IMPROVING THE PERFORMANCE OF RECOMMENDER SYSTEMS USING A PARALLEL CBIR METHOD

Egyetemi doktori (PhD) értekezés

Mohammadreza Azodinia

témavezető neve: Dr. András Hajdu

DEBRECENI EGYETEM

Természettudományi Doktori Tanács

Informatikai Tudományok Doktori Iskola

Debrecen, 2016

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Hungary, Debrecen, October 2016

Mohammadreza Azodinia

This is to certify that the thesis entitled "Improving the performance of recommender systems using a parallel content based image retrieval method" submitted by Mohammad Reza Azodinia to University of Debrecen for the award of the degree of Doctor of Philosophy is a bona fide record of the research work carried out by him under my supervision and guidance. The content of the thesis, in full or parts have not been submitted to any other Institute or University for the award of any other degree or diploma.

Hungary, Debrecen, October 2016

Dr. András Hajdu

IMPROVING THE PERFORMANCE OF RECOMMENDER SYSTEMS USING A PARALLEL CBIR METHOD

Értekezés a doktori (Ph.D.) fokozat megszerzése érdekében
a informatika tudományágban

Írta: Mohammadreza Azodinia okleveles informatikus

Készült a Debreceni Egyetem Informatikai Tudományok doktori iskolája
(Diszkrét matematika, adatfeldolgozás és vizualizáció programja) keretében

Témavezető: Dr. András Hajdu

A doktori szigorlati bizottság:

elnök: Dr. Pethõ Attila

tagok: Dr. Miklós Hoffmann

Dr. Gábor Fazekas

A doktori szigorlat időpontja: 2015. October 19.

Az értekezés bírálói:

Dr. ....

Dr. ....

Dr. ....

A bírálóbizottság:

elnök: Dr. ....

tagok: Dr. ....

Dr. ....

Dr. ....

Dr. ....

Az értekezés védésének időpontja: 201… . …… … .

Acknowledgment

I would like to express my special appreciation and thanks to my supervisor Dr. András Hajdu, you have been a tremendous mentor for me. I would like to thank you for encouraging my research and for allowing me to grow as a research scientist. Your advice on both research as well as on my career have been priceless. I would also like to thank my committee members, Professor János Sztrik, Dr. Marianna Zichar, and Dr. Gábor Szũcs for serving as my committee members even at hardship. I also want to thank you for letting my defense be an enjoyable moment, and for your brilliant comments and suggestions, thanks to you.

A special thanks to my family. I would like express appreciation to my beloved wife Dr. Sareh Abbasabadi who spent sleepless nights with and was always my support in the moments when there was no one to answer my queries. Words cannot express how grateful I am to my father, Mr. Enayatollah Azodinia, my mother, Mrs. Molood Sarkhosh, my father-in-law, Mr. Ebrahim Abbasabadi, mother-in-law, Mrs. Zahra Gharleghi for all of the sacrifices that you've made on my behalf. Your prayer for me was what sustained me thus far. I would also like to thank all of my friends who supported me in writing, and incented me to strive towards my goal.

Abstract

Nowadays, many technologies based on the computer science have rapidly improved the quality of every part of our lives. These technologies are utilized in many different areas, use different techniques, and are based on different concepts; however, they typically have a single goal in common that is to make life easier and improve its quality. Today, biggest companies are offering their services and products based on the computerized technologies. In the current competitive market, giving the users a better experience working with the website, application, or any other software that a company uses to communicate with its users can be highly valuable to that company.

Recommender systems are among the most valuable automatic tools that can improve the user experience with a more general system considerably when embedded inside that system. Since the mid-nineties and slightly after that, when the term "recommender system" coined by Resnick and Varian, researchers have conducted huge studies on this topic and the methods and approaches they have found are various and sundry. A recommender system serves the users in many ways, for instance, it shows the items to the users of a retailer's website in an order which is probably more compatible with their attitudes. One of the major factors that affects one's decisions and attitude towards an item is the image that describes the item or is highly related to it. Therefore, it seems intuitive that recommender systems consider images in their methods; however, this not the case in general. In this research we have proposed an approach that measures the similarity of the items considering both its common features, which are considered as numerical and categorical, and the image attached to that item. One big issue that treats the feasibility and practicality of such an approach is the difference between the computation time of similarity measurement considering these two categories of features. In order to make this system practical we have used a distributed approach which considerably reduces the computations related to the image part. The results

show the effectiveness of this approach despite the simple similarity measure that we have used to compute the similarity of images.

Table of Contents

Chapter 1 Introduction 3

1.1 Motivation and Overview _____ 4

1.2 The Organization of Thesis_____ 5

Chapter 2 Parallel Programming 6

2.1 Overview of parallel programming _____ 7

2.1.1 Beyond the Moor's Law 10

2.1.2 Multi-core processors 10

2.1.3 Parallel computing challenges 11

2.2 Performance measures_____ 13

2.2.1 Work and span 14

2.2.2 Speedup 15

2.2.3 FLOPS 21

2.2.4 Efficiency 22

2.3 Parallel systems_____ 23

2.3.1 SISD 23

2.3.2 MISD 24

2.3.3 SIMD 26

2.3.4 MIMD 27

Chapter 3 Image processing taxonomy 29

3.1 A General Categorization of Image Processing 30

3.2 Low Level Image Processing Algorithms _____ 30

3.2.1 Point Operators 35

3.2.2 Neighborhood Operators 37

3.2.3 Global Operators 41

3.3 Intermediate Level Image Processing_____ 41

3.4 High level image processing_____ 43

Chapter 4 Recommender Systems 45

4.1 Recommender Systems 46

4.2 Recommender System Paradigms 48

4.2.1 Content-Based Filtering 49

4.2.2 Collaborative Filtering 50

4.2.3 Hybrid Recommender Systems 55

Chapter 5 Content Based Image Retrieval 57

5.1 The Introduction of the CBIR 58

5.2 Architecture of CBIR Systems 59

5.3 Features utilized by CBIR systems 60

5.3.1 Color Features 61

5.4 Invariance 63

5.5 Distance Measures 64

5.5.1 Euclidean Distance 64

5.5.2 Mahalanobis Distance 64

5.5.3 Jaccard Distance 65

Chapter 6 Proposed method 66

6.1 Related Works 67

6.2 Proposed Method 69

6.3 Experimental results 76

Chapter 7 The Proposed Image Retrieval Methods 82

7.1 First Proposed Image Retrieval Method 83

7.1.1 The Proposed Method 83

7.1.2 Experimental Results 86

7.2 The Second Proposed Image Retrieval Method 88

7.2.1 Features 89

7.2.2 Weight adjustment 91

7.2.3 Ecperiments and Results 94

7.3 Utilization of the Image retrieval (IR) Methods 97

Chapter 8 Conclusion and Future Works 100

8.1 Summary _____ 101

8.2 Suggestion and future works _____ 102

References 103

Appendix A 115

Appendix B 116

List of Figures

Figure 2-1 Calculation of *work* for a typical program [20]. 14
Figure 2-2 Calculation of span for a typical program [20]. 15
Figure 2-3 Speedup curves. 17
Figure 2-4 The maximum speedups possible with some typical settings [26]. 19
Figure 2-5 The various speedup values corresponding to some typical values of α [18]... 20
Figure 2-6 The trend in improvement of performance of GPU and CPU over the past years, in terms of FLOPS [19]. 22
Figure 2-7 Flynn's taxonomy. 23
Figure 2-8 An SISD computer architecture. 24
Figure 2-9 An MISD computer architecture. 25
Figure 2-10 An SIMD computer architecture. 27
Figure 2-11 An MIMD computer architecture. 28
Figure 3-1 A sample of a low level point operator. 34
Figure 3-2 A sample of a low level neighborhood operator. 34
Figure 3-3A sample of a low level global operator. 34
Figure 3-4 An arbitrary image (left), along with the resulting image from thresholding (right) [43]. 35
Figure 3-5 An arbitrary image (left), along with the resulting image from brightness addition (right) [34]. 36
Figure 3-6 An arbitrary image (left), along with the resulting image from brightness subtraction (right) [34]. 36
Figure 3-7 An arbitrary image (left) along with its contrast adjusted counterpart (right) [34]. 37
Figure 3-8 An example of b) gray-level along with c) colored region labeling applied on an arbitrary image (a) [47]. 42
Figure 5-1 Architecture of CBIR system [105]. 60
Figure 6-1The general architecture of the system. 70
Figure 6-2 A sample chromosome with arbitrary values for α and β. 72
Figure 6-3 The parallelized version of the IMNCC method using MapReduce. 74
Figure 6-4 An item from the data set use in the experiments. 78
Figure 6-5 NRMSE metric measured for different approaches. 80
Figure 6-6 The overhead due to measuring the graphical similarity, i.e., considering the images explicitly. 80
Figure 6-7 Comparison of query time between proposed method and exhaustive search-based method for different number of items. 81
Figure 7-1 The block diagram of the layer extraction system. 85
Figure 7-2. A number of pictures selected from the dataset used to evaluate the proposed method. 87
Figure 7-3 The average precision graph depicted for the top 20 retrieved images. 95
Figure 7-4 The average time taken for the retrieval of images by different methods. 96
Figure 7-5 NRMSE metric measured for different approaches. 99

List of Tables

Table 4-1 A comparison between three collaborative filtering approaches. 55
Table 6-1 text-based features of items. 77
Table 6-2 Characteristics of each user. 78
Table 7-1The results of experiments conducted using the proposed method (Pr) and the color correlogram (CC) method. 88
Table 7-2 The orders of the images retrieved using each feature separately at the stage t. 92
Table 7-3 The orders of the images retrieved using each feature separately at the stage *t+1*. 92
Table 7-4 The rankings of the first 4 images according to each feature. 93
Table 7-5 Comparison of the precision of the proposed method with other methods for the top 20 retrieved images. 96

Chapter 1

Introduction

1.1 Motivation and Overview

An everlasting goal of the computerized systems has been providing users with a more desirable experience using the system. This goal has driven many researches in various domains of science. Regardless of how the user desirability is defined in a system, the unchanged fact is that the user desirability is one of the common goals among most of the systems using computer to provide a service. For instance, machine learning methods along with speech processing methods can help the experts to design answering machines which give people the illusion that the machine can really understand them, so they feel more comfortable communicating the system. Therefore, there is a cycle in which the improvements cause to further researches and the results of those researches in turn result in further improvements and increase the users' expectations, as well.

Recommender systems are among the most promising components that can increase the effectiveness of a system hiring them through improvement in the user experience. Since the mid-nineties and slightly after that, when the term "recommender system" coined by Resnick and Varian, researchers have conducted huge studies on this topic, and the methods and approaches they have found are various and sundry.

Despite the reasonable and undeniable achievements in the domain of recommender systems, many questions still remain unanswered and need to be addressed. One of these issues that, to the best of our knowledge, is neglected in most of the researches about recommender systems is related to the fact that a huge part of data around us is in the image format, however, the researchers usually ignore the images and focus solely on the text-based data. For instance, movies are proposed to users just based on some textually representable information extracted from other users or suchlike. This approach works well in many scenarios, nevertheless, this is not the case in general. To bring an example, one of the most important factors to people when deciding to choose a house is its appearance which is obviously a pictorial data.

Therefore ranking houses just based on text-based features does not seem to be appropriate enough.

In this research we have focused on the aforementioned issue, i.e., considering images as part of the data describing items which should be recommended by the recommender systems. The results can be considerably effective when used in many applications, e.g. ranking or suggesting users in social networks, houses on real estate websites, or vehicles, dresses, and even jewelries on the corresponding e-commerce websites.

1.2 The Organization of Thesis

This document can be partitioned into two main parts. The first part provides the required background knowledge which serves as the basis for the methods proposed during this research. Indeed, in the first part of this document which includes chapter 2 to chapter 5 we have respectively provided some basic information about the parallel processing, image processing, recommender systems, and image retrieval. The second part of this document is dedicated to the proposed method. Chapter 7 is all about our proposed method which utilizes some techniques introduced in the aforementioned domains. In this chapter, first, some information about the areas that recommender system have been used and their approaches are mentioned. Then, the proposed method is explained thoroughly. Finally, experimental results are provide to show the positive effects of the proposed method.

In order to improve the results achieved using the novel method proposed in chapter 7, we have used a CBIR component. As this component plays a significant role in the quality of the final results we proposed two methods which help to retrieve more relevant images regarding the query image. This chapter is fully dedicated to the illustration of these methods.

Finally, chapter 8 summarizes the contributions of this work, and provides some hints about the future studies that can be conducted based on this research.

Chapter 2

Parallel Programming

2.1 Overview of parallel programming

Many science and engineering disciplines are affected profoundly by parallel computing with regard to speed and capability. These disciplines include environmental and climate modeling, plasma and condensed matter physics, economics, image and video processing, bioinformatics and computational biology, jet construction, quantum chromo-dynamics, device and semiconductor simulation, speech recognition, seismology, thermodynamics, societal health and safety, earthquakes, auto assembly, geophysical exploration and geoscience, turbulence, materials science and computational nanotechnology, human/organizational system studies, atmospheric science, plate tectonics, stockpile stewardship, signals intelligence, defense, cosmology and astrophysics, and so forth[1]-[4].

For instance, consider astronomy, a natural science which is the study of celestial objects (such as stars, galaxies, planets, moons, asteroids, comets and nebulae), the physics, chemistry, and evolution of such objects, and phenomena that originate outside the atmosphere of Earth, including supernovae explosions, gamma ray bursts, and cosmic microwave background radiation, where one of the most striking paradigm shifts has occurred. According to NASA[1], a vast number of new, enormously detailed observations deep into the universe have been available to scientists for a while. These staggering new data are collected from such instruments as the Digital Sky Survey and the Hubble Space Telescope. Nonetheless, until recently it was quite hard, if not impossible to contrast these information and draw firm conclusions based on complicated mathematical theories, due to the complexity of these theoretical relations. However, thanks to massively parallel computers having large memories, now we witness an astonishing change in that. Indeed, comparing the theories with observed data is made available using High-performance computing (HPC) which has transformed the practice of cosmology

[1] National Aeronautics and Space Administration

deeply. Nowadays, high-quality observations, with incredible resolution, have become available at the same time as advances in parallel computing and realistic physics software which together have facilitated fast progress in improving our knowledge from the universe [6][5]. Equations of physics-based computation are now complemented by massive-data-driven computations. Another equally important point is that using parallel computing techniques is that they allow us to discard some theories as being incompatible with observations [7].

Furthermore, nowadays, in consequence of the broad availability of large-scale data sets which are collected from various domains including text, speech, image or video processing systems a vital demand to automate the process of extracting information from them is created. In brief, Data Mining and Knowledge Discovery are commonly defined as the extraction of patterns or models from observed data, usually, so as to make further and more complicated decisions and categorization possible. Examples of pattern recognition and data mining range from large collections of images clustering on Flickr, social connection mining on Facebook, to books recommendation by Amazon.

Knowledge discovery and data mining requires complex operations on the underlying data which can be very expensive in terms of computation and analysis time. This computation time can be reduced significantly through high performance parallel systems. Indeed, researchers have realized that parallel processing as a novel technique for scaling up the algorithms provides them with an the tools required for developing learning algorithms that are able to take advantage of the increasing availability of multi-processor and grid computing technology and consequently run faster. Using parallel computing not only increases the performance of current data mining systems, but also makes it possible to expand the borders of this domain beyond the borders of possibility before the emergence of HPC. For example, in text mining the size of the input data is huge and we have to deal with a high dimensional

data, so the computations have to be distributed across hundreds or thousands of machines in order to finish in a reasonable amount of time.

Another example can be found in the fields of computational biology bioinformatics [4], especially in molecular biology, which is aimed at gaining and understanding of how cells and systems of cells operate. The ultimate goal of bioinformatics is improving human health, longevity, and the treatment of diseases, which should be done at the cost of huge computations with enormous run-times. So far, computer simulations are the only approach for understanding the dynamics of macromolecules and their assemblies. However, as stated previously, gaining a clear understanding of the characteristics of protein interaction networks and protein-complex networks which are formed by all the proteins of an organism requires enormous computational efforts and resources. This demand is so high that even after employing knowledge-based constraints, protein folding problem[2] remains seemingly computationally intractable. The complexity that exists in molecular systems, can be considered in terms of both the number of molecules and the types of molecules [9].

Nowadays, this field of science enjoys modern parallel computation approaches to overcome the aforementioned difficulties. In fact, beside the utilization of statistical methods to process large data sets, a tremendous number of data mining and knowledge discovery algorithms are being developed and deployed, which have enabled the scientists to decode the human genome [8]. Accordingly, there has been a paradigm shift in the nature of biological computing with the. The ability to perform predictive simulations of biochemical processes has completely changed the researchers' ability to understand the chemical basis of biological functions, which

[2] Prediction of protein structures and computational modeling to comprehend the mechanism that transform gens into proteins

in turn has extremely improves their capability to treat diseases, design new medicines, and understand the mechanisms of genetic disorders besides its intrinsic value in basic biological researches.

2.1.1 Beyond the Moor's Law

Many scientists consider the Moore's Law the business model which drives the semiconductor industry and the exponential growth that continues nowadays. According to this law, the number of transistors in a dense integrated circuit doubles approximately every two years. However, as previously stated our ever-increasing demand for more processing has inclined the researchers to move toward the development of new approaches to go beyond what is expected according to Moore's law and to explore innovative High Performance Computing (HPC) architectures which are able to overcome the limitations of conventional computing systems. These approaches include:

- Multi-core processors
- Heterogeneous computing architectures

2.1.2 Multi-core processors

A multi-core processor is a single computing component with two or more independent actual central processing units which provide the whole processor with enhanced performance, reduced total power consumption, and more efficient simultaneous processing of multiple tasks. These processors are able to deliver considerable performance benefits for multi-threaded programs through increasing processing power with minimal latency, given the proximity of the processors. Multi-core processors show even more substantial advantages in applications such as CRM[3], ERP[4], ecommerce and virtualization which are highly multi-threaded

[3] Customer Relationship Management

[4] Enterprise Resource Planning

applications with huge demands for processing power. Indeed, this approach enjoys immediate multiple factors of computing density, while reducing per-processor power consumption and heat.

From the well-known examples of multiple-core processors we can mention for example AMD Phenom II X2 and Intel Core Duo –which have two cores, AMD Phenom II X4, Intel's i5 and i7 processors- with four cores-, AMD Phenom II X6 and Intel Core i7 Extreme Edition 980X-having six cores-, or even Intel Xeon E7-2820 and AMD FX-8350-which have eight cores [32].

2.1.3 Parallel computing challenges

Besides providing great opportunities, parallel computing faces reasonable limits which even seem prohibitive in some problem domains. Before proceeding further, in this sub-section, we quote "Top 10 issues in parallel computing" provided by a group of experienced parallel programmers at Intel [22].

1. Finding concurrent tasks in a program. How to help programmers "think parallel"?
2. Scheduling tasks at the right granularity onto the processors of a parallel machine.
3. The data locality problem: Associating data with tasks and doing it in a way that our target audience will be able to use correctly.
4. Supporting scalability, hardware: bandwidth and latencies to memory plus interconnects between processors to help applications scale.
5. Supporting scalability, software: libraries, scalable algorithms, and adaptive runtimes to map high level software onto platform details.
6. Synchronization constructs (and protocols) that let programmers write programs free from deadlock and race conditions.

7. Tools, API's and methodologies to support the debugging process.
8. Error recovery and support for fault tolerance.
9. Support for good software engineering practices: composability, incremental parallelism, and code reuse.
10. Support for portable performance. What are the right models (or abstractions) so programmers can write code once and expect it to execute well on the important parallel platforms?

Parallel computing demands the development of sophisticated data preprocessing, data compression, out-of-core processing, and compiling and message-passing techniques in order to facilitate transition from traditional two dimensional processing to contemporary N dimensional one, in addition to requirement of massive computing power. However, porting existing codes which are prepared for a sequential architecture to parallel architecture often requires massive effort [25]-[27].

As stated previously, it is not always possible or cost effective to translate a sequential code to its parallel counterpoint. Indeed, having a persistent programming model for scalable, parallel computers is essential to easily perform this transition. Nevertheless, software developers often generate highly optimized codes for specific models on parallel computers to exploit new hardware features and new parallel algorithms. This makes reaching a general unique programming platform even harder [38]. We should note that the principal goal of high performance computing has always been the development of software and algorithms that exploit the maximum processing power of the parallel architecture [28]. Due to lack of a single common API[5] exploitation of portability in parallel programming is hard to define

[5] Application Programming Interface

and difficult to achieve. Portable performance, which is defined as the capability to gain the highest performance possible on each parallel computing system from the same program image, is of high importance in the study of parallel computing [30].

Yet another challenge comes from the complexity of the problems which require various and unusual skills from the application developers.

All in all, the success of a parallel computing model depends highly on how sufficiently it overcomes these issues. This clearly explains the convergence in HPC computing to standard programming and architecture models as well as the reasons why hybrid computing, especially GPU computing, are proving themselves as major trends in the parallel computing approaches, and are gradually replacing conventional computing models [31].

2.2 Performance measures

Performance measures is composed of a set of metrics which can be employed to measure the quality of an algorithm. While time and space are adequate metrics for measurement of the quality of sequential algorithms, it becomes quite different when it comes to parallel algorithms. Two additional metrics which are essential for evaluating parallel algorithms are efficiency and speedup. Using this measures in addition to time and space we can gain a good understanding of quality of a parallel algorithm. These measures provide us with a theoretical analysis of the algorithm. Consequently, in order to gain an experimental evaluation of the algorithm, metrics such as memory bandwidth and floating point operations per second can be employed. These metrics define the performance of a parallel architecture when it is running a parallel algorithm. Furthermore, sometimes we cannot completely parallelize an algorithm. In such cases, having a theoretical estimate of the maximum speedup possible, which can be gained from the laws of Amdahl and Gustafson, is quite worthwhile.

Given a problem of size n, the running time of a parallel algorithm, using p processors, is denoted:

(2-1) $T(n, \mathrm{p})$

Theoretically, speedup and efficiency can be calculated based on the metrics *work* and *span*.

2.2.1 Work and span

The measure *work* and *span* provide us with a method to evaluate a parallel algorithm, as they determine the borders of parallel computing and introduce the notion of work [10]. Parallel algorithms not only have to be as fast as possible, but also have to introduce the least amount of additional effort required. The less extra effort we make, the more efficient a parallel algorithm will be.

Definition 1: Work is defined as the total amount of processor time required to complete the program using only one processor, and denoted by $T(n,1)$.

