University of Rajshahi **Rajshahi-6205** **Bangladesh.**

RUCL Institutional Repository **http://rulrepository.ru.ac.bd**

Department of Computer Science and Engineering PhD Thesis

2006

Fingerprint Identification System Using Artificial Neural Computing Models

Hossain, A.K.M. Akhtar

University of Rajshahi

http://rulrepository.ru.ac.bd/handle/123456789/546

Copyright to the University of Rajshahi. All rights reserved. Downloaded from RUCL Institutional Repository.

Fingerprint Identification System Using Artificial Neural Computing Models

A

Dissertation Submitted

in the Department of

Computer Science and Engineering of

University of Rajshahi

for the Degree of

Doctor of Philosophy

By

A.K.M. AKHTAR HOSSAIN

Pattern Recognition Lab
Department of Computer Science
and Engineering
University of Rajshahi
Rajshahi-6205
Bangladesh

JANUARY 2006

Declaration

I do, hereby, declare that the research work incorporated in this **Ph.D. Thesis** entitled ***Fingerprint Identification System Using Artificial Neural Computing Models*** has been done by me and the work described is entirely my own unless otherwise explicitly stated in the text. It is prepared for the degree of *Doctor of Philosophy*, in the Pattern Recognition Lab, Department of Computer Science and Engineering, Faculty of Science, University of Rajshahi, Rajshahi-6205, Bangladesh. No part of this research work has been submitted in any university or institute for any degree or award earlier.

Hossain 29.01.2006

(A.K.M. Akhtar Hossain)
Ph.D. Researcher & Assistant Professor
Department of Computer Science and Engineering
University of Rajshahi,
Rajshahi-6205,
Bangladesh.

S.K. Ahmed Kamal, M.Sc.(Engg.)
Ph.D. (Knowledge Engineering)
Professor,
Department of Applied Physics & Electronics,
University of Rajshahi,
Rajshahi-6205,
BANGLADESH.

E-mail: kamal**52**@yahoo.com
kamal**52**@gmail.com

Tel: (+88)0171376045

Date: 29.01.06

Certificate from the Supervisor

I do, hereby, certify that the Ph.D. thesis entitled ***Fingerprint Identification System Using Artificial Neural Computing Models*** has been composed by **Mr. A.K.M. Akhtar Hossain** under my supervision. It is based on the unique research work, which has been done by the author himself. To the best of my knowledge, this thesis or any part of this research is not submitted for any degree or award elsewhere.

(Dr. S.K. Ahmed Kamal)
Professor,
Department of Applied Physics and Electronics,
University of Rajshahi, Rajshahi-6205
Bangladesh.

Dedicated to the memory of

My respected & beloved

Mother Late Akhtarun Nesa.

Abstract

This thesis presents the idea of features extraction technique of an off-line fingerprint. Traditionally, minutiae, core, delta, crossover, bifurcation, ridge ending, pore, island, enclosure, bridge, dot features of a fingerprints are considered to identify a person. These features are available when the fingerprint images are good. The great demerit of those existing techniques is that, they fail to extract the features of poor fingerprint images. If the fingerprint image is blurred then it is difficult to extract those features from the fingerprint. In this research work is fully surveyed all the available methods and different problems related with poor fingerprint identification. Taking into consideration the merits and demerits of the available methods, a new method for feature extraction technique for the poor fingerprint images is proposed. This research is specially dealing with the problem of poorly input fingerprint identification. For these poor types of fingerprint images, features can be extracted by using proposed grid mapping feature extraction techniques. In this process, the fingerprint is processed through Grid Mapping Features (*GMF*) extraction with fixed square/rectangle cells, such as 16X16 square/rectangle areas. If one small square/rectangle cell of the grid map contains more than 45~55% black pixels, then its digital value is considered as 1 otherwise 0. These digital values are applied to the input of the neural network for training purpose using *Minimum Distance Error Rate BackPropagation (MDER-BP) algorithm*, which is proposed in this research work.

During the training period, the values of the nodes are updated and stored in a relational knowledge base. The matching part of the system identifies the Fingerprint of a person with the help of the previous experiential values, which were stored in the relational knowledge-base of the system.

The proposed *Grid Mapping Feature-based* fingerprint matching system gives an *acceptable accuracy* in off-line identification system for *poor fingerprint images*. It is also shown that the proposed *MDER-BP Algorithm* also possesses capability for training and matching the *poor fingerprint* images. Several factors are responsible for getting correct result through neural computing techniques. The convergence of the solution depends heavily on initialization with random numbers and accuracy of the

results depends on (i) spread factors (ii) learning rates, (iii) iterations and (iv) hidden units. Finally, it has been concluded that for recognition of fingerprint using the *MDER-BP Algorithm* the system shows better efficiency with respect to Adaptive Resonance Theory-2 for the poor fingerprint images.

Acknowledgements

Regarding the outcome of this **Ph.D. thesis**, I would like to express my deepest sense of gratitude and respect to my honorable teacher and Ph.D. supervisor, **Dr. S. K. Ahmed Kamal**, **M.Sc**.(Communication Engineering) Odessa Engineering University, **Ph.D.**(Knowledge Engineering) Donetsk National University, Ukraine, **Professor**, Department of Applied Physics and Electronics, University of Rajshahi, Bangladesh, for his guidance, advice, valuable suggestions regarding this research work, encouragement and every possible help throughout the work and preparation of this Ph.D. thesis.

I am also very much grateful to my honorable teacher **Professor Dr. A. R. Sarkar**, Chairman, Department of Computer Science and Engineering, University of Rajshahi, Bangladesh, for his moral support and useful official cooperation.

I am grateful to all of my honorable colleagues of the Department of Computer Science and Engineering, University of Rajshahi, Bangladesh, for their valuable suggestions and very much useful cooperation.

Finally, I am also very much grateful to my family members for their constant encouragement, cooperation and inspirations during the research work.

January, 2006.

The Author

Department of Computer Science
and Engineering,
University of Rajshahi,
Rajshahi-6205,
Bangladesh.

CONTENTS

CHAPTER FIVE

NEURAL NETWORK MODELS FOR FINGERPRINT IDENTIFICATION

CHAPTER SIX

FINGERPRINT VERIFICATION USING UNSUPERVISED NEURAL NETWORK

List of Figures

List of Tables

List of Publications Based on the Ph.D. Research Work

1. **Title:** "Fingerprint Verification System Using Minimum Distance Error Rate BackPropagation Algorithm", Scientific Journal founded in 1997, Bulletin of Donetsk National University, 004.932.721, ISSN 1817-2237, Ukraine, PP-439-444, **1/2005.**

2. **Title:** "Grid Mapping Features based Fingerprint Verification System", Proceedings of 8th International Conference on Computer and Information Technology, Islamic University of Technology (IUT), Gazipur, Dhaka, Bangladesh, PP-1052-1057, ISBN 984-32-2873-1, **ICCIT-2005.**

3. **Title:** "An Approach To Extract Fingerprint Feature Using Grid-Mapping Technique And To Match Through BackPropagation Neural Network", Scientific Journal founded in 1997, Bulletin of Donetsk National University, 004.932.721, ISSN 1817-2237, PP-321-328, Ukraine, **2/2005**.

4. **Title:** "Minutiae Features based Fingerprint Verification System using BackPropagation Neural Network", Rajshahi University Studies, Journal of Science, Part-B, Bangladesh, ISSN: 1681-0708, **2005**.

5. **Title:** "ART-2 Based Fingerprint Verification System", Proceedings of 7th International Conference on Computer and Information Technology, BRAC University, Dhaka, Bangladesh. PP-838-842, ISBN: 984-32-1836-1, **ICCIT-2004**.

6. **Title:** "Off-line Fingerprint Verification Using Supervised Neural Network", Proceedings of 6th International Conference on Computer and Information Technology, Jahangirnagar University, Dhaka-1342, Bangladesh, PP-586-589. ISBN: 984-584-005-1, **ICCIT-2003.**

Abbreviations used in the Thesis

2-D	Two Dimension
AAD	Average Absolute Deviation
Acc	Classification Accuracy
AFIS	Automated Fingerprint Identification Systems
ANN	Artificial Neural Network
ART	Adaptive Resonance Theory
ART-1	Adaptive Resonance Theory-1
ART-2	Adaptive Resonance Theory-2
ART-3	Adaptive Resonance Theory-3
AS	Automatic Setting
BP	BackPropagation
BPNN	BackPropagation Neural Network
C	Class
CCD	Charged Coupled Device
CPN	Counter Propagation Network
DB	Database
DNA	Deoxyribonucleic Acid
EER	Equal-Error-Rate
E-Field	Electric Field
F.B.I.	Federal Bureau of Investigation
GMF	Grid-Mapping Feature
IAFIS	Integrated Automated Fingerprint Identification Service
ID	Identification
KKT	Karush-Kuhn_Tucker
LEDs	Light-Emitting Diodes
LTM	Long-Term Memory
MDER-BP	Minimum Distance Error Rate BackPropagation
MLP	Multilayer Perceptron
NEC	National Electronics Corporation
PEs	Processing Elements
PIN	Personal Identification Number
RGB	Red Green Blue
RR	Reject Rate
SNN	Supervised Neural Network
STM	Short-Term Memory
SVM	Support Vector Machine
UNN	Unsupervised Neural Network

Chapter One
INTRODUCTION

Contents **Page No.**

1.1 Introduction to Fingerprint

Among all the biometric techniques, fingerprint-based identification is the oldest method, which has been successfully used in numerous applications. The uniqueness and permanence of the fingerprints are very well known. Archaeological artifacts prove that fingerprints were already used by the ancient Assyrians and Chinese as a form of identification of a person. The first scientific studies on fingerprint date from the late sixteen century, but the fundamentals of modern fingerprint identification methods were provided at the end of nineteenth century. The studies of Sir F. Galton and E. Henry led to formally accept fingerprints as valid signs of identity by law enforcement agencies.

The first Automated Fingerprint Identification Systems (AFIS) were developed in the 1950s by the F.B.I. (Federal Bureau of Investigation) in cooperation with the National Bureau of Standards, the Cornell Aeronautical Laboratory, Rockwell International Corp. Ten years later other AFISs were developed by NEC Technologies Inc. (Tokyo), Print rack Inc. (Anaheim, California), and Morpho System (Paris).

Fingerprint recognition is now-a-days the basic task of the Integrated Automated Fingerprint Identification Service (IAFIS) of the most famous police agencies. Ten-print

based identification and latent fingerprint recognition are the two main concerns of an IAFIS. In the former, the system should identify a person by the whole sequence of his/her ten-fingerprints, in the latter it has to identify a person through a latent fingerprint found on a *crime scene*. Technology advances in the 1980s in the areas of personal computing and optical scanners triggered non-forensic applications of fingerprint recognition methods (http://www.biometrika.it/eng/wp_fingintro.html).

