Big Data and Image Recognition

Petar Marinov

Abstract –**Nowadays, image recognition and Big Data are two facets that had practically nothing in common, but recent advances in several fields and mainly computer vision has led them to converge to perform tasks using deep machine learning such as detection, recognition and classification.**

In this report, an image recognition system based on Local Binary Pattern Histogram (LBPH) technique and Haar Cascade Classifier technique is presented to process real-time recognition of an object from camera. This system first prepares the data from which to extract information and perform recognition, these are photos and videos containing information on the image, then it uses Haar Cascade Classifier to perform image object detection. Thereafter, the system proceeds to the extraction of the characteristics of the points in the image to encode the shape of the latter in binary using the LBP technique. Finally the classification of the generated histograms is done which allows seeing if it is the right identified object or not.

Keywords –**boosting, feature extraction, image detection, image recognition.**

I. Introduction

To most people, a digital image is just an image. But for a computer, an image is a tapestry of pixels and the scientist has the possibility of deconstructing and analysing this image. Extracting information from still images and even videos, using image processing techniques, including algorithms, will be monetized in the near future.

Much previous research regarding automated facial identification has circumvented the problem of knowing which aspects of the image stimulus are necessary for recognition, considering predetermined measures to be appropriate and effective [4]. This suggested that an information-theoretic method of encoding and decoding objects could provide insight into the information content of these images, focusing on important local and global features. These features might or might not have a direct relationship with the intuitive idea of a given object. In the concept of information theory, the goal is to obtain useful information in an object image, then encode it according to the method used, and then compare the encoding of the object to a database of models that have were coded in the same way [1].

A. Computer vision

The goal of computer vision is to develop algorithms that allow the computer to “see”. Computer vision is therefore a comprehensive field that deals with a high level of programming by feeding input images/videos to automatically perform, and using deep machine learning, tasks such as detection, recognition and classification. The main role of computer vision is to determine whether an image contains a certain object. Algorithms automatically analyze images and extract information.

Automated computer vision is not yet reliable enough to detect or track objects. Therefore, in such cases, humans are always in the analysis loop to assess the situation. [2] Computer vision can improve internet search engine results. Technology can contextualize images in searches. Which means it can label and identify objects, and even settings in your uploaded photos.

B. Big Data

In recent years, more data is being produced at faster rates than ever before and in different formats, this has led to the emergence of a number of technologies that correct specific aspects of these problems hence Big Data.

Big Data literally means a set of massive data. This data can be structured, semi-structured or unstructured. This data is collected and analyzed in order to be used to extract information that can be used for different applications in machine learning projects.

Big Data analysis is based on three dimensions (3V-s):
— Verocity concerns the high speed with which data is created, stored and analyzed;
— Volume refers to the unlimited side of Big Data with a number of data that continues to grow;
— Variety represents the different sources of origin of the data.

Talking about Big Data, there are distinction between two types of data, structured data and unstructured data. The term Big Data refers to the significant volume of data, their types and the tools allowing their processing.

Structured data	is the data that was organized to facilitate processing and simplify analysis, it may be elements such as name, age which respect a defined format, it is this type of data that we use to fill out a form.
Unstructured data	are the data that do not do not have a defined format or structure, we are talking about photos, videos or text, they are more difficult to analyze, by relying on analytical tools we can make them speak

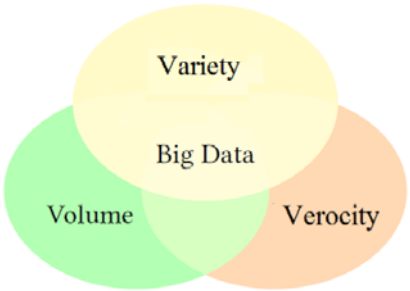

Figure 1: Big data and the 3 V

C. Object recognition

Object recognition consists of automatically recognizing an image using its objects, via a camera, a computer or a smartphone, and using software with its identification algorithms. Its applications have rapidly multiplied: surveillance video, biometrics, robotics, espionage,

automatic payment in cashless stores, image and video indexing, content search, etc.

D. Big Data and object image recognition

Recent advances in several related areas such as social media led object recognition and Big Data to converge. Upcoming applications such as object tagging in social networking are well coming and useful in crucial element in critical applications.

The main benefit of Big Data Analytics (BDA) is the significant increase in accuracy and performance improvement it can offer due to its ability to come up with a massive number of images for the task at hand. Popular BDA technologies include: Apache Hive, Apache Giraph and the already prominent works of Horton, Hadoop, etc.

E. Architecture of an image recognition system

The process of object detection and identification is a machine learning technique, by learning and extracting the features of the image. Matching these features with the tested images can identify the object or prevent these objects from recognizing each other.