As illustrated in Figure 2-1 Calculation of *work* for a typical program [20]. All the processing times should be considered in the calculation of work.

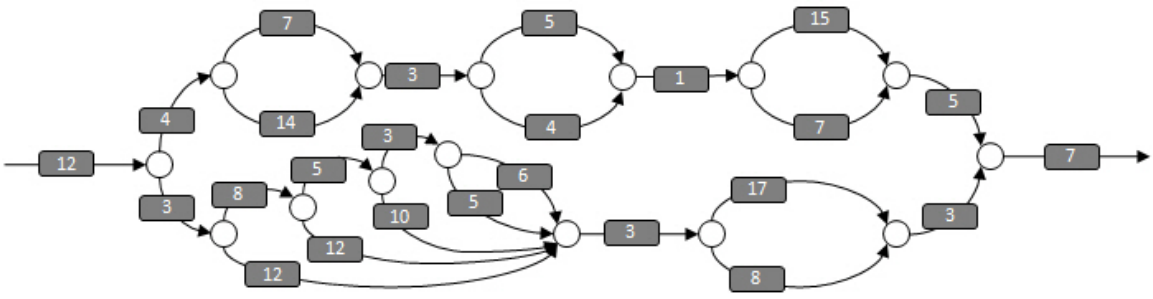


Figure 2-1 Calculation of *work* for a typical program [20].

Definition 2: Span, sometimes called the critical path length, is defined as the most expensive path that extends from the beginning to the end of the program; or equivalently, it is defined as total amount of processor time required to complete the program using an infinite number of processors, denoted by $T(n,\infty)$.

As illustrated in Figure 2-2 Calculation of span for a typical program [20]. Only the processing times along the most expensive path should be considered in the calculation of span.

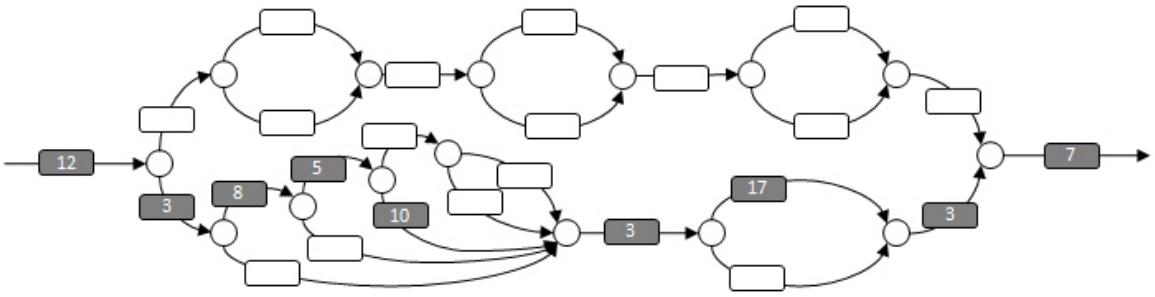


Figure 2-2 Calculation of span for a typical program [20].

These definitions pave the for definition of two important laws, namely *work law* and *span law* which determine two lower bound for $T(n,\mathrm{p})$.

Definition 3: According to *work law* the execution time of a parallel algorithm is never less than $\frac{1}{p}$ of its work. This definition can be stated by the following inequality:

(2-2) $$T(n,p) \geq \frac{T(n,1)}{p}$$

Definition 4: According to *span law* the execution time of a parallel algorithm is never less than its span. This definition can be stated by the following inequality:

(2-3) $$T(n,p) \geq T(n,\infty)$$

2.2.2 Speedup

It is highly desired to somehow measure how much improvement can be achieved through parallelization of an algorithm. This measurement can be taken using the notion of speedup.

Definition 5: *Speedup*, denoted by S_p, is defined as the ratio of the rate at which work is done when a job is run on *p* processors to the rate at which it is done using only one processor. This definition can be stated by the following equation:

(2-4) $$S_p = \frac{T_s(n, p)}{T(n, p)}$$

where $T_s(n, p)$ denotes the best execution time of a sequential algorithm for the same problem whose execution time using p processors is $T(n, p)$.

According to work law the following inequality always holds:

(2-5) $$S_p \leq p.$$

Ideally, we prefer $S_p = p$, which is called *perfect speedup* and is maximum theoretical value of speedup a parallel algorithm can achieve when n is fixed, however, in practice this is rarely achieved because of two prohibitive factors, namely memory bottlenecks and overhead increase as a function of p.

Definition 6: *Linear speedup*, occurs when the speedup increases linearly as a function of *p*, which means that the overhead of the algorithm is always in the same proportion with its execution time, for all *p*.

Definition 7: *Super linear speedup*, occurs when the speedup is more than p, using p processors in parallel computing, that is $S_p > p$.

Definition 7: *Sub linear speedup*, occurs when the speedup is less than p, using p processors in parallel computing, that is $S_p < p$.

All the various types of speedup which mentioned before, have been illustrated in Figure 2-3.

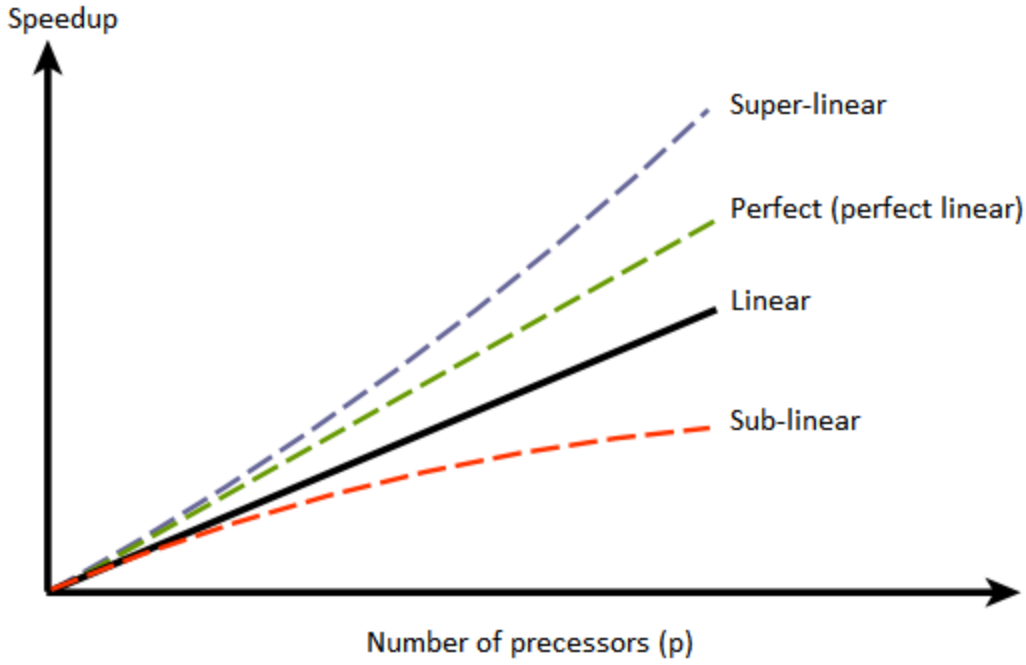


Figure 2-3 Speedup curves.

The possibility of super-linear speedup has been a source of debate among the researchers for decades. Super-linear speedup states that the computational power of parallel machines would be literally more than the sum of their parts [12], which has made it of high importance in the field of computer science. While some researchers contend that the notion of super-linear speed up is not possible [14][13], there is a plethora of scientists who believe that it is possible due to some reasons, including the presence of loop overhead in the single processors [15].

Sometimes it is not possible to fully parallelize an algorithm, which leads us to sub-linear speedup, so we need a partial speedup expression instead of the equation (2-4). These expressions which are called *laws of speedup* have been proposed by Amdahl and Gustafson.

2.2.2.1 Amdahl's Law (Strong scaling)

Amdahl's law [16], also known as Amdahl's argument, can be employed in order to find the maximum expected improvement to an overall system when only some parts

of the system are parallelized. According to this law, the expected overall speedup for a fixed size problem is given by the following equation:

(2-6) $$S(p) = \frac{1}{(1-r) + \frac{r}{p}}$$

where r is the proportion of a program that can be made parallel.

It can be simply implied from the equation (2-6) $S(p) = \frac{1}{(1-r) + \frac{r}{p}}$

that, if p tends to infinity, this equation becomes:

(2-7) $$S(p) = \frac{1}{(1-r)}$$

Amdahl's law is worthwhile for algorithms which need to scale their performance as a function of the number of processors, given a fixed problem size n. This type of scaling is known as *strong scaling*.

The maximum speedups possible with some typical settings have been illustrated in Figure 2-4.

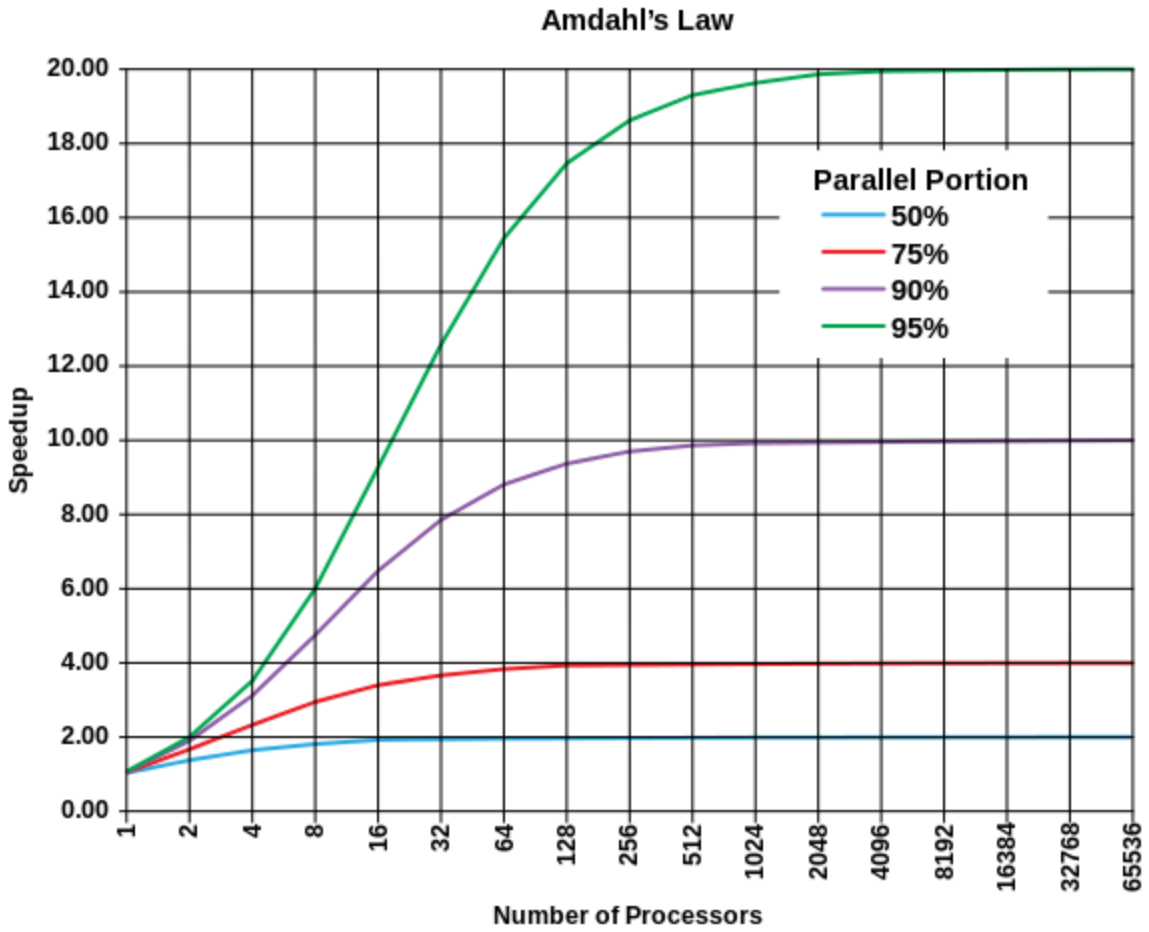


Figure 2-4 The maximum speedups possible with some typical settings [26].

2.2.2.2 Gustafson's Law (Weak scaling)

Gustafson's Law [19] which provides a counterpoint to Amdahl's law and is another useful measure for theoretical performance analysis that, in brief, contends that computations involving arbitrarily large data sets can be efficiently parallelized; and in contrast to Amdahl's law does not assume a fixed size of the problem. Indeed, Gustafson's law is based on a fixed-time model whereby work per processor is kept constant while p and n increase. This law assumes that, the time of a parallel program is composed of a sequential part s and a parallel part r which are executed by p processors, i.e.:

(2-8) $$T(p) = s + r$$

According to Gustafson's Law the speedup can be expressed by the following equation:

$$S(p) = \frac{s+rp}{s+r} = \frac{s}{s+r} + \frac{rp}{s+r}$$

$$s/_{s+r} = \alpha \quad \text{(The fraction of serial computation)}$$

$$\Rightarrow S(p) = \alpha + p(1-\alpha) \tag{2-9}$$

The various speedup values corresponding to some typical values of α have been shown in Figure 2-5.

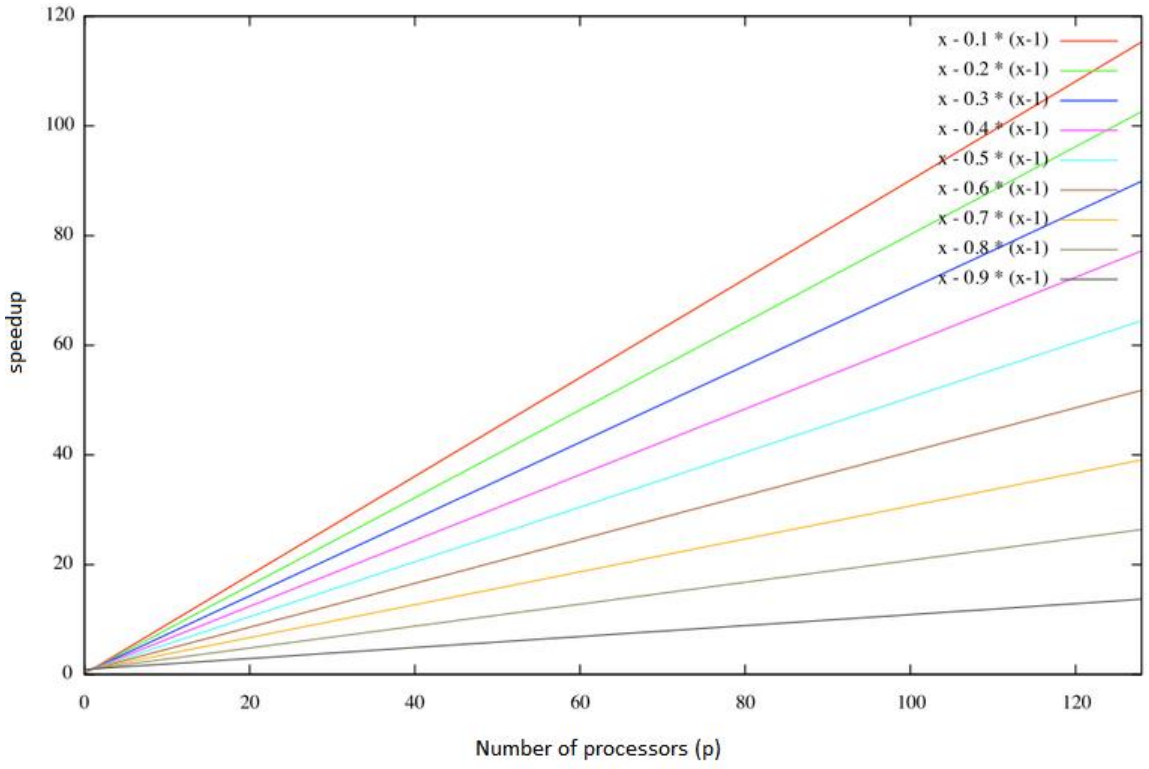


Figure 2-5 The various speedup values corresponding to some typical values of α [18].

Gustafson's law helps us to gain a deeper understanding of parallel computing and the notion of speedup. Regarding Gustafson's law a type of scaling known as *weak scaling* is defined whereby the problem size is not assumed to be fixed anymore, and we try to increase the work linearly as a function of p and n.

Besides the speedup, as one of the most important measures of parallel computing, there are also other metrics, such as the FLOPS[6], the efficiency, and so forth, which provide additional information about the quality of a parallel algorithm.

2.2.3 FLOPS

FLOPS is a measure of computer performance which as it is implied by its name measures the number of floating point operations per second. Based on this metric we can calculate a parameter which expresses the efficiency of the numerical computation relative to a given hardware:

(2-10) $$F_c = \frac{F_e}{F_h}$$

Where F_e denotes the FLOPS corresponding to the implementation of a given algorithm and F_h denotes the peak FLOPS of a specific hardware.

On June 10, 2013, China's Tianhe-2 was ranked the world fastest with a record of 33.86 PFLOPS[7]. So far, ExaFLOPS is remained out of reach, however it is believed that in the following years, using GPU-based hardware, the goal of EFLOPS scale will be achieved. Figure 2-6 illustrates how GPU and CPU performance has improved over the past years.

[6] **FL**oating-point **O**perations **P**er **S**econd

[7] PetaFLOPS (An updated list of the 500 most powerful super-computers in the world is available at the 'www.top500.org')

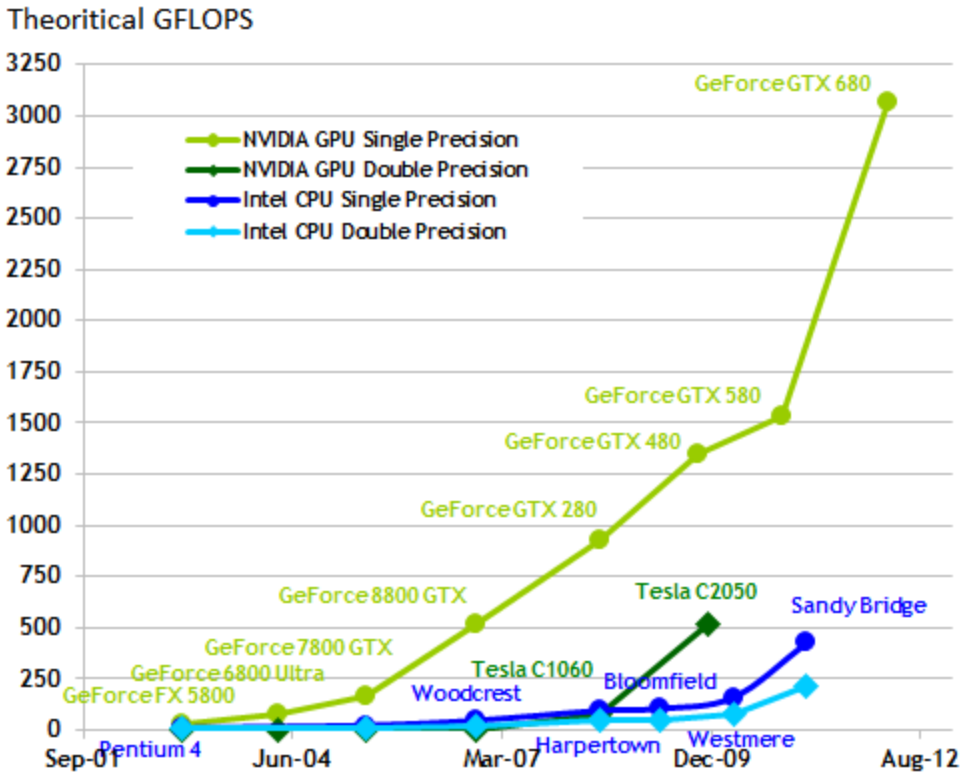


Figure 2-6 The trend in improvement of performance of GPU and CPU over the past years, in terms of FLOPS [19].

2.2.4 Efficiency

Efficiency of an algorithm using p processors, denoted by EF, is defined as:

(2-11) $$EF(n,p)=\frac{S_p}{p}=\frac{1}{p}.\frac{T_s(n,1)}{T(n,p)}.$$

According to Equation (2-11) $$EF(n,p)=\frac{S_p}{p}=\frac{1}{p}.\frac{T_s(n,1)}{T(n,p)}.$$

it is clear that $EF(n,p)\leq 1$. As it is difficult to achieve perfect speedup, maximum Efficiency ($EF(n,p)=1$) is almost unobtainable. This evaluation metric is of high importance because it indicates how well the hardware is uses. Moreover, efficiency states among the implementations such as cluster, supercomputer, workstation which should take priority, with the presence of limited resources.

2.3 Parallel systems

The lack of general consensus on a single definition for parallel computing is obvious, however a general term on which is able to gain a positive feedback from many scientists is that parallel computing is the simultaneous use of multiple computing resources to solve a computational problem. This is exactly the case that has happened to the categorization of computer architectures. Despite the huge efforts made over the years to reach a satisfactory classification, there is not such a thing as the perfect categorization. However, there are some categorizations which are widely accepted by many scientists. Among them, is Flynn's taxonomy, proposed by Michael J. Flynn in 1972 [21]. He analyzes the machines by the number of data streams and the number of different program streams processed at a given time and provides a coarse classification of parallel machines. Flynn's four classifications are shown in Figure 2-7.

	Single Instruction	Multiple Instruction
Single Data	SISD	MISD
Multiple Data	SIMD	MIMD

Figure 2-7 Flynn's taxonomy.

2.3.1 SISD

SISD[8] (Figure 2-8) refers to a computer architecture in which there is a single processor which executes a single instruction stream, to operate on data stored in a single memory. There is no parallelism at all in either instruction or data streams in SISD. During the process of computation the instruction stream generates an instruction and processor applies it on a datum of memory using data stream. After

[8] Single Instruction stream, Single Data stream

finishing the execution of this instruction, another instruction is generated to be applied on another datum and this process continues until the desired task is completed. All of the single core CPUs of the 1950s, which were based on the original Von Neumann architecture, as well as the Intel processors from 8086 to 80486 fall into this category[23][22].