The enormous interest roused by electronic commerce on Internet and, more in general, by the need of reliable techniques for authenticating the identity of a living person in a broad range of applications has greatly intensified the research efforts towards the development of low cost small-size fingerprint-based biometric systems.

With the advent of electronic banking, e-commerce, and smartcards and an increased emphasis on the privacy and security of information stored in various databases, automatic personal identification system has become a very important topic. The need to ensure that only the right people have authorization to high security accesses has led to the development of systems for automatic personal verification. There are many commercial systems designed for person identification worldwide. The most popular systems are based on fingerprint, ID card and signature recognition techniques.

Fingerprint is the rigid and furrow patterns on the tip of a finger (**Fig-1.1**). It is a distinctive feature and remains invariant over a person's lifetime, excepts for cuts and bruises. The two fundamental premises [1, 10] on which fingerprint identification is based are: (i) fingerprint details are permanent and (ii) fingerprints of an individual are unique.

Fingerprint authentication requires acquiring and digitizing a fingerprint image. The digital image of the fingerprint includes several unique features in terms of rigid bifurcation and rigid ending, collectively referred to as minutiae.

According to the method of acquisition of fingerprint data, there are two types of fingerprint verification system (i) *On-line* and (ii) *Off-line*.

Fig.1.1: A Sample Fingerprint image

The fundamental difference between the on-line and off-line fingerprint verification system is the nature of the fingerprint samples used in each system. In the on-line fingerprint verification system, data about the rigid bifurcation and rigid ending of fingerprint and pressure of the fingertip are collected as a function of time by using on-line scanner. The fingerprint samples are analyzed in a dynamic environment. This allows on-line systems to collect real-time information.

In the off-line fingerprint verification system, data about the rigid bifurcation, rigid ending and ***Grid-Mapping Feature (GMF)*** of fingerprints are collected from the static scanned image of the fingerprint. The system analyzes a digital image using either the raw pixel data or some sort of representation of the pixel data.

In this work, we focus on the implementation of off-line fingerprint verification system by using supervised neural network. The power of neural networks lies in their ability to recognize general patterns. This is particularly useful because there are general characteristics present in each fingerprint.

1.2 Biometrics

Biometrics, which refers to identify an individual based on his/her physical or behavioral characteristics, has the capability to reliably distinguish between an authorized person and an imposter. The biometrics characteristics are basically of two types (i)

Physiological Characteristics and (ii) Behavioral Characteristics (**Fig.1.2**). Since biometric characteristics are distinctive, can not be forgotten or lost, and the person to be authenticated needs to be physically present at the point of identification, biometrics is inherently more reliable and more capable than traditional knowledge-based and token-based techniques (http://www.biometrika.it/eng/wp_biointro.html).

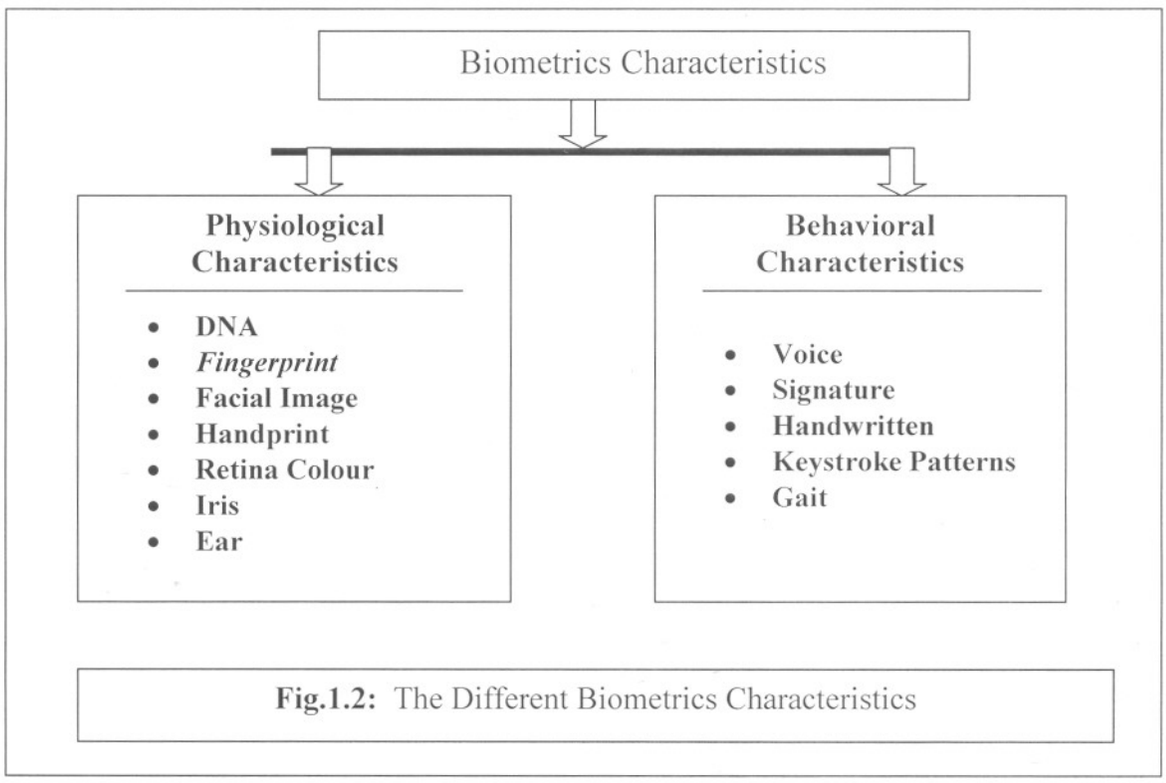


Fig.1.2: The Different Biometrics Characteristics

However, among all biometrics (e.g., fingerprint, facial image, handprint, retina, iris, signature, voice, etc.), fingerprint-based identification is one of the most mature and proven techniques.

Why we choose fingerprint to identify a person?

- *Fingerprint is unique.*
- *Fingerprint is not changed during his or her life time.*
- *It is only changed by cutting or burning.*
- *But other biometrics feature, such as face, is changed with the increase of age. Sometimes Retina's colour is also changed due to the same reason.*

- *The DNA test is also unique. But it is costly and time consuming.*
- *On the other hand fingerprint identification system is less costly and faster.*

1.3 Biometrics System

A biometric system is essentially a pattern recognition system that recognizes a person by determining the authenticity of a specific physiological and/or behavioral characteristic possessed by that person [1]. An important issue in designing a practical biometric system is to determine how an individual is recognized. Depending on the application context, a biometric system may be called either a verification system or an identification system:

- A verification system authenticates a person's identity by comparing the captured biometric characteristic with her own biometric template(s) pre-stored in the system. It conducts one-to-one comparison to determine whether the identity claimed by the individual is true. A verification system either rejects or accepts the submitted claim of identity.
- An identification system recognizes an individual by searching the entire template database for a match. It conducts one-to-many comparisons to establish the identity of the individual. In an identification system, the system establishes a subject's identity (or fails if the subject is not enrolled in the system database) without the subject having to claim an identity.

The term authentication is also frequently used in the biometric field, sometimes as a synonym for verification; actually, in the information technology language, authenticating a user means to let the system know the users identity regardless of the mode (verification or identification). Throughout this work we use the generic term recognition where we are not interested in distinguishing between verification and identification.

The block diagrams of a verification system and an identification system are depicted in **Fig.1.3**; user enrollment, which is common to both tasks is also graphically illustrated. The enrollment module is responsible for registering individuals in the biometric system

database (system DB). During the enrollment phase, the biometric characteristic of an individual is first scanned by a biometric reader to produce a raw digital representation of the characteristic. A quality check is generally performed to ensure that the acquired sample can be reliably processed by successive stages. In order to facilitate matching, the raw digital representation is usually further processed by a feature extractor to generate a compact but expressive representation, called a template. Depending on the application, the template may be stored in the central database of the biometric system or be recorded on a magnetic card or smartcard issued to the individual. The verification task is responsible for verifying individuals at the point of access. During the operation phase, the user's name or PIN (Personal Identification Number) is entered through a keyboard (or a keypad); the biometric reader captures the characteristic of the individual to be recognized and converts it to a digital format, which is further processed by the feature extractor to produce a compact digital representation.

The resulting representation is fed to the feature matcher, which compares it against the template of a single user (retrieved from the system DB based on the user's PIN). In the identification task, no PIN is provided and the system compares the representation of the input biometric against the templates of all the users in the system database; the output is either the identity of an enrolled user or an alert message such as "user not identified." Because identification in large databases is computationally expensive, classification and indexing techniques are often deployed to limit the number of templates that have to be matched against the input.

Depending on the application domain, a biometric system could operate either as an on-line system or an off-line system. An on-line system requires the recognition to be performed quickly and an immediate response is imposed (e.g., a computer network logon application). On the other hand, an off-line system usually does not require the recognition to be performed immediately and a relatively long response delay is allowed (e.g., an employee background check application).

Typically, on-line systems are fully automatic and require that the biometric characteristic be captured using a live-scan scanner, the enrollment process be unattended, there be no (manual) quality control, and the matching and decision be fully automatic. Off-line systems, however, are typically semi-automatic, where the biometric acquisition could be through an off-line scanner (e.g., scanning a fingerprint image from