This revolutionary identification technology is progressing rapidly, it is being installed almost everywhere, it is more reliable than the human eye. An object recognition system is then capable of detecting an object and identifying it in a simple image or video using machine learning algorithms.

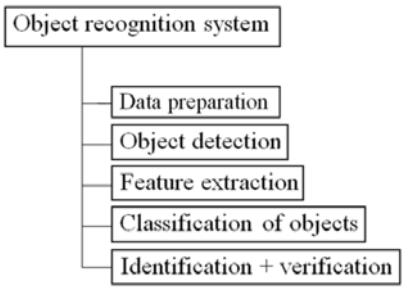

Fig. 2: Object recognition system

Some algorithms focus only on high-resolution images; some of them focus on low resolutions. Recently, researchers have focused on different frontal views of images, from different angles, different lighting. Despite the diversity of these algorithms, they respect almost the same architecture and the same workflow of the steps.

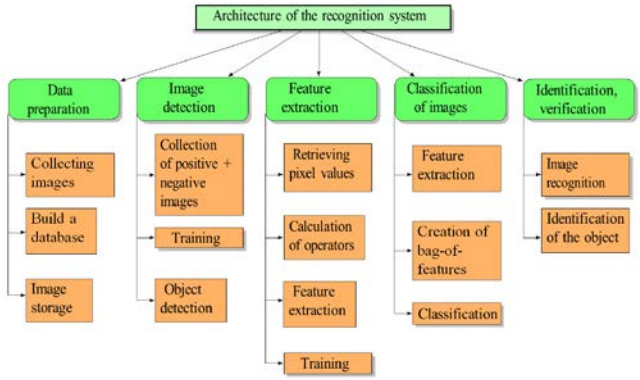

Fig. 3: Architecture of an object recognition system

An object recognition system must first prepare the data from which to extract information and perform recognition. This data is in the form of photos and videos containing information. Then it uses appropriate deep learning algorithms to detect the object, subsequently the system proceeds to extract the features of the object points to encode it shape in binary. Finally, object classification which makes it possible to identify and verify the object in a scene.

Data preparation

The data can be simple images or sequences of images (video) loaded from a database, a surveillance camera or others. The data must be cleaned and converted to grayscale in order to be used efficiently and noiselessly.

Object detection

Object detection is the cornerstone of all object analysis algorithms, including object alignment, modeling, re-illumination, recognition, verification or authentication, parts tracking or recognition, and many more. So computers can clearly identify the object, after which they begin to truly understand scene's thoughts and intentions using machine learning [3].

Object detection, is an important part of the recognition as the first step of the automatic recognition. Detection can be done using deep learning algorithms. This step is based on training the model in order to recognize and detect the object sought in the data. Before attaching a name to an object, the software must first understand that it is in the presence of a specific object. For this the machine has to be educated by telling it that it is a specific object in millions of images containing objects. The system then understands the features of the object whether from the front or in profile.

Feature extraction

This step consists of isolating the elements of the object that will be measured and detecting basic feature points to reconstruct the 3D shape of the object. To do this, the distance between the elements of the object is calculated.

The importance of the features for the recognition cannot be overstated. Many recognition systems need features in addition to the holistic object e.g., Eigenfaces [4] and Fisherfaces [5]. Features can be of different types: region [6], [7], key point [8], and contour [9].

Key point features provide a more accurate and consistent representation for alignment purposes than region-based features, with lower complexity and computational overhead than extracting edge features. Three types of feature extraction methods can be distinguished [10]:

— (1) generic methods based on edges, lines and curves;
— (2) model-based methods that are used to detect object features;
— (3) structural matching methods which take into account geometric constraints on the features.

Classification of objects

Image classification is a classic problem in the fields of image processing, computer vision and machine learning.

Classification is a systematic arrangement into groups and categories based on its characteristics. Image classification emerged to bridge the gap between computer vision and human vision by training the computer with the data. Image classification is achieved by differentiating the image into the prescribed category based on the vision content.

Image classification refers to a computer vision process that classifies an image based on its visual content. For example, an image classification algorithm can be designed to indicate whether or not an image contains a human figure.

Although object detection is trivial for humans, robust image classification remains a challenge in computer vision applications.

Identification and verification

Identification means that the system is able to recognize the identity of the object detected in the image or video. Using machine learning techniques, the system can choose the object that best matches the objects detected using the model created from the database. This correspondence can be carried out using mathematical calculations such as Euclidean distance or maximum likelihood.