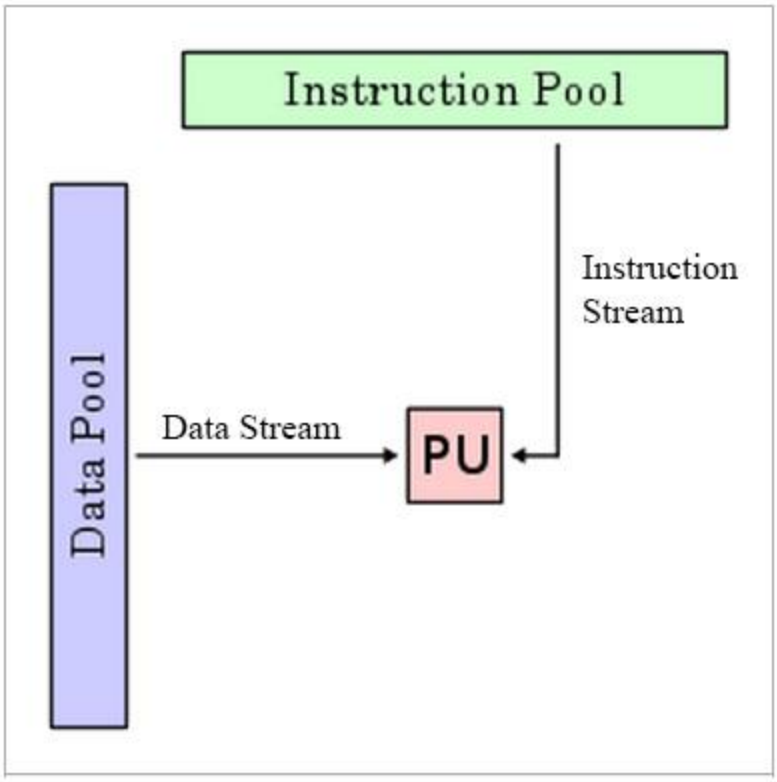


Figure 2-8 An SISD computer architecture.

2.3.2 MISD

Computers which belong to this category (illustrated in Figure 2-9) have more than one processors each of which has its own control unit and is able to handle only one instruction but apply it to many data streams simultaneously, therefore a kind of parallelism is made possible in these computer architectures.

It is worth mentioning that some scientists contend that there is no such thing as an MISD[9] machine, and believe that these machines are just a variant of SIMD machines. MISD computers are not very common and usually are implemented for specific purposes, especially in fault-tolerance computers executing the same instructions redundantly in order to detect and mask errors , for example in space shuttle flight computers [23][22].

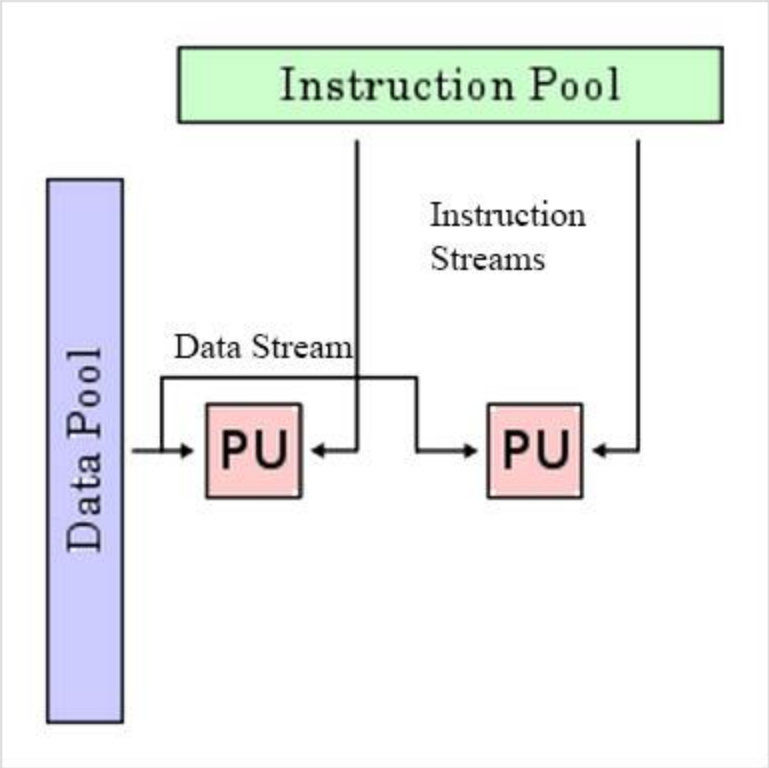


Figure 2-9 An MISD computer architecture.

[9] Multiple Instruction streams, Single Data stream

2.3.3 SIMD

The computers which work based on SIMD[10] architecture, have more than one processors, each of which has its own local memory unit. Each memory unit contain its corresponding processor's data. All the processors execute the same instruction at the same time, which are issued by a control processor. These computers were the first systems to be implemented with a huge amount of processors, and were among the first systems to provide computational power in the GFLOPS order. The instructions executed with such computers as well as the data used by them can be either simple or complex. These computers which are sometimes also known as processor-array machines exploit data parallelism [24].

Vector computers in the 70's and 80's were the first to implement MISD; vector processors such as IBM 9000, Cray X-MP are considered as MISD computers. This architecture is particularly applicable to common tasks such as adjusting the contrast in a digital image, adjusting the volume of digital audio or even tasks like applying a mathematical model to different parts of the problem domain.

Last but not least, most contemporary GPUs work with many SIMD batches simultaneously so they are considered as an evolved SIMD architecture [23].

[10] Single Instruction stream, Multiple Data streams

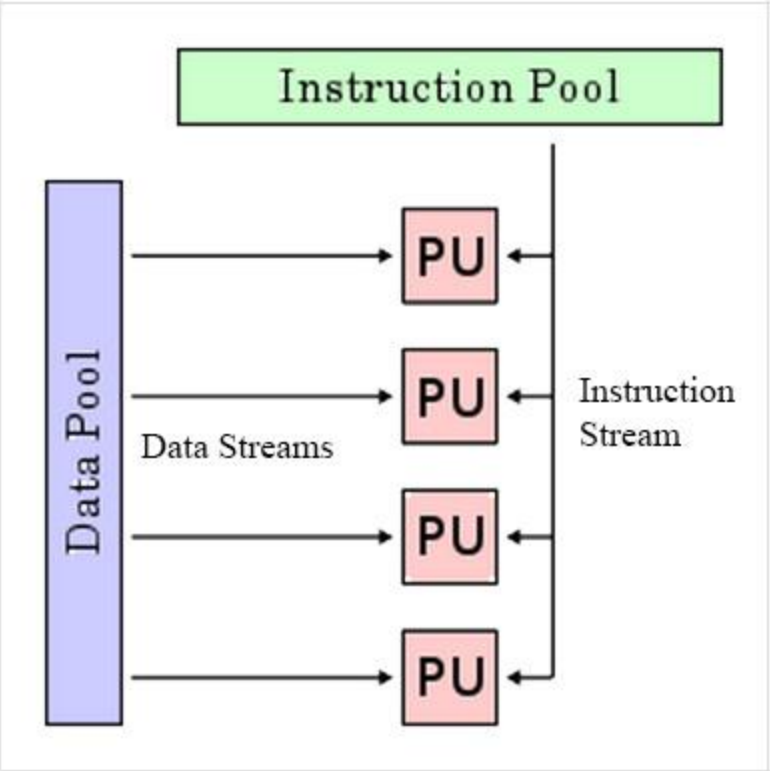


Figure 2-10 An SIMD computer architecture.

2.3.4 MIMD

MIMD[11] architecture includes most modern general purpose computers (Intel, AMD and ARM multi-cores), also newer GPU architectures are partially implementing this architecture. These machines are parallel computer structures composed of multiple independent processors. In this architecture, which is the most flexible architecture and the most common type of parallel machines in use, each processor runs under the control of instruction stream of its own control unit over the data of its own data stream. In fact, at any time, each processor may be executing different instructions on different pieces of data. Communication between processors is carried out either by means of a shared memory or through an interconnection network.

[11] Multiple Instruction streams, Multiple Data streams

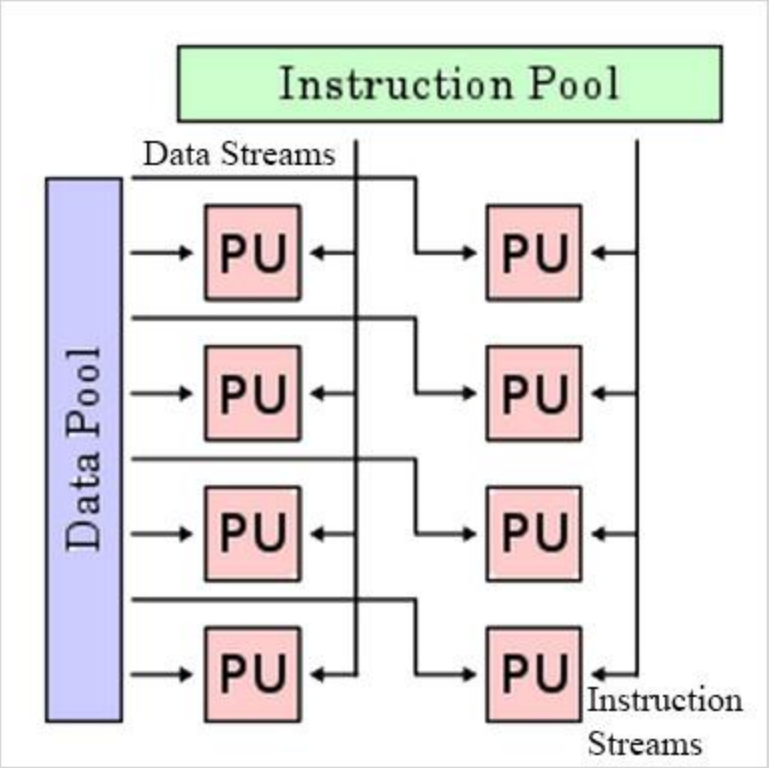


Figure 2-11 An MIMD computer architecture.

Chapter 3

Image processing taxonomy

3.1 A General Categorization of Image Processing

Image processing include a various and sundry range of tasks which can easily be categorized in many different ways. Roughly, these algorithms can be categorized into three general groups:

- Low level image processing algorithms, which usually convert image data into image data;
- Intermediate image processing algorithms, are more complex than low level operations and derive abstractions from the image pixels,
- High level image processing algorithms, which are knowledge-based algorithms that, in general, extract the interpretation of the information extracted from the lower level operations.

3.2 Low Level Image Processing Algorithms

Low level image processing algorithms, such as edge detection, noise reduction, object labelling, template matching, or corner detection, directly operates directly on stored images to improve or enhance them. The principal purpose of these operators is to prepare the images for improved human perception, measurement, automatic analysis, or interpretation. The input to these algorithms is an image and the output of the algorithm is an image, as well.

An image can roughly be defined as an array, or a matrix, of square pixels[12] arranged in columns and rows. These images may come directly from digital cameras or from indirect data, produced in synthetic aperture radar, seismic, MRI, etc. No matter how

[12] Picture elements

the image is acquired, a variety of operations can be performed on it which often begins with low-level processing.

Using individual pixel values, these operators modify the input image. For instance, a usually operation which is carried out on the pixels of an image is noise removal or reduction, also known as noise cleaning or noise reduction. Real images are very noisy, so we perform to remove or reduce these unwanted variations produced by the sensors or environment, which alter the actual pixel values.

Another example is the edges detection algorithm which identifies object boundaries within image to capture important events and changes in properties of the world. Indeed, the points at which image intensity changes sharply are typically organized into a set of curved line segments termed edges. Edges typically occur on the boundary between two regions, and after performing edge detection operation, these parts of the image are highlighted in the output image.

It is worth mentioning that sometimes to improve the effectiveness of a low level operation, one or more other low level operations should be performed on the same image. For example, usually, before performing edge detection on an arbitrary image, it requires to undergo the noise reduction process because it is very difficult to find edges in noisy images.

It is very common to transform an image from the original spatial domain into the frequency domain, because a big portion of image processing algorithms are carried out more effectively and efficiently when used in the frequency domain. These transformations operate upon all pixel values and convert them from one domain to another. Examples include Fourier transform, Hilbert transform, Wavelet transform, Hilbert-Huang transform, and many more [33]. Indeed, these transforms comprise an indispensable part of the image processing. For example, frequency domain analysis and Fourier transforms are the cornerstones of signal and system analysis.

Anyone who has spent any time on the internet, at some point has probably experienced delays in downloading web pages, which happens because of different factors such as the network overall traffic or the amount of data on the page which was requested at that moment. As image data are usually comprising a considerable portion of each webpage, downloading them often takes much time. For instance, visiting a shopping website, one may look for a particular purchase, whose image is among a huge database or archive of images. Downloading each image completely and then deciding that it does not belong to the item for which one is looking, can be time-consuming, while it was possible to make this determination (i.e., whether or not the image suits our purposes) with much less quality than the full image possesses. Certainly it is quite easy to reject most of the images with only a very rough idea of their content. What experienced web designers usually do in such circumstances, is using a reduced-size version of the pictures, also known as thumbnail. In technical perspective, this reduced size image is nothing but the original image which has undergone a transformation like Fourier transform and down-sampling procedure.

One may conclude from the examples that these low level operators are very simple arithmetic operations such as applying a filter on image, however it is not true. For example, template matching is a well-known technique whose goal is finding small parts of an image which match a template image requires some sophisticated operations based on cross-correlation.

It is quite common that pixels which form straight lines or curves need to be localized. Hough transform, which is designed to detect lines, using the parametric representation of a line, is an efficient algorithm which manipulate this issue effectively [39]. It uses pixel coordinates to identify the parameters of the equations of the line. The classical Hough transform was only concerned with the identification of lines in the image, nevertheless, later it has been extended to identifying positions

of arbitrary shapes whose parametric equation is known, especially circles or ellipses. An interesting merit of Hough transform is that it can give robust detection under noise and partial occlusion. Furthermore, as mentioned previously, many low level image processing algorithms have a positive effect on each other. In this case, edge detection is often used as preprocessing to Hough preprocessing to Hough transform.

Last but not least, in low level image processing algorithms, in most applications, operate on all the pixels of an image or all neighborhoods of some pixels, which make parallelism of huge benefit to these algorithms.

The low level image operations can be further classified into three categories:

- Point operations, whose output value at a specific coordinate is dependent only on the input value at that same coordinate. A typical point operator is depicted in Figure 3-1.
- Neighborhood operations, whose output value at a specific coordinate is dependent on the input values in the neighborhood of that same coordinate. A typical neighborhood operator is depicted in Figure 3-2.
- Global operations, whose output value at a specific coordinate is dependent on all the values in the input image. A typical global operator is depicted in Figure 3-3.

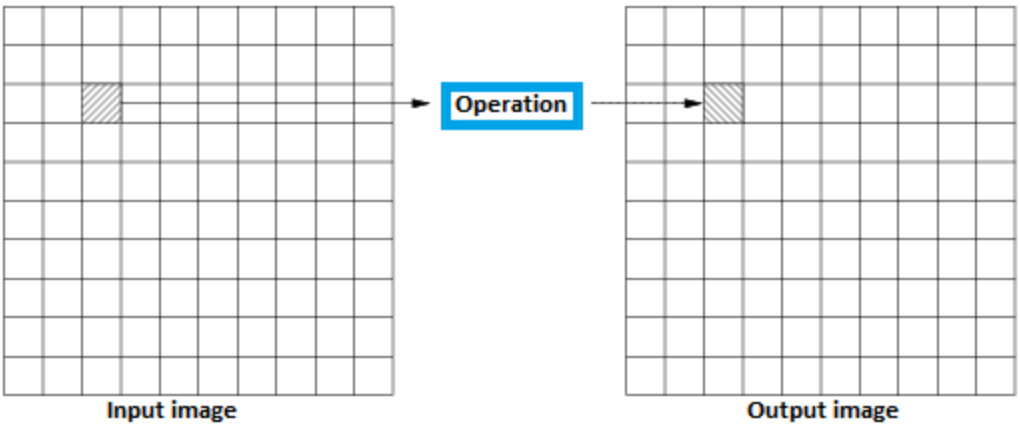


Figure 3-1 A sample of a low level point operator.

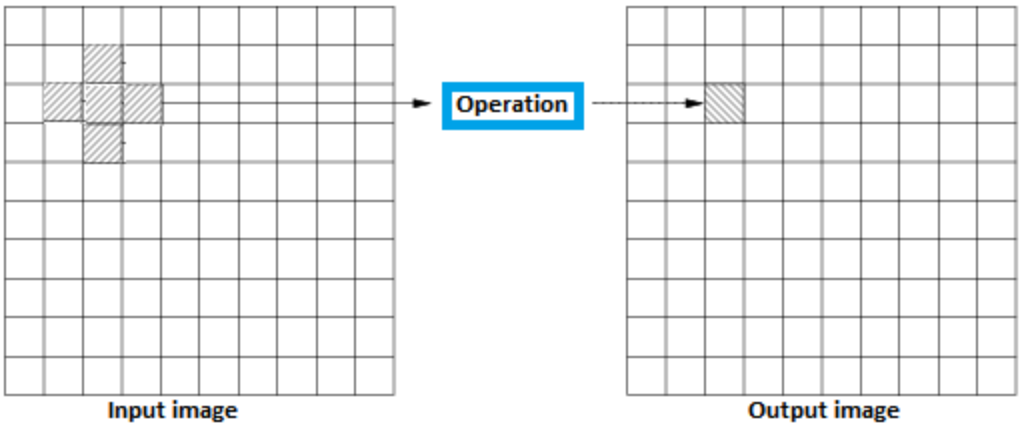


Figure 3-2 A sample of a low level neighborhood operator.

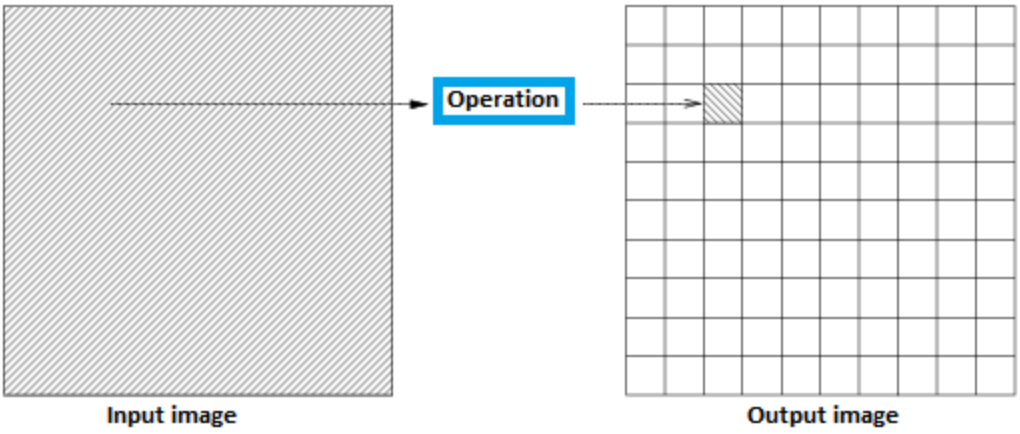


Figure 3-3A sample of a low level global operator.

In what follows in this section, we have explained each of these types briefly.

3.2.1 Point Operators

As stated before, these operators produce an image through a pixel-wise manipulation of the original image, i.e., each pixel of the output image depends just on the value of its corresponding pixel in the input image and require no data from other pixels to process an input pixel into an output pixel.

These operators are considered to be the simplest image operators regarding the computing time and complexity of the algorithm. Also, these algorithms are embarrassingly parallel, i.e., their computational graph is disconnected, which makes them almost perfect for parallelization especially through data decomposition paradigm.

A vast majority of operators such as noise reduction, smoothing, brightness adjustment, contrast adjustment, thresholding, or edge detection fall into this category. As these operators act as the basis of many other higher level image processing algorithms, we briefly discuss some of them.

Thresholding an image, which is the simplest method of image segmentation, is the process of making the corresponding input pixels above a certain threshold level white and others black. The effect of this operator on an arbitrary image has been illustrated in Figure 3-4.

Figure 3-4 An arbitrary image (left), along with the resulting image from thresholding (right) [43].

Brightness adjustment concerns with subtraction or addition of the brightness of an image through which a constant is added or subtracted from the luminance of all sample values in the input image to build the output image. The effects of brightness addition and subtraction are shown, respectively in Figure 3-5 and Figure 3-6.

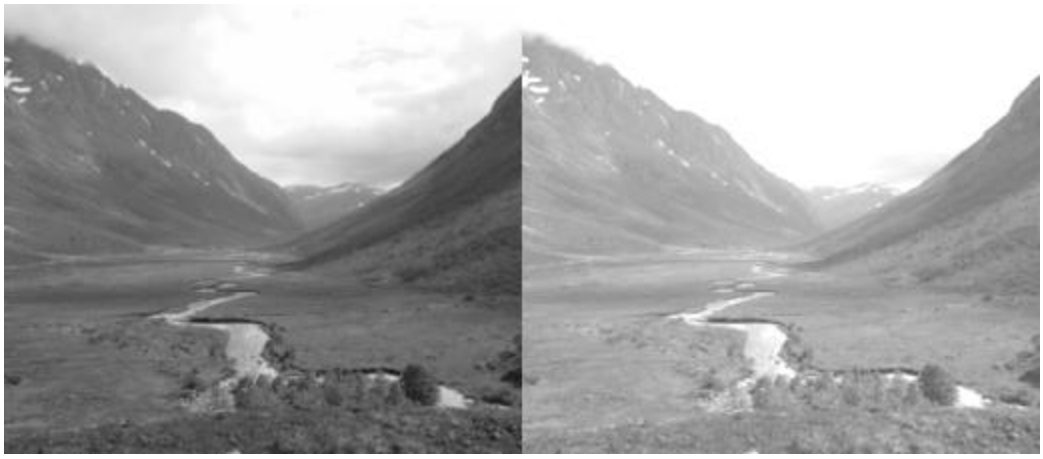


Figure 3-5 An arbitrary image (left), along with the resulting image from brightness addition (right) [34].

Figure 3-6 An arbitrary image (left), along with the resulting image from brightness subtraction (right) [34].

Contrast adjustment is equivalent to expanding or compressing the histogram around the midpoint value. This operation is simply implemented with a linear mapping of the brightness value of each pixel. An arbitrary image along with its contrast adjusted counterpart are depicted in Figure 3-7.

Figure 3-7 An arbitrary image (left) along with its contrast adjusted counterpart (right) [34].

3.2.2 Neighborhood Operators

Neighborhood operators apply to all pixels in a region of the image data around a specific pixel. As other low level operators, the result of these operators can be interpreted as images. Neighborhood operators are pretty much common and have various applications ranging from blurring, sharpening, noise reduction, and feature identification to measuring the similarity of two images, and edge detection. All of these operators employ the information of the source pixel and the value of pixels in the neighborhood surrounding it to calculate the corresponding value in the destination pixel. Some rather complicated tasks such as object recognition, image restoration, and image data compression fall into this category.