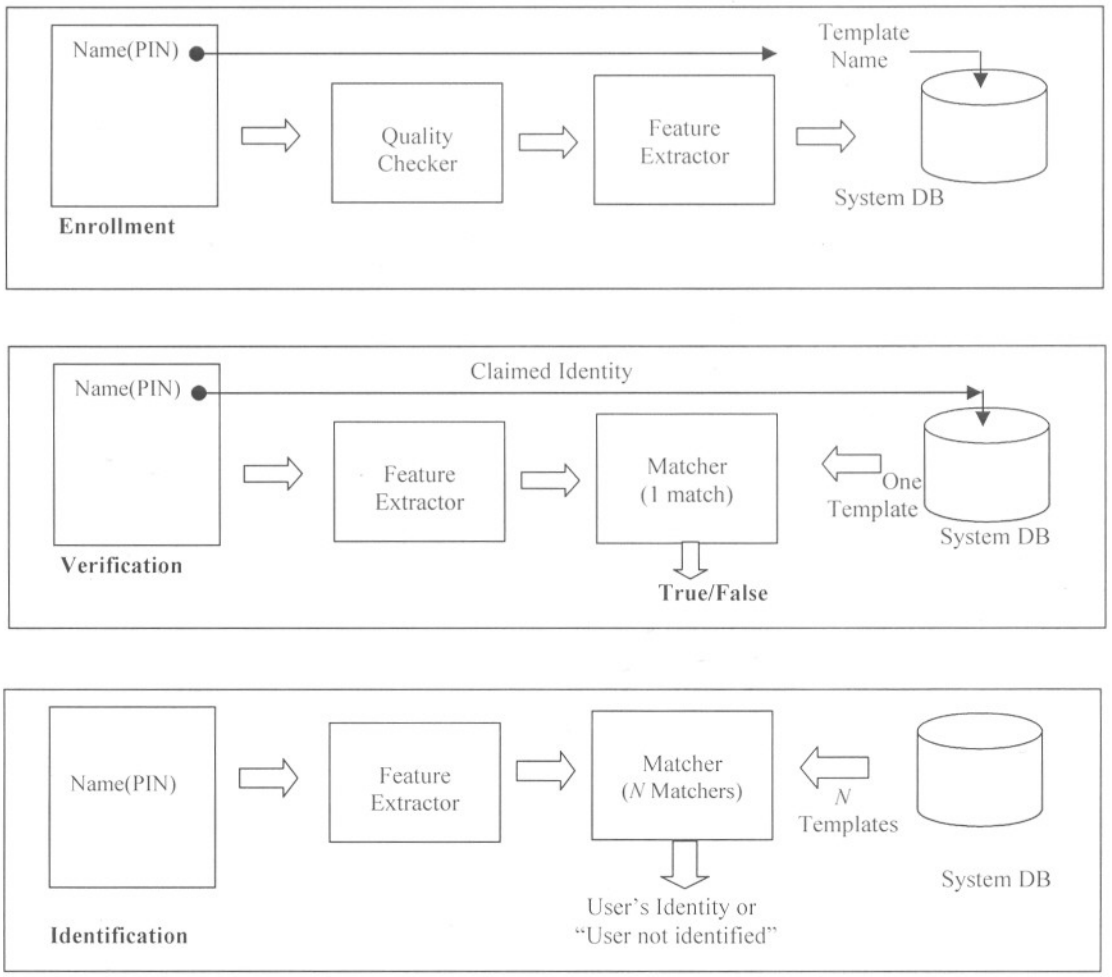


Fig.1.3: Block Diagrams of Enrollment, Verification, and Identification System of Fingerprints

a latent or inked fingerprint card), the enrollment may be supervised (e.g., when a criminal is "booked," a forensic expert or a police officer may guide the fingerprint acquisition process), a manual quality check may be performed to ensure good quality

acquisition, and the matcher may return a list of candidates which are then manually examined by a forensic expert to arrive at a final (human) decision.

1.4 Fingerprint Formation

Fingerprint formation is like the growth of capillaries and blood vessels in angiogenesis. The pattern is not strictly determined by the genetic code but by small variables in growth factor concentrations and hormones within the tissues. There are so many variables during fingerprint formation that it would be impossible for two to be alike. However, it is not totally random, perhaps having more in common with a chaotic system than a random system (http://www.newscientist.com/lastword/article.jsp?id=lw474).

Fingerprints are fully formed at about seven months of fetus development and finger ridge configurations do not change throughout the life of an individual except due to accidents such as bruises and cuts on the finger tips. This property makes fingerprints a very attractive biometric identifier.

1.5 Fingerprint Representation

The popular fingerprint representation schemes have evolved from an intuitive system developed by forensic experts who visually match the fingerprints. These schemes are either based on predominantly local landmarks (e.g., minutiae-based fingerprint matching systems [2, 4] or exclusively global information (fingerprint classification based on the Henry system [3, 5, 6]). The minutiae-based automatic identification techniques first locate the minutiae points and then match their relative placement in a given finger and the stored template [2]. A good quality inked fingerprint image contains between 60 to 80 minutiae, but different fingerprints and different acquisitions of the same finger have different numbers of minutiae. A graph-based representation [7, 8, 9] constructs a nearest neighbour graph from the minutiae patterns. The matching algorithm is based on inexact graph matching techniques. The point pattern-based representation [2, 11] considers the minutiae points as a two-dimensional pattern of points. Correlation-based techniques [12, 13] consider the gray level information in the fingerprint as features and match the global patterns of ridges and valleys to determine if the ridges align.

The global representation of fingerprints (e.g., whorl, left loop, right loop, arc, and tented arch) is typically used for indexing [3, 5, 6, 10] and does not offer good individual discrimination. Further, the indexing efficacy of existing global representations is poor due to a small number of categories (typically five) that can be effectively identified automatically and highly skewed distribution of the population in each category. The global representation schemes of the fingerprint used for classification can be broadly categorized into four main categories: (i) Knowledge-based, (ii) Structure-based, (iii) Frequency-based, and (iv) Syntactic. The knowledge-based fingerprint representation technique uses the locations of singular points (core and delta) to classify a fingerprint into five major classes (whorl, left loop, right loop, arch, and tented arch) [3, 6]. A knowledge-based approach tries to capture the knowledge of a human expert by deriving rules for each category by hand-constructing the models. Structure-based approach uses the estimated orientation field in a fingerprint image [14, 15]. Frequency-based approaches use the frequency spectrum of the fingerprints for representation [16]. Hybrid approaches combine two or more approaches for representation [17, 18].

There are two major shortcomings of the traditional approaches to fingerprint representation. For a significant fraction of the population, the automatic extraction of representations based on an explicit detection of complete ridge structures in the fingerprint is difficult.

The widely used minutiae-based representation does not utilize a significant component of the rich discriminatory information available in the fingerprints. Local ridge structures cannot be completely characterized by minutiae. Further, minutiae-based matching has difficulty in efficiently and robustly matching two fingerprint images containing different numbers of unregistered minutiae points. Some applications such as smart cards will also benefit from a compact representation. *In this research work, we have proposed new technique to extract features of fingerprint images, called Neighborhood-Minutiae feature extraction for good fingerprint images. We have also proposed another feature extraction technique for very poor fingerprint images, named Grid-Mapping Feature (GMF) extraction (Detail in Chapter-2, section- 2.6).*

1.6 Fingerprint Classification

Large volumes of fingerprint are collected and stored everyday in a wide range of applications; including forensics, access control and driver license registration [19]. Automatic identity recognition based on fingerprints requires that the input fingerprint be matched with a large number of fingerprints stored in a database (the FBI database currently contains more than 630 million fingerprints [20]). To reduce the search time and computational complexity, it is desirable to classify these fingerprints in an accurate and consistent manner such that the input fingerprint needs to be matched only with a subset of the fingerprints in the database. Fingerprint classification is a technique used to assign a fingerprint into one of the several pre-specified types already established in the literature (and used in forensic applications) which can provide an indexing mechanism. Fingerprint classification can be viewed as a coarse level matching of the fingerprints.

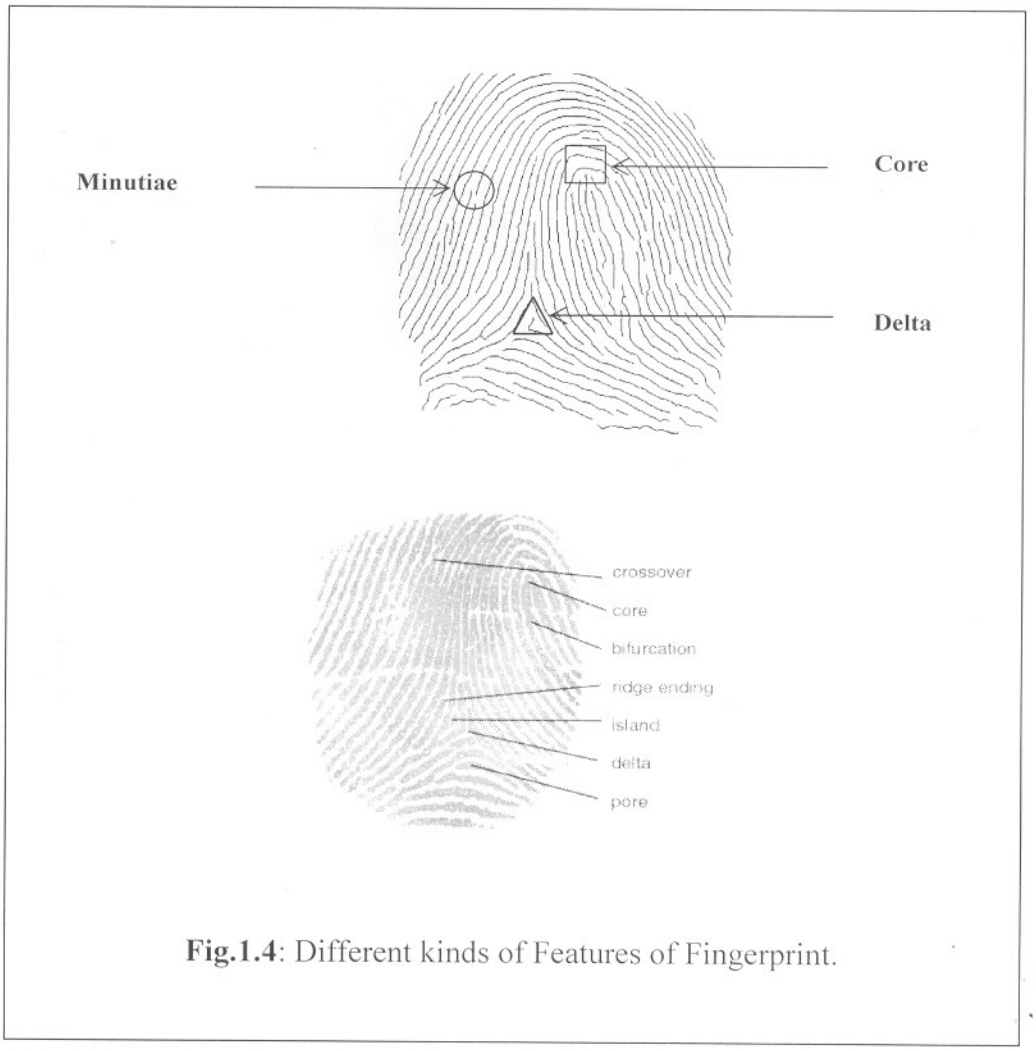


Fig.1.4: Different kinds of Features of Fingerprint.

An input fingerprint is first matched to one of the pre-specified types and then it is compared to a subset of the database corresponding to that fingerprint type. To increase the search efficiency, the fingerprint classification algorithm can classify a fingerprint into more than one class.

For example, if the fingerprint database is binned into five classes, and a fingerprint classifier outputs two classes (primary and secondary) with high accuracy, then the identification system will only need to search two of the five bins, thus decreasing the search space 2.5 folds. Continuous classification of fingerprints is also very attractive for indexing where fingerprints are not partitioned in non-overlapping classes, but each fingerprint is characterized with a numerical vector summarizing its main features. The continuous features obtained are used for indexing fingerprints though spatial data structures and for retrieving fingerprints by means of spatial queries [21].