II. Approach for object recognition using Big Data

The process of object detection and identification is a machine learning technique, learning and extracting the object features. Matching these features with the tested images can identify the object. There are several challenges and variable parameters in detection and identification like lighting, different poses, changing expressions, poor quality input images, etc.

There are several different perspectives on the object detection and recognition system; some of the projects focus only on high-resolution images; some of them focus on low resolutions. Recently, researchers have focused on the different frontal views of the images, on the different angles, on the different lighting, etc. [11].

The proposed recognition approach has four main steps: an image acquisition module, a feature extraction module, classifier database learning module and a classification module. Initially, the datasets are collected by the image acquisition module. Then, a series of features are extracted by applying the feature extraction module. These features are used to analyze landmarks that represent information about object identity. In the following process, the classifier is trained to recognize the object. In the last module, the system recognizes the image and retrieves information about the object from the database.

A. Data preparation

Like any system using machine learning, a database is needed to train the classification files.

There are public databases whose main functionality is use for object recognition research tests. It is possible to build own database.

A large database can increase the accuracy and efficiency of the algorithm as well as make the training more robust and precise.

These images need to be saved in annotated folders so that they can help to identify each object in the image.

These images will serve training data so that the LBP object recognition can learn the characteristics of each individual object.

An image is a two-dimensional optical projection, but the world we wish to make sense of visually is three-dimensional. In this respect, vision is “inverse optics”: it is necessary to invert the 3D → 2D projection to find the properties of the object in the image space; note that the 2D → 3D inversion of such a projection is strictly impossible mathematically.

At this stage, the dataset is pre-processed for the feature extraction process. The images in the dataset were converted to grayscale images and then normalized these images for good recognition results.

B. Object detection

Haar modules are used to detect the local features in a given input image. Here, input image refers to the digital image captured by the camera.

Haar Cascade classifier

Although there are many different algorithms for performing object detection, each has its own weaknesses and strengths. Some use contours and others are even more complex involving models, neural networks or filters. These algorithms suffer from the same problem: they are expensive in terms of calculation time.

Viola and Jones [12] designed Haar Classifiers algorithm, to quickly detect any object, using cascades of AdaBoost classifiers. It is basically a machine learning based approach in which a cascaded function is trained from a large number of positive and negative images. Based on the training, it is then used to detect the objects in the other images.The basis of the Haar classifier's object detection lies in Haar-like features and not pixels. They rather than using the intensity values of a pixel, use the change in contrast values between adjacent rectangular groups of pixels. Contrast differences between groups of pixels are used to determine relatively light and dark areas. Two or three adjacent groups with relative contrast variance form a Haar-like feature.

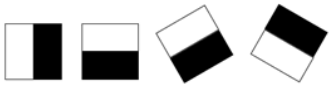

Figure 4 – Feature of the edges

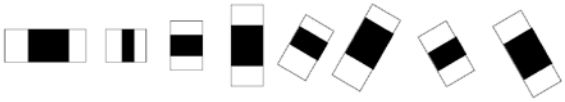

Figure 5 – Features of the lines

Figure 6 – Features of the centres

Haar features can easily be scaled by increasing or decreasing the size of the group of pixels being examined. This allows features to be used to detect objects of different sizes. Haar Cascade classifier is used in this work.

Training the classifier to detect an object

Detecting object features requires the Haar classifier cascades to be trained first. In order to train the classifiers, the AdaBoost algorithm and Haar feature algorithms need to be implemented.

Boosting

Boosting is a class of ensemble machine learning algorithms that involve combining the predictions of many weak learners.

A weak learner is a very simple model, although it has some skill on the dataset. Boosting was a theoretical concept long before a practical algorithm could be developed, and the AdaBoost (adaptive boost) algorithm was the first successful approach to the idea.

In general, boosting provides a clean interface between boosting and the underlying weak learning algorithm: in each round, the boosting algorithm provides a set of

weighted data to the weak learner, and the weak learner provides a predictor in return.

The only way for the boosting algorithm to learn from the data is to call the base learning algorithm. However, if the base learner is simply called repeatedly, always with the same set of training data, we cannot expect anything interesting to happen; Instead, the same is expected, or nearly the same, base classifier to be produced over and over again, so that this small classifier is gained over the base learner's execution only once. The boosting algorithm improving the base learner somehow manipulates the data it provides.

As boosting remains independent of the weak learner, it is known as meta-learning algorithm.