These operators can be further categorized into two categories: linear filters and non-linear filters.

Generally speaking, a filter is a process which removes or enhances some aspects of an arbitrary image. We usually consider two-dimensional filtering techniques in image processing as an extension of one-dimensional signal processing theory with exactly the same principles and rules. However, compared to signal processing that was applied to analog or continuous time domain processing long time ago, almost

all contemporary image processing techniques which are somewhat common in a vast group of applications involve discrete or sampled signal processing.

Linear filters

The *homogeneity* and *additivity* properties together are called the *superposition* principle which itself is the basis for definition of linear systems, including linear filters. A system is called linear if it satisfies the properties of superposition.

The property of homogeneity states that for a given input, x, in the domain of the function f, and for any real number a,

(3-1) $$f(ax) = af(x)$$

The property of additivity states that for given inputs, x_1 and x_2, in the domain of the function f,

(3-2) $$f(x_1 + x_2) = f(x_1) + f(x_2)$$

The superposition principle for two-dimensional signals is depicted in the Figure 3.8.

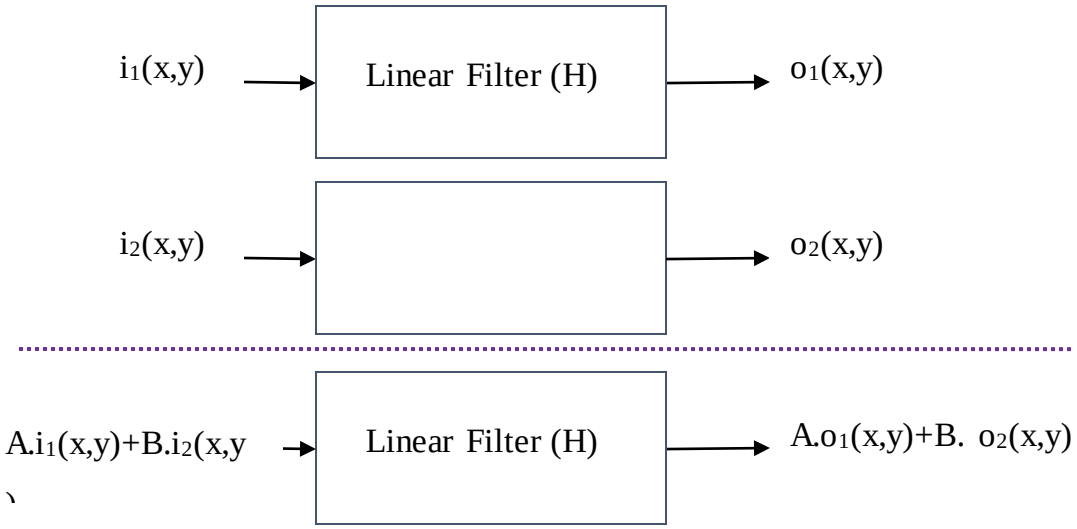


Figure 3-8 The superposition principle for two-dimensional signals.

In image processing jargon, a linear filter is implemented using two-dimensional convolution which performs between a source image and a two-dimensional impulse response, which is commonly known as convolution kernel and is usually considerably smaller than the source or input image, in order to produce a destination image. This operation is state as,

$$IM[x,y] = (im * h)[x,y]$$
$$IM[x,y] = \sum_{i \in V(x,y)} \sum_{j \in V(x,y)} h(i,j) \times im[x-i, \text{y}-\text{j}] \quad (3\text{-}3)$$

Non-linear filters

In general, non-linear filter is a system whose output is not a linear function of its input. These filters exist in both continuous-domain and discrete-domain.

Despite the adequacy of the linear image enhancement tools in many cases, significant advantages in image enhancement can be attained if non-linear techniques are applied to an image. Indeed non-linear filters preserve the edges and details of images and remove the noise while methods based on linear operators tend to blur and distort them. Furthermore, nonlinear image enhancement tools are less susceptible to noise. This quality is very important because noise is always present as a consequence of physical randomness of image acquisition systems which involves acquisition of images in the environments with high or low temperatures, illuminations, humidity, etc.

As per [35] non-linear filters can be divided into four categories based on their behavior:

- L-type filters
- R-type filters
- M-type filters

3.2.2.1 L-type filters

The output of these filters can be expressed by a fixed linear combination of the order statistics. Given an input signal X, the output Y_k of such a filter can be stated like this:

(3-4) $$Y_k = \sum_{i=1}^{n} a_i . \mathrm{Z}_{k_i}$$

where n is the window size, a_i is the data-dependent coefficient, that is the i'th element of a set of constant weights, and Z_{k_i} is the i'th sample of the order statistics among the 2N+1 samples that are inside the window centered at k.

3.2.2.2 R-type filters

The output of these filters can be expressed by a fixed linear combination of the order statistics. Given an input signal X, the output Y_k of such a filter at index k of window size 2N+1 can be stated like this:

(3-5) $$Y_k = \text{order statistics of } \{\mathrm{f}(\mathrm{X_i})\}$$

where $\mathrm{f}(\mathrm{X_i})$ can be any arbitrary linear function of the input signal X_i.

3.2.2.3 M-type filters

The output of these filters can be calculated by solving an equation. Given an input signal X, the output Y_k of such a filter at index k of window size 2N+1 can be achieved through solving this equation:

(3-6) $$\sum_{i=1}^{n} \psi(x_i - Y_k) = 0,$$

where ψ is an odd, continuous, and sign-preserving function.

3.2.3 Global Operators

Global operators use the whole source image to produce the resulting output image. A representative example for this kind of low level operators is discrete Fourier transform (DFT). Many other well-known transforms including DCT[13], FFT[14], or Walsh-Hammond Transform fall into this category. These operators, along with neighborhood operators, are more computationally expensive than point operators.

3.3 Intermediate Level Image Processing

The operators that belong to this category take in one or more regions of pixels, represented by one or more N by N arrays of pixel data and then produce more compact data structures or slightly more abstract forms, such as lists or graphs, which are descriptive of the image. This implies that these operators also work on the image at the pixel level, however the main difference is that they reduce the amount of data that is available in the input image in the process of producing the output image. Consequently while the input data achieved from the source image is positionally related to the source data, the same does not hold for the output data.

Some representative examples of this category of image processing operations are Connected-component labelling (CCL), perceptual grouping, image measurement, and object tracking.

Since the origin of computer science, algorithms of connected-component labeling, also known as connected-component analysis, blob extraction, region labeling, blob discovery, or region extraction [44], have been widely used in many fields [45], including image processing [46]. In the field of image processing, a CCL algorithm

[13] Discrete Cosine Transform

[14] Fast Fourier Transform

typically scans an input image and groups its pixels into components based on pixel connectivity. The pixels' connectivity is defined based on positioning of the pixels and their intensity, that is, all pixels in a connected component have similar pixel intensity values and are somehow connected with each other. After determination of all groups, each pixel is labeled with a gray-level or a color according to the component it was assigned to, which the latter case is called color labeling [48]. An example of region labeling applied on an arbitrary image has been illustrated in Figure 3-8. In this figure both gray-level and color labeling are presented.

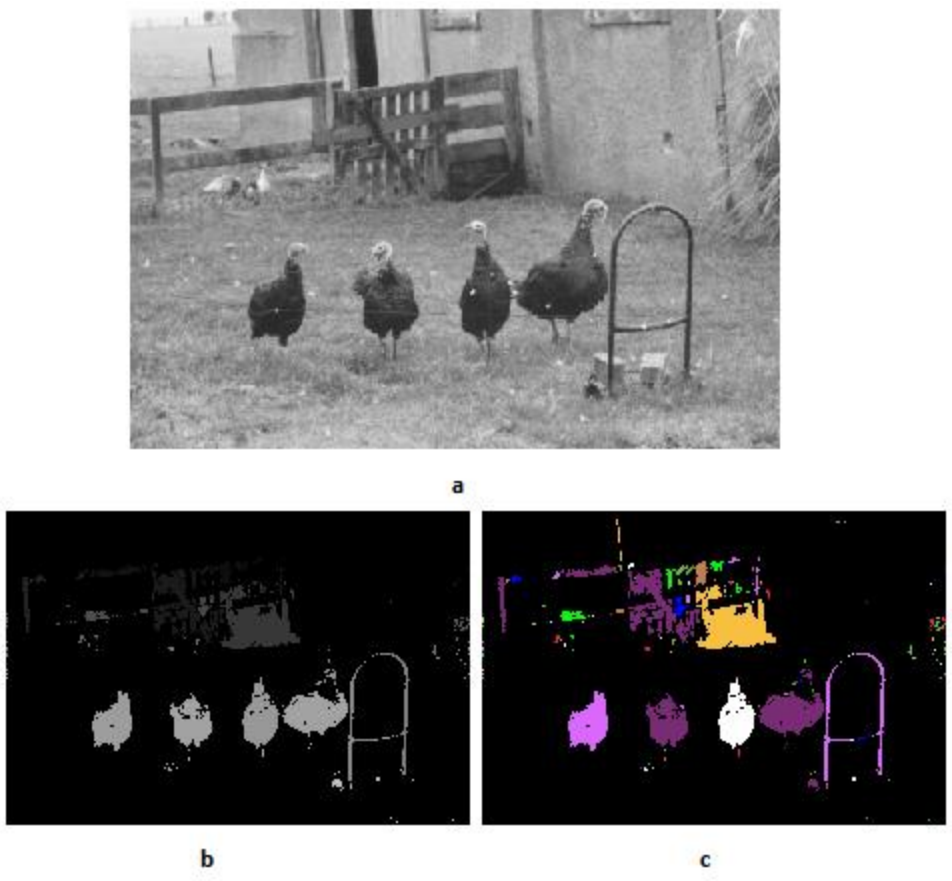


Figure 3-8 An example of b) gray-level along with c) colored region labeling applied on an arbitrary image (a) [47].

The idea of perceptual grouping for computer vision is based on the human visual ability to spontaneously derive relevant groupings or structures from source images

and is aimed at reducing ambiguity presented in image data or in initial segmentation and to increase the robustness and efficiency of subsequent processing steps. This is achieved through organization of image primitives into higher level primitives thus explicitly representing structure contained in the image data. The origins of perceptual grouping are rooted in the work of Gestalt psychologists who described, among others, the ability of the human visual system to organize parts of the retinal stimulus to "Gestalten", into organized structures [43][42]. Though the significance of this operation is discerned in the mid-80th [39]-[41], recently, methods based on perceptual grouping have produced promising results in various vision applications and have proved to be extendable to even more complex applications [37][36]. Nevertheless, these methods suffer from a prohibitive low computational speed on current workstations, despite their reasonable accuracy. To overcome this problem, using parallelized methods is inevitable, which is further improved using some algorithms for dynamic load balancing in [38].

3.4 High level image processing

High level image processing refers to sophisticated knowledge-based methods which are aimed at achieving a rich interpretation from the high level information extracted from the intermediate level processing. Computer vision and its subcategories are among the most well-known members of this category. For example, scene analysis and image understanding are regarded high-level image processing operations.

Computer vision concerns reconstruction, interpretation, and understanding a three dimensional scene from two dimensional image(s), and its ultimate goal is to model and replicate the human vision. Computer vision is highly depended or based on four pillars, namely:

- Human vision,

- Computer science,
- Pattern recognition,
- Signal processing.

It is worth mentioning that many researchers, especially those who work in the field of computer vision, consider some intermediate operators such as segmentation as parts of computer vision, while many consider such operators as a basis for computer vision and limit the computer vision to very high level operations.

Chapter 4

Recommender Systems

4.1 Recommender Systems

Along with astonishing achievements in customizations of applications, recommender systems have gained wide-spread acceptance during the last decade [49]. These systems have attracted increased public interest which has provided the e-commerce based business with new opportunities to raise their sales [51][50].

Great companies like Google[1], Netflix[2] and Amazon[3] and websites like imdb[4] are considered as good examples that employ an extensive range of different types of recommender systems which has had a huge positive effect in their user's experiences.

Indeed, Recommender systems are among the most valuable components that can improve the user experience with a more general system considerably when embedded inside that system. Since the mid-nineties and slightly after that, when the term "recommender system" coined by Resnick and Varian, researchers have conducted huge studies on this topic and the methods and approaches they have found are various and sundry.

The main objective of recommender systems is to reduce the complexity for users, sifting through very large datasets containing information about some items and selecting the most desired ones. For instance, when a user is going to buy a book on a website like Amazon, he usually faces enormous book titles which makes it difficult to choose a book quickly that suits the user's attitude. However, using a recommender system, the retailer website offers the user some books which are

[1] www.google.com

[2] www.netflix.com

[3] www.amazon.com

[4] www.imdb.com

supposed to be more attractive to him, and therefore limits his choices which in turn makes the decision making easier.

Each recommender system is typically expected to offer a common set of tasks as follows [52]:

- Obviously, the main functionality that a recommender system has to offer is to recommend a list of items, which are considered by the system as the most useful for the specific user.
- In addition, the system should be able to predict the desirability or rating of an item from a user's perspective when the item is requested by the user.
- Further, the system should be able to provide useful and effective recommendations when the user has limited the set of items through setting some restrictions.

In order to get this functionality, recommender systems usually follow a general three step mechanism [53]:

1- The users' preferences are provided to the recommender system, which can be down in two ways:

   a. Users provide explicit information, such as rating, in order to show the desirability of the items from their own perspective. Indeed, in this case, system offers the users one or more mechanisms to unequivocally express their interests in objects and evaluate them, usually using a rating system [69].

   b. Users' attitude is obtained implicitly, without interventions of users and through the actions made by the users in the application. Namely, this evaluation is performed without the user being aware, through capture of information obtained from. For instance users' attitude

towards a song can be roughly related to the times they listen to it, or when the users access to an online journal article, based on the time they spend on reading, the system could automatically infer how much the content have been interesting to them.

2- A user preference profile is created for all the users, using the information collected in the first step.

3- The system produces recommendations for the current users, which have an entry in users' preferences list, and even unseen users, based on the preference profile created in the previous step.

It is worth to mention that each of the approaches to achieve information about user's preferences has its own advantages and disadvantages. The first approach which achieves the information explicitly may face problems because it mistakenly assumes that users' attitudes are fixed through time, which is clearly a wrong assumption and can degrade the performance of the recommender system to some extent. The second approach which tries to achieve the information implicitly, nevertheless, has its own problem because realistically there is no information available about the new users that enter the system.

There are various methods and algorithms which can be utilized by recommender systems in order to produce personalized recommendations. All of these approaches can be categorized under three distinct paradigms which are explained in the next section.

4.2 Recommender System Paradigms

Recommender systems can be roughly divided into three main categories. These categories are formed based on the type of information the recommender system uses to predict the desirability of items to users [54].

4.2.1 Content-Based Filtering

The main goal of Content-Based Filtering (CBF), also called cognitive filtering, is to offer the users the items to which they have shown interest before [55]. In this approach, which is deeply rooted in information retrieval [56], similarities between the user's list of desired products, and items from the whole set of items which are still unknown are computed. Similarities between items are measured based on specific features attributes which have already been chosen by the system designer(s).

In other words, the system creates and maintains a profile which is composed of historical information about the users' preferences toward items and recommends them items which are similar to the items they liked in the past. Typically, these systems use learning methods such as Bayesian classification ([57]-[60]) or k-nearest neighbors ([61]-[64]).

Although the general framework remains the same among the content-based recommender systems are highly depended on the kind of items, or the context of use, in order to qualify the content of the items they have to use some specific methods. In fact, the set of features these systems use to represent each item for their computations, such as similarity measurement, or even the similarity measures vary based on the context of use. For instance, the most representative words of the web pages, or the number of links to a web page from other websites can be used by a system which works with webpages [65]; while TF-IDF weights [67][66] are usually used by systems which work documents [68].

One of the key advantages of content-based recommender systems is that they do not need any domain knowledge and the only information they require to make a recommendation is the user's preference history toward some of the items from the

item set to which new items belong. And the key challenge these systems face is that they are only able to work when feature extraction is feasible and items can be represented using meaningful attributes.

4.2.2 Collaborative Filtering

Collaborative Filtering (CF), also called social filtering [69], is another mainstream approach utilized by recommender systems. According to this approach the recommendations are made based on collecting and analyzing some reasonable information about the system users' behaviors and attitudes. The methods belonging to this category predict the desirability of the items to the users based on the similarity of the users to each other. Indeed, the items recommended to a user are those preferred by similar users.

This approach has its roots in human behavior. People usually rely on the recommendations made by those who have similar attitudes as theirs, to choose a movie to rent, a tourist site to visit, etc. This technique tries to simulate the collaboration that happens between users in the real world, i.e. they share opinions about items and make suggestions [78].

CBF recommender systems identify the users who share the same filling and their taste can be leveraged to predict a specific user's attitude towards an item. To be more precise, the similarities between users are computed based upon their rating profiles. Then, the products that most similar users have liked or the most relevant products are suggested to the users. There are many situations where one should make a decision and of the widely used approaches is getting advice from people whose opinion is valued by them. The same idea is employed by the CBF recommender systems.

One of the main pros of the collaborative filtering is that, as opposed to collaborative filtering, it does not require machine analyzable content. As a result, CF-based

approaches are capable of predicting the desirability of complex items such as music or movie accurately without requiring an "understanding" of the item itself. However, a key challenge which CF-based approaches should deal with is measuring the user and item similarity.

Indeed, because collaborative filtering systems have shown very acceptable and high quality results, despite the fact that they require minimal information they have become the most prominent representatives of recommender systems. These systems are used by well-known retailers such as Amazon and TiVo in order to make suggestions to their customers [71][70].

There are many approaches to implement collaborative filtering, for example, methods based on associative rule [72], or Bayesian classifier [73]-[76], or horting [77]. In fact, Collaborative filtering approaches can be categorized as memory-based or model-based. These approaches are explained in more details in the following sections.

4.2.2.1 Memory-based Collaborative Filtering

Memory-based collaborative filtering approach, utilize the entire user-item database by storing them as-is into memory in order to generate a prediction. When the system tries to make suggestions to a user, the ratings of the similar users which are already stored into the database are used. Usually these algorithms use only a portion of users, known as neighbors instead of the entire database. Recommendation for the active user[1] can be generated based on items or users. The first case is known as item-based collaborative filtering while the second case is known as user-based collaborative filtering.

[1] The user for whom recommendations are made

User-based collaborative filtering

Because the user-based collaborative filtering methods are easy-to-implement and highly effective, they so popular that the term "collaborative filtering" is commonly used as a synonym for user-based collaborative filtering. Early generations of recommender systems that use the item-based collaborative filtering to approach the suggestion making problem include the GroupLens [79] and Ringo [80]. For instance, the GroupLens employs the user rating data to calculate the similarity or weight between users or items. Based on these stored similarity values, predictions or recommendations are made.

User-based CF algorithm, first calculate the similarity, $S_{u,v}$, between the users *u* and *v* who have both rated the same items. Subsequently, the similarity, $S_{i,j}$, between the two co-rated items of the users are calculated.

The similarity between users or items can be measured in many different ways, some of which are represented here.

Pearson correlation, derived from a linear regression model, measures the extent to which two variables linearly relate with each other. According to this method $S_{u,v}$ is:

(4-1) $$S_{u,v} = \frac{\sum_{i \in I} (r_{u,i} - \bar{r_u}).(r_{v,i} - \bar{r_v})}{\sqrt{\sum_{i \in I} (r_{u,i} - \bar{r_u})^2 . \sum_{i \in I} (r_{v,i} - \bar{r_v})^2}} .$$

where I is the set of items, $r_{u,i}$ denotes the rating of user *u* for item *i*, and $\bar{r_u}$ denotes the average rating of user *u* for all the items rated by *u*.

Another approach to calculate the similarity between two users is the cosine similarity which measures the cosine of the angles of two vectors representing the users' rating to items in the same order:

$$S_{u,v} = \frac{\sum_{i \in I} r_{u,i} \cdot r_{v,i}}{\sqrt{\sum_{i \in I} r_{u,i}^2 \cdot \sum_{i \in I} r_{v,i}^2}} . \quad (4\text{-}2)$$

Item-based collaborative filtering

In contrast to the user-based collaborative filtering algorithm, the item-based approach [81]-[83] considers the set of items which the target users have rated and computes how similar they are to the target item in order to selects *k* most similar items. At the same time their corresponding similarities are also computed. After finding the most similar items, the weighted average of the target users' ratings on these similar items should be taken in order to generate the prediction.

Among the most important merits are the item-based collaborative filtering are slightly low computational complexity and its ability to decouple the model computation from the prediction generation.

One of the most important steps in item-based CF is computing the item-item similarity in order to select the most similar ones. The similarity measures that mentioned in the previous section can be used to this aim.

4.2.2.2 Model-based collaborative filtering

Model-based collaborative filtering algorithms the ratings are not directly used in order to generate the recommendations. Indeed, in this approach a model is developed using data mining, machine learning algorithms based on the ratings to make predictions [85]. The models are first trained using a training set, and then they can be used to generate the predictions.

Typically, classification methods, such as Bayesian networks, association rule mining or latent semantic analysis, can be used as collaborative filtering models when the user ratings are categorical, and regression models and SVD methods can be used when the ratings are numerical.

4.2.2.3 Hybrid collaborative filtering

Another approach to collaborative filtering is the hybrid collaborative filtering which combines the memory-based and the model-based CF methods in order to overcome the shortcomings of each approach. Hybrid collaborative filtering, not only, improves the prediction performance, but also, it overcomes some of the major collaborative filtering problems such as sparsity and loss of information. Many commercial recommender systems, such as Google news recommender system utilize this this approach. Nevertheless, these improvements come at the cost increased complexity and more expensive implementation.

The main pros and cons of the three approaches are summarize in the following table [84].

Table 4-1 A comparison between three collaborative filtering approaches.

Approach	Pros	Cons
Memory based	- Easy implementation - Incremental addition of new data - Being aware of content of the items being recommended are not required	- Dependency to human ratings - Performance degradation due to data sparsity - Cannot handle new users and items - Does not scale well with data size
Model based	- Handle the scalability sparsity, and other issues better. - More accurate predictions - Having an intuitive rationale for recommendations	- Building the model has a rather high complexity - Precision of the prediction comes at the cost of a lower scalability
Hybrid method	- Overcomes the shortcomings of the other approaches	- More complexity and implementation cost

4.2.3 Hybrid Recommender Systems

Hybrid approach as its name suggests, combine both collaborative and content-based filtering under one single framework to overcome the shortcomings of either approach and take advantage of the strength of each approach. In fact, there are

numerous ways for combining collaborative and content-based approaches, which includes:

- Using each method separately and combining them in order to achieve the ultimate recommendation.
- Implementing a collaborative filtering approach and adding some content-based features to achieve recommendations of higher quality.
- Implementing a content-based approach and adding some collaborative filtering features to achieve recommendations of higher quality.
- Implementing a model which unifies both approach.

