AI in Image Processing

A Project Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of

Bachelor of Technology

by

Kaushal Kishore
(111601008)

under the guidance of

Dr. Chandra Shekar

INDIAN INSTITUTE
OF TECHNOLOGY
PALAKKAD

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

<u>CERTIFICATE</u>

*This is to certify that the work contained in this thesis entitled "**AI in Image Processing**" is a bonafide work of **Kaushal Kishore (Roll No. 111601008)**, carried out in the Department of Computer Science and Engineering, Indian Institute of Technology Palakkad under my supervision and that it has not been submitted elsewhere for a degree.*

Dr. Chandra Shekar

Assistant Professor

Department of Computer Science & Engineering

Indian Institute of Technology Palakkad

Acknowledgements

Write acknowledgements, if your want to.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Image processing libraries these days (eg. Open CV) uses the conventional methods which have the possibility to be outperformed by methods which leverage the power of artificial intelligence. Some recent research have shown that some of these AI based methods are able to perform atleast as good as conventional approaches. The aim of this project is to implement, apply and possibly improve upon the existing approaches in Digital Image Processing and Computer Vision. These common tasks can include (not limited to) applications like: Image Compression, Denoising, Super Resolution, Flow Estimation, Object Detection, etc.

1.1 Image Processing

Image processing is manipulating an image in order to enhance it or extract information from it. It is widely useed in medical visualization, biometrics, self-driving vehicles, gaming, surveillance, and law enforcement. It can used in various ways: visualization, restoration, imformation retrieval, pattern recognition, etc.

General approach of image processing involves eight key phases: image acquisition, image enhancement, image restoration, color space transformation, compression or decom-

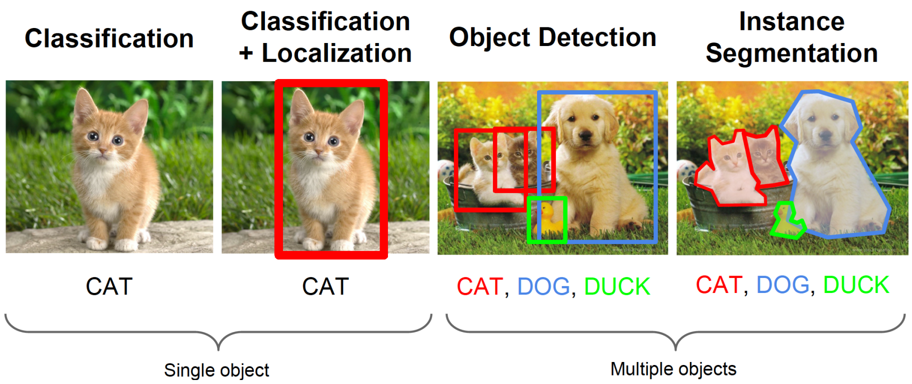

Fig. 1.1: Examples of pattern recognition
Source: www.cs.cornell.edu

pression, morphological processing, recognition, and representation. It is very difficult to carry out these steps manually on a very big data, this is where AI and ML algorithms become very helpful.