There are two main types of features in a fingerprint: (i) global ridge and furrow structures which form special patterns in the central region of the fingerprint, and (ii) minutiae/local ridge and furrow minute details (**Fig.1.4**). A fingerprint is classified based on only the first type of features and is uniquely identified based on the second type of features (ridge endings and bifurcations, also known as minutiae). **Fig.1.4** shows ridges, minutiae, orientation field and singular points in a fingerprint image [10, 22]. *In this work, a new concept is developed to learn and classify the fingerprint with Minimum Distance Error Rate BackPropagation neural network* [53]. *This system is knowledge based, which will be more efficient techniques to identify off-line fingerprint.*

1.7 Fingerprint Verification System:

A biometrics system can be operated in two modes: (i) *verification mode* and (ii) *identification mode* [10]. In the verification mode, a biometric system either accepts or rejects a user's claimed identity while a biometric system operating in the identification mode establishes the identity of the user without a claimed identity. Fingerprint *identification mode* is a more difficult problem than fingerprint verification because a huge number of comparisons need to be performed in identification. In this research work, we have focused on a biometric system operating in a *verification mode* (**Fig.1.5**)

Rajshahi University Library
Documentation Section
Document No. D. 2513
Date 24/5/06

and an indexing scheme (fingerprint classification) that can be used in an identification system. A number of civilian applications operate in verification mode on a regular basis and perform identification only at the time of the user registration to check the integrity of the database (e.g., finding duplicates).

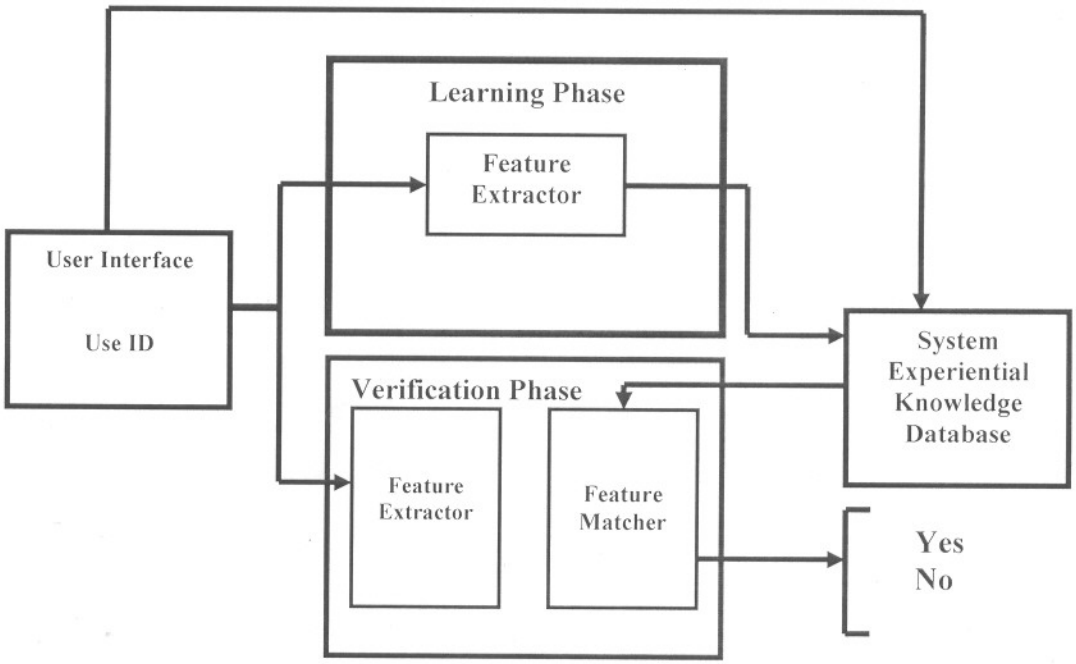


Fig.1.5: Block Diagram of Off-line Fingerprint Verification System

A typical verification system can be divided into two modules/phases: (i) *enrollment/ learning Phase* and (ii) *verification/testing phase*. The enrollment module scans the fingerprint of a person through a sensing device and then stores a representation (called template) of the fingerprint in the database. The verification module is invoked during the operation phase.

The same representation which was used in enrollment/learning phase is extracted from the input fingerprint and matched against the template of the claimed identity to give a "yes/no" answer. On the other hand, an identification system matches the input fingerprint with a large number of fingerprints in the database and as a result, fingerprint classification is effective only in an identification system and is not an issue in a

verification system. In this work, we have used the term "identification" in a loose sense for both the fingerprint verification and identification problems and the exact meaning of the term can be resolved based on the context.

1.8: History of Fingerprints

One of the reasons fingerprint identification is so promising is that, unlike iris scanning or other relatively untested biometrics, Europe, the U.S.A. and other countries have extensive real-world experience with fingerprint identification. We are all aware of the use of fingerprint identification in law enforcement. Less well known is the ancient history of fingerprint identification for commerce and the long history of the science of fingerprints in China, Europe and the U.S.A [1, 23].

- Pre-historic picture writing found in NOVA SCOTIA shows a hand with ridge patterns.
- In ancient China thumbprints were used on clay seals to prove identity in financial transactions.
- 1686 Marcello Malpighi, a professor of anatomy at the University of Bologna, wrote about ridges loops, and spirals in fingerprints.
- 1823 Professor Purkinji from the University of Breslau described nine basic fingerprint patterns. These pattern descriptors are still used today.
- 1823 Dr. Henry Faulds wrote an article describing fingerprints as a means of personal identification. He is credited with the first fingerprint identification in law enforcement by obtaining a conviction based on correctly identifying a greasy print left on an alcohol bottle.
- 1882 Gilbert Thompson of the U.S.A. Geological survey used his own fingerprint on a document to prevent forgery.
- 1892 Sir Francis Galton, a British anthropologist, published the first fingerprint classification system and established the individuality and permanence of fingerprints. The "minutia points" Galton Identified are still used today.

- 1901 Scotland Yard adopted the Galton-Henry fingerprint identification system, an adaptation of Galton's observations by Sir Edward Henry, chief commissioner of the London metropolitan police.
- 1903 The New York State prison system began the first systematic use of fingerprints in the U.S.A. for identifying known criminals.
- 1904 The U.S.A. Army first began fingerprints to identify enlisted personnel.
- 1904 Juan Vucetich of the Buenos Aires Police published his system of fingerprint identification, which helped him identify a murderer by studying fingerprints left on a door-post. His method is still used today.
- 1905-1930 Low enforcement agencies across the Europe and the U.S.A. turned to fingerprints for personal identification. Many began to send copies of their fingerprint cards to the National Bureau of Criminal Identification established by the International Association of Police Chiefs.
- 1919 The U.S.A. Congress established the Identification Division of the *Federal Bureau of Investigation* (F.B.I.). The National Bureau and Leavenworth consolidated their files to from the nucleus of the current F.B.I. fingerprint files.
- 1946-1971 The F.B.I. had processed 200 million fingerprint cards.
- 1971- 2004 Different kinds of method had been introduced to identify a person by his/her fingerprints by many scientists all over the world.
- 2001- (**Lee** and **Gaensslen**). Fingerprint recognition was formally accepted as a valid personal identification method and became a standard routine forensics. Fingerprint identification agencies were set up worldwide and criminal fingerprint databases were established.

1.9: Objectives of the research

A fingerprint identification system involves several stages. First, the fingerprint image needs to be acquired and scanned into a digital representation. There is a loss of information when the three-dimensional fingerprint is scanned into a two-dimensional digital image. Placement of the finger on ink-pad and to press onto the paper and to get a good image is very difficult. It has a challenge for the feature extraction algorithm to

reliably extract a robust representation from these images. Due to the noise present in the fingerprint image because of inexact sensing process, there may be false features detected or important features missed.

The matching algorithm should recover the invariant information from the features such that it outputs a high score when matching impressions of the same finger and a low score when matching the impressions of different fingers. If the fingerprint image is of poor quality, a fingerprint enhancement algorithm should be used to improve the quality of the image. However, it is very difficult to design a fingerprint enhancement algorithm that is robust to all types of noise in the sensed fingerprint.

For these poor types of fingerprint images, features can be extracted by using *grid mapping techniques (chapter-3).* In this system, the preprocessed fingerprint is grid mapping with fixed square/rectangle cells, such as 16x16 square/rectangle areas. If each small square/rectangle cell contains more than 45~55% (this percentage may be varied with respect to the grayscale of the fingerprint images) black pixels, then it is considering as digital value 1 otherwise 0. These digital values are applied to the input of the neural network for training purpose using *Minimum-Distance Error Rate BackPropagation algorithms* [53]. During the training period, the values of the nodes are updated and stored in a relational knowledge base. The verification part of the system identifies the Fingerprint of a person with the help of the previous experiential values, which was stored in the relational knowledge-base. *The main objectives on this research work are to investigate poor fingerprint images of criminals for their identification.*

We also proposed a new technique named the *nearest neighborhood minutiae features detection technique*, is mainly based on minutia points for the good fingerprint images. The reference minutia point is considered on the basis of its importance. The *proposed feature* extraction technique is used for where the high security is required. The technique is simple but more authentic. In *chapter-2*, the *nearest neighborhood minutiae features detection technique* is described in detail. In these figures four factors are considered. These are *linear distances between the selected minutiae, related radial angle, related*

position angle, the direction of the local ridge of the minutiae. In this work, we do not work with nearest neighborhood minutiae features detection technique because we work with only very poor fingerprint images.

1.10: Outlines of the Research Work

In this work, we have focused on an alternative method of extraction of feature of poor fingerprint images and also develop an algorithm called "Minimum Distance Error Rate BackPropagation [53]" neural network for training and matching the fingerprint images. In the following section is provided the chapter-based brief discussions on the basis of this research work.

Chapter-One (Introduction): It is described the Biometric System and its classification. The use of fingerprint to recognize a person on the basis of historical and modern scientific point of views, its formation, representation, classification and properties of fingerprint and a verification system are studied quite widely.

Chapter-Two (The Unique Characteristics of Fingerprints): In this chapter, the clear idea about different kinds of fingerprint features that are presently used is provided. The Minutiae-based and Global ridge-based features are described and it also described their classification widely. In this chapter is proposed a new technique for extraction feature called *GMF* for poor fingerprint images.

Chapter-Three (Grid Mapping Feature Extraction Technique): Feature extraction technique with proper tools to extract the feature for poor fingerprint image is studied.

Chapter-Four (Fingerprint Classification Techniques): In this chapter, it is focused on different types of pattern classifying techniques. It is tried to establish neural network as a fingerprint classifier which is proved to be better than the other classifying technique for poor fingerprint images.

Chapter-Five (Neural Network Models for Fingerprint Identification): Here it is described the basic concept of neural network, classification of neural network, and supervised neural network. Different kinds of supervised neural work have been pointed out and specially is given the idea on BackPropagation neural network. An algorithm named *Minimum Distance Error Rate BackPropagation* **(MDER-BP)** *Algorithm* [53] to classify (Patterns) fingerprints is proposed and developed.

Chapter-Six (Fingerprint Verification Using Unsupervised Neural Network): In this chapter, is given the idea about unsupervised neural network models like Adaptive Resonance Theory (ART). It is described ART-1 and ART-2 models widely. Finally ART-2 Network for fingerprint verification system is simulated with developed algorithm.