The weighting scheme (how α is defined in Algorithm) can be varied to modify the overall boosting algorithms. In this section, AdaBoost is implemented which chooses α as:

$$\alpha_t = \frac{1}{2} ln \frac{1+\gamma_t}{1-\gamma_t}$$

where γ is the edge of $h_t(x)$ defined as:

$$\gamma_t = \sum_{i=1}^{N} D_t(i) y_i h_t(x_i)$$

AdaBoost and Random forest algorithms

The AdaBoost algorithm [13] is the first practical boosting algorithm, and remains one of the most widely used and studied, with applications in many fields. It involves using very short (single-level) decision trees as weak learners that are added sequentially to the set. On data from many real-world applications, AdaBoost also proves to be very effective comparatively to other methods like Random Forest, which follows an important methodology in machine learning.

AdaBoost is based on single-depth trees with the allocation of different weights for each node, unlike Random Forest which is represented with a tree having a long depth.

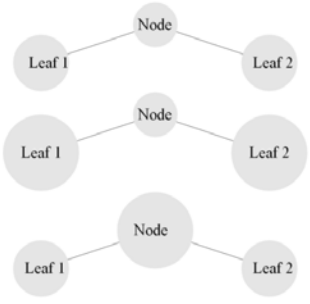

Figure 7: AdaBoost algorithm tree

Each subsequent model attempts to correct the predictions made by the model before it in the sequence. This is achieved by weighting the training dataset to place more emphasis on training examples where previous models made prediction errors (Figure 8). AdaBoost proceeds in rounds or iterative calls to the basic learner. To choose the training sets provided to the base learner in each round, AdaBoost maintains a distribution over the training examples. The distribution used in the t-th round is denoted D_t, and the weight it attributes to the training example i is denoted $D_t(i)$.

This weight is a measure of the importance of correctly classifying example i on the current round. Initially, all weights are set equally, but in each round the weights of misclassified examples are increased so that, in effect, difficult examples get successively higher weights, forcing the basic learner to focus your attention on them [13].

Enter : Data base $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$, where $x_i \in X, y_i \in Y = \{-1, +1\}$
Basic learning algorithm E;
Number of learning cycles *T*;
Number of samples *n*

Treat :

1. $D_1(x) = 1/n$ # initialization weight distribution
2. for $t = 1,\dots,T$;
3. $h_t = E(D, D_t)$; # get weak hypothesis h_t: X → $\{-1, +1\}$ from *D* currently being in D_t
4. $\varepsilon_t = P_{x\sim D_t}(h_t(x_i) \neq y_t)$; # evaluate h_t error
5. ***if*** $\varepsilon_t > 0.5$ ***then break***
6. $\alpha_t = \frac{1}{2} \ln \frac{1-\varepsilon_t}{\varepsilon_t}$; # determine the weight of h_t
7. $D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} exp(-\alpha_t) & if\ h_t(x_i) = y_t \\ exp(\alpha_t) & if\ h_t(x_i) \neq y_t \end{cases} = \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i))}{Z_t}$
update the distribution, where Z_t is a normalization factor so that D_{t+1} to be a distribution.
8. ***end***
Output : $H(x) = \text{sign}(\sum_{t=1}^{T} \alpha_t h_t(x))$

Figure 8: AdaBoost algorithm

The learning algorithm involves starting with a decision tree, finding examples in the training dataset that have been misclassified, and adding more weight to those examples. Another tree is trained on the same data, although now weighted by classification errors. This process is repeated until a desired number of trees is added.

Unlike the Adaboost algorithm, the Random Forest algorithm which is a substantial modification of bagging constructs a large collection of decorrelated trees and then averages them. On many problems, the performance of random forests is similar to boosting, and they are simpler to train and tune. So, random forests are popular and are implemented in a variety of packages.

Figure 9: Random forest

Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest unlike Adaboost which assigns different weights to the nodes. The generalization error for forests

gradually converges to a limit as the number of trees in the forest becomes large. This error for tree classifiers depends on the strength of the individual trees in the forest and the correlation between them.

So, AdaBoost is an undeniably powerful algorithm and Random Forest is at least as good, if not iteratively fitting even noisy data sets until every data point in the training set was fit without error.

The statistical view of boosting understands AdaBoost as a stepwise optimization of an exponential loss, which requires regularization of the tree size and control of the number of iterations as well as the weight. On the other hand, a Random Forest is not an optimization; it seems to work best with large trees and as many iterations as possible.

Training the Adaboost classifiers is done as follows (Figure 10): each classifier collects a new set of negative samples for each step, and the number of negative samples sets the limit for the size of the set. It uses information from previous steps to determine which of the "candidate samples" are misclassified.

For a learning classifier to be efficient and accurate in its predictions, it must meet three conditions:
— (1) it must be trained on "sufficient" training examples;
— (2) it should provide a good fit to these training examples (meaning low training error);
— (3) it should be "simple".