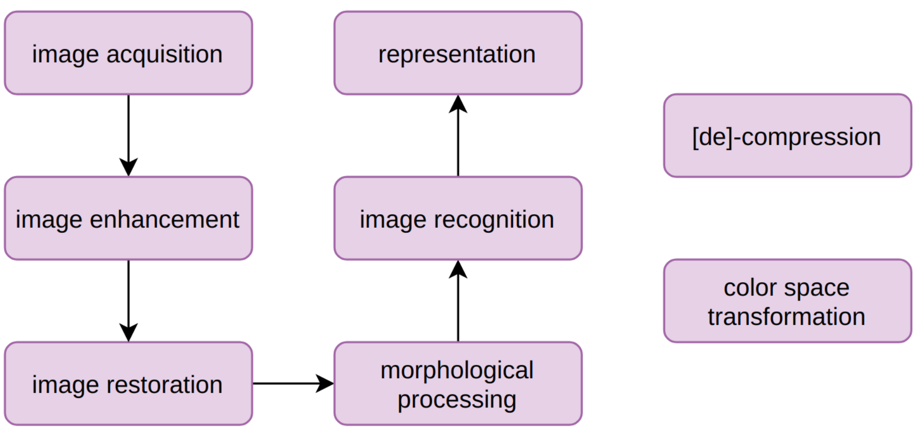

Fig. 1.2: Key phases of image processing

1.2 Recent advances

Modern AI algorithms have enabled computer to perform detection, segmentation, recognition, compression, extraction, generation, and discrimination. Every year a state of the art model is invented to solve the existing problem in a better way. It is now an established fact that machines are now better than humans in counting, classifying and segmenting instances.

1.3 Organization of The Report

Chapter 2

Image Compression

A data compression algorithm transforms the data to occupy a less space. The original data is encoded by a program called encoder, to a compressed representation using a fewer number of bits. Decoder is responsible for decompressing the compressed representation.

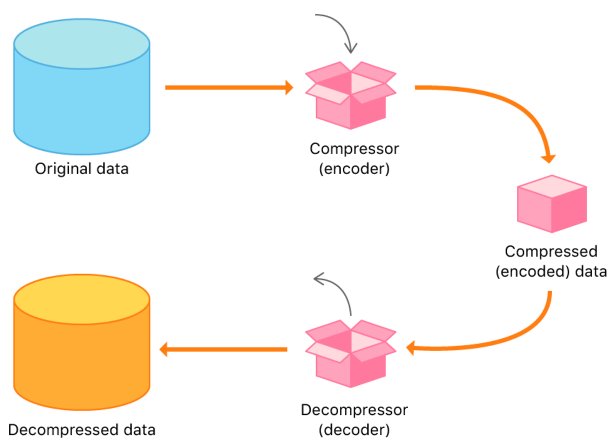

Fig. 2.1: Compression phases
Source: `https://developer.apple.com/documentation/compression`

Image compression is very crucial in order to reduce the size of disk space used as well as reduce the amount of internet bandwidth used while loading images. It's also important to compress images for people accessing the internet via low bandwidth connections.

2.1 Lossy vs. Lossless

The compression technique where the decompressed data is exactly same as original data is called as lossless compression otherwise it is known as lossy compression technique because some information is lost during coding-encoding phase. Two well-known codecs for image compression are JPEG and PNG. PNG is lossless and JPEG is lossy.

Lossy	Lossless
A compression that permits reconstruction only of an approximation of the original data, though usually with an improved compression rate.	A class of data compression that allows the original data to be perfectly reconstructed from the compressed data.
Reduces the quality.	Does not reduce the quality.
Data reduction is higher.	Data reduction is lower.
Commonly used to compress multimedia data such as audio (MP3), video and image (JPEG) files.	Used commonly for text, data files, etc.

Table 2.1: Lossy vs. Lossless

2.2 Dimensionality Reduction PCA

Principal components analysis (PCA) is one of a family of techniques for taking high-dimensional data, and using the dependencies between the variables to represent it in a more tractable, lower-dimensional form, without losing too much information. PCA is one of the simplest and most robust ways of doing such dimensionality reduction.

PCA is mathematically defined as an orthogonal linear transformation that transforms the data to a new coordinate system such that the greatest variance by some scalar projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on.