Chapter-Seven (Fingerprint Matching Algorithms): Here it is focused on the different kinds of matching algorithm that has been presently used for fingerprint matching. The practical use of the developed **MDER-BP** *Algorithm for matching* the poor fingerprint image is shown.

Chapter-Eight (Implementation of Fingerprint Identification System): It is described the proper idea about the practical implementation of proposed GMF extraction techniques. It is also described the simulation of GMF features with proper fingerprint training and matching (*MDER-BP)* algorithm. The implementation of fingerprint identification system has been described step by step. The screen shots are taken for every practical result for showing reliability of the research work. The excremental results have been shown at end of this chapter.

Chapter-Nine (Discussion & Conclusion): In this chapter, it is discussed specially on the experimental results on the basis of chapter-8 and also on the whole research work. It is described the overall merits and demerits of this work. It is also focused the difficulties that are faced during the research work. Finally, the work is concluded with acceptable

percentage of identification for poor fingerprints images. At last, the direction for the development of the present research is proposed for future researchers.

Chapter Two

THE UNIQUE CHARACTERISTICS OF FINGERPRINT

Contents **Page No.**

2.1 Introduction

Fingerprints are extremely complex. In order to read and classify them certain defining characteristics are considered, many of which have been established by law enforcement agencies as they have created and maintained larger and larger databases of fingerprints. There are two types of features of fingerprint presently used to identify a person. These are (i). Minutiae- based features or local features and (ii) Global ridge–based features.

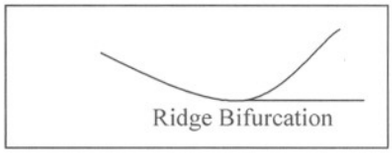
2.2 Minutiae-based Features or Local features

The Minutiae-based features or Local features are also known as Minutia points [1,10, 22]. They are the tiny, unique characteristics of fingerprint ridges that are used to identify a person. The fingerprint ridges are not continuous or straight ridges. Instead they are broken, forked, changed directionally, or interrupted. The points at which ridges end, fork, and changed are called minutia points, and these minutia points provide unique, identifying information.

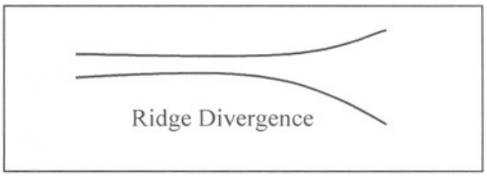
There are *five characteristics of minutia points* in fingerprints (**Fig.1.5**).

(1). Type- There are different types of minutia points. The most common are ridge endings and ridges bifurcations [22].

- Ridge Ending – occurs when a ridge ends abruptly.
- Ridge Bifurcation – the point at which a ridge divides into two or more branches.



- Ridge Divergence – the spreading apart of two lines which have been running parallel.



- Dot or Island – a ridge that is so short that it appears as a point.
- Enclosure – a ridge that divides into two and then re-unites to create an enclosed area of ridge-less skin.
- Short Ridge – an extremely short ridge, but not so short that it appears as a Dot or Island.

(2). Orientation – Each minutia point faces a particular direction. This is the orientation of the minutia point.

(3). Spatial frequency – Spatial frequency refers to how far apart the ridges are in the neighborhood of the minutia point.

(4). Curvature – The curvature refers to the rate of change of ridge orientation.

(5). Position – The position of the minutia point refers to its x, y location, either in an absolute sense or relative to fixed points like the Delta and Core points.

2.3 Global Ridge–based Features

Global ridge-based features are those characteristics which can be seen with the naked eye. Global features include:

2.3.1 Basic Ridge Patterns

2.3.2 Pattern Area

2.3.3 Core

2.3.4 Delta

2.3.5 Type Lines

2.3.6 Ridge Count

Fig.2.1: Global ridge-based features

2.3.1 Basic Ridge Patterns

Over the years those who work with fingerprints have defined groupings of prints based on patterns in the fingerprint ridges (**Fig.2.1**). This categorization makes it easier to search large databases of fingerprints and identify individuals. The basic ridge patterns

are not sufficient for identification but they help narrow the search. Certain product based identification on "optical correlation" of global ridge patterns, or matching one fingerprint pattern image with another. *DigitalPersona* believes that positive identification must be based on verification of minutia points in addition to global features (*www.digitalpersona.com*).

There are a number of basic ridge pattern groupings, in which three of the most common are *loop*, *arch* and *whorl*.

2.3.1.1 Loop

The loop is the most common type of fingerprint pattern and accounts for about 65% of all fingerprints. There are basic three types of loop. These are left loop, right loop and twin loop (**Fig.2.2**).

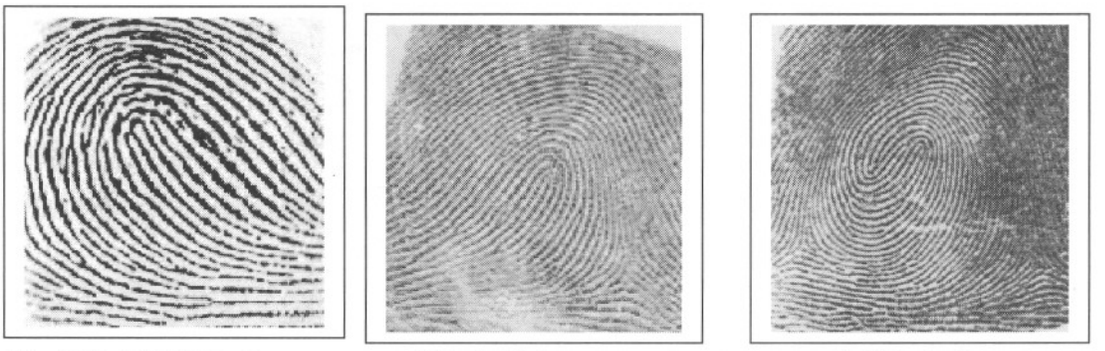


Fig.2.2: (a) Right Loop (b) Left Loop (c) Twin Loop

2.3.1.2 Arch

The Arch pattern is a more open curve than the loop. There are two types of arch patterns the plain arch and the tented arch (**Fig.2.3**).

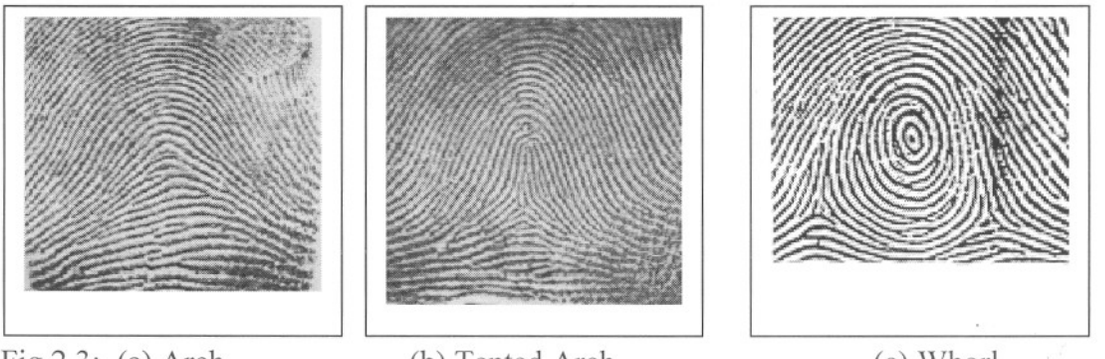


Fig.2.3: (a) Arch (b) Tented Arch (c) Whorl

2.3.1.3 Whorl

Whorl patterns occur in about 30% of all fingerprints and are defined by at least one ridge that makes a complete circle (Figure-2.3(c)).

2.3.1 Pattern Area

The pattern area is the part of the fingerprint that contains all the global features. Fingerprints can be read and classified based on the information in the pattern area (**Fig.2.4**). Certain minutia points that are used for final identification might be outside the pattern area.

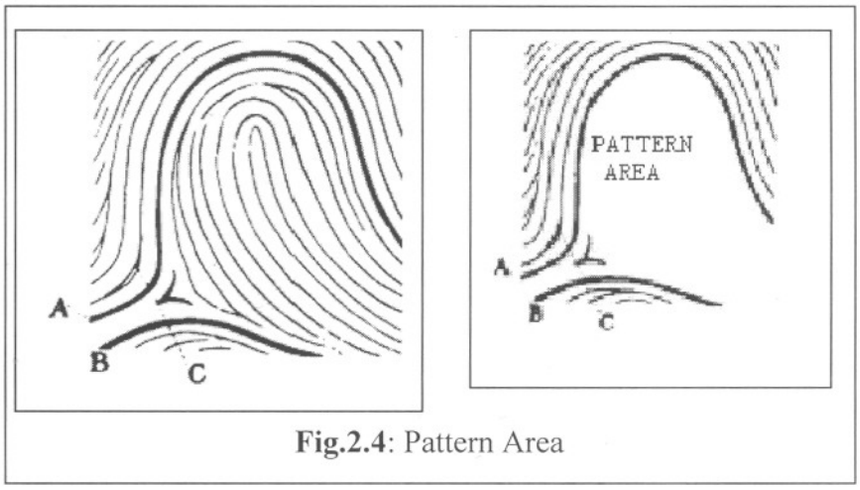


Fig.2.4: Pattern Area

2.3.2 Core

The core or core point, located at the approximate centre of the fingerprint impression, is used as a reference point for reading and classifying the fingerprint (**Fig.2.5**).

2.3.3 Delta

The delta is the point on the first bifurcation, abrupt ending ridge, meeting of two ridges, dot fragmentary ridge, or any point upon a ridge at or nearest the centre of divergence. It is a definite fixed point used to facilitate ridge counting and tracing (**Fig.2.5**).

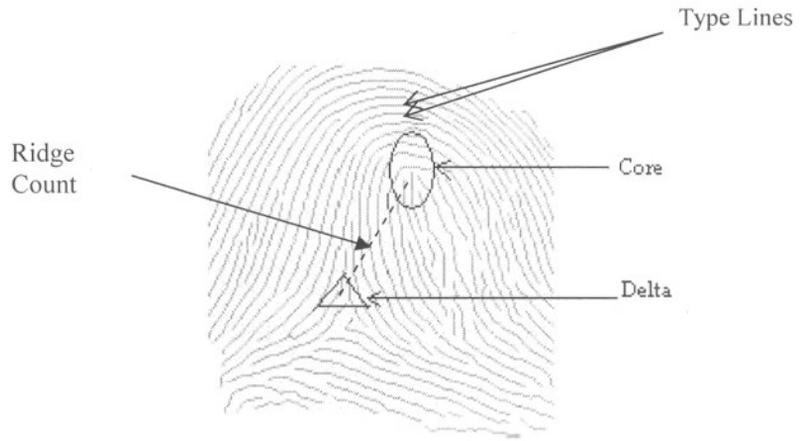


Fig.2.5: Core and Delta feature of a fingerprint Image

2.3.4 Type Lines

Type Lines are the two innermost ridges that start parallel, diverge and surround or tend to surround the pattern area. When there is a definite break in a type line, the ridge immediately outside that line is considered to be its continuation.

