# Face Recognition with Decision Tree-based Local Binary Patterns

Daniel Maturana, Domingo Mery and Álvaro Soto
*Departamento de Ciencias de la Computación*
*Pontificia Universidad Católica*
*Santiago, Chile*
*Email: {dimatura, dmery, asoto}@uc.cl*

## I. Introduction

Face recognition algorithms commonly assume that face images are well aligned and have a similar pose – yet in many practical applications it is impossible to meet these conditions. Therefore extending face recognition to less constrained face images has become an active area of research.

To this end, face recognition algorithms based on properties of small regions of face images – often known as local appearance descriptors or simply local descriptors – have shown excellent performance on standard face recognition datasets. Examples include the use Gabor jets [**?**], SURF [**?**], SIFT [**?**], HOG [**?**] and histograms of Local Binary Patterns [**?**]. A comparison of various local descriptor-based face recognition algorithms may be found in Ruiz del Solar et al [**?**].

Among the different local descriptors in the literature, histograms of Local Binary Patterns (LBP) [**?**] have become popular for face recognition tasks due to their simplicity, computational efficiency and robustness to illumination variation.

The success of LBPs has inspired several variations. These include local ternary patterns [**?**], elongated local binary patterns [**?**], multi scale LBPs [**?**], centralized binary patterns [**?**], patch based LBPs [**?**], among others. However, these are specified a priori without any input from the data itself, except in the form of cross-validation to set parameters. Our key contribution is a method to explicitly learn discriminative descriptors from the training data, which is based on a connection between LBPs and decision trees.

In this work we consider the task of *closed set face identification*. In this task we are given a gallery of identified face images and for any given unidentified probe image we return one of the identities from the gallery.

### A. Face recognition with Local Binary Patterns

For any intensity image, we consider a pixel $c$ and a vector $\mathbf{n}$ composed by its $s$ neighbors

$$\mathbf{n} = (n_1, \dots n_s)$$

in an arbitrary order.

LBPs were introduced by Ojala et al [?] as a fine scale texture descriptor. In LBPs, neighbors are values sampled at equally spaced points on a circle of radius $r$ centered on the the pixel $c$, using bilinear interpolation if the sample points do not correspond to the center of a pixel.

Figure 1. The $LBP$ operator thresholds each pixel against its neighboring pixels and interprets the result as a binary number. In the bottom image each gray-level value corresponds to a different local binary pattern.

The LBP operator assigns a decimal number to a pair $(c, \mathbf{n})$. This number is calculated as

$$b = \sum_{i=1}^{s} 2^{i-1} I(c, n_i)$$

where

$$I(c, n_i) = \begin{cases} 1 & \text{if } c \leq n_i \\ 0 & \text{otherwise} \end{cases}$$

This can be seen as assigning a 0 to each neighbor that is larger than the center pixel, a 1 to each neighbor smaller than the center pixel, and interpreting the result as a number in base 2. It is clear that with $s$ neighbors, there are $2^s$ possible LBP values.

Ahonen et al [?] introduce the use of LBPs for face recognition with the following basic procedure:

1) Partition the face image in a grid with equally sized cells, the size of which is an parameter.
2) For each grid cell, apply the LBP operator to each pixel in the grid cell and create a histogram of the LBP values (with $2^s$ bins) and concatenate the histograms into a single vector.
3) Classify a probe face with the identity of the nearest neighbor in the gallery, where the nearest neighbor distance is calculated with $\chi^2$ distance between the histograms of the corresponding face images.

The first step is illustrated in figure **??**.

### B. Local Binary Patterns as Decision Trees

The aforementioned histograms of LBPs may be seen as quantizing each pair $(c, \mathbf{n})$ with a specially constructed binary decision tree. The tree with $s$ levels, where all the nodes at level $l$ compare the center pixel with neighbor $n_l$. That is, if $c < n_l$ the vector is assigned to the left node at level $l - 1$; otherwise, it is assigned to the right node at

level $l-1$. Since the tree is complete, at level 0 we have $2^s$ leaf nodes. Each of these nodes corresponds to one of the $2^s$ possible LBP. In fact, seen as a binary number, each LBP encodes the path taken by $(c, \mathbf{n})$ through the tree; for example, in an LBP with $s = 8$, `11111101` corresponds to a a $(c, \mathbf{n})$ pair which has taken the left path at level $l = 1$ and taken the right path at all other levels.

This equivalence suggests the possibility of using standard decision tree induction algorithms to learn discriminative LBP-like descriptors from the training data. Once we have a tree, for any set of pairs $(c, \mathbf{n})$ we may build a histogram from the tree, with the contents of each histogram bin corresponding to a leaf node. The idea of using decision trees to construct discriminative quantizers is in the spirit of [**?**], who use random forests to quantize SIFT descriptors.

Thus we may use the same basic algorithm from Ahonen outlined above, but using trained decision trees in place of the standard LBP operator in step 2.

