Histogram of oriented gradients (HOG)

Aditya Raj Patidar (0801CS171006)
Raghav Parsai (0801CS171060)
Vijayant Gadria (0801CS171091)

December 13, 2020

Contents

Chapter 1

Introduction

1.1 Feature extraction

Feature extraction is a part of the dimensionality reduction process, in which an initial set of the raw data is divided and reduced to more manageable groups. So when you want to process it will be easier. The most important characteristic of these large data sets is that they have a large number of variables. These variables require a lot of computing resources to process them. So Feature extraction helps to get the best feature from those big data sets by selecting and combining variables into features, thus effectively reducing the amount of data. These features are easy to process, but still able to describe the actual data set with the accuracy and originality.

1.2 Histogram Of Oriented Graphics

Histogram of Oriented Gradients or HOG, is a feature descriptor that is often used to extract features from image data. It is widely used in computer vision tasks for object detection.

- The HOG descriptor focuses on the structure or the shape of an object. ,HOG is able to provide the edge direction as well. This is done by extracting the gradient and orientation (or you can say magnitude and direction) of the edges
- Additionally, these orientations are calculated in 'localized' portions. This means that the complete image is broken down into smaller regions and for each region, the gradients and orientation are calculated. We will discuss this in much more detail in the upcoming sections
- Finally the HOG would geientations of the pixel values, hence the name 'Histogram of Oriented Gradients'
- generate a Histogram for each of these regions separately. The histograms are created using the gradients and or

Chapter 2

Mathematical Formulation

2.1 Formulation

2.1.1 Calculate the Gradient Images

To calculate a HOG descriptor, we need to first calculate the horizontal and vertical gradients Gradients are the small change in the x and y directions
Approximate the two components Ix and Iy of the gradient of I by central differences1 :
$Ix(r, c) = I(r, c + 1) - I(r, c - 1)$ and $Iy(r, c) = I(r - 1, c) - I(r + 1, c)$.

$$\text{Total Gradient Magnitude} = \sqrt{[(G_x)^2+(G_y)^2]}$$

2.1.2 Calculate the orientation (or direction)

Calculate the orientation (or direction) for the same pixel. We know that we can write the tan for the angles:

$$\tan(\Phi) = Gy / Gx$$

Hence, the value of the angle would be:

$$\Phi = \mathrm{atan}(Gy / Gx)$$

If we divide the image into 8×8 cells and generate the histograms, we will get a 9 x 1 matrix for each cell.

2.1.3 Normalize Gradient

To normalize this matrix, we will divide each of these values by the square root of the sum of squares of the values. Mathematically, for a given vector

V = [a1, a2, a3, ….a36]

We calculate the root of the sum of squares:

k = √(a1)2+ (a2)2+ (a3)2+ …. (a36)2

And divide all the values in the vector V with this value k:

$$\text{Normalised Vector} = \left(\frac{a_1}{k}, \frac{a_2}{k}, \frac{a_3}{k}, \dots \frac{a_{36}}{k} \right)$$

The resultant would be a normalized vector of size 36×1.

We would have 105 (7×15) blocks of 16×16. Each of these 105 blocks has a vector of 36×1 as features. Hence, the total features for the image would be 105 x 36×1 = 3780 features.

Chapter 3

Algorithm

3.1 HOG

Algorithm 1 Histogram of oriented gradients

```
Input : Image
1. Evaluate gradient $(G_x, G_y)$ for each pixel in cell $C$.
2. Calculate magnitude $M = \sqrt{[(G_x)^2+(G_y)^2]}$ and orientations $\Phi = \text{atan}(Gy / Gx)$
3. For each cell histogram is built using bilinear interpolation.
4. Normalize the block of cells using L2 normalization.
       V = [a1, a2, a3, ….a36] ($a_i$ : magnitude in particular bin)
       k = √(a1)2+ (a2)2+ (a3)2+ …. (a36)2
```

$$\text{Normalised Vector} = \left(\frac{a_1}{k}, \frac{a_2}{k}, \frac{a_3}{k}, \ldots \frac{a_{36}}{k} \right)$$

Chapter 4

Documentation of API

4.1 Methods

hog(*image,pixels_per_cell,cells_per_block,bins,visualize*)

Parameters

Image : input image
pixels_per_cell : tuple : number of pixels in one cell
cells_per_block : tuple : number of cells in one block
bins : int : number of orientations
visualize : bool : if set to true will return a visualized image.

returns feature vector (nd_array)

calculate_magnitude_orientation(img)

