Fourier Transform
(Image Feature Extraction)

Prateek Narsinghani (0801CS171057)
Yash Sonkiya (0801CS171094)

Contents

Introduction

1. Feature extraction

Feature extraction involves reducing the number of resources required to describe a large set of data. When performing analysis of complex data one of the major problems stems from the number of variables involved. Analysis with a large number of variables generally requires a large amount of memory and computation power, also it may cause a classification algorithm to overfit to training samples and generalize poorly to new samples. Feature extraction is a general term for methods of constructing combinations of the variables to get around these problems while still describing the data with sufficient accuracy.

2. Fourier Transform

The Fourier Transform is a mathematical technique that transforms a function of time, $x(t)$, to a function of frequency, $X(\omega)$. It is closely related to the Fourier Series.

The Fourier Transform is an important image processing tool which is used to decompose an image into its sine and cosine components. The output of the transformation represents the image in the Fourier or frequency domain, while the input image is the spatial domain equivalent. In the Fourier domain image, each point represents a particular frequency contained in the spatial domain image.

The Fourier Transform is used in a wide range of applications, such as image analysis, image filtering, image reconstruction and image compression. Using different types of filters, we can make images less noisy or can be used in edge detection.

The Fourier transform was proposed by Jean Baptiste Joseph Fourier

3. Discrete Fourier Transform

As we are only concerned with digital images, we will use the Discrete Fourier Transform (DFT). When a signal is discrete and periodic, we do not need the continuous Fourier transform. Instead, we use the discrete Fourier transform, or DFT.

The DFT is the sampled Fourier Transform and therefore does not contain all frequencies forming an image, but only a set of samples which is large enough to fully describe the spatial domain image. The number of frequencies corresponds to the number of pixels in the spatial domain image, *i.e.,* the image in the spatial and Fourier domain are of the same size.

Mathematical Formulation

The Fourier transform of a continuous-time signal x(t) may be defined as ay be defined

$$X(\omega) = \int_{-\infty}^{\infty} x(t) e^{-j\omega t} dt, \qquad \omega \in (-\infty, \infty).$$

The DFT, on the other hand, replaces the infinite integral with a finite sum:

$$X(\omega_k) \triangleq \sum_{n=0}^{N-1} x(t_n) e^{-j\omega_k t_n}, \qquad k = 0, 1, 2, \ldots, N-1,$$

For a square image of size N×N, the two-dimensional DFT is given by:

$$F(k,l) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f(i,j)\; e^{-\iota 2\pi \left(\frac{ki}{N} + \frac{lj}{N}\right)}$$

where f(a,b) is the image in the spatial domain and the exponential term is the basis function corresponding to each point F(k,l) in the Fourier space. The equation can be interpreted as: the value of each point F(k,l) is obtained by multiplying the spatial image with the corresponding base function and summing the result.

The basis functions are sine and cosine waves with increasing frequencies, i.e. F(0,0) represents the DC-component of the image which corresponds to the average brightness and F(N-1,N-1) represents the highest frequency.

In a similar way, the Fourier image can be re-transformed to the spatial domain. The inverse Fourier transform is given by:

$$f(a,b) = \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} F(k,l)\; e^{\iota 2\pi \left(\frac{ka}{N} + \frac{lb}{N}\right)}$$

To obtain the result for the above equations, a double sum has to be calculated for each image point. However, because the Fourier Transform is separable, it can be written as

$$F(k,l) = \frac{1}{N} \sum_{b=0}^{N-1} P(k,b)\; e^{-\iota 2\pi \frac{lb}{N}}$$

where

$$P(k,b) = \frac{1}{N}\sum_{a=0}^{N-1} f(a,b)\ e^{-\iota 2\pi \frac{ka}{N}}$$

Using these two formulas, the spatial domain image is first transformed into an intermediate image using N one-dimensional Fourier Transforms. This intermediate image is then transformed into the final image, again using N one-dimensional Fourier Transforms. Expressing the two-dimensional Fourier Transform in terms of a series of 2N one-dimensional transforms decreases the number of required computations.

Algorithm

Since image is a 2d object we can calculate the DFT of it by first calculating DFT column wise then row wise.

Algorithm: - Fourier Transform
Input: - An image

1. First find the DFT of the image column wise (first calculate the transpose) using the formula

$$P(k,b) = \frac{1}{N} \sum_{a=0}^{N-1} f(a,b)\ e^{-\iota 2\pi \frac{ka}{N}}$$

2. Then find the DFT of the image across row wise (on the result obtained from the first step) using the formula

$$F(k,l) = \frac{1}{N} \sum_{b=0}^{N-1} P(k,b)\ e^{-\iota 2\pi \frac{lb}{N}}$$

3. Then the components with low frequency are shifted to center and high frequency components at the corners.
4. The magnitude spectrum is the adjusted by taking log of the transformed image.

Documentation of API

Package organization

from feature_extraction import fourier_transform

Methods

fourier_transform(image)

Arguments: -

an image

Output: -

magnitude_spectrum: the Fourier Transform magnitude spectrum (shifted and adjusted magnitude)

Example

```
img = cv2.imread("/content/a_0.png",0)
magnitude_spectrum = fourier_transform(img)

plt.subplot(121),plt.imshow(img, cmap = 'gray')
plt.title('Input Image'), plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(magnitude_spectrum, cmap = 'gray')
plt.title('Magnitude Spectrum'), plt.xticks([]), plt.yticks([])
plt.show()
```

Learning Outcome

- From this we learnt how to extract features of image
- We learnt how to calculate Fourier transform of discrete object such as image
- We learnt how to calculate DFT of a 2d image
- Fourier transform can be used to detect edges in a image
- Fourier Transform can be used to make high pass and low pass filters

References

- https://ccrma.stanford.edu/~jos/st/Introduction_DFT.html
- https://homepages.inf.ed.ac.uk/rbf/HIPR2/fourier.htm
- https://www.robots.ox.ac.uk/~sjrob/Teaching/SP/l7.pdf
- https://medium.com/@hicraigchen/digital-image-processing-using-fourier-transform-in-python-bcb49424fd82
- https://youtu.be/v743U7gvLq0