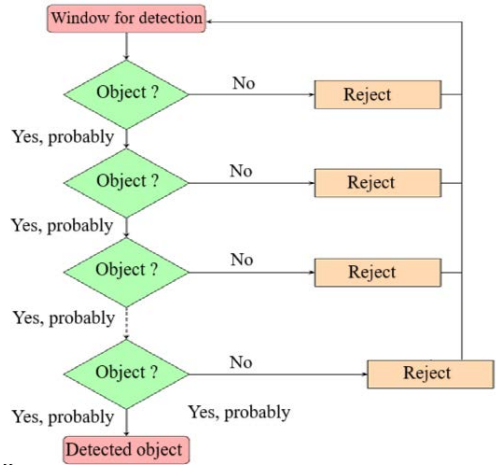

Fig. 10: Training classifiers with AdaBoost

The final classifier is:

$$f(x) = \text{sign}(\sum_t \alpha_t h_t(x))$$

$$D_T(i) = 1/n \frac{\exp(-\sum_t \alpha_t y_t h_t(x_i))}{\prod_{t=1}^{T-1} Z_t}$$

$$\prod_t Z_t = 1/n \sum_i \exp(-y_i f(x_i)) = \prod_t \sqrt{1-4\gamma_t^2} \leq e^{-2\sum_t \gamma_t^2}$$

$$\gamma_t = 1/2 - e_t,$$

if $\gamma_t \geq \gamma > 0$. The training error drops exponentially, so we obtain the best possible classifier.

For each predictor to satisfy decision strains, this is just a feature-threshold pair. For each characteristic fi, there can be several possible thresholds τ. Given an input instance to be classified, a decision stem corresponding to the characteristic fi and the threshold τ will predict +1 if the input instance has a characteristic value fi greater than the threshold τ; otherwise, it predicts −1.

Determining appropriate threshold:

To create the different thresholds for each characteristic fi, it is necessary to:
1. sort the training examples by their fi values;
2. remove duplicate values;
3. construct thresholds halfway between successive characteristic values.

Two thresholds are added for each feature: one below all observed values for that feature and one above all values for that feature. By removing duplicate values, the number of thresholds to check is reduced.

Optimal decision strain:

The optimal threshold $\tau*$ for fi corresponds to the threshold maximizing:

$$\tau^* = \underset{\tau}{\text{argmax}} \left|\frac{1}{2} - error(\tau)\right|$$

where *error*(.) is calculated as the sum of the weights ($D_t(i)$) of the misclassified instances. Since dealing with absolute values, if one threshold has an error of 0.2 and another threshold produces an error of 0.9, the latter is preferred.

C. Feature extraction

The LBP histogram

Definition

The histogram with local binary patterns is a technique used in the field of image recognition. This technique uses a block of 3*3 pixels extracted from an image, and it is particularly interested in the central pixel to be able to calculate the LBP operator.

Images are represented as rectangular arrays of numbers representing image intensities at particular locations. Each element of such an array is called a pixel. A color image can be represented as three separate matrices called "color planes", containing red, green and blue components as monochromatic images. An image with a slanted edge might look like:

TABLE 1. REPRESENTATION OF AN IMAGE IN PIXELS

0	0	1	12	10	17	15	18	1	0
0	2	5	15	14	18	12	20	25	0
0	1	17	20	15	20	14	16	56	60
0	8	22	18	20	35	28	40	43	46
0	10	25	27	32	38	40	41	39	46
0	12	20	30	40	38	30	38	46	40
0	5	15	36	38	40	30	89	77	76
5	12	11	10	42	55	45	56	60	72
20	15	13	20	18	60	63	90	98	74

The LBP operator is based on the values of the 9 pixels which constitute this block, namely, the central pixel and the 8 neighboring pixels or even the 26 neighbors if we have an XY grid. This feature extraction technique subsequently

helps to make the classification which will be used to create a deep learning model.

Below an image detection algorithm based on LBPH is presented. The LBPH algorithm is the combination of Local Binary Patterns (LBP) and Histogram of Oriented Gradients (HOG). LBP is a simple but powerful way to extract and label pixels from an image. Using LBPH, it is easily represent objects in images with a simple vector.

Calculation of the LBPH operator:

In this section it is explained how the LBP operator calculates the values of each pixel in the image to create considered model using the local binary pattern histogram.