Chapter 5

Content Based Image Retrieval

5.1 The Introduction of the CBIR

Nowadays, digital images are commonly utilized in many areas from biology, and medicine, to face or finger print recognition. Therefore, effective methods for searching and retrieval of images have witnessed considerable interest. The traditional image retrieval systems worked based on keywords and captions representing the images [94][95]. These captions were generated manually which was quite inefficient and expensive. The manual keyword generation process even is not able to capture every noticeable keyword that describes the image. In order to overcome the major deficiencies of the conventional approaches, content-based image retrieval (CBIR), also known as query by image content (QBIC) and content-based visual information retrieval (CBVIR), was introduced. The term "content-based" alludes that the contents of the image are used in the retrieval process rather than the metadata such as keywords, or captions associated with the image which were traditionally used.

Most CBIR system rely on low-level image features, i.e., color, texture and shape which are extracted from individual images and arranged in some predetermined way forming an appropriate feature vector (FV). When the system receives a query image it is compared to these features using a distance metric. The images with the minimum distance to the query are displayed as the result. In order to increase the overall efficiency of the system, the low-level features are usually stored in a database. There are plenty of distance metrics reported in the literature which are presented in the following sections.

The main advantages of the CBIR systems is that the retrieval process is automatic and there is no need to manually provide metadata for images. However, one of the difficulties the CBIR systems have is that images are complex to manage. Also, image indexing is not a trivial task. Furthermore, they require a mapping from the

high-level user perceptions into low-level image features, which is known as the "semantic gap" problem. A survey on this concept presented in [103].

5.2 Architecture of CBIR Systems

A typical architecture of a CBIR system is depicted in Figure 5-1. As mentioned before, the system is able to receive a query image and return the most similar images to the query image which are stored in the image database. The queries are processed based on the low-level features of the images. The query-processing module is responsible for extracting the features of the query image and comparing it to the stored images. Once the similarity between the query image and the stored images are computed based on a similarity measure, the stored images are ranked based on their distance to the query image and the most similar ones are retrieved.

In order to increase the efficiency of the system, the features vectors representing each image are extracted and stored in the database. This process, which is done by data insertion subsystem, is usually performed off-line. It is worth mentioning that, like other forms of databases, image databases are typically indexed too. The image indexing is performed based on the feature vectors extracted from each image, and utilizes structures such as M-tree [97] or Slim-tree [98] to accelerate the similarity computation and retrieval processes.

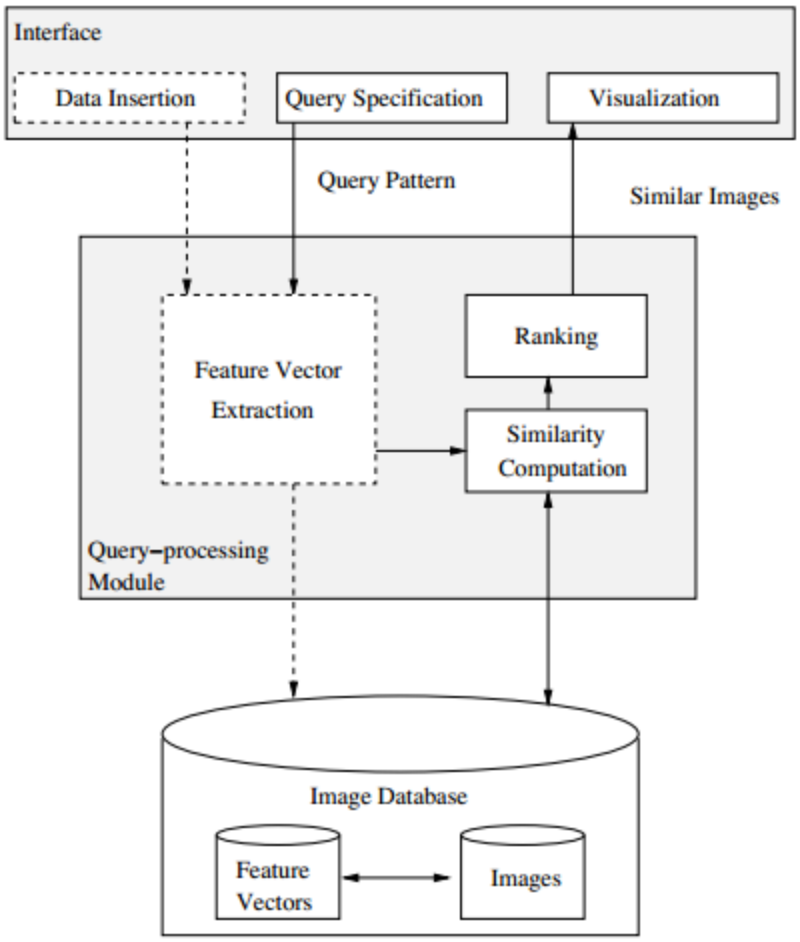


Figure 5-1 Architecture of CBIR system [105].

5.3 Features utilized by CBIR systems

Typically, each CBIR system extracts some features, such as color, edge, texture, and shape, from each image to describe it. Each feature represents a specific visual property of the image either globally for the whole image or locally for its regions or objects within it [104][105]. These low level features are described in the following sections.

5.3.1 Color Features

One of the most widely used features extracted from images is the color feature which is invariant with respect to scaling, translation and rotation of an image. There are some key approaches for extracting these features which are illustrated here.

5.3.1.1 Color Moments

The mean, variance and skewness of an image which are known as color moments (respectively, known as the first, second, and third color moments) characterize the color distribution in an image.

The first moment can be calculated by using the following formula:

(5-1) $$\mu_i = \sum_{j=1}^{N} \frac{1}{N} p_{ij}$$

where N is the number of pixels in the image and p_{ij} is the value of the j^{th} pixel of the image at the i^{th} color channel.

The second color moment can be calculated by using the following formula:

(5-2) $$\sigma_i = \sqrt{(\frac{1}{N} \sum_{j=1}^{N} (p_{ij} - \mu_i)^2)}$$

The third color moment can be calculated by using the following formula:

(5-3) $$\gamma_i = \sqrt[3]{(\frac{1}{N} \sum_{j=1}^{N} (p_{ij} - \mu_i)^3)}$$

5.3.1.2 Color histogram

A color histogram denotes a representation of the statistical distribution of colors in an image and disregard the spatial location of the colors where each histogram bin corresponds to a color in the quantized color space. The number of bins is specified by the number of colors in the image.

The color histogram can be measured for any kind of color space. The number of elements in a histogram is specified by the number of bits in each pixel of the image. Indeed, an *n*-bit image has 2^n values (or intensities) from the interval $[0, 2^n - 1]$.

Although the color histogram is relatively invariant regarding the translation and rotation about the viewing axis, and varies only slowly with the angle of view, it is significantly affected by changes in illumination, shading, highlight and inter reflections. To overcome this drawbacks, Gevers et al. [109], proposed a robust histogram for object recognition that is invariant to the aforementioned changes. Domke and Aloimonos [110], proposed a method to create color histogram that makes the histogram invariant under any mapping of the surface that is locally affine, and thus a very wide class of viewpoint changes or deformations. The main idea is that pixels in the two images can be 'weighted' using the gradients in different color channels. So, a deformation of an image region will change both the image and the weights equally.

5.3.1.3 Color coherence vector

Color histograms are a computationally efficient solution that have been widely used in many CBIR systems. However, they do not contain any spatial information which may defect the performance of the system facing different images with similar color distribution. Furthermore, they are affected by changes in overall image brightness.

Color coherence vector (CCV) [111] overcomes these drawbacks by partitioning each bin of histogram into two types: coherent, and incoherent. If a pixel belongs to large similarly-colored contiguous region it is categorized as coherent, otherwise it is considered as incoherent. So, the values of the pixels are partitioned into corresponding bins according to this definition.

5.4 Invariance

A typical CBIR system should be able to retrieve most of the most similar images to the query image. This means that these systems, not only, should retrieve the most similar images using proper similarity measures, but also, if an image is similar to the query image it should be retrieve, no matter if it has transformed geometrically, its view or even illumination has changed .

As mentioned before, CBIR systems utilize low-level features extracted from the images to describe them. These features are later used to compute the similarity between images. Nonetheless, not every low-level feature is geometrical transformation invariant. It is worth to mention that, invariance to changes in illumination, shading, highlights, inter reflections, as well as invariance to various kinds of noise are important too. Therefore, using these features will decrease the overall performance and accuracy of the CBIR system. There are many approaches presented in the literature to overcome this problem, from which we mention some.

To cope with the change of view when comparing two images, a typical solution is to simulate the original image to every possible view, extract the ordinary features. These features can be used in future matchings. Affine-SIFT (ASIFT), proposed by Morel and Yu [99][100], is a promising image matching framework which is based on this approach. However, the most important drawback of ASIFT is that its computational complexity is high. To overcome this drawback Yu et al. [102] proposed the iterative SIFT (ISIFT), for different viewpoint images. ISIFT estimates the geometric transformation between the image pairs iteratively. The estimated model is then used to simulate the query image. The simulated image and the query image are matched and results are converted to the original images.

Yao et al. [101] proposed a retrieval technique which extends histogram intersection method to retrieve the texture regions from a database of natural images. Indeed, they proposed two types of local edge patterns (LEP) histograms:

- LEPSEG for image segmentation,
- LEPINV for image retrieval.

While LEPSEG is sensitive to variations in rotation and scale, the LEPINV is invariant to variations in rotation and scale. It is worth to mention that their method performs well only on texture based retrieval. The experiments were performed on images that had been subjected to translation, rotation and/or scaling.

5.5 Distance Measures

Distance measures can be considered as the basis of every CBIR system. The similarity of images are measured using distance measures / metrics. The distance measures that exist in the literature are various and sundry, so we mention some of the most widely used methods.

5.5.1 Euclidean Distance

Thanks to its simplicity, among all the image metrics, Euclidean distance is the most commonly used.

Given two M by N images, $I = (i^1, i^2, ..., i^{MN})$ and $J = (j^1, j^2, ..., j^{MN})$, the Euclidean distance is computed by the following equation:

$$d_E{}^2(I, J) = \sum_{k=1}^{MN} (i^k - j^k)^2 \tag{5-4}$$

where i^k and j^k are the points belonging to each image.

5.5.2 Mahalanobis Distance

The Mahalanobis distance, also known as quadratic distance, is one of the most effective distance measures because it takes into account the correlation in original data.

It is worth mentioning that the main drawback of Mahalanobis distance is that the computation time is large and limited samples cause poor results.

Given two M by N images, $I = (i^1, i^2, ..., i^{MN})$ and $J = (j^1, j^2, ..., j^{MN})$, the Mahalanobis distance is computed by the following equation:

(5-5) $$d_M{}^2(I, J) = (\mu_I - \mu_J)^T \Sigma^{-1} (\mu_I - \mu_J)$$

where μ_I and μ_J are first color moment at pixel I and J and Σ^{-1} is the covariance matrix of the respective mean values.

5.5.3 Jaccard Distance

The Jaccard distance measures the asymmetric information on binary variables and the compares the vectors components. Due to its low computational complexity, Jaccard distance is a popular method in shape recognition. However, despite its strength that has made it popular, Jaccard distance is sensitive to image transformations.

Chapter 6

Proposed method

6.1 Related Works

The majority of recommender systems research papers are limited to movie, shopping, document, book, TV program, music, and suchlike fields [86].

Putting together the concepts of knowledge discovery and the collaborative filtering, Carrer-Neto et al. conducted a study [87] which presents a hybrid system that allows the suggestions to be made based on the semantic linkage between the contents and the users' preferences, as well as inheriting recommendations from the users' social network.

In order to capture implicit preference information, Lee et al. [88] have proposed a new CF-based recommendation methodology for mobile Web music, called CoFoSIM. The main characteristics of CoFoSIM are:

- Capturing the users' preference information by using a mobile Web usage mining technique called mWUM. This technique applies data mining methods on the mobile Web log data in order to discover the customer behavior patterns.
- Representing customer preference on an ordinal scale to decrease the estimation error.
- Applying a well-known consensus model called the CK method to enhance the quality of recommendations. Experiments with real mobile Web customers prove the superior performance of CoFoSIM compared to the existing CF algorithms in the mobile Web environment.

Yu et al. [89] have proposed a method to merge users' profiles in order to provide recommendations about TV programs. The rationale behind taking this approach has been the fact that people usually watch TV together. In order to provide program recommendation, this method merges all user profiles to construct a common user profile as its first step. Thereafter, a recommendation approach is used to generate a

common program recommendation list for the group according to the merged user profile. The evaluation results prove the effectiveness of their method when used to recommend TV programs to groups of people.

Valdéz et al. [90] have conducted a research to define and capture some parameters which helped them to analytically find the correlation between the explicit and implicit feedbacks on recommender systems. They have provided a more advanced method for taking feedbacks from users of electronic books which captures their preferences implicitly instead of explicitly rating, which may hinder the user experience.

Serrano-Guerrero et al. [91] have proposed a fuzzy linguistic recommender system as a tool for communicating researchers interested in common research lines, which is based on the Google Wave[1] capabilities. The proposed system automatically suggests several researchers and useful resources for each wave. Their proposed recommender system utilizes information related to user preferences and resource description along with the wave description, which are provided by the systems users and the administrators of the waves.

Zaiane [92] has suggested a recommender system for e-learning purposes. The author has used data mining techniques in order to build a software agent which is able to recommend on-line learning activities or shortcuts in a course website based on the learners' access history. The aim of the system is to improve users' experience through providing them the ability to better navigate the on-line materials by finding the relevant resources faster, and assisting the online learning process.

Castro et al. [93] have proposed a fuzzy logic based recommender system which provides the users with knowledge about the existing relationships among the

[1] https://wave.google.com

attributes that describe a given product category. Their system is applicable to B2C[1] portals. They have developed a prototype of a B2C e-commerce portal to demonstrate the benefits of their proposed system. It is worth to mention that their system is based on multi-agent systems foundation whose specifications are defined by the FIPA[2] committee.

We have mentioned some recommender systems used in various fields, but one thing that all of these systems have in common is that they do not take images into account. As we mentioned, images are not trivial and play a significant role in the process of decision making in many scenarios. In order to overcome this shortcoming we have conducted the current study.

6.2 Proposed Method

As implied from the previous section, images, despite their undeniable effect on users' decisions and desirability and acceptability of a computer system which works on data that has at least a visual part, have not taken enough attention from the researchers who work on recommender systems. Suppose you search for a car and everything except the shape of the cars participate in the ranking of items. To address this issue, we have proposed a method to take the image part of each item's information into account in order to propose the most desirable items to users. It is worth mentioning that not only we have introduced such a parameter in our rankings, but also we have tried to use as much information to give offers as possible.

A general view of the proposed system is depicted in Figure 6-1.

[1] Business-to-Consumer

[2] Foundation for Intelligent Physical Agents

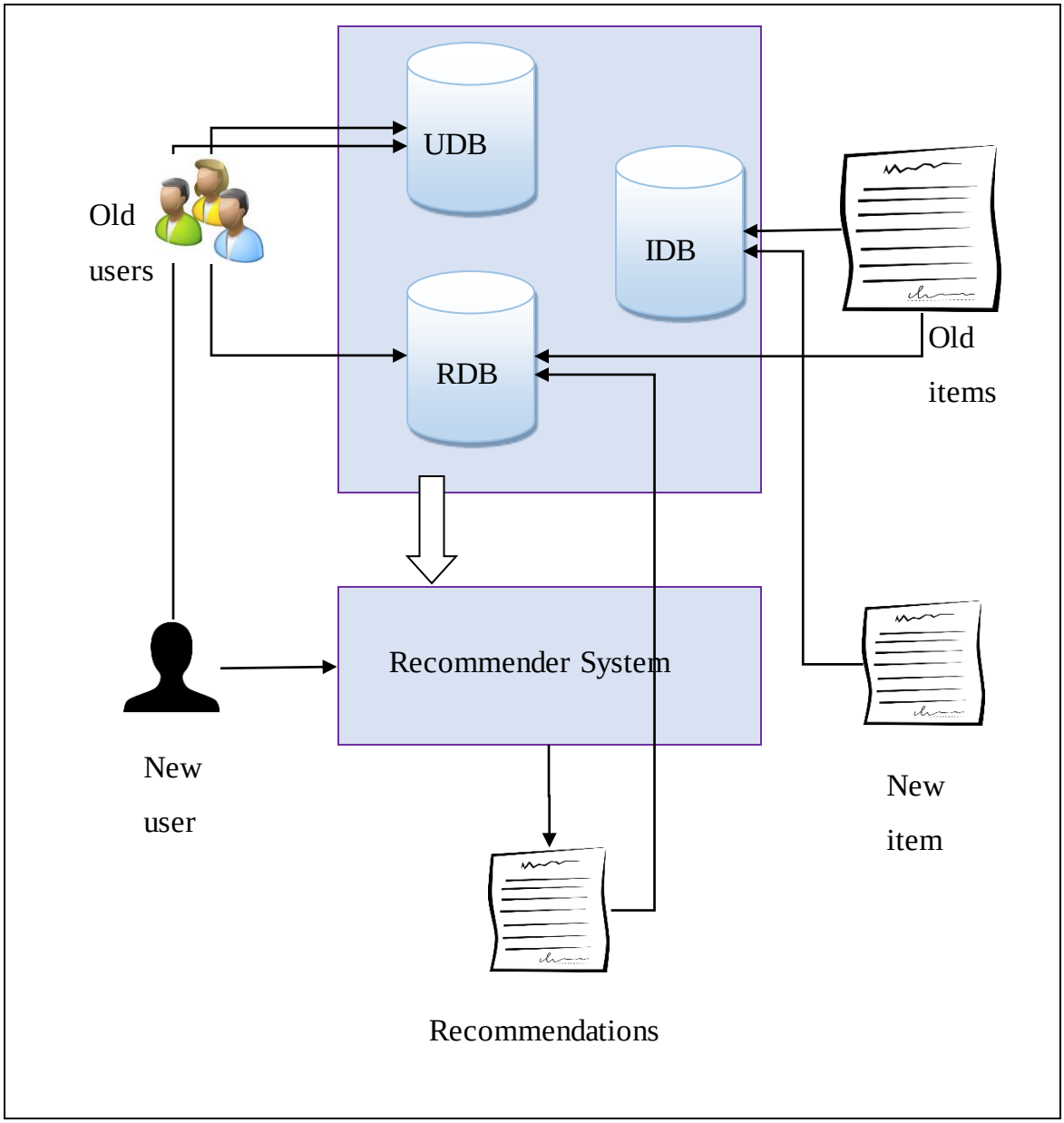


Figure 6-1The general architecture of the system.

As the depicted in the above figure, one of the main parts of the system is storage part. The storage part is constituted from some distributed databases that contain three main entities of the system:

- Users (stored in UDB);
- Items (stored in IDB);

- Users' ratings (stored in RDB).

When a new item is added up to the system, it will be stored in the proper part of IDB. As it can be alluded from the term 'proper', there is a mechanism for storing each item into the databases. In our proposed approach each item should be stored in a database which contains the items having most similarity to the new item.

When a new user arrives into the system, along with adding her to the proper part of the UDB, the system should compute the best recommendations for the user, i.e. the items which receive the highest ratings from the recommender system. The users' future rating will be added to the RDB. Again, the most similar users are stored in the same parts of the distributed database.

In order to store the image parts of the items in the proper databases we have used the distributed Kd-Tree implemented by MapReduce architecture [114]. We have briefly discussed the Kd-Tree and MapReduce in Appendix A and B respectively. It should be noted that the IDB consists of two type of databases:

- Image databases (IMDB)
- Text-based databases (TDB)

IMDB contains the image part of each item, while the TDB contains the other features of an item. For example, when the items are flowers the IMDB will contain the picture of each flower, while the TDB may contain some information about that flower. Therefore, IMDB has a Kd-Tree structure, and TDB has its own structure. We have proposed an approach to store the non-image part each item that we call descriptive attributes. Text-based parts are stored in tree structure where each node corresponds to one of the feature vector of the items. Given an item $I = \{IMG, a^1, a^2, ..., a^D\}$, where IMG denotes the image that belong to the item, and a^i denotes the i^{th} descriptive attribute, each node of the tree corresponds to one of the attributes from a^1 to a^D. In other terms, each node is split based on an attribute. The order of attributes in the tree is determined based on the degree of separation made

by that attribute. Indeed, if an attribute separates the data into more equal categories it will be located closer to the root. Therefore, the root of the tree splits the whole data more equally compared to any other node.

Our proposed method, which falls into the hybrid recommender systems category is comprised of a CF component and a CBF component, which are modified properly to work with items that have an image part. This modification is made in the definition of the similarity metric used for item-item comparisons. We define the similarity of two items i_1, i_2 using the following formula [126]:

$$Sim(i_1, i_2) = \alpha.grSim(i_1, i_2) + \beta.trSim(i_1, i_2) \quad (6\text{-}1)$$

where *grSim*(.) refers to a function which measures the similarity of two items based on their graphical feature, which is the only picture of the item in our case, *trSim*(.) refers to a function which measures the similarity of two items based on their text-based features, and α and β are two weighting parameters chosen from the interval [0,1]. We have assumed that each item has a graphical feature set (images) and a text-based feature set (categorical, Boolean, numerical, etc.). It is worth to mention that we have assumed that each item has only one image.

While the values of α and β can be found by trial and error, in our proposed method, the near-optimal values for these parameters are found through optimization using Genetic Algorithm (GA) [117]. In our proposed method, each chromosome is comprised of two floating point gens, as illustrated in Figure 6-2.

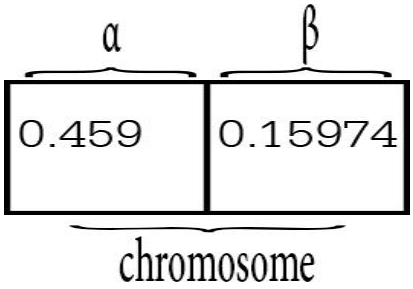


Figure 6-2 A sample chromosome with arbitrary values for α and β.

