its variant in a FR system. We have been evaluated on three fontal faces datasets. Each dataset has unique properties that represent challenges when work with face images. It has been proved that $\boldsymbol{LBP_{8,1}^{u2}}$ $(\boldsymbol{6 \times 7})$ overpowered the rest in all test. Further tests in LFW showed that the lack of skewness invariant as well as bur, facial expression and noise has a huge drawback on all tested features performance. High error rate in LFW dataset shows that we need to overcome these problems before apply $\boldsymbol{LBP}$ features on real-world application. We also find out that $\boldsymbol{PLBP}$ can be as well as $\boldsymbol{LBP_{8,1}^{u2}}$ $(\boldsymbol{6 \times 7})$ from user perspective. Next, the key contribution is that our system can stand against the increasing in the size of dataset and its properties without sudden drop in MAP. Finally, LSH is an ideal choice when designing a FR system that works with large database. However, our system also has some limitations. We have observed that skewness in face image a great influence in retrieval performance and our approach shows its weakness in real-world condition that will be considered in the next works.

In the future, we will consider the fusion of different features such as HOG, learning feature… As well as, we also study face alignment and 3D face modeling to improve the performance in real-word dataset.

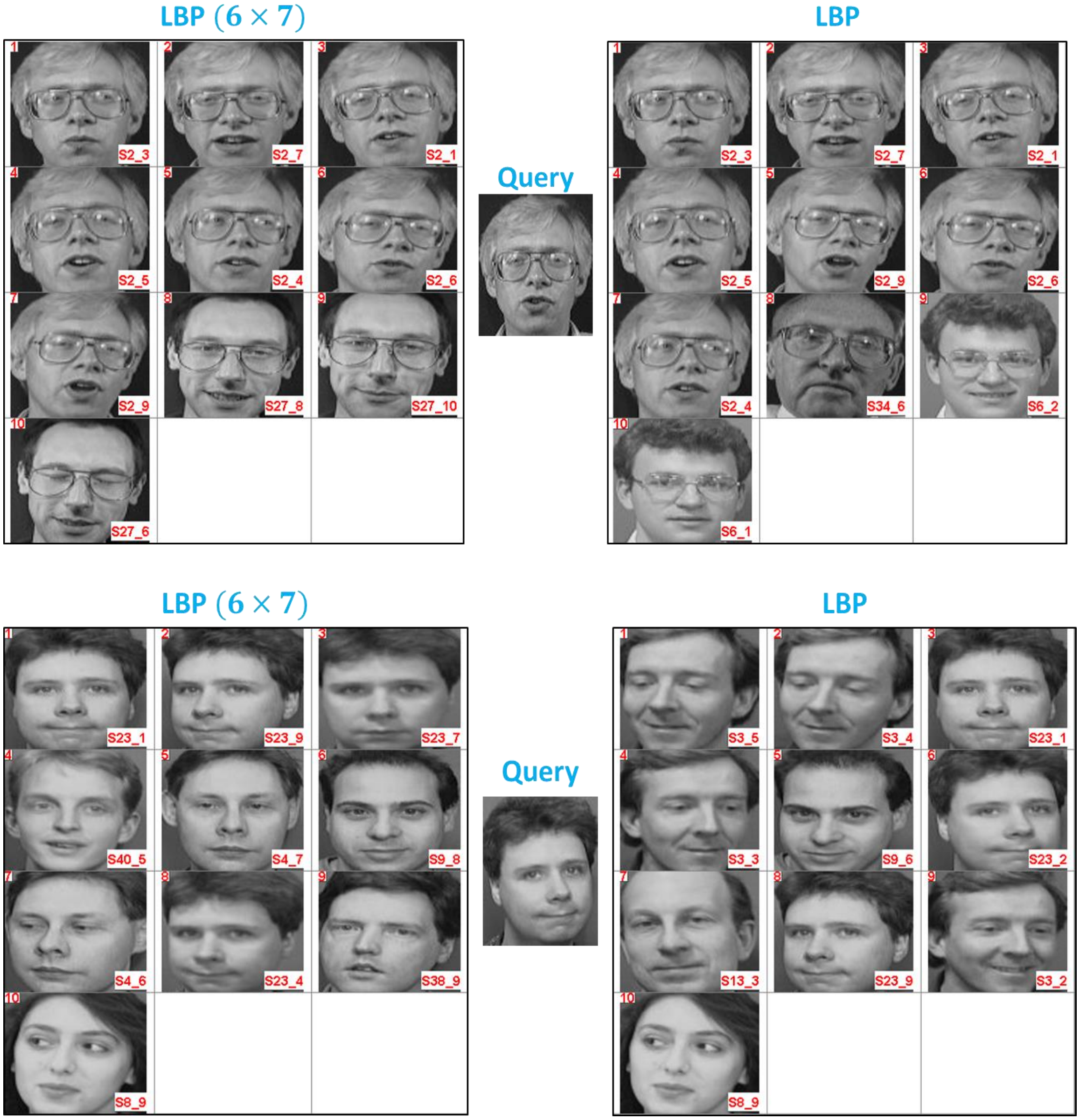

Fig. 11. Some results of the queries from our system on $\boldsymbol{LBP_{8,1}^{u2}}$ $(\boldsymbol{6 \times 7})$ and $\boldsymbol{LBP_{8,1}}$ operator

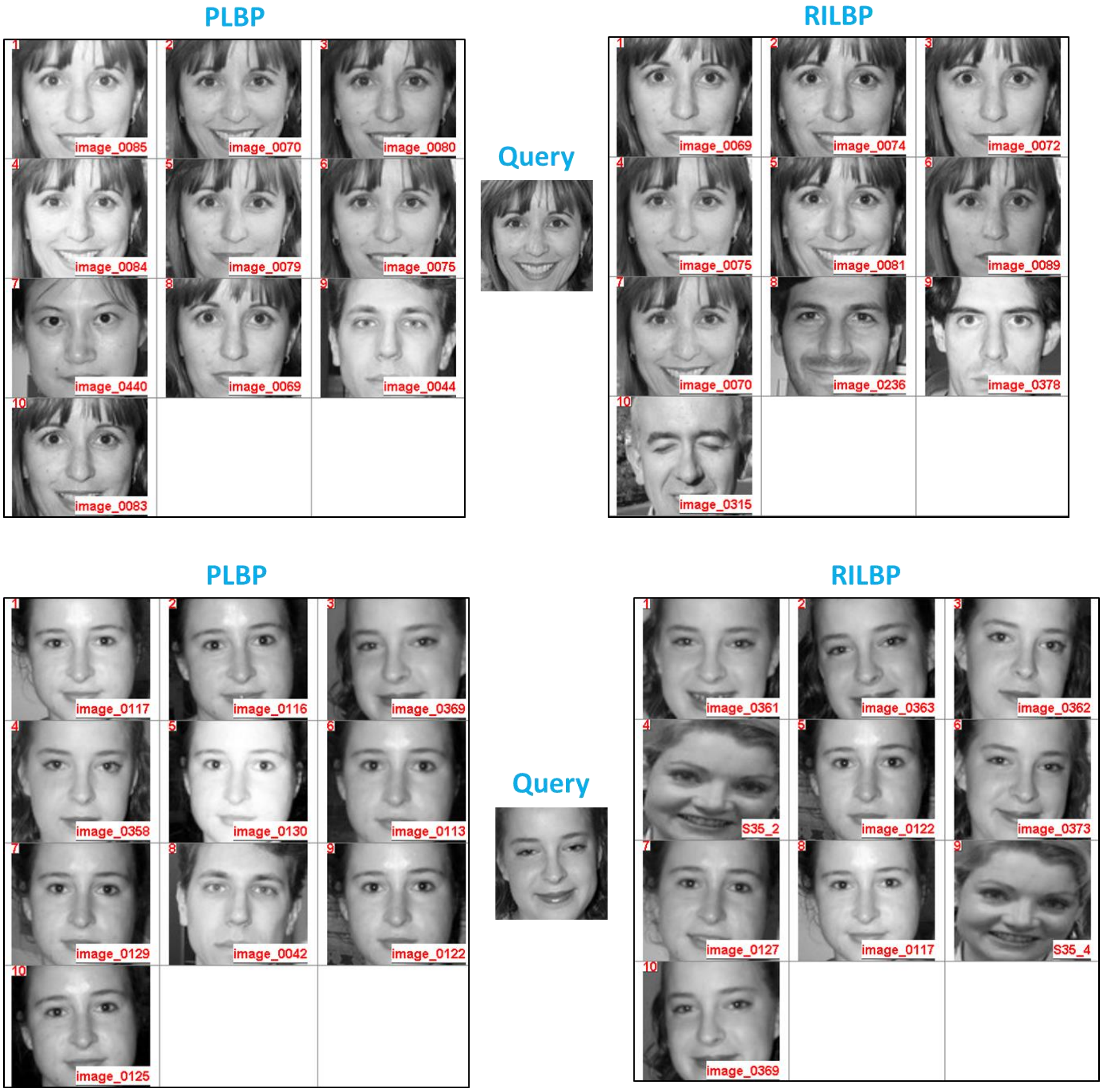

Fig. 12. Some results of the queries from our system on $PLBP$ and $RILBP$ operator

References

[1] Eakins, John P., and Margaret E. Graham. "Content-based image retrieval, a report to the JISC Technology Applications programme." 1999.