The LBP operator is defined as follows:

For a block of 9 pixels 3*3, it can be defined as follows:

$$LBP = \sum_n^7 s(i_n - i_c)2^n$$
$$s(i_n - i_c) = 1sii_ni_c \geq 0\,,\ 0sii_n - i_c < 0$$

with ic the value of the central pixel, and in the value of the neighboring pixels which are the pixels around the central pixel knowing that n going from 0 to 7. If the difference of the values ic is negative then $s(i_n - i_c) = 1$ otherwise $s(i_n - i_c) = 0$. These pixel values can be the intensity or brightness values and they are used to do the calculation of the LBP operator to maximize the variation or features that are relevant to identity.

In the matrix, the features extracted from the image are used to compare the values of neighboring pixels with the values of central pixels to finally generate a binary code.

To detect objects in an RBG (color) photo, the image is converted to a grayscale image. For each pixel $P_{i,j}$, it is a vector which contains three values representing the degree of red, blue and green. So, the RBG image is converted to a *y*-scale image by:

$$G_{i,j} = (0.2989, 0.5870, 0.1140)^T.P_{i,j},$$

where $G_{i,j}$ represents the corresponding pixel in the image. The LBP operator compares the central value with its P neighboring values on the circle of radius R and assigns 1 to the neighboring value, if the central value is greater than the neighbor, it assigns the value 0 otherwise. Below the LBP operator for a pixel is demonstrated with the order of the pixel values (a):

i_0	i_1	i_2
i_7	i_c	i_3
i_6	i_5	i_4

(a) Order of the pixel values

5	9	1
4	4	6
7	2	3

(b) Pixels extracted from image

1	1	0
1		1
1	0	0

(c) Binary values by LBR operator

Let consider a simple example of a block of 3*3 pixels extracted from an image (b) and calculate LBP operator which transforms these values into binary values forming a single value of the central pixel (3).

TABLE 2. CALCULATION OF THE LBP OPERATOR

For n=0 :	$LBP = s(i_0 - i_c)2^0 = s(5-4)2^0 = 1$
For n=1 :	$LBP = s(i_1 - i_c)2^1 = s(9-4)2^1 = 1$
For n=2 :	$LBP = s(i_2 - i_c)2^2 = s(1-4)2^2 = 0$
For n=4 :	$LBP = s(i_4 - i_c)2^4 = s(3-4)2^4 = 0$
For n=5 :	$LBP = s(i_5 - i_c)2^5 = s(2-4)2^5 = 0$
For n=6 :	$LBP = s(i_6 - i_c)2^6 = s(7-4)2^6 = 1$
For n=7 :	$LBP = s(i_7 - i_c)2^7 = s(4-4)2^7 = 1$

The calculated binary values can be presented as follows: Transition of these binary values from 0 to 1 or from 1 to 0 show that there are edges or contour which subsequently allows to define and detect the desired object. Therefore this is an invariant edge illumination descriptor.

LBP operator reduces the influence of lighting and returns the texture by considering each pixel of an image that excludes boundary pixels. Then, the binary values/bits calculated successively from i_0 to i_7 are recovered, i.e. rotated in the direction of the needle of a clock, to transform them into one byte to form the binary LBP number. This sequence of 8 bits forms a binary number generated from the pixel values are converted into a decimal number to obtain the value of the central pixel in order to form the system:

7	6	5	4	3	2	1	0
1	1	0	1	0	0	1	1

The formula dives:

$$1.2^0 + 1.2^1 + 0.2^2 + 1.2^3 + 0.2^4 + 0.2^5 + 1.2^6 + 1.2^7 = 211$$

So 211 is the generated LBP code. By the LBP result, a histogram for this image is generated and a data vector to describe the patterns in the original image is formed.

Production of the histogram

This calculation must be done for each pixel of the image by implementing the same instructions in order to obtain the corresponding LBP codes and replacing it in the pixel of the image concerned. These obtained values are transformed into a histogram depending on the number of appearances or frequency of each number and looking at the statistics the characteristics of the distribution of these decimal numbers are obtained (256 different values - between 0 and 255).

The whole picture is not directly considered. The image is divided into several image pieces. The values of 256 positions for each image element are calculated given N_x and N_y, representing the number of cells in the horizontal and vertical direction respectively. Then all histograms are concatenated from all image chunks to form a more meaningful histogram.

Subsequently, uniform local binary models are organized that have only 59 different possible values instead of 256, which really represent robust statistics. Then histograms for

each image are made to find the similarity between them, and if the similarity is high with an image of known training one can predict that it is the searched object.