2.3.5 Ridge Count

The Ridge Count is most commonly the number of ridges between the Delta and the Core. To establish the ridge count, an imaginary line is drawn from the Delta to the Core and each ridge that touches this line is counted.

2.4 Previous History of Latest Research on Fingerprints

The early individuality studies typically focused on a predominantly minutiae-based representation; some studies explicitly factored in fingerprint class (e.g., right loop, left loop, whorl, arch, tented, etc.) information. The type, direction, and location of minutiae were the most commonly used features in the early individuality studies. See **Table 2.2** for a comparison of the features used in fingerprint individuality models. The types of minutiae used varies from one study to other: some studies used two minutia types (ending and bifurcation) whereas others used as many as 13 types of events (e.g., empty

cell, ridge ending, ridge fork, island, dot, broken ridge, bridge, spur, enclosure, delta, double fork, trifurcation, multiple events) [24]. The later models considered addition features to determine the probability of occurrence of a particular fingerprint configuration (e.g., ridge counts [25], sweat pores [26].

Most of the early individuality studies examined the distinctiveness of a portion/feature of the fingerprint. Under simplifying assumptions (e.g., implicit assumptions about statistical independence of events and the corresponding event distributions are identical), these studies estimated the distinctiveness of the entire fingerprint (total pattern variation) by collating the distinctiveness in the feature extracted from fingerprints (total feature variation).

The fingerprint individuality problem was first addressed by Galton in 1892 [27], who considered a square region spanning six-ridges in a given fingerprint. He assumed that, on an average, a fingerprint can be covered by 24 such six-ridge wide independent secure regions. Galton estimated that he could correctly reconstruct any of the regions with a probability of $\frac{1}{2}$, by looking at the surrounding ridges. Accordingly, the probability of a specific fingerprint configuration, given the surrounding ridges is $\left(\frac{1}{2}\right)^{24}$. He multiplied this conditional (on surrounding ridges) probability with the probability of finding the surrounding ridges to obtain the probability of occurrence of a fingerprint as

$$P(Fingerprint\,Configuration) = \frac{1}{16} \times \frac{1}{256} \times \left(\frac{1}{2}\right)^{24} = 1.45 \times 10^{-11} \qquad (2.1)$$

where $\frac{1}{16}$ is the probability of occurrence of a specific fingerprint type (such as arch, tented arch, left loop, right loop, double loop, whorl, etc.) and $\frac{1}{256}$ is the probability of occurrence of the correct number of ridges entering and exiting each of the 24 regions. **Eq.(2.1)** gives the probability that a particular fingerprint configuration in an average size fingerprint (containing 24 regions defined by Galton) will be observed in nature. Roxburgh [28], Pearson [30], and Kington [29] objected to Galton's assumption that the

probability of occurrence of any particular ridge configuration in a six-ridge square is $\frac{1}{2}$, and claimed that **Eq.(2.1)** grossly underestimated the fingerprint individuality (i.e., overestimated the probability of occurrence). Pearson [30] argued that there could be 36 (6×6) possible minutiae locations within one of Galton's six-square regions, leading to a probability of occurrence of a particular fingerprint configuration of

$$P(Fingerpr\,\text{int}\quad Configuration) = \frac{1}{16}\times\frac{1}{256}\times\left(\frac{1}{36}\right)^{24} = 1.09\times10^{-41} \qquad (2.2)$$

A number of subsequent models (Henry [31], Balthazard [32], Bose, Wentworth and Wilder [33], Cummins and Midlo [34], and Gupta [35]) are interrelated and are based on a fixed probability, p, for the occurrence of a minutiae. They computed the probability of a particular N-minutiae fingerprint configuration as

$$P(Fingerpr\,\text{int}\,Configuration) = p^N \qquad (2.3)$$

In the following, are provided the values of p used in various studies. In most cases, the authors do not present any details on how they arrived at their choices of p.

- Henry [31] chose $p = \frac{1}{4}$ and added 2 to the number of minutiae, N, if the fingerprint type and core-to-delta count could be determined from the given (latent) fingerprint.
- Balthazard [32] also set $p = \frac{1}{4}$, under the assumption that there are four types of equally likely minutiae events: (i) fork (bifurcation) to the right, (ii) fork to the left, (iii) ending to the right, and (iv) ending to the left.
- Bose adopted $p = \frac{1}{4}$, under the assumption that there are four possibilities in each square region of one ridge-interval width in a fingerprint: (i) a dot, (ii) a fork, (iii) an ending, and (iv) a continuous ridge.
- Wentworth and Wilder [33] chose $\frac{1}{50}$ as the value of p.

- Cummins and Midlo [34] adopted the same value of p as Wentworth and Wilder, but introduced a multiplicative constant of $\frac{1}{31}$ to account for the variation in fingerprint pattern type.
- Gupta [35] estimated the value of p as $\frac{1}{10}$ for forks and endings, and $\frac{1}{100}$ for the less commonly occurring minutiae types, based on 1,000 fingerprints. He also used a fingerprint-type-factor of $\frac{1}{10}$ and correspondence-in-ridge-count-factor of $\frac{1}{10}$.

Because of the widely varying values of p used in the above studies, the probability of a given fingerprint configuration also dramatically varies from one model to the other.

Roxburgh [28] proposed a more comprehensive analysis to compute the probability of a fingerprint configuration. His analysis was based on considering a fingerprint as a pattern with concentric circle, one ridge interval apart, in a polar coordinate system. Roxburge also incorporated a quality measure of the fingerprint into his calculations. He computed the probability of a particular configuration to be:

$$P(Fingerpr\,\text{int}\,Configuration) = \left(\frac{C}{P}\right)\left(\frac{Q}{RT}\right)^{N} \tag{2.4}$$

where P is the probability of encountering a particular fingerprint type and core type, Q is a measure of quality (Q=1.5 for an average quality print, Q=3.0 for a poor quality print), R is the number of semicircular ridges in a fingerprint (R=10), T is the corrected number of minutiae types (T=2.412), and C is the number of possible positions for the configuration (C=1). Amy [36] (Stoney [25]) considered the variability in minutiae type, number, and position in his model for computing the probability of a fingerprint configuration. He further recognized that K multiple comparisons of the fingerprint pair (e.g., each hypothesized orientation alignment, each reference point correspondence) increase the possibility of false association which is given by

$$P(\text{False Association}) = 1 - (1 - P(\text{Fingerprint Configuration}))^{K} \tag{2.5}$$

Kingston's model [29], which is very similar to Amy's model, computes the probability of a fingerprint configuration based on the probabilities of the observed number of minutiae, observed positions of minutiae, and observed minutiae types as follows:

$$\text{P(Fingerprint Cofiguration)} = \left(e^{-y}\right)\left(y^{N} / N!\right)\left(P_1\right)\prod_{i=2}^{N}\left(P_i\right)\frac{(0.082)}{\left[S-(i-1)(0.082)\right]} \qquad (2.6)$$

Where y is the expected number of minutiae in a region of given size S (in mm^2) and P_i is the probability of occurrence of a particular minutiae type.

Most of the models discussed above simply assume those fingerprints are being matched manually. The probability of observing a given fingerprint feature is estimated by manually extracting the features from a small number of fingerprint images. Champod and Margot [37] used an AFIS to extract minutiae from 977 fingerprint images scanned at a relatively high resolution of 800 *dpi*. They generated frequencies of minutiae occurrence and minutiae densities after manually verifying the thinned ridges produced by the AFIS to ensure that the feature extraction algorithm did not introduce errors. They considered minutiae only in concentric bands (five ridges wide) above the core and acknowledge that their individuality estimates were conservative (i.e., provided an upper bound). As an example [10], they estimated the probability of occurrence of a seven-minutiae configuration (five ending and two bifurcations) as 2.25×10^{-5}.

Table 2.1: Fingerprint features used in different models.

Author	Fingerprint features used
Galton (1892)	ridges, minutiae types
Pearson (1930)	ridges, minutiae types
Henry (1900)	minutiae locations, types, core-to-delta ridge count
Balthazard (1911)	minutiae locations, two types, and two directions
Bose (1917)	minutiae locations and three types
Wentworth & Wilder (1918)	minutiae locations
Cummins & Midlo (1943)	minutiae locations, types, core-to-delta ridge count

Author	Fingerprint features used
Gupta (1968)	minutiae locations and types, types, ridge count
Roxburgh (1933)	minutiae locations, two minutiae types, two orientations, fingerprint and core types, number of possible positioning, area, fingerprint quality
Amy (1948)	minutiae locations, number, types, and orientation
Trauring (1963)	minutiae locations, two types, and two orientations
Kingston (1964)	minutiae locations, number, and types
Osterburg et al. (1980)	minutiae locations and types
Stoney et al. (1986)	minutiae locations, distribution, orientation, and types, variation among prints from the same source, ridge counts, and number of alignments
S.Prabhakar (2001)	minutiae locations, core, orientation, rotation, template and variation among prints from the same source, number of alignments, matching with filterbank

Table: 2.2: Comparison of probability of a particular fingerprint configuration using different models. For a fair comparison we do not distinguish between minutiae types. By assuming that an average size fingerprint has 24 regions (R=24) as defined by Galton, 72 regions (M=72) as defined by Osterburg et al., and has 36 minutiae on an average (N=36), we compare the probability of observing a given fingerprint configuration in the third column of the table. The probability of observing a fingerprint configuration with N=12, and equivalently, R=8, is given in braces in the third column.