Regarding the details of the tree training process, we use a simple version of Quinlan's classic ID3 algorithm [**?**] to induce the trees. The trees are recursively built top-down by splitting the pairs $(c, \mathbf{n})$ in each node with a decision of the form $c < n_i$, with $1 \leq i \leq s$. The decision is chosen greedily at each node via the usual entropy gain criterion:

$$\arg\max_i \Delta H(n_i) = H_n - p_r H_r - p_l H_l$$

Where $H_n$ is the entropy of the current node, $H_r$ and $H_l$ are the entropies of the left and right nodes induced by the decision $n_i$. Likewise, and $p_r$ and $p_l$ are the proportion of $(c, \mathbf{n})$ pairs that go to the left and right nodes according to the decision. This criterion favors splits that discriminate between pairs from each class. The data is split until a minimum number of pairs per node or a maximum depth are reached.

It is also worth noting that for this algorithm the neighborhood $\mathbf{n}$ is defined somewhat differently than for LBPs. We use a square neighorhood centered around $c$, and insted of samples taken along a circle we consider all pixels inside the square as part of the neighborhood.

The main parameters of this algorithm are the size of the neighborhood $\mathbf{n}$ of $c$ to explore, and the maximum depth of the trees. The first parameter is determined by the side length in pixels, of a square centered on the pixel $c$. All the pixels within this square are considered as potential split candidates. The second parameter, tree depth, determines the size of the resulting histograms. Smaller histograms are desirable for space and time efficiency, but there is a possible trade-off in accuracy with respect to larger histograms.

One benefit of this scheme is that we may explore larger neighborhoods than those used by LBPs, since an LBP corresponds to a complete tree to $2^s$ values – our scheme may use a subset of those $2^s$ values. Furthermore, we may train a separate tree for each grid cell, taking into account the fact that different features may be discriminative in different facial regions.

While we have explored alternatives to the ID3 algorithm, such as the use of random forests, these have yielded results no better than ID3 and we omit these results.

Table I
RESULTS FOR AT&T-ORL DATABASE

| Method | Accuracy(%) |
|---|---|
| $Eig$ | 94 |
| $LBP_8^1$ | 96.1 |
| $LBP_8^2$ | 96.4 |
| $DTID3_2^1$ | 96.25 |
| $DTID3_2^2$ | 95.25 |
| $DTID3_3^2$ | **97.5** |
| $DTID3_2^3$ | 95 |
| $DTID3_3^3$ | 95.2 |
| $DTID3_4^3$ | 95.2 |
| $DTID3_5^3$ | 96.25 |
| $DTID3_3^4$ | 97.4 |

## II. METHODS

We have implemented Ahonen's procedure outlined above with the LBP algorithm, as well as our proposed variation with decision trees replacing the standard LBP operator.

For Ahonen with LBP we report results using the standard settings of $s = 8$ with $r = 1$ and $r = 2$, as well as the also standard partitioning of the image into an $8 \times 8$ grid.

For our decision tree-based variation, We use the same $8 \times 8$ grid as above. We present results with various combinations of neighborhood sizes and depths to give a notion of how the algorithm behaves under different settings.

We also evaluate the classic Eigenfaces holistic algorithm [**?**]. The main parameter of the algorithm is the number of eigenfaces. We report the best result, obtained with 40 eigenfaces.

We evaluate our algorithm on the AT&T-ORL dataset, a well known and freely available face recognition dataset. This dataset consists of photographs of 40 subjects, each represented by 10 images. To evaluate recognition accuracy we perform 10 experiments, randomly choosing 5 images per individual as gallery and 5 as probes in each experiment. We report the mean accuracy obtained over the 10 rounds.

## III. RESULTS

Table **??** summarizes the mean accuracies obtained for each algorithm, where $Eig$ corresponds to the where $LBP_r^s$ corresponds to LBPs with $s$ sampling points and radius $r$, and $DTID3_d^r$ corresponds to ID3-learned decision trees with maximum depth $d$ and square side-length $2r$.

## IV. Discussion

Both algorithms perform very similarly on this database. The best result, by a small margin, is obtained using the $DTID3$ algorithm with trees of maximum depth $d = 3$ and square side-length 4.

We note that since each LBP histogram has $2^8 = 256$ bins, and one is created for each of the 64 grid cells, the resulting concatenated histogram has $256 * 64 = 16384$ bins. In our algorithm, since the trees are usually grown to their maximum depth, a tree with $d$ will usually create a histogram of size $2^d$, so the concatenated histogram will have $2^d * 64$ bins.

With this observation we can see our algorithm is capable of creating histograms that are slightly more discriminative than those made by LBPs with only a small fraction of the bins. For example, $DTID3_3^2$ has $2^3 * 64 = 512$ bins yet achieves a higher accuracy than the LBP histograms, with 16384 bins. The difference is reduced if we consider a variation of LBPs, known as uniform LBPs [?], which produce smaller histograms – yet the $DTID3$ histograms are still smaller.

While the space savings are substantial, it is somewhat disappointing the accuracy difference isn't larger. Moreover, we can observe that creating larger trees, and therefore larger histograms, does not increase accuracy. It seems there is a "diminishing returns" effect at work, or the larger trees may be over-fitting the data. This will be object of further study.

## V. Conclusions

We have proposed a novel method that uses the training data to create discriminative LBP-like descriptors. The first results indicate this algorithm achieves accuracies comparable to those obtained with classical LBP methods at a fraction of the space requirements. However, further experimentation on other databases is required. We also hope to explore variations on our proposal, such as the use of random forests, to ascertain whether the accuracy can be improved.