The fitness function of each chromosome is calculated through a slightly complicated process. First a random set U_{eval} of users are selected from the user dataset by uniform probability. A subset I_{eval} of items are selected from the items, as well. For each user U_i the rate of each item I_j is calculated by CF component R_{CF} and CBF component R_{CBF} separately. Then the average of R_{CF} and R_{CBF} is considered as the temporary result of the system R_{TMP}. This temporary result is compared to the rating R_{real} of that item given by the user U_i. The fitness function is calculated through the following formula:

$$fitness(C) = \sqrt{\sum_{(u,i)\in U_{eval}\times I_{eval}} (\mathrm{R}_{real} - \mathrm{R}_{TMP})^2}$$

It should be notified that the chromosome determines the weight of qualitative and quantitative features in measuring the similarity of two items. So the better these weights are set, the lower is the error in estimation and fitness value will be smaller, as we try to minimize the fitness function.

In order to find the text-based similarity of two items, first we categorize numerical features; then simple matching coefficient (SMC) in order to measure the similarity of two items:

(6-2) $$SMC(i_1, i_2) = \frac{\text{Number of matching attributes beween } i_1 \text{ and } i_2}{\text{Number of textural attributes}}$$

Let X and Y denote the intensity values of two $M \times N$ images. In order to find the graphical similarity of images, we have used the method introduced in [115] called IMNCC[1] :

[1] IMage Normalized Cross-Correlation

(6-3) $$IMNCC(X,Y) = \frac{\sum_{i=1}^{MN}\sum_{j=1}^{MN} g_{ij} x_i y_j}{\sqrt{\sum_{i=1}^{MN}\sum_{j=1}^{MN} g_{ij} x_i x_j \sum_{i=1}^{MN}\sum_{j=1}^{MN} g_{ij} y_i y_j}}$$

where g_{ij} is defined the same way used in IMED[1] [115]:

(6-4) $$g_{ij} = \frac{1}{2\pi\sigma^2}\exp\left(\frac{-dist(P_i, P_j)^2}{2\sigma^2}\right)$$

where $dist(P_i, P_j)$ denotes the spatial distance between the pixels i and j.

The algorithm that computes this similarity measure using MapReduce is illustrated in Figure 6-3.

Parallel version of IMNCC

- Map(ID, Vector row):
  - for each pixel in in the row :
    - EmitIntermediate ($g_{ij}x_i y_j$, $g_{ij}x_i x_j$, and $g_{ij}y_i y_j$).
- Reduce(ID, Iterator values):
  - Term1 = Term2 = Term3 = 0;
  - for each value (a,b,c) in values:
    - Term1 += a;
    - Term2 += b;
    - Term3 += c;
  - Emit (a/sqrt(b.c))

Figure 6-3 The parallelized version of the IMNCC method using MapReduce.

[1] IMage Euclidean Distance

As we mentioned previously, the proposed system uses a combination of the CBF and the CF methods. The role of the CBF component is to record the users' preferences. This component maintains a profile for each user which is built to indicate the type of item this user likes. In other words, this component tries to recommend items that are similar to those that a user liked in the past. The CF component is used to predict what users will like based on their similarity to other users. As mentioned in the previous chapters the combination of the components can be carried out in many ways from which we have used the weighted approach, i.e. the score of these components are combined numerically. Considering the fact that the main motivation of conducting this study has been to consider the images in the processes of generating the recommendations along with the text-based features this combination is carried out without any complication. In fact, the average of the scores provided by each component is considered as the ultimate score for each item which represents the desirability of the item from the active user's point of view.

In other words, the process of producing the recommendations is composed of the following steps:

Finding neighbor: the most similar users to the active user form a proximity-based neighborhood with him. Therefore, for each user u_a a group of users $N_{u_a} = \{u_1,...,u_{N_a} | u_i \in \mathrm{U}\}$ are selected so that each user u_i has the maximum similarity to u_a compared to every other user from the set $U - N_{u_a}$. In our notations, N_a denotes the number of neighbors the user u_a. In our settings this number is equal for all the users. It is worth to mention that finding the optimal N_a is another issue that is not addressed in this study, and we have simply set it to 10.

Producing Social Recommendation: we have proposed a very fast method for this step which is as follows. First, for each user u_j the item i_j^{best} received the most score from that user, which is recorded in his profile, is put into a list $R_S = \{i_1^{best},...,i_{N_a}^{best}\}$. N

items that occur most often in R_S are selected and inserted to a list of tuples. The first entry of each tuple is one of the selected items and the second entry corresponds to its score. The score of each item is equal to the average of its score among the all of its occurrences in the list R_S.

Producing Profile-based Recommendation: This is step is quite straight forward. *M* items having received the most scores from the active user are selected into a list R_P. This list contains the items and their corresponding scores the same way discussed for the list R_S.

Producing the Final Recommendation: In this step the results of the two categories of recommendation, i.e., R_S and R_P are merged into a final list *R* which contains *K* items that have the most scores. The items can be selected from each list. In the case that an item is present in both of the lists, its score is considered to be the average of the corresponding scores on either of the lists.

Clearly, *K* must be less than $M + N$, however, either of *M* and *N* can be greater than the other one. Indeed we can adjust the effect of each approach on the final recommendation using the amount of these parameters. In our experiments *M* and *N* are both set to $2 \cdot K$.

6.3 Experimental results

In order to evaluate the proposed method we have conducted a comprehensive set of experiments.

A considerable issue in the area which we have put our focus in this research is the lack of a sensible general purpose dataset. Consequently, we decided to generate our dataset which contains all the information required for both running the system and evaluating it. In this regard, we gathered information of 10000 houses from some local real estate agents and online real estate websites and asked 250 different people to give 500 items a rank from 1 to 5 based on the pictures and other features of the

house with overlaps in items they have ranked. The education level, occupation and gender of the evaluators were intentionally quite different, to reflect a more realistic evaluation.

The proposed method was implemented on 4 machines running Hadoop on them. Each machine has 4GB RAM DDR 2, an Intel Core 2 due CPU and 500GB HDD.

The text-based features of the items of the data set used in our experiments are illustrated in Table 6-1.

Table 6-1 text-based features of items.

Feature	Type	Details
Area	Integer (quantized)	Square meters
Bedrooms	Integer	-
Type	Categorical	House, Apartment, Office, Villa
Price	Integer	Currency (IRT)
Loan	Integer	Currency
City	Categorical	41 cities
Suburb	Categorical	659 suburbs

Furthermore, we recorded some key characteristics of each person who participated in the evaluation process. These characteristics are presented in Table 6-2.

Table 6-2 Characteristics of each user.

Feature	Type	Details
ID	Numerical	Discrete
Gender	Boolean	2 genders
Age	Integer (quantized)	5 categories
Occupation	Categorical	13 categories
Education level	Categorical	5 categories
Family Size	Numerical	-
IsHeadofFamily	Boolean	-

In order to provide a better understanding of the items used our experiments, an item from this data set is depicted in Figure 6-4.

Area	Bedrooms	Type	Price	Loan	City	Suburb
200	3	Villa	57M	0	Sari	Chams

Figure 6-4 An item from the data set use in the experiments.

In order to evaluate the accuracy of the proposed method we have used the Root Mean Square Error (RMSE) metric. A subset of items was represented to a set of new users and their ratings were recorded. It is worth mentioning that some users had already have rated the items, but in cases that the item was new to the users, they were asked to rate it. This process repeated exactly using the proposed method and the ratings were recorded, with the exception that each item's rating was calculated anew by the recommender system even if the user's rating was available in the dataset. Then RMSE was calculated using the formula (6-5).

(6-5) $$RMSE = \sqrt{\frac{1}{| U_{test} \times I_{test} |} \sum_{(u,i) \in U_{test} \times I_{test}} (\hat{r}_{u,i} - r_{u,i})^2}$$

We used NRMSE metric to evaluate the precision of our proposed method in combining image part of item with its other features when used by a recommender system. The NRMSE is simply RMSE normalized by the range of the ratings, i.e. r_{max}-r_{min}. Lower NRMSE corresponds to more accurate recommender systems. In the above formula, $r_{u,i}$ denotes the rating given to the item i by user u and $\hat{r}_{u,i}$ denotes the rating predicted by the system for user u and item i.

(6-6) $$NRMSE = \frac{RMSE}{r_{\max} - r_{\min}}$$

In the following picture Graphical is same as our proposed approach without considering the text-based features, and Text-based is same as our approach without considering the image of an item.

Figure 6-5 NRMSE metric measured for different approaches.

As we mentioned before, traditional recommender systems either do not consider the images that belong to items at all, or consider them implicitly through the users' ratings. So far, the results have shown the significant improvements due to introducing images into recommender systems, which proves that considering images explicitly in these systems is quite promising. The plot depicted in Figure 6-6 proves that it is possible and practical.

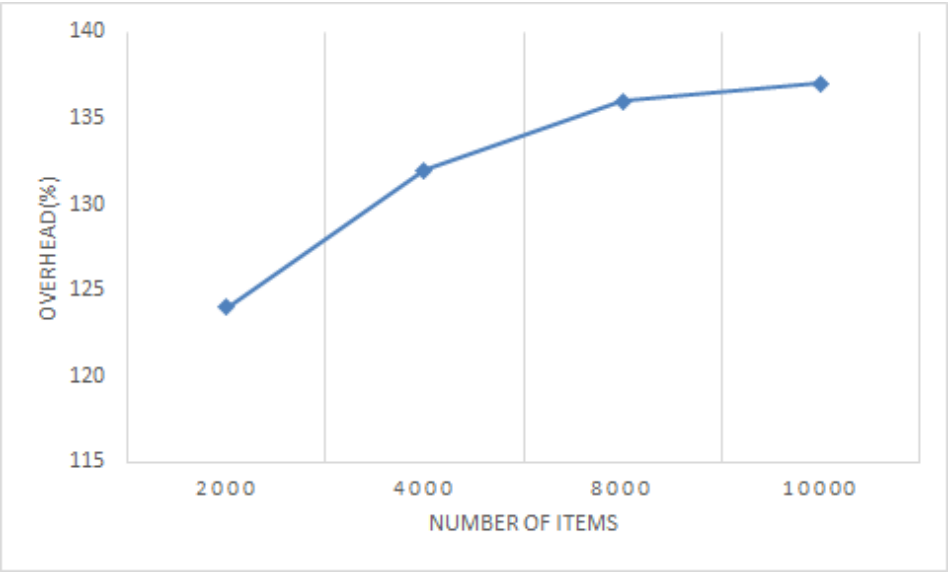


Figure 6-6 The overhead due to measuring the graphical similarity, i.e., considering the images explicitly.

Indeed, the results show that using the distributed parallel approach that we have proposed for comparing the images of items still the system is fully practical, and the overhead has never exceeded 140 percent.

In order to show the effectiveness of our distributed similarity computation approach, we have implemented the whole system again, and the only difference is that the search method that used to retrieve the similar images is exhaustive search. The results are depicted in the following figure.

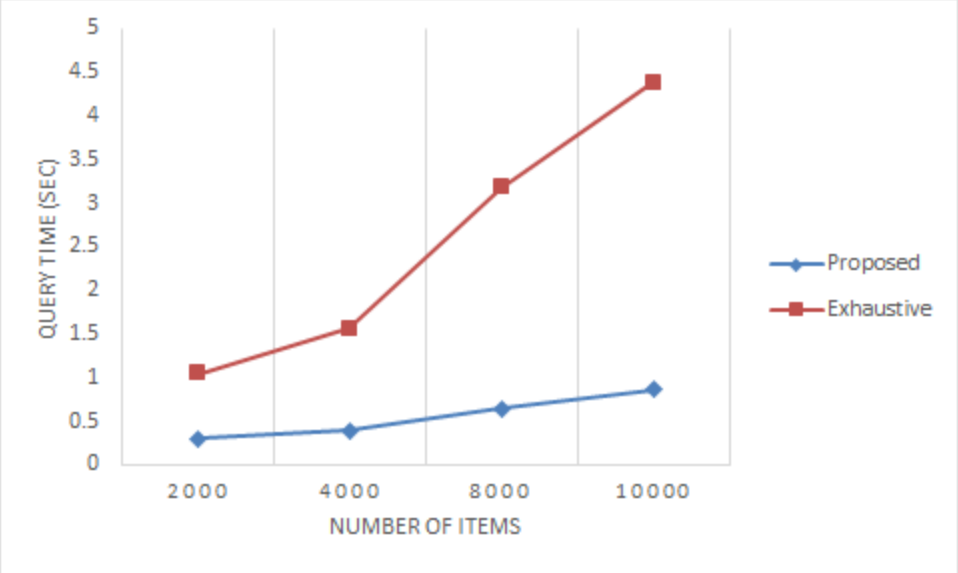


Figure 6-7 Comparison of query time between proposed method and exhaustive search-based method for different number of items.

This plot shows that the distribution of similarity measurement process that is proposed in this dissertation has a considerably positive effect on the processing time of each query. It should be noted that in the Exhaustive approach we did not use the Kd-Tree structure as well.

Chapter 7

The Proposed Image Retrieval Methods

7.1 First Proposed Image Retrieval Method

According to the fact that our ultimate goal is to use the image retrieval method in a system dealing with humans, not machines, we proposed a method for the retrieval of images which is developed based on human visual system. The proposed method, first decomposes each image into different frequency bands, each of which represents different details of the image. In order to compare two images, the features of the corresponding sub-bands are used. To this aim, complex wavelet transform has been used, which like Gabor filters covers almost the entire frequency space, and have a high computational speed. In addition, the HSV color space is quantized so that matches the human visual system to high extent. Our proposed method was applied on a dataset including about 3000 images and the results compared to those of Color Correlogram method, which proved the superiority of the proposed method.

7.1.1 The Proposed Method

In content based image retrieval, some features of images, such as color, texture and shape are extracted and used to index the images. Swain and Ballard introduced the color histogram method in 1991 [91]. This method counts the number of pixels which have colors that belong to each of a fixed list of colors that are made through quantization of the colors. An important advantage of histograms is that they are invariant to rotation and translation around an axis perpendicular to the image, and vary only slowly with the angle of view. Therefore a three dimensional item can be retrieved easily despite a small change in the angle of view. Nevertheless, the disadvantage of this technique is that images with completely different shapes may have similar color histograms, which may occur more frequently as the number of images in the dataset increases.

In order to overcome the problems of the color histogram, the color coherence vector (CCV) method was proposed [92]. This method adds some extra information to the typical histogram, which correspond to the spatial positions of the pixels. According

to this approach, the pixels which belong to each bin are classified based on their local features either as coherent or incoherent and the number of pixels which belong to each category are stored. Coherent pixels are part of a big connected component, whereas incoherent pixels are part of a small connected component. Despite the superiority of the CCV method over the color histogram, it still has a major disadvantage that is the extraction of feature vectors takes too long.

In a study conducted by Huang et al. [93] color correlogram has been used to characterize each image. The rationale behind using color correlogram has been to consider the spatial relations of pixels and to overcome the shortcomings of the color histogram. This approach is based on the assumption that the spatial correlation of each pair of colors in different images are different. This approach has proved effective, however, it has the same prohibitive problem as what the CCV suffers from, i.e. suffering from a high computational complexity.

The best metric to evaluate an image retrieval system is the evaluation of the system made by its users. Consequently, using the same methods as what humans use to find similar images should be effective in order to increase the acceptability of such systems [115]. There are different spatial frequency channels in human vision each of which function as a band-pass filter. Therefore, this vision system can be considered as a filter bank which is comprised of filters each of which receive and process a specific part of images. High details parts of an image and its edges correspond to high frequencies and parts of the image with smooth variations, which usually include the regions inside the objects or backgrounds, correspond to low frequencies. Therefore, parts of an image which have different frequencies may belong to the same object or to different objects. That being said, human visual system works like this: first, a frequency analysis is carried out on the image that is formed on the retina, which decomposes the image into different frequency layers. Thereafter, the equivalent parts of images are compared with each other which forms the basis for the general evaluation of the similarity of two images.

Our proposed method [127] is comprised of two main parts. The first part determines how the filter bank should be implemented and applied to decompose the input image into layers, and the second part proposes a method to extract features from each layer. The block diagram which illustrates how these layers are extracted has been depicted in Figure 7-1.

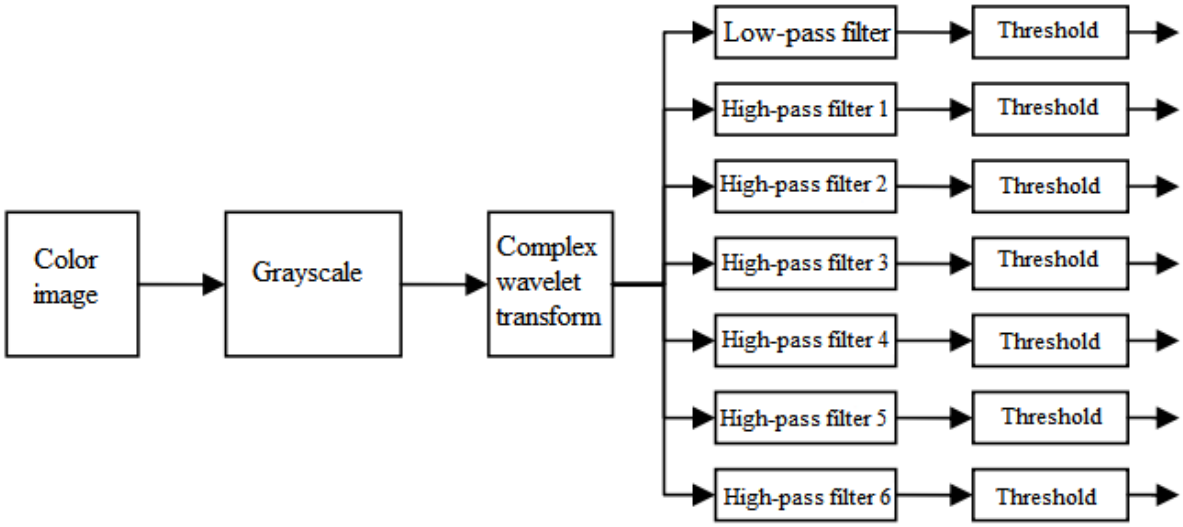


Figure 7-1 The block diagram of the layer extraction system.

With the extraction of these layers, each layer represents a different range of spatial frequency. The first layer determines the regions with low-frequency which correspond to backgrounds or the interior regions of objects, while the other 6 layers represent the regions which include more details in. If the variation range of the value of the pixels after the application of wavelet transform filters are set to be in the interval [0,1], all the threshold can be set to 0.5. For the first layer, the pixels whose value is lower than 0.5 and for the rest of the layers the pixels whose value is higher than 0.5 are taken into account and the other pixels are neglected. Moreover, in order to decrease the length of the feature vector, which results in a lower computation complexity, we can use only 3 high-pass filters out of the 6 high-pass filters mentioned before. In this case, only the first, third, and fifth filters are uses.

Thereafter, color features of each layer are extracted using ordinary color histogram method. At this stage the RGB color space is converted to HSV color space, because it complies better with human perception of colors. Then, this space is quantized into 166 distinct values. This quantization is based on [125] according to which 18 distinct values for H, 3 distinct values for both S and V and 4 distinct values for gray level are used. The reason why we have quantized the color space as described is that it is proved that this approach is quite similar to the human visual system perception of colors. Therefore, 7 histograms are created for the 7 layers as mentioned previously. Furthermore, the reason why complex wavelet transform is used is that they are very effective in extraction of texture features from images, which can be witnessed in Gabor filters [116], and at a higher speed compared to Gabor filters.

In the proposed method, the complex wavelet transform is applied to the image up to 4 levels. In addition, the values of mean, standard deviation, and kurtosis are calculated for the sub-images resulting from each of the 6 high-pass filters. Therefore, a vector composed of 72 entries is achieved for each image which is added to the color feature of layers which mentioned previously.

7.1.2 Experimental Results

In order to evaluate the proposed method, it has been implemented using Matlab running on a PC with an Intel CORE 2 Due 2.20 GHz processor and 4GB of RAM, on which the extraction of features from an image takes about 2 seconds. A collection including 3000 images borrowed from Hamshahri dataset was used in our experiments. The main reason for using this dataset was the fact that this dataset is extracted from an online news website so the images have come with very precise details about the category or categories the image belong to and each image is supported with a list of keywords which describe its contents. This gave us the ability to precisely verify the effectiveness of the proposed method. Although the Hamshahri dataset includes 36 main categories with many more sub-categories, we

used 5 main categories in our evaluations including sports, buildings, animals, weather forecast, and accidents.

Figure 7-2. A number of pictures selected from the dataset used to evaluate the proposed method.

A number of pictures which belong to this dataset have been illustrated in Figure 7-2.

Two common metrics, namely precision, and recall have been used to evaluate the proposed method. These metrics have used in various applications and their definitions have been adapted to make them suit the application properly. We have used the definitions given in [118]:

(7-1) $$P_j = \frac{\text{Number of retrieved and relevant elements in the first j positions}}{\text{j}}$$

(7-2) $$R_j = \frac{\text{Number of retrieved and relevant elements in the first j positions}}{\text{Total number of relevant elements in the collection}}$$

We have used the aforementioned tags provided for each image in order to verify whether the image is relevant to the query or not.

According to the equations (7-1) and (7-2), P_j shows an almost decreasing behavior when j increases while the opposite is true for R_j. As we mentioned before, color correlogram is one of the best methods in CBIR, so we have compared our proposed method with this method. The results have been shown in Table 7-1.

Table 7-1The results of experiments conducted using the proposed method (Pr) and the color correlogram (CC) method.

Category	P_5	P_{10}	P_{15}	P_{20}	R_5	R_{10}	R_{15}	R_{20}	Method
Sports	1	1	1	0.94	0.22	0.47	0.79	0.8	Pr
	1	0.94	0.81	0.84	0.17	0.39	0.71	0.78	CC
Buildings	1	1	1	1	0.14	0.33	0.59	0.63	Pr
	0.89	0.74	0.86	0.86	0.12	0.28	0.45	0.57	CC
Animals	1	0.99	0.99	0.98	0.1	0.21	0.31	0.46	Pr
	1	0.96	0.83	0.76	0.09	0.21	0.25	0.9	CC
Weather Forecast	1	0.99	0.99	0.87	0.14	0.22	0.32	0.37	Pr
	0.96	0.92	0.85	0.86	0.15	0.19	0.28	0.31	CC
Accidents	0.98	0.98	0.98	0.98	0.04	0.09	0.21	0.29	Pr
	0.81	0.82	0.89	0.85	0.03	0.06	0.11	0.14	CC

As it is implied from the above table the proposed method shows a consistent superiority over the color correlogram method which is a competent method in the domain of CBIR.