Author	P(Fingerprint Configuration)	N=36, R=24, M=72 (N=12, R=8, M=72
Galton (1892)	$\frac{1}{16}\times\frac{1}{256}\times\left(\frac{1}{2}\right)^{R}$	1.45×10^{-11} (9.54×10^{-7})

Author	P(Fingerprint Configuration)	N=36, R=24, M=72 (N=12, R=8, M=72
Pearson (1930)	$\frac{1}{16}\times\frac{1}{256}\times\left(\frac{1}{36}\right)^{R}$	1.09×10^{-41} (8.65×10^{-17})
Henry (1900)	$\left(\frac{1}{4}\right)^{N=2}$	1.32×10^{-23} (3.72×10^{-9})
Balthazard (1911)	$\left(\frac{1}{4}\right)^{N}$	2.12×10^{-22} (5.96×10^{-8})
Bose (1917)	$\left(\frac{1}{4}\right)^{N}$	2.12×10^{-22} (5.96×10^{-8})
Wentworth & Wilder (1918)	$\left(\frac{1}{50}\right)^{N}$	6.87×10^{62} (4.10×10^{-21})
Cummins & Midlo (1943)	$\frac{1}{31}\times\left(\frac{1}{50}\right)^{N}$	2.22×10^{63} (1.32×10^{-22})
Gupta (1968)	$\frac{1}{10}\times\frac{1}{10}\times\left(\frac{1}{10}\right)^{N}$	1.00×10^{-38} (1.00×10^{-14})
RoxbRcÿ (1933)	$\frac{1}{1000}\times\left(\frac{1.5}{10\times2.412}\right)^{N}$	3.75×10^{-47} (3.35×10^{-18})
Trauring (1963)	$(0.1944)^{N}$	2.47×10^{-26} (2.91×10^{-9})
Osterburg et al. (1980)	$(0.766)^{M-N}(0.234)^{N}$	1.33×10^{-27} (3.05×10^{-15})
Stoney (1985)	$\frac{N}{5}\times0.6\times(0.5\times10^{3})^{N-1}$	1.2×10^{-80} (3.5×10^{26})

Osterburg et al. [24] defined fingerprints into discrete cells of size 1 *mm*×1 *mm*. They computed the frequencies of 13 types of minutiae events (including an empty cell) from 39 fingerprints (8,591 cells) and estimated the probability that 12 ridge ending will match

between two fingerprints based on an average fingerprint area of 72 mm^2 as 1.25×10^{-20}. Sclove [38] modified Osterburg et al.'s model by incorporating the observed dependence of minutiae occurrence in cells and came up with an estimate of probability of fingerprint configuration that is slightly higher than that obtained by Osterburg et al. Stoney and Thornton [25] criticized Osterburg etal.'s and Sclove's models because these models did not consider the fingerprint ridge structure, distortions, and the uncertainty in the positioning of the grid. Stoney and Thornton [25] critically reviewed early fingerprint individuality models and proposed a detailed set of fingerprint features that should be taken into consideration. These features included ridge structure and description of minutiae location, ridge counts between pairs of minutiae, description of minutiae distribution, orientation of minutiae, variation in minutiae type, variation among fingerprints from the same source, number of positions (different translations and rotations of the input fingerprint to match with the template), and number of comparisons performed with other fingerprints for identification.

Stoney's [39] model is different from other models in that it attempts to characterize a significant component of pairwise minutiae dependence. Stoney [39] and Stoney and Thornton [25] studied probabilities of occurrences of various types of minutiae, their orientation, number of neighboring minutiae, and distances/ridge counts to the neighboring minutiae. Given a minutiae set, they calculated the probability of a minutiae configuration by conjoining the probabilities of the individual events in the configuration. For instance, they proposed a linear ordering of minutiae in minutia configuration and recursively estimated the probability of a n-minutiae configuration from the probability of a (n-1) – minutiae configuration and the occurrence of a new minutiae of certain type/orientation at a particular distance/ridge counts from its nearest minutiae within the (n-1) – minutiae configuration. The model also incorporated constraints due to connective ambiguity and due to minutia-free areas. The model corrected for the probability of false association by accounting for the various possible linear orderings which could initiate/drive the search for correspondence. A sample calculation for computing the probability of a false association using Stoney's model is given below.

$$P(\text{False Association}) = 1 - \left(1 - 0.6 * \left(0.5 \times 10^{-3}\right)^{(N-1)}\right)^{\left[\frac{N}{5}\right]} \approx \frac{N}{5} \times 0.6 * \left(0.5 \times 10^{-3}\right)^{(N-1)} \quad (2.7)$$

For the sake of simplicity, we have considered only a rudimentary version of Stoney's model for the above computation; it is arbitrarily assumed that the probability of a typical starting minutia is 0.6, a typical neighboring minutia places an additional constraint of 5×10^{-3} on the probability, and there are no constraints due to connective ambiguity, minutia-free areas or minutia-free borders are assumed. Finally, it is (arbitrarily) assumed that one in every five minutia can potentially serve as a starting point for a new search.

Stoney and Thornton identified weaknesses in their model and acknowledged that one of the most critical requirements, i.e., consideration of variation among prints from the same source, is not sufficiently addressed in their model. Their tolerances for minutiae position were derived from successive printings under ideal conditions and are far too low to be applicable in actual fingerprint comparisons.

The models discussed above (including Amy's model of false association due to multiple comparisons) concentrated mainly on measuring the amount of detail in a single fingerprint (i.e., estimation of the probability of a fingerprint configuration). These models did not emphasize the intra-class variations in multiple impressions of a finger.

Trauring [40] was the first to concentrate explicitly on measuring the amount of detail needed to establish correspondence between two prints from the same finger using an AFIS and observed that corresponding fingerprint features could be displaced from each other by as much as 1.5 times the inter-ridge distance. He further assumed that (i) minutiae are distributed randomly, (ii) there are only two types of minutiae (ending and bifurcation), (iii) the two types of minutiae are equally likely, (iv) the two possible orientations of minutiae are equally likely, and (v) minutiae type, orientation, and position are independent variables. Trauring computed the probability of a coincidental correspondence of N minutiae between two fingerprints to be:

$$P(\text{Fingerprint Correspondence}) = (0.1944)^N \quad (2.8)$$

Stoney and Thornton's [25] criticism of the Trauring model is that he didn't consider ridge count, connective ambiguity, and correlation among minutiae location. Further, they claim that Trauring's assumption that the minutiae types and orientations are equally probable is not correct. The probabilities of observing a particular minutiae configuration from different models are compared in **Table 2.2**.

There have been a few studies which empirically estimate the probability of finding a fingerprint in a large database that successfully matches the input fingerprint. Meagher et al. [41] (for more details see Stiles [42] matched about 50,000 rolled fingerprints belonging to the same fingerprint class (left loop) with each other to compute the impostor distribution. However, the genuine distribution was computed by matching each fingerprint image with itself; this ignores the variability present in different impressions of the same finger. Further, they assumed that the impostor and the genuine distributions follow a Gaussian distribution and computed the probability of a false association to be 10^{-97}. This model grossly underestimates the probability of a false association because it does not consider realistic intra-class variations in impressions of a finger (see also, Stoney et al. [25] and Wayman [43]).

2.5 Nearest Neighborhood Minutiae Features Extraction Technique

From the above discussion it has been clear that existing models for good fingerprint images are not sufficient. *We propose a new technique to extract feature from good fingerprint image, named nearest neighborhood minutiae features. The nearest neighborhood minutiae features detection technique is mainly based on minutia points.* The reference minutia point is considered on the basis of its importance. This method applied for good fingerprint images. The proposed feature extraction technique is used for where the high security is required. The technique is simple but more authentic. The following two figures show how the features are considered. In these following two figures two factors are considered. One is the angular distance between nearest minutiae

points and the other is the linear distance between centre minutia points (which is mentioned by the user) with the nearest minutiae points.

Minutiae matching are certainly most well-known and widely used method for fingerprint matching. The minutia position such as (x, y) coordinates local ridge direction are available for each minutia. These minutiae features could be used directly to match the fingerprint. In this research, we proposed a new technique, which is based on linear distance, angular distance and related position angle between nearest neighborhood minutiae points.

Let us consider T and P be the representation of the template and input fingerprint respectively. Unlike in correlation-based techniques, where the fingerprint represent coincides with the fingerprint image, here the representation is a feature vector (of variable length) [10] whose elements are the fingerprint minutiae. Most common minutiae matching algorithms consider each minutia as a triplet $M = \{x, y, \theta\}$ that indicates the x, y minutia coordinates and the minutia angle θ;

$$T = \{M_1, M_2, M_3, .., M_k\} \tag{2.9}$$

$$M_i = \{x_i, y_i, \theta_i\}, \quad \text{where } i = 1, 2, 3, \ldots\ldots, k. \tag{2.10}$$

$$P = \{M_1', M_2', M_3', ..., M_q'\} \tag{2.11}$$

$$M_j = \{x_i', y_i', \theta_i'\}, \text{ where } j = 1, 2, 3, \ldots\ldots, q. \tag{2.12}$$

Where k and q denote the number of minutiae in T and P, respectively. A minutiae M_j' in P and a minutia M_i in T are considered "matching" if the spatial distance (*ld*) between them is smaller than a given tolerance ε_0 and the direction (angular) difference (*ad*) between them is smaller than an angular tolerance θ_0 .

$$ld(M_j', M_i) = \sqrt{(x_j' - x_i)^2 + (y_j' - y_i)^2} \le \varepsilon_0 \tag{2.13}$$

$$ad(M_j', M_i) = \min(|\theta_j' - \theta_i|, 360^0 - |\theta_j' - \theta_i|) \le \theta_0 \tag{2.14}$$

Eq.2.14 takes the minimum of $|\theta_j' - \theta_i|$ and $360^0 - |\theta_j' - \theta_i|$ because of the circularity of angles (the difference between angles of 2^0 and 368^0 is only 4^0). The tolerance boxes (or

hyper-spheres defined by ε_0 and θ_0 are necessary to compensate for the unavoidable errors made by feature extraction algorithms and to account for the small plastic distortions that cause the minutiae positions to change.

In order to overcome this problem, our proposed theorem describes bellow:

We consider M is the given one minutia point the local structure of the minutiae point is described by the following vectors.

$$Local_M = \{(ld_1, ld_2, ld_3, ..., ld_N), (\alpha_1, \alpha_2, \alpha_3, ..., \alpha_N), (\beta_1, \beta_2, \beta_3, ..., \beta_N), R\} \quad (2.15)$$

Where N is the number of neighborhood minutiae points taken into consideration during matching. ***R*** is the direction of the local ridge of the minutia M (**Fig.2.6a**). ld_n (n = 1, 2, 3, ….., N) describe the linear distance between the selected minutia M and its nth nearest neighborhood, α_n (n = 1, 2, 3, ….., N) is the related radial angle between M and its nth nearest neighbor, and β_n (n = 1, 2, 3, ……., N) represents the related position angle of the nth nearest neighborhood of the minutiae points.

The linear distances between the selected minutia M and its *nth* nearest neighborhood are calculated as follows:

$$ld_1 = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}$$

$$ld_2 = \sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2}$$

..

..

$$ld_n = \sqrt{(x_n - x_0)^2 + (y_n - y_0)^2} \quad (2.16)$$

The related radial angle between M and its *nth* nearest neighborhood are calculated as follows:

$$\alpha_n = diff(R_n, R), \quad n = 1,2,3,...., N \quad (2.17)$$

The related position angle of the *nth* nearest neighborhood of the minutiae points are calculated as follows:

$$\beta_n = diff\left(\arctan\left(\frac{y_n - y_0}{x_n - x_0}\right), R\right) \quad (2.18)$$

The function *diff*( ) calculates the difference of two angles and converts the result to range $\{0, 2\pi\}$.