Parameters

img : input image

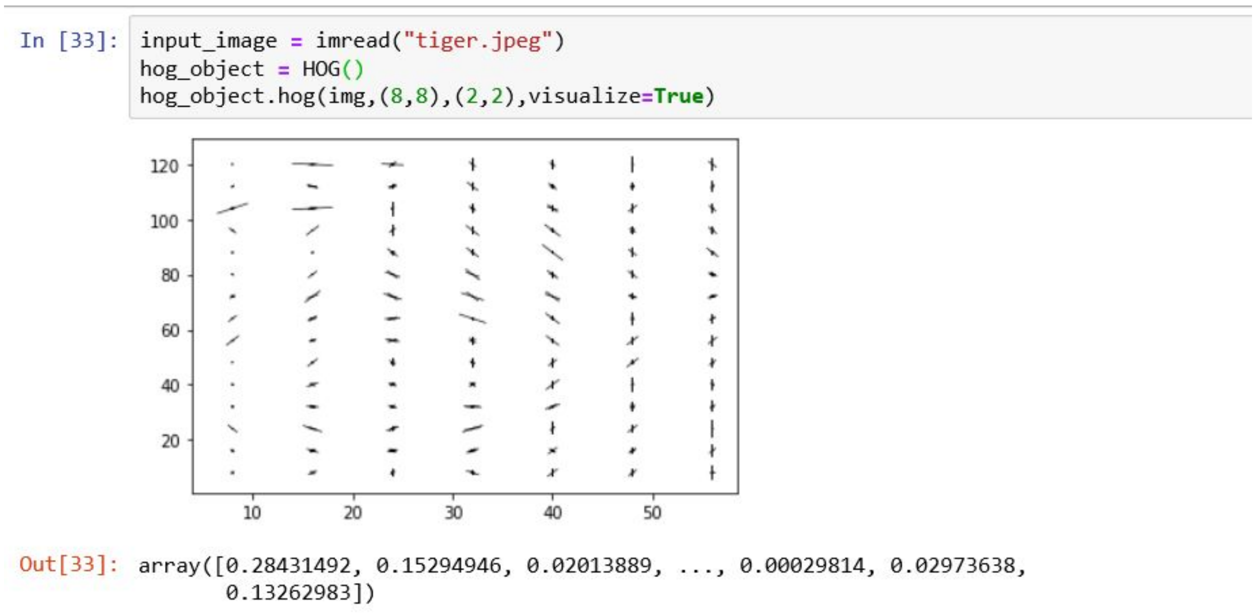
returns magnitude and orientation for each pixels

Chapter 5

Example

5.1 Example 1

input_image = imread("tiger.jpeg")
hog_object = HOG()
hog_object.hog(img,(8,8),(2,2),visualize=True)



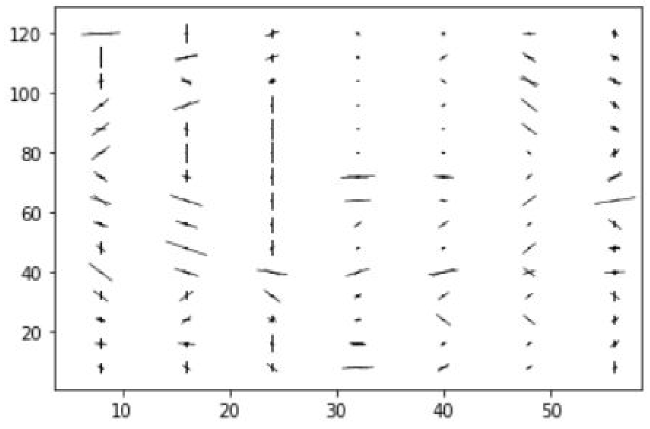
```
In [33]: input_image = imread("tiger.jpeg")
         hog_object = HOG()
         hog_object.hog(img,(8,8),(2,2),visualize=True)
```

```
Out[33]: array([0.28431492, 0.15294946, 0.02013889, ..., 0.00029814, 0.02973638,
                0.13262983])
```

5.2 Example 2

input_image = imread("person.jpeg")
hog_object = HOG()
hog_object.hog(input_image,(8,8),(2,2),visualize=True)

```
In [6]: input_image = imread("person.jpeg")
        hog_object = HOG()
        hog_object.hog(input_image,(8,8),(2,2),visualize=True)
```



```
Out[6]: array([0.68307886, 0.46331277, 0.29091325, ..., 0.02467806, 0.11635696,
               0.32306623])
```

Chapter 6

Learning Outcomes

● Successfully analysed and implemented the Concepts of HOG using two methods of fit and transform.
● Realised the Use of HOG and it's concepts in calculation gradient for each pixel in cell.Evaluate magnitude and orientations
● Used the methods in the package on our locally generated data and got satisfactory results as expected from the analysis of the mathematical equation

References

- https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/
- https://en.wikipedia.org/wiki/Histogram_of_oriented_gradients