7.2 The Second Proposed Image Retrieval Method

Firstly, we should mention that in the proposed system, some general low-level features are extracted from the images contained in the database of images. The database is represented by $X = \{X_1, X_2, ..., X_N\}$ where, X_i refers to the i^{th} image. For

every image X_i there is a feature vector F_i. Therefore, the image database contains a database of feature vectors $F = \{F_1, F_2, ..., F_N\}$ corresponding to X. We extracted four types of low-level visual features, three of which are color features and one of them is a texture feature.

7.2.1 Features

7.2.1.1 Color Moments

The mean, variance and skewness of an image which are known as the color moments (respectively, known as the first, second, and third color moments) characterize the color distribution in an image.

The first moment can be calculated by using the following formula:

(7-3) $$\mu_i = \sum_{j=1}^{N} \frac{1}{N} p_{ij}$$

where N is the number of pixels in the image and p_{ij} denotes the value of the j^{th} pixel of the image at the i^{th} color channel.

The second color moment can be calculated by using the following formula:

(7-4) $$\sigma_i = \sqrt{(\frac{1}{N} \sum_{j=1}^{N} (p_{ij} - \mu_i)^2)}$$

The third color moment can be calculated by using the following formula:

(7-5) $$\gamma_i = \sqrt[3]{(\frac{1}{N} \sum_{j=1}^{N} (p_{ij} - \mu_i)^3)}$$

7.2.1.2 Color Histogram

A color histogram denotes a representation of the statistical distribution of the colors in an image, and disregards the spatial location of the colors. Each bin of the histogram corresponds to a color in the quantized color space. The number of the bins is specified by the number of colors in the image.

The color histogram can be measured for any kind of color space. The number of elements in a histogram is specified by the number of bits in each pixel of the image. Indeed, an n-bit image has 2^n values (or intensities) lying in the interval $[0, 2^n - 1]$.

Although the color histogram is relatively invariant under the translation and rotation about the viewing axis and varies just slightly with the angle of the view, it is significantly affected by the changes in the illumination, shading, highlights, and the inter reflections. To overcome these drawbacks, Gevers et al. [110] proposed a robust histogram for object recognition that is invariant to the aforementioned changes. Domke and Aloimonos [110] proposed a method to create a color histogram which makes the histogram invariant under any mapping of the surface that is locally affine, which includes a very wide range of the viewpoint changes or deformations. The main idea is that the pixels in the two images can be weighted using the gradients in different color channels. Therefore, a deformation of a region of the image will change both the image and the weights equally. We have used the method proposed in [118] in this study.

7.2.1.3 Edge histogram

Edges are among the important features of images. In the MPEG-7 standard the edge histogram is used in order to capture the edge distribution of the image. However, its drawback is that it contains only the local edge distribution with 80 bins. In this paper, we have used the method proposed in [119], which generates two other types of the edge histograms: the global, and semi-global histogram bins.

7.2.1.4 Gabor features

The texture features of each image are captured using the Gabor filter. The Gabor filter provides optimal joint resolution in both frequency and spatial domains [120]. We have used 30 filters with five different scales and six different orientations [121]. We first create a grayscale equivalent for each image and normalize it so as to have dimensions of 256×256 to speed up the computation using the Fast Fourier transform

(FFT) method. After filtering the images, their means and standard deviations are computed which provide 60 features for each image.

7.2.2 Weight adjustment

In this proposed method, the query vector is broken into several multi-query vectors in regard to every feature space to perform the short-term learning. Initially, the user feeds the system with a query image Q, then, the features of the query image are extracted- these features are illustrated in the following section. The extracted feature vector is then used by the similarity function to retrieve *P* relevant images from the stored images. The images are then partitioned into two categories by the user: the relevant images, and the irrelevant images. At the next stage, the multi-query approach is used. This is done by clustering the images that are categorized as relevant, and using each cluster centre as a new query. Subsequently, the minimum distance of each image from the cluster centres is considered as its similarity to the query. We have used the WPGMC[1] algorithm to perform the clustering.

In order to combine the results of the retrieval based on each feature *i*, the ranking of each image in the database is computed when images are retrieved based solely on that feature. This ranking is then used to adjust the weight of each feature. Suppose we have *N* images in the database, and we offer the user only the *M* first retrieved images. First, the rankings of each of these *M* images are added together. Then according to the difference between the summation achieve at the stage *t* and the stage *t*+1 the new weights are computed. It should be noted that initially the weight of all the features are equal. In order to shed more light on this process, we bring an example. Suppose the database contains 8 images, i.e., $\{A, B, C, D, E, F, G, H\}$, and the system shows the user the first 4 images, i.e., $\{I_1, I_2, I_3, I_4\}$. Furthermore, suppose we

[1] Weighted Pair Group Method Centroid

use three features f_1, f_2, f_3. Also, the first four features retrieved at the stage *t* are A, B, C, and D, and the first four features retrieved at the stage *t*+1 are E, F, G, and H. The orders of the images retrieved using each feature separately at the stages *t* and *t*+1 are represented in Table 7-2 and Table 7-3 respectively.

Table 7-2 The orders of the images retrieved using each feature separately at the stage t.

Retrieval based on each feature		
f_1	f_2	f_3
C	A	F
E	F	D
F	B	E
H	G	G
B	C	C
D	D	B
A	E	H
G	H	A

Table 7-3 The orders of the images retrieved using each feature separately at the stage *t+1*.

Retrieval based on each feature		
f_1	f_2	f_3
A	C	G
B	D	H
H	A	E
C	F	B
G	B	A
F	G	F
E	E	C
D	H	D

Considering Table 7-2 and Table 7-3, and the aforementioned assumptions, we can deduce the results shown in Tale 7-4.

Table 7-4 The rankings of the first 4 images according to each feature.

Stage		Image	The ranking based on each feature		
			f_1	f_2	f_3
t	I_1	A	7	1	8
	I_2	B	5	3	6
	I_3	C	1	5	5
	I_4	D	6	6	2
	The feature-wise sum of the rankings		19	15	21
	The normalized feature-wise sum of the rankings		19/55	15/55	21/55
$t+1$	I_1	E	7	7	3
	I_2	F	6	4	6
	I_3	G	5	6	1
	I_4	H	3	8	2
	The feature-wise sum of the rankings		21	25	12
	The normalized feature-wise sum of the rankings		21/58	25/58	12/58

It is worth mentioning that we call the sum of feature-wise ranking of the selected images the *feature-wise sum of the rankings*. The *normalized feature-wise sum of the rankings* for each feature is computed by dividing the feature-wise sum of the rankings for that feature by the summation of the feature-wise sum of the rankings.

As implied from Table 7-4, the normalized total ranking of the first 4 images considering only feature f_1 has decayed from 0.34 to 0.36 so its corresponding weight should decrease accordingly. On the other hand, the normalized total ranking of the first 4 images considering only feature f_3 has improved from 0.38 to 0.2,

therefore, its corresponding weight should increase accordingly. It is clear from this example that the values of the feature-based normalized rankings always sum up to 1. This is exactly the case for the weights of each feature in the similarity function. Therefore, considering the sign of the changes in these values, the total changes will always be equal to 0. Consequently, the difference between the normalized feature-wise sum of rankings of that feature at the stages t and t+1 is added the weight of each feature. That is, the weight of the feature f_i at the stage t+1 is computed using the following formula:

$$w_i^{t+1} = w_i^t + \left(\sum_{j=1}^{M} r_{ij}^t - \sum_{j=1}^{M} r_{ij}^{t+1}\right) \tag{7-6}$$

where r_{ij}^t is the ranking of the j^{th} relevant image when it is retrieved using only the i^{th} feature, and M is the number of the images that are given to the user. It should be reminded that the relevance of the images is determined by the user.

7.2.3 Experiments and Results

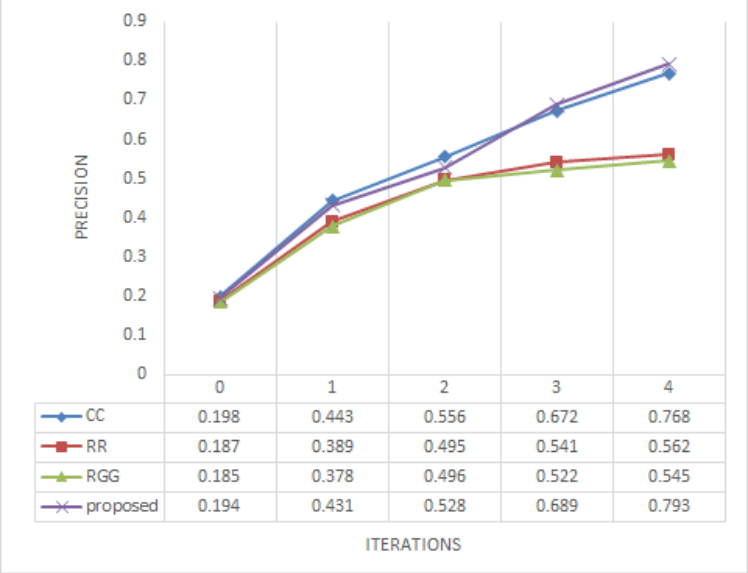
In order to evaluate the proposed method, it has been implemented using Matlab running on a PC with an Intel CORE 2 Due 2.20 GHz processor and 4GBs of RAM. A collection composed of 20000 JPEG formatted images borrowed from the Hamshahri2 dataset was used in our experiments [122]. Furthermore, 50 query images chosen from 50 different semantic groups were examined in the experiments. Each of these 50 images was used in a scenario wherein the image was given to the user as the query image, then the system offered the user a predefined number of images as the relevant images. Thereafter, the user was asked to provide a feedback and to decide which images were relevant and which ones were not. We have compared our proposed method with the following methods:

- CC: The Classifier Combination, which is based on the multi-query approach.

- RR: This method uses the Rocchio's method [97] for the query vector refinement and uses the Rui's method [123] for the distance function refinement;
- RGG: This method uses the Rocchio's method [97] for the query vector refinement and uses the method proposed by Guldogan & Gabouj [124] for the distance function refinement.

The precision of the proposed method is compared to the precisions of the other methods and the results are depicted in Figure 7-3. It is worth mentioning that the results achieved at each iteration are averaged over all the categories. The following equation is used to compute the precision of each method:

(7-7) $$\text{Precision} = \frac{\text{The number of the relevant images in the first j positions of the retrieved images}}{j}$$



	0	1	2	3	4
CC	0.198	0.443	0.556	0.672	0.768
RR	0.187	0.389	0.495	0.541	0.562
RGG	0.185	0.378	0.496	0.522	0.545
proposed	0.194	0.431	0.528	0.689	0.793

Figure 7-3 The average precision graph depicted for the top 20 retrieved images.

The average time for the retrieval of images by means of each method is shown in Figure 7-4

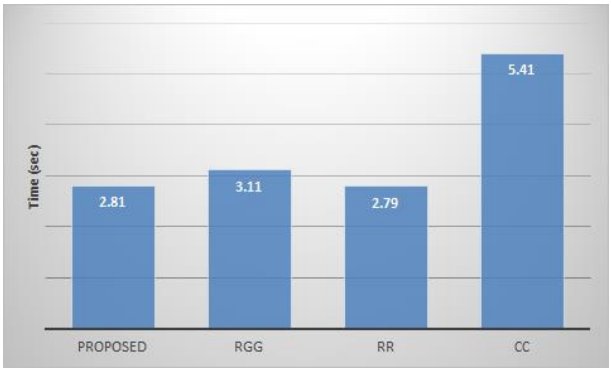


Figure 7-4 The average time taken for the retrieval of images by different methods.

It can be alluded from the above results that the proposed method not only produces more accurate results, but also achieves the results in a reasonable time.

Furthermore, the precisions of each method for some randomly selected semantic groups are shown in Table 7-5. The best result is shown by bold face for each case.

Table 7-5 Comparison of the precision of the proposed method with other methods for the top 20 retrieved images.

Semantic group	CC	RR	RGG	proposed
Sports	0.8185	0.5643	0.5912	**0.9261**
Buildings	0.7203	0.6485	0.7943	**0.8527**
Animals	**0.9918**	0.9561	0.8924	0.9840
Weather Forecast	0.5470	0.7511	**0.7519**	0.7455
Accidents	0.4149	0.2206	0.2418	**0.5613**
Cars	0.8651	0.8769	0.8133	**0.8774**

According to Table the proposed method outperforms the other well-known methods in most cases.

7.3 Utilization of the Image retrieval (IR) Methods

Besides the advantages in using the proposed methods solely in the image retrieval domain, they can be utilized to improve the proposed method illustrated in the previous chapter. In this section the application of these methods in our novel method introduced and discussed in the previous chapter is presented. Indeed, we avoid repeating the materials and just focus on the experiments carried out with and without the methods proposed in this chapter. The settings and the results of the experiments are presented after the illustration of the mechanism required to adopt each of the information retrieval methods so as to be used in the proposed recommender system.

There are some commonalities between the mechanisms required to apply the image retrieval methods presented in this chapter into our recommender system. In fact the general framework to make this work is the same for both methods, however, due to the process of relevance feedback there are some steps that are specific to the second method.

In the previous chapter we used the equation (6-1) in order to measure the similarity of the items, which was calculated in order to find the most similar items to some specific items which we call the proposal set $R = \{i_1,...,i_N\}$. In order to use the proposed image retrieval methods, this scenario should be slightly changed. Indeed, the items having most similarity to the items of the proposal set regarding just the graphical features are retrieved using the IR method. In this step, the items are manipulated as they are just images and the text-based features are simply neglected. It should be mentioned that for each item i_j , *n* items are retrieved. From the items retrieved according to the item i_j , *m* items having the maximum similarity regarding just the text-based features are selected. In this approach the parameters *n* and *m* play almost the same role as the parameters α and β which mentioned in the equation (6-1).

In order to apply the second method to our recommender system, the items should be categorized into some semantic groups. To avoid the unintended bias and achieve the most realistic results, the users that participate in the process of providing the feedbacks should be distinct from the users of the recommender system.

Some adjustments carried out to make an equal condition for all the experiments are as listed below:

- All the implementations are carried out using the Matlab software running on a PC with an Intel CORE 2 Due 2.20 GHz processor and 4GBs of RAM.
- We first create a grayscale equivalent for each image and normalize it so as to have the dimensions of 256×256.
- None of the users participated in the RF process of the second proposed IR method have taken part in any process of the proposed recommender system.
- A database containing 2300 houses from some local real estate agents and online real estate websites were used for the evaluations. In order to make sure the ratings reflect the view points of the users more realistically, the people participated in the process of rating the houses were real customers.
- The database of users contains 178 users each of which have rated at least 8 items and at most 58 items. The average number of ratings provided by each user is almost 21.
- As we have not yet proposed the parallel version of the methods discussed in this chapter, despite the fact that some parts of each method can be simply performed in a parallel fashion, no parallelization is carried out on any of the methods which are mentioned in this section.
- There are 5 semantic groups considered for the items which correspond to the different kinds of the properties existed in the data set of items.

In order to measure the effectiveness of the proposed methods presented in this chapter, the quality of the recommendations is measured in terms of *NRMSE* metric introduced in the previous chapter which was defined in equation (6-6).

Figure 7-5 NRMSE metric measured for different approaches.

In Figure 7-5, M1 and M2 denote utilization respectively the first IR method and the second IR method, while ch6 denotes the approach discussed in chapter 6. According to the results it is clear that using either of the methods illustrated in this chapter has a positive effect on the overall quality of the recommender system.

Chapter 8

Conclusion and Future Works

8.1 Summary

An everlasting goal of the computerized systems has been providing users with a more desirable experience using the system. This goal has driven many researches in a vast area of science. The user's desirability has shown itself in many different forms in various fields, but one thing that remains unchanged is itself as the ultimate goal. For instance, machine learning methods along with speech processing methods help the experts to design answering machines which give people the illusion that the machine can really understand them so they feel more comfortable using the system. This makes a cycle which causes to further researches and the results of those researches in turn cause further improvements and increase the users' expectations, as well.

Recommender systems are among the most valuable components that can improve the user experience with a more general system considerably when embedded inside that system. Since the mid-nineties and slightly after that, when the term "recommender system" coined by Resnick and Varian, researchers have conducted huge studies on this topic and the methods and approaches they have found are various and sundry.

A big part of data around us is in image format and people use these images in many of their decisions. The popularity of an item, in many cases, depends highly on its visual quality. For instance, the shape of a car has a significant influence on the attitude of potential customers toward it. Recommender systems try to provide people with recommendations resulted from an automatic process which is aimed at giving the users a better experience working with system, and perhaps improve the system owner's sales. As images are quite important in users' decisions, in this research we have proposed a distributed method to take images into account when trying to give the user a recommendation, which despite its apparent advantages has not found a fair amount of attention so far. In order to improve the quality of the

results even further, two methods for image retrieval are proposed which help the recommender system to achieve items with more relevant images to what the active user expects.

8.2 Suggestion and future works

Our main focus in this study was on suggesting a new practical approach to improve the performance of recommender systems that work on the items containing images that describe them or have a considerable impact on the users' attitude towards the items. We believe that this study can be a considered as the starting point a significant variety of researches in the future, from which we mention some:

- Trying to improve the quality of the system using similarity measures that take the shapes into account;
- Extending the proposed method to the case that items have more than one descriptive image attached to them;
- Generating a data set which contains millions of items with images associated to them, along with the user's rankings and testing the proposed method using this data set to achieve an even better evaluation.
- Generating a data set containing items which belong to different semantic groups.
- Trying to combine the proposed CBIR approach with supervised learning methods. Also, trying to use and evaluate other distance measures to compute the distance between the feature vectors.
- Trying to propose the parallel version of the methods discussed in chapter 7.

References

[1] Asanovic, K., Bodik, R., Catanzaro, B. C., Gebis, J. J., Husbands, P., Keutzer, K., Patterson, D. A., Plishker, W. L., Shalf, J., Williams, S. W., and Yelick, K. A. *The landscape of parallel computing research: A view from Berkeley*. Technical Report UCB/EECS-2006-183, Electrical Engineering and Computer Sciences, University of California at Berkeley, December 2006.

[2] Barney, B. *Introduction to parallel computing*, Nov 2007.

[3] NVIDIA. *CUDA community show case*, May 2008.

[4] Zomaya, A. Y. *Parallel Computing for Bioinformatics and Computational Biology*. Wiley Series on Parallel and Distributed Computing, Wiley-Interscience, NY, USA, 2005.

[5] Barnes, D. G., and Fluke, C. J. *Incorporating interactive three-dimensional graphics in astronomy research papers*. J. New Astronomy, Vol 13(8), pp. 599-605, 2008.

[6] Fraedrich, R., Schneider, J., and Westermann, R. *Exploring the millennium run-scalable rendering of large-scale cosmological datasets*. IEEE Transactions on Visualization and Computer Graphics, Vol. 15, pp. 1251-1258, 2009.

[7] Springel, V., White, S., Jenkins, A., Frenk, C., Yoshida, N., Gao, L., Navarro, J., Thacker, R., Croton, D., Helly, J., Peacock, J., Cole, S., Thomas, P., Couchman, H., Evrard, A., Colberg, J., and Pearce, F. *Simulations of the formation, evolution and clustering of galaxies and quasars*. Nature, Vol. 435, pp. 629-636, Jun 2005.

[8] Cherls, J., and Janin, J. Protein. *Docking algorithms: Simulating molecular recognition*. Current Opinion in Structural Biology, Vol 3(2), pp. 265-269, 1993.

[9] Waterman, M., Arratia, R., and Galas, D. *Pattern recognition in several sequences: Consensus and alignment*. Bulletin of Mathematical Biology, Vol 46(4), pp. 515-527, 1984.

[10] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition, 2001.

[11] https://software.intel.com/sites/products/documentation/doclib/iss/2013/compiler/cpp-lin/GUID-83D8631F-8ECA-498A-A091-CF952FE77A24.htm

[12] J. L. Gustafson. *Fixed time, tiered memory, and super linear speedup*. The 5th Distributed Memory Computing Conference (DMCC5), pp. 1255-1260, 1990.

[13] J. R. Smith. *The design and analysis of parallel algorithms*. Oxford University Press, Inc., New York, NY, USA, 1993.

[14] V. Faber, O. M. Lubeck, and A. B. White, Jr. *Super linear speedup of an efficient sequential algorithm is not possible*. J. Parallel Comput., Vol 3(3), pp. 259-260, July 1986.

[15] D. Parkinson. *Parallel efficiency can be greater than unity*. J. Parallel Computing, Vol 3(3), pp. 261-262, 1986.

[16] G. M. Amdahl. *Validity of the single processor approach to achieving large scale computing capabilities*. In Proceedings of the April 18-20, 1967, spring joint computer conference, AFIPS '67 (Spring), pp. 483-485, New York, NY, USA, 1967.

[17] G.M. Amdahl, *Validity of the Single Processor Approach to Achieving Large-Scale Computing Capabilities*, Federation of Information Processing Soc. Spring Joint Computer Conf. (AFIPS 07), AFIPS Press, pp. 483-485, 1967.

[18] J. L. Gustafson, *Reevaluating Amdahl's law*, Communications of the ACM, Vol. 31(5), pp. 532-533, May 1988.

[19] X.H. Sun, J.L. Gustafson. *Toward a better parallel performance metric, Parallel Computing*. Vol. 17(10-11), pp. 1093-109, Dec. 1991.

[20] Nvidia-Corporation. Nvidia CUDA C Programming Guide, 2012.

[21] M. J. Flynn, *Some Computer Organizations and Their Effectiveness*. IEEE Transactions on Computer, Vol. 21, pp. 948-960, 1972.

[22] I. Prabhudev, A. B. Gawali, S. A. Betkar, *Architecture of parallel processing in computer organization*, American Journal of Computer Science and Engineering, Vol. 1(2), pp. 12-17, 2014.

[23] C. A. Navarro, N. Hitschfeld-Kahler, L. Mateu, *A Survey on Parallel Computing and its Applications in Data-Parallel Problems Using GPU Architectures*, Vol. 15(2), pp. 285-329, 2014.

[24] P. Carlin, M. Chandy, and C. Kesselman. *The compositional C++ language definition*. Technical report, 1993.

