Lecture 15
Local Binary Patterns (LBP) & Histogram of Oriented Gradient (HoG)
Local Binary Patterns (LBP)
LBP

• Mainly designed for monochrome still images
  – Have been extended for color (multi channel)
  – Videos ...

LBP

• The local binary pattern operator is an image operator which transforms an image into an array or image of integer labels describing small-scale appearance (textures) of the image.
• These labels directly or their statistics are used for further analysis.
• It is assumed that a texture has locally two complementary aspects, a pattern and its strength
• local binary pattern operator works in a 3×3 pixel
• The pixels in this block are
  – thresholded by its center pixel value,
  – multiplied by powers of two (Decimal)
  – then summed to obtain a label for the center pixel
  – 256 different labels
1. Local Binary Pattern (LBP)

- Description of pixels neighbourhood
- Binary short code to describe neighbourhood
- Operates by taking difference of central pixel with neighbouring pixels
- Mathematically

\[ LBP_{R,P} = \sum_{p=0}^{P-1} s(g_p - g_c) \cdot 2^p \]

where,

- neighborhood pixels \((g_p)\) in each block
- is thresholded by its center pixel value \((g_c)\)
- \(p\) → sampling points (e.g., \(p = 0, 1, \ldots, 7\) for a 3x3 cell, where \(P = 8\))
- \(R\) → radius (for 3x3 cell, it is 1).

Coordinates of “\(g_c\)” is (0,0) and of “\(g_p\)” is \((x + R\cos(2\pi p/P), y - R\sin(2\pi p/P))\)

Binary threshold function \(s(x)\) is,

\[ s(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases} \]
## Computation of Local Binary Pattern

### Neighborhood of a gray-scale image

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

### Binary code for \( g_c \)

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

### Component-wise multiplication

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>128</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>32</td>
<td>16</td>
</tr>
</tbody>
</table>

### Representation

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>128</td>
<td>?</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>16</td>
</tr>
</tbody>
</table>

### Example of how the \( LBP \) operator works

\[
\sum \text{Sum} = 146
\]
Computation of LBP

Binary Pattern: 1 0 0 1 0 1 0 1 (MSB) 0 1 0 0 1 0 0 (LSB)

Code/Weight (2^p): 1 x 2^7 0 x 2^6 1 x 2^5 0 x 2^4 1 x 2^3 0 x 2^2 0 x 2^1 1 x 2^0

= 128 = 0 = 32 = 0 = 8 = 0 = 0 = 1

LBP: 1 + 0 + 0 + 8 + 0 + 32 + 0 + 128 = 169
Fig. 2.3  Different texture primitives detected by the LBP
STANDARD LBP Filter
Advanced LBP \((P,R)\)

\(P = \text{Pixels}\)
\(R = \text{Radius}\)

LBP(8,1)  \hspace{1cm}  LBP(16,2)  \hspace{1cm}  LBP(20,4)
LBP Advantages and disadvantages

Advantages
• High discriminative power
• Computational simplicity
• Invariance to grayscale changes and
• Good performance.

Disadvantages
• Not invariant to rotations
• The size of the features increases exponentially with the number of neighbours which leads to an increase of computational complexity in terms of time and space
• The structural information captured by it is limited. Only pixel difference is used, magnitude information ignored.
LBP: UNIFORM PATTERNS
uLBP

- Uniformity measure $U$ ("pattern") is the number of bitwise transitions from 0 to 1 or vice versa.
- A local binary pattern is called uniform if its uniformity measure is at most 2. i.e transitions between 0 and $1 \leq 2$.
Example

- 00000000 (0 transitions)
- 01110000 (2 transitions)
- 11001111 (2 transitions)
- 11001001 (4 transitions)
- 01010011 (6 transitions)
uLBP

• In uniform LBP mapping there is a separate output label for each uniform pattern and all the non-uniform patterns are assigned to a single label.
• Why Omit non-uniform patterns?
Reasons for omitting non-uniform patterns

• most of the local binary patterns in natural images are uniform
• Ojala et al. noticed that in texture images, uLBP account for
  – 90% of all patterns using the (8,1)
  – 70% in the (16, 2) neighborhood.
• Facial images
  – 90.6% of the patterns in the (8, 1)
  – 85.2% of the patterns in the (8, 2)
• Uniform Value can be found using eq. below

\[ U(LBP_{p,R}) = |s(g_{p-1} - g_c) - s(g_0 - g_c)| + \sum_{p=1}^{P-1} |s(g_p - g_c) - s(g_{p-1} - g_c)| \]