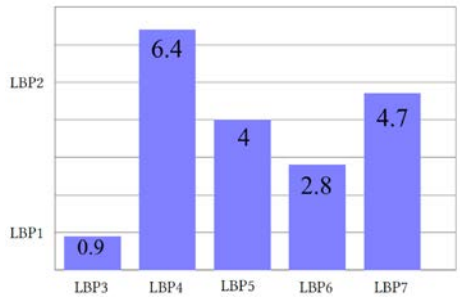

Fig. 11: LBP histogram

The advantage of this technique is that it is lighting invariant, i.e. if the lighting of the image or scene is changed, all pixel values will increase and the relative difference between they will stay the same, so the binary pattern will stay the same regardless of the lighting value in general.

C. Classification

This step consists of making the classification which is carried out by calculating the similarities between direct histograms.

The system performs an object recognition process. First, it detects all objects in the image, then it extracts object features from the input test image. Once this input feature vector is made, it is compared to the image dataset model trained using the Haar Cascade classifier.

If the input test feature vector matches the trained model, it recognizes the object and retrieves the information about that image from the database. The system can detect multiple objects in the image if a new image enters the system, it recognizes the object from the information in the database [14]. The trained classifier aims to match the image with images from the trained model which is calculated by one of the following methods:

The intersection of histograms:

$$D(S,M) = \sum_i (min(S_i, Mi))$$

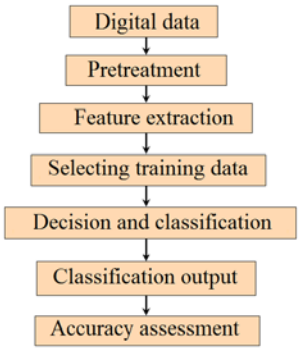

Figure 12: Classification process

Log-likelihood statistic:

$$L(S,M) = -\sum_i S_i log M_i$$

Chi-square statistic (X^2):

$$X^2(S,M) = \sum_i \frac{(S_i - M_i)^2}{S_i + M_i}$$

with *S* and *M* two LBP histograms.

The search for the two histograms which are the most similar can also be done using the calculation of the Euclidean distance as follows:

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$$

Given the feature representation of an object sample, the classification step aims to calculate a score for the sample and, according to a decision threshold, accept or reject the sample. Similarity measurement methods, such as normalized correlation (NC) [15], are the simplest and most popular classifiers. More complex statistical models such as neural networks (NN) [16] or support vector machines (SVM) [17] are also used. NC, NN or SVM are discriminative approaches.

Identification

After having matched the histograms in order to classify the objects and recognize them, an identification is produced which is the final step of the recognition process. For this purpose the entire directory of images is browsed, which is done in the form of a tree in the object database. Two tables are built, the first contains the set of photos and the second contains the identity of each object which is the name of the directory. If there are images in the directory, the name of the folder containing the image will be retrieved and store it in the first table which identifies the objects and all the corresponding images will be saved in the second image table.

This identification process is performed for all the image files in the database and by the trained classifiers matching the stored data with the detected object the object can be easily identified.

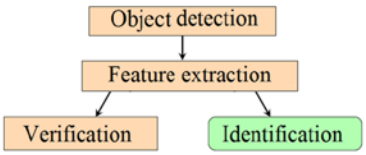

Fig. 13: Object identification

D. Results and prospects

The considered method is quite simple in terms of extracting local binary patterns and gives satisfactory results in terms of time and precision. The results produced by the developed prototype show that an object can be detected and recognized in a few fractions of seconds and in real time.

Good results are obtained from various experimental analyses of this technique, they also provide acceptable results for occlusion, pose variation and illumination.

The true pose variation occurs when the subject's position is at a different angle to the camera, and the system can recognize the object even if in the dataset only frontal images are stored as shown. As illustrated above, the test was done with three different angular variations, one at approximately 30° to the left, the second is a frontal position and the third

at approximately 30° to the right and we always the obtained results are the same, which is the identification of the object in real time.

However, there are imperfections, each time the lighting in the room was insufficient, the recognition rate decreased, since there was a significant number of false positives. Another challenge was that of hardware, object recognition requiring high-performance computer hardware and more particularly a high definition camera with high resolution.
The accuracy of this approach is worth about 80% with low image resolution. Better results can be obtained with a high image resolution and a large database.

Additionally, the number of images per object in the training set database having more diverse pose and lighting variations plays a vital role in determining recognition accuracy and can increase the precision and efficiency of the approach.

As machine learning is very important nowadays, there are many areas where this work can be extended.

The proposed system could identify even faces in real time and be integrated with Google Maps as an example application, to track any subject of interest. Additionally, object recognition systems can also be used in the development of automated attendance systems.

III. Conclusion

Object recognition technology has come a long way over the past twenty years. Today, machines are capable of automatically verifying identity information for secure transactions, for surveillance and security tasks, for building access control, etc. These applications generally operate in controlled environments and recognition algorithms can take advantage of environmental constraints to achieve high recognition accuracy. However, next-generation object recognition systems are going to have widespread application in smart environments – where computers and machines are more like helpful assistants.