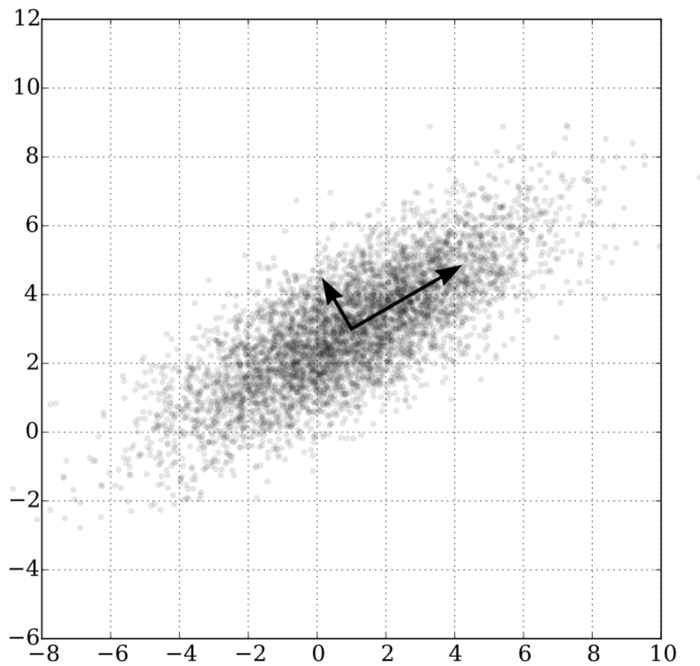

Fig. 2.2: PCA of a multivariate Gaussian distribution centered at (1,3) with a standard deviation of 3 in roughly the (0.866, 0.5) direction and of 1 in the orthogonal direction. **Source:** `https://en.wikipedia.org/wiki/Principal_component_analysis`

Let W be a $d \times d$ matrix whose columns are the principal components of X. The transformation $T = XW$ maps a data vector $x_{(i)}$ from an original space of d variables to a new space of d variables which are uncorrelated over the dataset. However, not all the principal components need to be kept. Keeping only the first L principal components, produced by using only the first L eigenvectors, gives the truncated transformation:

$T_L = XW_L$

where the matrix T_L now has n rows but only L columns. In other words, PCA learns a linear transformation $t = W^T x, x \in \Re^d, t \in \Re^L$, where the columns of $d \times L$ matrix W form an orthogonal basis for the L features (the components of representation t) that are decorrelated. By construction, of all the transformed data matrices with only L columns, this score matrix maximises the variance in the original data that has been preserved, while minimising the total squared reconstruction error $\left\|TW^T - T_L W_L^T\right\|_2^2 \, or \, \|X - X_L\|_2^2$.

The basic steps for computing the PCA is as follows:

1. Standardize the d-dimensional dataset.
2. Construct the covariance matrix.
3. Decompose the covariance matrix into its eigenvectors and eigenvalues.
4. Sort the eigenvalues by decreasing order to rank the corresponding eigenvectors.
5. Select L eigenvectors which correspond to the L largest eigenvalues, where L is the dimensionality of the new feature subspace $L \leq d$.
6. Construct a projection matrix W_L from the "top" L eigenvectors.
7. Transform the d-dimensional input dataset X using the projection matrix W_L to obtain the new L-dimensional feature subspace.

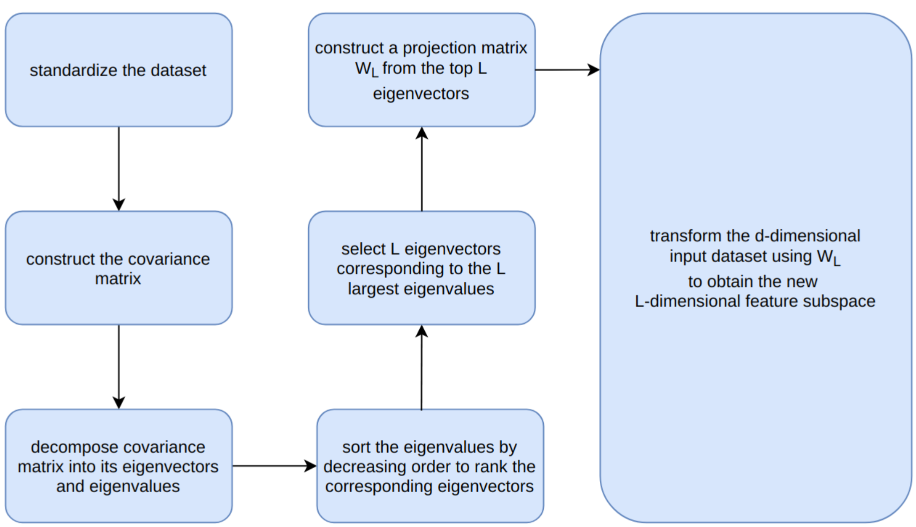

Fig. 2.3: Steps for dimensionality reduction using PCA

2.3 Image compression using PCA

We will follow the steps mentioned in the previous section for dimensionality reduction using PCA. The results of the experiments are tabulated below:

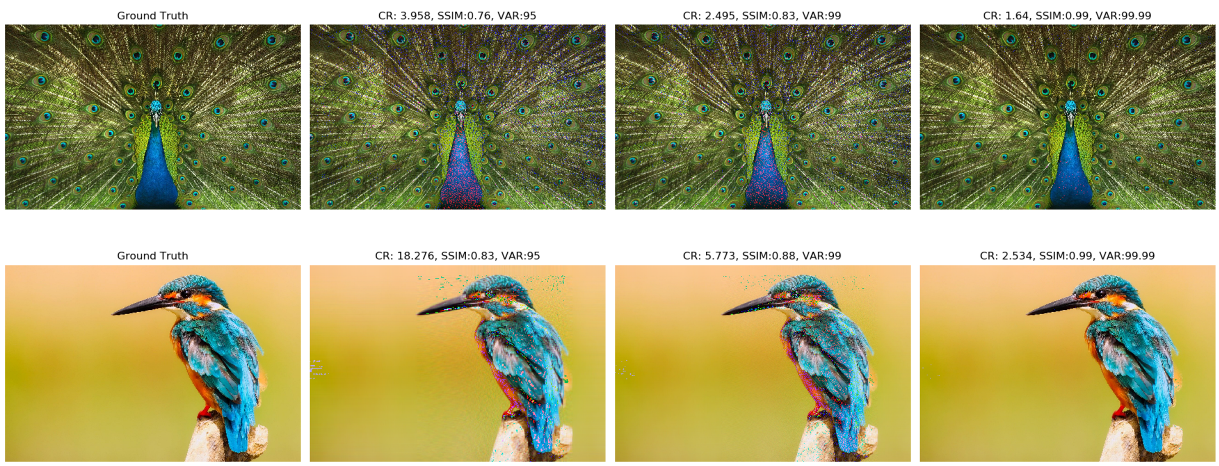

Fig. 2.4: CR = Compression Ratio, SSIM = Structural Similarity Index, VAR = Variance. Results of compression using the PCA method. Higher variance leads to more number of principal components and higher is the reconstructed image quality and lower the compression rate.

Code for the above test can be found here: @KishoreKaushal/ImageCompression/

It is clear from the above experiment that higher variance leads to more number of principal components and higher is the reconstructed image quality and lower the compression rate. Infact, the first L principal components is selected to get atleast the given number of variance.

In the next chapter we will discuss the JPEG compression algorithm.

Chapter 3

JPEG

3.1 Overview

JPEG, which stands for Joint Photographic Experts Group (the name of the committee that created the JPEG standard) is a lossy compression algorithm for images. A lossy compression scheme is a way to inexactly represent the data in the image, such that less memory is used yet the data appears to be very similar. This is why JPEG images look almost the same as the original images they were derived from most of the time unless the quality is reduced significantly, in which case there will be visible differences.

The JPEG algorithm takes advantage of the fact that humans can't see colors at high frequencies. These high frequencies are the data points in the image that are eliminated during the compression. JPEG compression also works best on images with smooth colour transitions.

JPEG is a commonly used method of lossy compression for digital images. The degree of compression can be adjusted, allowing a the tradeoff between storage size and image quality with a compression ratio 10:1 but with little perceptible loss in image quality.