• If \( U \leq 2 \) it is uniform else non-uniform LBP
• Uniform LBP has \( P*(P-1)+2 \) output values
- A total of 58 binary patterns for (8, 1) neighbourhood
- ‘r’ and ‘n’ shows rotation and No. of 1s respectively
uLBP Advantages and disadvantages

**Advantages:**

- considers only the smooth patterns that account for the majority ((90%) for (8,1) and (70%) for (16, 2) neighbourhood) of the total binary patterns
- Only “uniform” patterns are fundamental patterns of local image texture.
- The uniform LBP gives better performance than LBP due to statistical properties of these patterns
- Lower dimensionality of features

**Disadvantages:**

- No rotation Invariant
ROTATION INVARAINCE
Rotation Invariance
Fig. 2.5  Effect of image rotation on points in circular neighborhoods
LBPri

- Rotations of a textured input image cause the LBP patterns to translate into a different location and to rotate about their origin.
**LBPr**

\[ LBP_{P,R}^{r_i} = \min_i ROR(LBP_{P,R}, i) \]

- Where \( ROR(x,i) \) represents circular bitwise right rotation of \( x \) by \( i \) steps.
- 8-bit LBP codes 10000010b, 00101000b, and 00000101b all map to the minimum code 00000101b.
- LBPr is rotation invariant
Example

- An 8-bit patterns $10000001$, $00110000$ and $00001100$ are mapped to a minimum code of $00000011$
- It does not apply to a sequence containing all zeros or all ones

**Advantages:**
- Invariant to scale and rotation

**Disadvantages:**
- Two different images can be misclassified as the same class if they are composed of micro-patterns
4. Rotated LBP

- Problems in RILBP are resolved by RLBP
- Circularly shifting the weights of LBP operator
- Utilize magnitude of difference to find dominant direction in neighbourhood
- Dominant direction is the maximum difference of neighbouring pixels from central pixel
- Dominant direction is set as reference

\[
D = \arg \max_{p \in (0, 1, \ldots, P-1)} |g_p - g_c| \quad \text{(Dominant Direction)}
\]

\[
\text{RLBP}_{R,P} = \sum_{p=0}^{P-1} s(g_p - g_c) \cdot 2^{\text{mod}(|p-D|, P)} \quad \text{(Rotated LBP)}
\]
Advantages
• Invariant to rotation
• High discriminative power

Disadvantages
• Large feature vector size
• Computational complexity
Histogram of Oriented Gradient (HoG)
Histogram of Oriented Gradients (HoG)

[Dalal and Triggs, CVPR 2005]
Histogram of Oriented Gradients (HoG)
Histogram of Oriented Gradients (HoG)

- First used for application of person detection [Dalal and Triggs, CVPR 2005]
- Cited since in thousands of computer vision papers
• Tested with
  – RGB
  – LAB
  – Grayscale

• Gamma Normalization and Compression
  – Square root
  – Log
- Histogram of gradient orientations
  - Orientation
  - Position

- Weighted by magnitude
Input image $\rightarrow$ Normalize gamma & colour $\rightarrow$ Compute gradients $\rightarrow$ Weighted vote into spatial & orientation cells $\rightarrow$ Contrast normalize over overlapping spatial blocks $\rightarrow$ Collect HOG’s over detection window $\rightarrow$ Linear SVM $\rightarrow$ Person / non-person classification

**R-HOG**

Cell

Block

**C-HOG**

Center Bin

Block

Radial Bins, Angular Bins

$L_1 - norm: v \rightarrow \frac{v}{||v||_1 + \epsilon}$

$L_1 - sqrt: v \rightarrow \frac{\sqrt{v}}{||v||_1 + \epsilon}$

$L_2 - norm: v \rightarrow \frac{v}{\sqrt{||v||_2^2 + \epsilon^2}}$

$L_2 - hys: L2$-norm, plus clipping at .2 and renomalizing
HoG Based feature Vector
Gradient Vector Calculation

Final Descriptor Size
7 x 11 x 9 x 4 = 2772
Acknowledgements

- Digital Image Processing”, Rafael C. Gonzalez & Richard E. Woods, Addison-Wesley, 2002
- Peters, Richard Alan, II, Lectures on Image Processing, Vanderbilt University, Nashville, TN, April 2008
- Some slides are taken from Dr. Ali Hassan machine Learning Course