To achieve this goal, computers must be able to reliably identify objects nearby in a way that fits naturally into the pattern of normal human interactions. They should not require special interactions and should conform to human intuitions about when recognition is likely. This implies that future intelligent environments should use the same modalities as humans and have roughly the same limitations. These goals now appear to have been achieved - however, substantial research remains to be done to make person recognition technology work reliably, under widely varying conditions using information from single or multiple modalities.

References

[1] D. Alzubaydi, M. Husein, „Mobile face recognition application using eigen face approaches for Android“, Al-Mustansiriyah Journal of Science, vol. 30, no 1, p. 119, 2019, DOI:10.23851/mjs.v30i1.540

[2] D. Gutierrez, „Where are we with computer vision“, https://insidebigdata.com/2018/01/03/ where-are-we-with-computer-vision/

[3] A. Kumar, A. Kaur, M. Kumar. Face detection techniques: a review, Artificial Intelligence Review, vol. 52, issue 1, 2019, DOI:10.1007/s10462-018-9650-2

[4] R.Jb. Tushar, A.K.N. Balasubramanya, M.S. Natarajanb Vinay Aa, V.S. Shekhara, „Cloud based big data analytics framework for face recognition in social networks using machine learning“, Procedia Computer Science, vol. 50, pp. 623-630, 2015, DOI:10.1016/j.procs.2015.04.095

[5] D. Kriegman, P.N. Belhumeur, J.P. Hespanha, „Face recognition: Eigenfaces vs. fisherfaces: Recognition using class specific projection“, IEEE Trans. on Pattern Analysis and Machine, 1997.

[6] Y.S. Ryu, S.Y. Oh, „Automatic extraction of eye and mouth fields from a face image using eigenfeatures and multilayer perceptrons“, Pattern Recognition, vol. 34, no 12, pp. 2459-2466, 2001, DOI:10.1016/S0031-3203(00)00173-4.

[7] D. Cristinacce, T. Cootes, „Facial feature detection using adaboost with shape constraints“, 2003, DOI:10.5244/C.17.24.

[8] L. Wiskott, J.M. Fellous, N. Krüger, C. Malsburg, „Face recognition by elastic bunch graph matching“. In: G. Sommer, K. Daniilidis, J. Pauli, $Computer Analysis of Images and Patterns“, CAIP, Lecture Notes in Computer Science, vol. 1296, 1997, Springer. https://doi.org/10.1007/3-540-63460-6_150

[9] T.F. Cootes, G.J. Edwards, C.J. Taylor, „Active appearance models“, IEEE Trans. On Pattern Analysis and Machine Inelligence, vol. 23, no. 6, 2001.

[10] R. Wirza, O.K. Rahmat, E. Bagherian, „Facial feature extraction for face recognition: a review“, International Symposium on Information Technology, 2008, Publisher: IEEE, DOI: 10.1109/ITSIM.2008.4631649.

[11] F. Ali, D. A. Ghaffar, H. Memon, F. Deeba, A. Ahmed, „Lbph based enhanced real-time face recognition“, International Journal of Advanced Computer Science and Applications(IJACSA), vol. 10, issue 5, 2019, DOI: 10.14569/IJACSA.2019.0100535

[12] P. Viola, M. Jones, „Rapid object detection using a boosted cascade of simple features“, Computer Vision and Pattern Recognition Conference, vol. 1, pp. 511–518, 2001, Kauai, Hawaii.

[13] R. E. Schapire, Y. Freund, „Boosting foundations and algorithms“, 2012, MIT Press Direct, DOI: https://doi.org/10.7551/mitpress/8291.001.0001

[14] P. Nimbal, G. Krishna, Sh. Abhishek, P. Singh, S. lKumar, S Manvi, „Face recognition system based on lbph algorithm“, International Journal of Engineering and Advanced Technology (IJEAT), Vol. 8, issue 5S, pp. 26-30, 2019.

[15] J. Kittlerp Y. Li, J. Matas, „On matching scores of lda-based face verification“, Proceedings of the British Machine Vision Conference, 2000, BMVC 2000, Bristol, UK.

[16] S. Marcel. „A symmetric transformation for lda-based face verification“, Proceedings of Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004, Seoul, Korea, DOI: 10.1109/AFGR.2004.1301532.

[17] J. Kittler, K. Jonsson, J. Matas, Y.P. Li. „Learning support vectors for face verification and recognition“, Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition, 2000, Grenoble, France, DOI: 10.1109/AFGR.2000.840636.