Fig. 3.1: A photo of a European wildcat with the compression rate decreasing and hence quality increasing, from left to right.
Source: `https://en.wikipedia.org/wiki/JPEG`

3.2 Algorithm

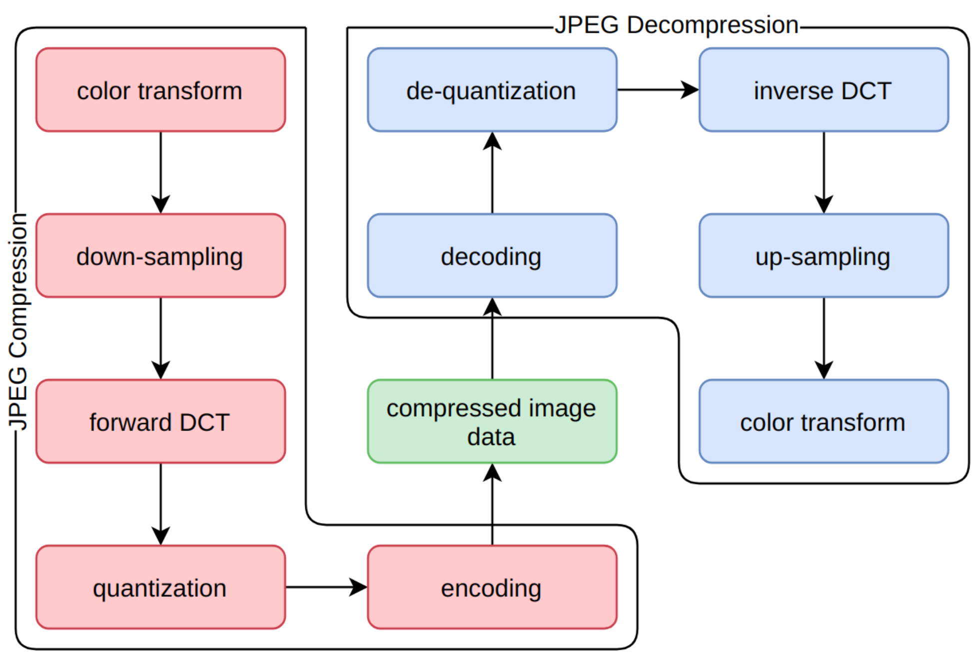

Fig. 3.2: JPEG Schematic

3.2.1 Splitting

JPEG uses transform coding, it is largely based on the following observations:

- Usually image contents change relatively slowly across images, i.e., it is unusual for intensity values to alter up and down several times in a small area, for example, within an 8×8 image block. A translation of this fact into the spatial frequency domain, implies, generally, lower spatial frequency components contain more information than the high frequency components which often correspond to less useful details and noises.
- Experiments suggest that humans are more immune to loss of higher spatial frequency components than loss of lower frequency components. Human vision is insensitive to high frequency components.

Hence, first step is to split the image into 8×8 blocks non-overlapping pixel blocks. If the image cannot be divided into 8×8 blocks then the block is zero padded.

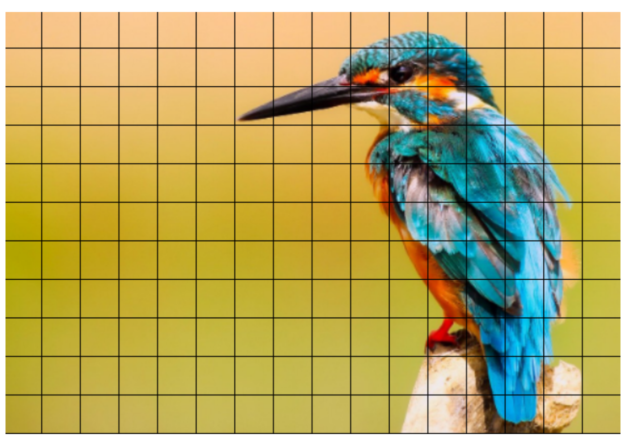

Fig. 3.3: Splitting into 8×8 blocks

3.2.2 Color Space Transform

JPEG makes use of $[Y, Cb, Cr]$ model instead of $[R, G, B]$ model. There is a reason for using the color space. The human eye is more sensitive to luminance than to chrominance.

$Y = 0.299R + 0.587G + 0.114B$
$Cb = -0.1687R - 0.3313G + 0.5B + 128$
$Cr = 0.5R - 0.4187G - 0.0813B + 128$
$R = Y + 1.402(Cr - 128)$
$G = Y - 0.34414(Cb - 128) - 0.71414(Cr - 128)$
$B = Y + 1.772(Cb - 128)$

Table 3.1: RGB and YCbCr conversion table

3.2.3 Down-sampling

After the color space transformation, the image is downsampled. Downsampling happens in such a way that Y is taken for each pixel whereas Cb and Cr is taken for 2×2 blocks.