[2] Craw, Ian, H. Ellis, and J. Rowland Lishman. "Automatic extraction of face-features." Pattern recognition letters 5.2, pp 183-187, 1987.

[3] Bach, Jeffrey R., Santanu Paul, and Ramesh Jain. "A visual information management system for the interactive retrieval of faces." Knowledge and Data Engineering, IEEE Transactions on 5.4, pp 619-628, 1993.

[4] Turk, Matthew A., and Alex P. Pentland. "Face recognition using eigenfaces." Computer Vision and Pattern Recognition, IEEE Computer Society Conference on. IEEE, 1991.

[5] Pentland, Alex, Baback Moghaddam, and Thad Starner. "View-based and modular eigenspaces for face recognition." Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on. IEEE, 1994.

[6] Gupta, Amarnath, and Ramesh Jain. "Visual information retrieval." Communications of the ACM 40.5, 70-79, 1997.

[7] Ojala, Timo, Matti Pietikäinen, and David Harwood. "A comparative study of texture measures with classification based on featured distributions." Pattern recognition 29.1, pp 51-59, 1996.

[8] Ahonen, Timo, Abdenour Hadid, and Matti Pietikainen. "Face description with local binary patterns: Application to face recognition." Pattern Analysis and Machine Intelligence, IEEE Transactions on 28.12, pp 2037-2041, 2006.

[9] Liao, Shengcai, et al. "Learning multi-scale block local binary patterns for face recognition." Advances in Biometrics. Springer Berlin Heidelberg, pp 828-837, 2007.

[10] Ahonen, Timo, Abdenour Hadid, and Matti Pietikäinen. "Face recognition with local binary patterns." Computer vision-eccv 2004. Springer Berlin Heidelberg, pp 469-481, 2004.

[11] Hadid, Abdenour, Matti Pietikäinen, and Timo Ahonen. "A discriminative feature space for detecting and recognizing faces." Computer Vision and Pattern Recognition, 2004.

[12] Feng, Xiaoyi, Abdenour Hadid, and Matti Pietikäinen. "A coarse-to-fine classification scheme for facial expression recognition." Image Analysis and Recognition. Springer Berlin Heidelberg, pp 668-675, 2004.

[13] Zhao, Guoying, and Matti Pietikainen. "Dynamic texture recognition using local binary patterns with an application to facial expressions." Pattern Analysis and Machine Intelligence, IEEE Transactions on 29.6, pp 915-928, 2007.

[14] Sun, Ning, et al. "Gender classification based on boosting local binary pattern." Advances in Neural Networks-ISNN 2006. Springer Berlin Heidelberg, pp 194-201, ,2006..

[15] Mu, Yadong, et al. "Discriminative local binary patterns for human detection in personal album." Computer Vision and Pattern Recognition, 2008.

[16] Takala, Valtteri, Timo Ahonen, and Matti Pietikäinen. "Block-based methods for image retrieval using local binary patterns." Image analysis. Springer Berlin Heidelberg, pp 882-891, 2005.

[17] Ojala, Timo, Matti Pietikäinen, and Topi Mäenpää. "Gray scale and rotation invariant texture classification with local binary patterns." ECCV, 2000.

[18] Qian, Xueming, et al. "PLBP: An effective local binary patterns texture descriptor with pyramid representation." Pattern Recognition 44.10, pp 2502-2515, 2011.

[19] Datar, Mayur, et al. "Locality-sensitive hashing scheme based on p-stable distributions." Proceedings of the twentieth annual symposium on Computational geometry. ACM, 2004.

[20] Rüger, Stefan. "Multimedia information retrieval." Synthesis Lectures on Information Concepts, Retrieval, and Services 1.1, pp 1-171, 2009.

[21] Box, G. E. P.; Muller, Mervin E. A Note on the Generation of Random Normal Deviates. Ann. Math. Statist. 29, no. 2, 610-611, 1958.

[22] F. Samaria and A. Harter "Parameterisation of a stochastic model for human face identification" 2nd IEEE Workshop on Applications of Computer Vision, December 1994.

[23] Markus, W. (n.d.). Face 1999 (Front). Retrieved April 11, 2016, from http://www.vision.caltech.edu/html-files/archive.html.

[24] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. “Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments”. University of Massachusetts, Amherst, Technical Report 07-49, October, 2007.