[25] Buck, I., Foley, T., Horn, D., Sugerman, J., Fatahalian, K., Houston, M., and Hanrahan, P. *Brook for GPUs: Stream computing on graphics hardware*. SIGGRAPH '04: ACM SIGGRAPH 2004 Papers, ACM, pp. 777-786, New York, NY, USA, 2004.

[26] Chatterjee, S., Blelloch, G. E., and Zagha, M. *Scan primitives for vector computer*s. The 1990 ACM/IEEE Conference on Supercomputing (Los Alamitos, CA, USA), Supercomputing '90, IEEE Computer Society Press, pp. 666-675, 1990.

[27] Harris, M. *Mapping computational concepts to GPUs*. In ACM SIGGRAPH 2005 Courses (New York, NY, USA), SIGGRAPH '05, ACM., 2005

[28] Geist, A., and Lucas, *R. Major computer science challenges at Exascale*. Int. J. High Perform. Comput. Appl. Vol. 23, pp. 427-436, November 2009.

[29] Dzatko, D., and Shanley, T. *AGP System Architecture*. 2nd ed. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[30] Nickolls, J., Buck, I., Garland, M., and Skadron, K. *Scalable parallel programming with CUDA*. J. Queue - GPU Computing, Vol. 6(2), pp. 40-53, March 2008.

[31] Stone, J. E., Gohara, D., and Shi, G. Opencl: *A parallel programming standard for heterogeneous computing systems*. Computing in Science and Engineering, Vol. 12, pp. 66-73, 2010.

[32] J.-G. Lee, E. Jung, and W. Shin. *An asymptotic performance/energy analysis and optimization of multi-core architectures*. 10th International Conference, ICDCN 09, pp. 85-90, India, 2009.

[33] R.N. Bracewell. *The Fourier Transform and its Applications*. McGraw-Hill, 3rd edition, 2000.

[34] http://pippin.gimp.org/image_processing/chap_point.html

[35] A. Bovik, *Handbook of Image and Video Processing*. New Yor Academic, 2000

[36] A. Huertas, C. Lin, and R. Nevatia. *Detection of buildings from molecular views of aerial scenes using perceptual grouping and shadows*. Image Understanding Workshop, pp. 253-260, 1993.

[37] H. Lu and J. Aggarwal. *Applying perceptual organization to the detection of man-made objects in non-urban scenes*. J. Pattern Recognition, pp. 835-853, 1992.

[38] G.N.S. Prasanna and B.R. Musicus. *Generalized multiprocessor scheduling and applications to matrix computation*s. IEEE Transactions on Parallel and Distributed Systems, Vol. 7(6), pp. 650-664, June 1996.

[39] D. G. Lowe, *Perceptual Organization and Visual Recognition*, Kluwer Academics, Boston, 1985.

[40] A. P. Witkin and J. M. Tenenbaum, *On the Role of Structure in Vision*. In Human and Machine Vision, Jacob Beck and Barbara Hope and Azriel Rosenfeld (eds), pp. 481-543, Academic Press, New York, 1983.

[41] David Marr, W.H. Freeman and Company. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. NY, pp. 29-61, 1982.

[42] K. Koffka, *Principles of Gestalt Psychology*, Harcourt Brace, New York, 1935.

[43] M. Wertheimer. *Laws of Organization in Perceptual Forms*. in A Source Book of Gestalt Psychology, W. D. Ellis (ed), pp. 71-88, Harcourt Brace, 1938.

[44] W. Lim. *Fast algorithms for labeling connected components in 2-D arrays.* Tech. Rep. No. 86.22, Thinking Machine Corp., Cambridge, Mass., 1986.

[45] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, 2nd edition*. Cambridge, MA, USA: MIT Press, 2001, Chapter 21: Data structures for Disjoint Sets, pp. 498–524.

[46] A. Rosenfeld. *Connectivity in digital pictures*. Journal of the ACM, Vol. 17(1), pp. 146-160, 1970.

[47] http://homepages.inf.ed.ac.uk/rbf/HIPR2/label.htm

[48] E. Davies. *Machine Vision: Theory, Algorithms and Practicalities*. Academic Press, 1990, Chap. 6.

[49] Y. S. Fong, C. A. Pomalaza-Räez, and X. H. Wang. *Comparison study of nonlinear filters in image processing application*s. Opt Eng. (Bellinghatn), Vol. 28(7), pp. 749-760, 1989.

[50] Schafer, B., KonsTan, J., and Reidel, J. *Recommender systems in e-commerce*. The 1st ACM Conference on Electronic Commerce. ACM Press, Denver, CO, USA, pp. 158-166, 1999.

[51] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. 2000a. *Analysis of recommendation algorithms for e-commerce*. The 2nd ACM Conference on Electronic Commerce. ACM Press, Minneapolis, MN, USA, pp. 158-167.

[52] Schafer, B., Frankowski, D., Herlocker, J., Sen, S. *Collaborative Filtering Recommender Systems*. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.): The Adaptive Web: Methods and Strategies of Web Personalization. Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag, Berlin, Heidelberg, New York, 2007

[53] M. Bilgic and R. J. Mooney. *Explaining recommendations: Satisfaction vs. promotion*. In Beyond Personalization Workshop, IUI, 2005.

[54] A. Gunawardana, G. Shani. *A survey of accuracy evaluation metrics of recommender tasks*. Journal of Machine Learning Research, Vol. 10, pp. 2935-2962, 2009.

[55] Goldberg, D., Nichols, D., Oki, B., and Terry, D. *Using collaborative filtering to weave an information tapestry*. Communications of the ACM, Vol 35(12), pp. 61-70, 1992.

[56] Baudisch, P. *Dynamic information filtering*. Ph.D. thesis, Technische Universität Darmstadt, Darmstadt, Germany, 2001.

[57] X. Yang, Y. Guo and Y. Liu. *Bayesian-inference based recommendation in online social networks*. The 30th Annual IEEE International Conference on Computer Communications (INFOCOM 2011), pp. 551-555, 2011.

[58] Ghani, R. and Fano, A. *Building recommender systems using a knowledge base of product semantics*. The Workshop on Recommendation and Personalization in E-Commerce (RPEC). Springer-Verlag, Malaga, Spain, 2002.

[59] J. Wang, A. P. de Vries, and M. J. T. Reinders. *Unifying user-based and item-based collaborative filtering approaches by similarity fusion*. SIGIR '06 Proceedings of the 29th annual international Conference on Research and development in information retrieval, pp. 501-508, USA, 2006.

[60] Sollenborn, M. and Funk, P. *Category-based filtering and user stereotype cases to reduce the latency problem in recommender systems*. The 6th European Conference on Case-based Reasoning. LNCS, Vol. 2416, Springer-Verlag, Aberdeen, UK, pp. 395-405, 2002.

[61] Pazzani, M. *A framework for collaborative, content-based and demographic filtering*. Artificial Intelligence Review, Vol 13(5-6), pp. 393-408, 1999.

[62] Alspector, J., Kolcz, A., and Karunanithi, N. *Comparing feature-based and clique-based user models for movie selection*. 3rd ACM Conference on Digital Libraries. ACM Press, Pittsburgh, PE, USA, pp. 11-18, 1998.

[63] Mukherjee, R., Dutta, P., and Sen, S. *MOVIES2GO: A new approach to online movie recommendation*. The IJCAI Workshop on Intelligent Techniques for Web Personalization. Seattle, WA, USA, 2001.

[64] Ferman, M., Errico, J., van Beek, P., and Sezan, I. *Content-based filtering and personalization using structured metadata*. The 2nd ACM/IEEE-CS Joint Conference on Digital Libraries. ACM Press, Portland, OR, USA, pp. 393-393, 2002.

[65] M. Balabanovic and Y. Shoham, Fab*: Content-Based Collaborative Recommendation*. Comm. ACM, Vol. 40(3), pp. 66-72, 1997.

[66] Baeza-Yates, R. and Ribeiro-Neto, B. *Modern Information Retrieval*. Addison Wesley, Reading, MA, USA, 1999.

[67] Balabanovic, M. and Shoham, Y. *Fab: Content-based, collaborative recommendation*. Communications of the ACM, Vol. 40(3), pp. 66-72, 1997.

[68] Resnick, P., Iacovou, N., Suchak, M., Bergstorm, P., and Riedl, J. *GroupLens: An open architecture for collaborative filtering of net news.* The ACM 1994

Conference on Computer-Supported Cooperative Work. ACM, Chapel Hill, NC, USA, pp. 175-186, 1994.

[69] Jawaheer, G. et al., *Comparison of implicit and explicit feedback from an online music recommendation service*. The 1st international workshop on information heterogeneity and fusion in recommender systems, Barcelona, Spain, 2010.

[70] Linden, G., Smith, B., and York, J. *Amazon.com recommendations: Item-to-item collaborative filtering*. IEEE Internet Computing, Vol 7(1), January 2003.

[71] Ali, K. and van Stam, W. *TiVo: Making show recommendations using a distributed collaborative filtering architecture*. 04 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM Press, Seattle, WA, USA, pp. 394-401, 2004.

[72] D. Lemire, H. Boley. *RACOFI: a rule -applying collaborative filtering system*, in: A. Ghorbani, S. Marsh (Eds.) International Workshop on Collaboration Agents: Autonomous Agents for Collaborative Environments, NRC, Halifax, Nova Scotia, Canada, 2003.

[73] Breese, J., Heckerman, D., and Kadie, C. *Empirical analysis of predictive algorithms for collaborative filtering*. The 14th annual Conference on Uncertainty in Artificial Intelligence. Morgan Kaufmann, Madison, WI, USA, pp. 43-52, 1998.

[74] Lang, K. *NewsWeeder: Learning to filter netnews*. In Proceedings of the 12th International Conference on Machine Learning. Morgan Kaufmann, San Mateo, CA, USA, 331–339, 1995.

[75] Lam, W., Mukhopadhyay, S., Mostafa, J., and Palakal, M. *Detection of shifts in user interests for personalized information filterin*g. The 19th ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, Z¨urich, Switzerland, pp. 317–325, 1996.

[76] Miyahara, K. and Pazzani, M. *Collaborative filtering with the simple Bayesian classifie*r. The 6th Pacific Rim International Conference on Artificial Intelligence. Melbourne, Australia, 679–689, 2000

[77] Aggarwal, C., Wolf, J., Wu, K.-L., and Yu, P. *Horting hatches an egg: A new graph-theoretic approach to collaborative filtering*. The 5th ACM SIGKDD International

Conference on Knowledge Discovery and Data Mining. ACM Press, pp. 201-212 San Diego, CA, USA, 1999.

[78] A. García-Crespo, J. Chamizo, I. Rivera, M. Mencke, R. Colomo-Palacios, J. M. Gómez-Berbís. *SPETA: social pervasive e-tourism advisor*, J. Telematics and Informatics, Vol. 26, pp. 306-315, 2009.

[79] Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L., and Riedl, J. *GroupLens: Applying collaborative filtering to Usenet news*. Communications of the ACM, Vol. 40(3), pp. 77-87, 1997.

[80] Shardanand, U. and Maes, P. *Social information filtering: Algorithms for automating "word of mouth"*. The ACM CHI Conference on Human Factors in Computing Systems. ACM Press, Denver, CO, USA, pp. 210-217, 1995.

[81] Karypis, G. *Evaluation of item-based top-n recommendation algorithms*. The 10th ACM CIKM International Conference on Information and Knowledge Management. ACM Press, Atlanta, GA, USA, pp. 247-254, 2001.

[82] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. *Item-based collaborative filtering recommendation algorithms*. The Tenth International World Wide Web Conference. Hong Kong, China, 2001.

[83] Deshpande, M. and Karypis, G. *Item-based top-n recommendation algorithms*. ACM Transactions on Information Systems, Vol. 22(1), pp. 143-177, 2004.

[84] Xiaoyuan Su and Taghi M. Khoshgoftaar. *A survey of collaborative filtering techniques*. Advances in Artificial Intelligence, Vol. 2009, pp. 1-20, 2009.

[85] G. Adomavicius and A. Tuzhilin. *Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions*. IEEE transactions on knowledge and data engineering, Vol. 17(6), pp. 734-749, 2005.

[86] D.H. Park, H.K. Kim, I.Y. Choi, J.K. Kim. *A literature review and classification of recommender Systems research*. Expert Systems with Applications, Vol. 39 (11), pp. 10059-10072, 2012.

[87] W. Carrer-Neto, M.L. Hernández-Alcaraz, R. Valencia-García, F. García Sánchez. *Social knowledge-based recommender system, Application to the movies domain*. Expert Systems with Applications, Vol. 39(12), pp. 10990-11000, 2012.

[88] S.K. Lee, Y.H. Cho, S.H. Kim. *Collaborative filtering with ordinal scale-based implicit ratings for mobile music recommendations*. Information Sciences, Vol. 180(11), pp. 2142-2155, 2010.

[89] Z. Yu, X. Zhou, Y. Hao, J. Gu. *TV program recommendation for multiple viewers based on user profile merging*. User Modeling and User-Adapted Interaction, Vol. 16(1), pp. 63-82, 2006.

[90] E.R. Núñez-Valdéz, J.M. Cueva-Lovelle, O. Sanjuán-Martínez, V. García-Díaz, P. Ordoñez, C.E. Montenegro-Marín. *Implicit feedback techniques on recommender systems applied to electronic books*. Computers in Human Behavior, Vol. 28(4), pp. 1186-1193, 2012

[91] J. Serrano-Guerrero, E. Herrera-Viedma, J.A. Olivas, A. Cerezo, F.P. Romero. *A google wave-based fuzzy recommender system to disseminate information in University Digital Libraries 2.0*. Information Sciences, Vol. 181(9). Pp. 1503-1516, 2011.

[92] O. Zaiane. *Building a recommender agent for e-learning systems*. The International Conference on Computers Education (ICCE'02), Vol. 1, pp. 55-59, 2002.

[93] Z. Huang, D. Zeng, H. Chen. *A comparison of collaborative filtering recommendation algorithms for e-commerce*. IEEE Intelligent Systems, Vol. 22(5), pp. 68-78, 2007.

[94] V. E. Ogle and M. Stonebraker. *Chabot: Retrieval from Relational Database of Images*. IEEE Computer, Vol. 28(9), pp. 40-48, Sep1995.

[95] H. Lieberman, E. Rosenzweig, and P. Singh. *Aria: An Agent for Annotating and Retrieving Images*. IEEE Computer, Vol. 34(7), pp. 57-62, 2001.

[96] Richard da Silva Torres and A. X. Falcao. *Content Based Image Retrieval: Theory and Application*. RITA, Vol 8(2), pp. 168-189, 2006.

[97] P. Ciaccia, M. Patella, and P. *M-tree: An Efficient Access Method for Similarity Search in Metric Spaces*, The 23rd International Conference on Very Large Data Bases, pp. 426-435, Athens, Greece, 1997.

[98] C. Traina, B. Seeger, C. Faloutsos, and A. Traina. *Fast Indexing and Visualization of Metric Datasets Using Slim-Trees*. IEEE Transactions on Knowledge and Data Engineering, Vol. 14(2), pp. 244-260, 2002.

[99] J. M. Morel and G. Yu. *Asift: A new framework for fully affine invariant image comparison*. SIAM J. Imag. Sci., Vol. 2(2), pp. 438-469, Apr. 2009.

[100] G. Yu and J. Morel. *A fully affine invariant image comparison method*. IEEE ICASSP, pp. 1597-1600, 2009.

[101] Cheng-Hao Yao and Shu-Yuan Chen. *Retrieval of translated, rotated and scaled color textures*. Pattern Recognition, Vol. 36, pp. 913-929, 2003.

[102] Y. Yu, K. Huang, W. Chen, and T. Tan. *A novel algorithm for view and illumination invariant image matching*. IEEE Trans. Image Processing, Vol. 21, pp. 229-240, 2012.

[103] Ying Liu, Dengsheng Zhang, Guojun Lu. *Region based image retrieval with high-level semantics using decision tree learning*. Pattern Recognition, Vol. 41, pp. 2554-2570, 2008.

[104] Selvarajah S. and Kodituwakku S. R. *Analysis and Comparison of Texture Features for Content Based Image Retrieval*. International Journal of Latest Trends in Computing, Vol. 2(1), pp. 108-113, 2011.

[105] Kodituwakku S. R, Selvarajah S. *Comparison of Color Features for Image Retrieval*. Indian Journal of Computer Science and Engineering, Vol. 1(3), pp. 207-211, 2010.

[106] Rahman M. M., Bhattacharya M. P. and Desai B. C. *A framework for medical image retrieval using machine learning and statistical similarity matching techniques with relevance feedback*. IEEE Trans. Inform. Technol. Biomed., Vol. 11(1), pp. 58-69, 2007

[107] Md. Mahmudur Rahman A, Prabir Bhattacharya B, Bipin C. Desai. *A unified image retrieval framework on local visual and semantic concept-based feature spaces*. Journal of Visual Communication and Image Representation, Vol. 20(7), pp. 450-462, 2009.

[108] Hatice Cinar Akakin and Metin N. Gurcan. *Content-Based Microscopic Image Retrieval System for Multi-Image queries*. IEEE Transactions on Information Technology in Biomedicine, Vol. 16(4), pp. 758-769, 2012.

[109] Theo Gevers and Harro Stokman. *Robust Histogram Construction from Color Invariants for Object Recognition*. IEEE Transactions on Pattern Analysis And Machine Intelligence, Vol. 26(1), pp. 113-118, 2004.

[110] J. Domke, Y. Aloimonos. *Deformation and viewpoint invariant color histograms*. Proc. BMVC (British Machine Vision Conference), Edinburgh, UK, September 2006.

[111] G. Pass, R. Zabih, and J. Miller. *Comparing images using color coherence vectors*. ACM Conf. on Multimedia, pp. 65-73, 1996.

[112] Prabir Bhattacharya, Md. Mahmudur Rahman, Bipin C. Desai. *Image Representation and Retrieval Using Support Vector Machine and Fuzzy C-means Clustering Based Semantical Spaces*. The 18th International Conference on Pattern Recognition (ICPR'06), pp. 1162-1168, 2006.

[113] Jeffrey S. Beis and David G. Lowe. *Shape indexing using approximate nearest neighbor search in high–dimensional spaces*. Computer vision and pattern recognition conference, pp. 1000-1006, Puerto Rico, June 1997.

[114] J. Dean and S. Ghemawat. *Usenix*. *MapReduce: Simplified data processing on large clusters*. The 6th Symposium on Operating Systems Design and Implementation (OSDI 04), Vol. 6, pp. 137-179, San Francisco, CA, 2004.

[115] A. Nakhmani, A. Tannenbaum. *A New Distance Measure Based on Generalized Image Normalized Cross-Correlation for Robust Video Tracking and Image Recognition*. Pattern recognition letters, Vol. 34(3), pp. 315-321, 2013.

[116] http://lucene.apache.org/hadoop

[117] Eiben AE, Smith JE. *Introduction to evolutionary computing*. Springer, Berlin, 2003.

[118] H. Muller, W. Muller, D.M. Squire, S. M. Mailent and T. Pun. *Performance evaluation in content-based image retrieval: overview and proposals*, Pattern Recognition Letters, Vol. 22, pp.593-601, 2001.

[119] Park DK, Jeon YS, .Won CS. *Efficient use of local edge histogram descriptor*. The 2000 ACM workshops on Multimedia, pp. 51-54, 2000.

[120] B. S. Manjunath et al., *Introduction to MPEG-7*, Wiley, New York, 2002.

[121] Manjunath BS, Ma WY. *Texture feature for browsing and retrieval of image data*. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 18(8), pp. 837-842, 1996.

[122] A. AleAhmad, H. Amiri, E. Darrudi, M. Rahgozar and F. Oroumchian. *Hamshahri: A Standard Persian Text Collection*. Knowledge-Based Systems, Vol. 22(5) pp. 382-387, 2009.

[123] Rui, Y., Huang, S., Ortega, M., Mehrotra, S. *Relevance feedback: a power tool for interactive content-based image retrieval*. IEEE Transactions on Circuits and Video Technology, Vol. 8(5), pp. 25-36, 1998.

[124] Guldogan, E. & Gabbouj, M. *Dynamic feature weights with relevance feedback in content-based image retrieval*. 24th International Symposium on Computer and Information Science, pp. 56-59, 2009.

[125] J.R. Smith and S. F. Chang. *VisualSeek: A fully automated content-based image query system*. MULTIMEDIA '96 the 4th ACM international conference on Multimedia, pp. 87-98, 1996.

[126] M. Azodinia, A. Hajdu. *A recommender system that deals with items having an image as well as quantitative features*. The 9th IEEE international symposium on intelligent signal processing (WISP 2015), Vol. 1, pp. 1-6, 2015.

[127] M. Azodinia, A. Hajdu. *A method for image retrieval using combination of color and frequency layers*. International journal of computer applications (IJCR), Vol. 118(3), pp. 10-13, 2015.

Appendix A

Kd-Tree

Kd-Tree is a promising data-structure for finding nearest-neighbor queries for image descriptors is theKD-Tree [113]. Indeed, Kd-Tree is a form of balanced binary search tree. Typically, Kd-Tree stores elements that have many features and are represented by high-dimensional vectors**.** The height of a Kd-Tree corresponding to a data set with N elements is 2^N.

As implied by its name, Kd-Tree is a tree-shaped structure, where at each node is split based on the median in the dimension with the greatest variance in the data set.

At the root of the tree, the data is split into two halves by a hyperplane orthogonal to a chosen dimension at a threshold value. Each of the two halves of the data is then recursively split in the same way to create a fully balanced binary tree.

Regularly, each leaf node of the Kd-Tree corresponds to a single data point, however, there are implementation where the leaf nodes may contain more than one data point.

Given a query image, a descent down the Kd-Tree requires $\log 2^N$ comparisons and leads to a single leaf node. By comparing the query vector with the partitioning value, it is easy to determine to which branch of the tree the query vector belongs.

When the search reaches to the leaf nodes, the data point associated with this first leaf node is the first candidate for the nearest neighbor.

Appendix B

MapReduce

MapReduce is a programming model and an associated implementation introduced by Google to support distributed processing of large data sets on large clusters of computers. This model should be fed with a set of input key/value pairs in order to produce a set of output key/value pairs. The user of this model has two main tasks, which are providing the following procedures, along with the configuration of the underlying machines:

1) Map: takes an input pair and produces a set of intermediate key/value pairs. The library collects together all intermediate pairs with the same key I and feeds these to the Reduce function.
2) Reduce: takes an intermediate key I and a set of values associated with it and "merges" these values together and creates zero or more output pairs.