Sometimes this steps is not carried out.

3.2.4 Discrete Cosine Transform

The key to the JPEG baseline compression process is a mathematical transformation known as the Discrete Cosine Transform (DCT). The DCT is in a class of mathematical operations that includes the well known Fast Fourier Transform (FFT), as well as many others. The basic purpose of these operations is to take a signal and transform it from one type of representation to another. For example, an image is a two-dimensional signal that is perceived by the human visual system. The DCT can be used to convert the signal (spatial information) into numeric data ("frequency" or "spectral" information) so that the image's information exists in a quantitative form that can be manipulated for compression.

The DCT is performed on an $N \times N$ square matrix of pixel values, and it yields an $N \times N$ square matrix of frequency coefficients. (In practice, N most often equals 8 because a larger block, though would probably give better compression, often takes a great deal of time to perform DCT calculations, creating an unreasonable tradeoff. As a result, DCT implementations typically break the image down into more manageable 8×8 blocks.) The

DCT formula looks somewhat intimidating at first glance but can be implemented with a relatively straightforward piece of code.

$$DCT[i,j] = \frac{1}{\sqrt{2N}}\, C[i]\, C[j] \, \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} I[x,y]\, cos[\frac{(2x+1)i\pi}{2N}]\, cos[\frac{(2y+1)j\pi}{2N}] \tag{3.1}$$
$$where\ C(x) = \frac{1}{\sqrt{2}}\ if\ x = 0,\ else\ 1\ if\ x > 0$$

$$\begin{bmatrix} 140 & 144 & 147 & 140 & 140 & 155 & 179 & 175 \\ 144 & 152 & 140 & 147 & 140 & 148 & 167 & 179 \\ 152 & 155 & 136 & 167 & 163 & 162 & 152 & 172 \\ 168 & 145 & 156 & 160 & 152 & 155 & 136 & 160 \\ 162 & 148 & 156 & 148 & 140 & 136 & 147 & 162 \\ 147 & 167 & 140 & 155 & 155 & 140 & 136 & 162 \\ 136 & 156 & 123 & 167 & 162 & 144 & 140 & 147 \\ 148 & 155 & 136 & 155 & 152 & 147 & 147 & 136 \end{bmatrix} \tag{3.2}$$

$$\begin{bmatrix} 140 & 144 & 147 & 140 & 140 & 155 & 179 & 175 \\ 144 & 152 & 140 & 147 & 140 & 148 & 167 & 179 \\ 152 & 155 & 136 & 167 & 163 & 162 & 152 & 172 \\ 168 & 145 & 156 & 160 & 152 & 155 & 136 & 160 \\ 162 & 148 & 156 & 148 & 140 & 136 & 147 & 162 \\ 147 & 167 & 140 & 155 & 155 & 140 & 136 & 162 \\ 136 & 156 & 123 & 167 & 162 & 144 & 140 & 147 \\ 148 & 155 & 136 & 155 & 152 & 147 & 147 & 136 \end{bmatrix}$$

3.3 Conclusion

In this chapter, we proposed a distributed algorithm for construction of xyz. The complexity of this algorithm is $O(n \log n)$. Next chapter presents another distributed algorithm which has linear time complexity based on xyz.

Chapter 4

Algorithm II

The algorithm presented in previous chapter has $O(n)$ time complexity. We further propose another distributed algorithm in this chapter based on xyz which has linear time complexity.

4.1 Construction

Write ...

4.2 Improved Method

Write...

4.3 Conclusion

In this chapter, we proposed another distributed algorithm for XYZ. This algorithm has both time complexity of $O(n)$ where n is the total number of nodes. In next chapter, we conclude and discuss some of the future aspects.

Chapter 5

Conclusion and Future Work

write results of your thesis and future work.

References