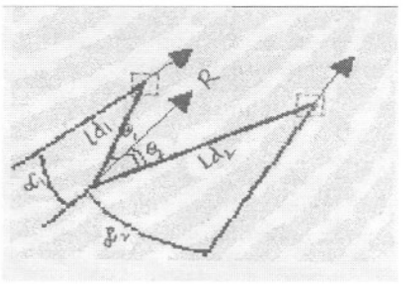


Fig.2.6:(a) Angular Distance

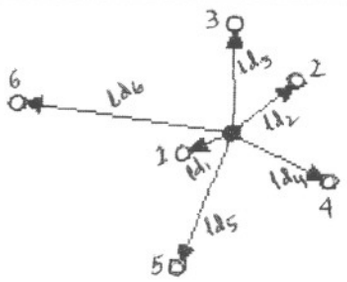


Fig.2.6:(b) Linear Distance

The proposed matching algorithm calculates how similar the neighborhood of one minutia in the input fingerprint is to that of one in the stored template. If it is similar enough, then these two minutiae are taken as a matched minutiae pair. After each minutia in the input fingerprint is checked, the total number of matched minutiae pair is used to calculate the final matching score. The selections of the reference point as well as the number of neighborhood points are very much important for the system performance.

In this research work, the reference point is selected manually and the neighborhood is defined as *7 nearest neighbors* for each minutia as shown in **Fig.2.6b**. When two minutiae are compared, the relative position and angles of their *7 nearest neighbor's* minutiae are investigated.

Now rewriting **Eq.2.15**, we obtain another form of the local feature vector. Consider the one local minutia point M of an input fingerprint image.

$$Local_M = \{\{ld_1, \alpha_1, \beta_1\}, \{ld_2, \alpha_2, \beta_2\}, \{ld_3, \alpha_3, \beta_3\}, ..., \{ld_N, \alpha_N, \beta_N\}, R\} \quad (2.19)$$

The stored template of the local minutia M' is describing as follows:

$$Local_{M'} = \{\{ld'_1, \alpha'_1, \beta'_1\}, \{ld'_2, \alpha'_2, \beta'_2\}, \{ld'_3, \alpha'_3, \beta'_3\}, ..., \{ld'_N, \alpha'_N, \beta'_N\}, R'\} \quad (2.20)$$

Now to make a decision whether it is matched with M or not. The condition to match with the reference minutia point i.e., the local ridge directions of the two minutiae, if $|R - R'| > \varepsilon_R$, then $Local_M$ and $Local_M^{/}$ *are* not matched.

If $|R - R'| \leq \varepsilon_R$ then the neighborhood minutiae will check the following conditions:

$$\begin{vmatrix} ld_i & - & ld'_j \end{vmatrix} \leq \varepsilon_{ld}$$
$$\begin{vmatrix} \alpha_i & - & \alpha'_j \end{vmatrix} \leq \varepsilon_{\alpha} \qquad (2.21)$$
$$\begin{vmatrix} \beta_i & - & \beta'_j \end{vmatrix} \leq \varepsilon_{\beta}$$

If the jth neighborhood of the template minutia $Local_M^{/}$ are satisfied with ith neighborhood of the minutia then it is recognized the person. Otherwise, result will be unrecognized or false rejection.

2.6 Grid-Mapping Feature Extraction Technique

Traditionally, minutiae, core, delta, crossover, bifurcation, ridge ending, pore, island, enclosure, bridge features of a fingerprints are considered to identify a person **Fig.1.7**. However, these features are accurately recognized when the when the features are prominent, during scanning the fingerprint images. But in some cases due to some different reasons it is not always available good fingerprint images. In such cases there is very difficult to recognize these features with proper technique to get accurate information about the fingerprint.

In this research, we propose an alternative crude method Grid-Mapping Feature (GMF), which can be applied for feature extraction when all the traditional method fail to recognize very poor fingerprint images. GMF extraction process, the features of poor

fingerprint images are extracted in the following way. The fingerprint image is transformed to 300X360-pixel image by using image-scaling process. In this process, the preprocessed fingerprint is grid mapping with fixed square/rectangle cells, such as 16x16 square/rectangle areas. If each small square/rectangle cell contains more than 45~55% black pixels, then it is considering as digital value 1 otherwise 0. An extraction algorithm is used to extract grid mapping features from the gray scale fingerprint image by examining the neighborhood pixels (**Fig.2.6**). *In **Chapter -3** the GMF extraction process is described widely.*

Grid Mapping Feature (GMF) Extraction Algorithm:

Step 1: Input the image of a fingerprint.

Step2: Define the number of m x m matrices.

Step3: Calculate the binary value for each cell.

A. Determine each pixel's RGB value and get the sum of RGB value.
B. Check whether or not the sum of RGB value is near the RGB value of BLACK pixel.
C. If the sum of RGB value is near the RGB value of BLACK pixel then increase the Index counter.
D. Repeat steps A, B, C for each pixel of the cell.
E. If the number of the pixel is above the 45~55% then put the binary value 1 otherwise 0.
F. Return binary value of the cell.

Step 4: Put the cell value into a file.

Step 5: Repeat step 3 & 4 for each cell.

Step6: Exit.

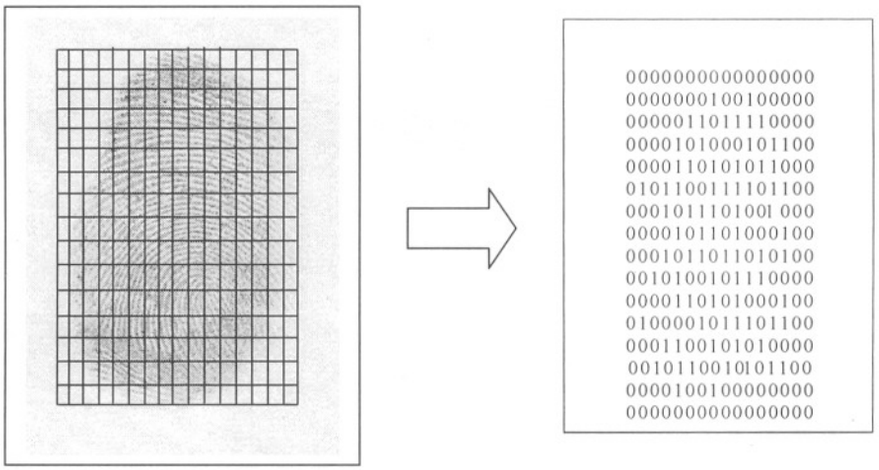


Fig.2.7: GMF extraction for a sample fingerprint image.

2.7 Fingerprints as Oriented Texture

An *oriented texture* is a texture characterized by a dominant orientation (*direction*) at each point. This type of texture is formed by the ridges of a fingerprint. It is highly desirable to be able to segment these textures into regions of homogeneous orientation. The automated analysis of fingerprints is still a very active area of research. The segmentation of the fingerprint image into regions of homogeneous orientation is an important step in locating features of a fingerprint that can be used to find globally similar fingerprints in a large database, before applying a more detailed matching algorithm to them.

The usual first step in the analysis of such a texture is to move a small neighbourhood over the texture image, and determine the dominant orientation at each position of the neighbourhood. These dominant orientations are then stored in a new image which summarizes the texture. Usually, an orientation "strength" image is also calculated. Most

of the algorithms used to calculate the orientation description image can only calculate one principal orientation at each point (http://www.prip.tuwien.ac.at/~hanbury/praktika).

The smooth flow pattern of ridges and valleys in a fingerprint can be viewed as an *oriented texture* field [44]. The image intensity surface in fingerprint images is comprised of ridges whose directions vary continuously, which constitutes an *oriented texture*. Most textured images contain a limited range of spatial frequencies, and mutually distinct textures differ significantly in their dominant frequencies [45, 46, 47]. Texture regions possessing different spatial frequency, orientation, or phase can be easily discriminated by decomposing the image in several spatial frequency and orientation channels. For typical fingerprint images scanned at 500 dpi, there is very little variation in the spatial frequencies (determined by inter-ridge distances) among different fingerprints. This implies that there is an optimal scale (spatial frequency) for analyzing the fingerprint texture. Every pixel in a fingerprint image is associated with a dominant local orientation and local measure of coherence of the flow pattern. A symbolic description of a fingerprint image can be derived by computing the angle and coherence at each pixel in the image. Fingerprints can be represented/matched by using quantitative measures associated with the flow pattern (*oriented texture*) as features.

A Gabor filterbank is one of the well-known techniques to capture useful information in specific bandpass channels as well as to decompose this information into orthogonal components in terms of *spatial frequencies*. The four main steps in our representation extraction algorithm are: (i) determine *a reference point* for the fingerprint image, (ii) *tessellate* the region around the reference point, (iii) *filter* the region of interest in eight different direction using a bank of Gabor filters (*eight directions are required to completely capture the local ridge characteristics in a fingerprint while only four directions are required to capture the global configuration [3]*), and (iv) compute the *Average Absolute Deviation* from the mean (AAD) of gray values in individual sectors in filtered images to define the *feature vector*, also called the FingerCode (similar to the IrisCode introduced by Daugman [48].

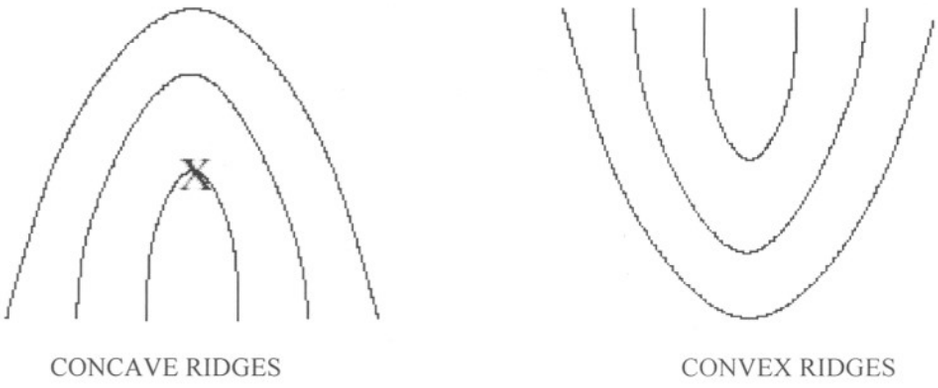


Fig.2.8: Concave and convex ridges in a fingerprint image when the finger is positioned upright. The reference point is marked by X.

2.7.1 Reference Point Location

Fingerprints have many conspicuous landmarks and any combination of them could be used for establishing a reference point. The reference point of a fingerprint as the point of maximum curvature of the concave ridges (see **Fig.2.8**) in the fingerprint image. The reference point location algorithm performs extremely well for good quality fingerprint images of whorl, left loop, right loop, and arch types. This algorithm has higher error in consistently location the reference point in the arch type fingerprints due to the absence of singular points in arch type fingerprint images. The algorithm fails for very poor quality fingerprints because of the errors in orientation field estimation.

2.7.2 Tessellation

Let $I(x, y)$ denote the gray-level at pixel (x, y) in an $M \times N$ fingerprint image and let (x_c, y_c) denote the reference point (**Fig.2.9**). The region of interest in the fingerprint is defined [10] as the collection of all the sectors S_i, where the i^{th} sector S_i is computed in terms of parameters (r, θ) as follows:

$$S_i = \{(x, y) \mid b(T_i + 1) \le r < b(T_i + 2), \\ \theta_i \le \theta < \theta_{i+1}, 1 \le x \le N, 1 \le y \le M\}, \tag{2.22}$$

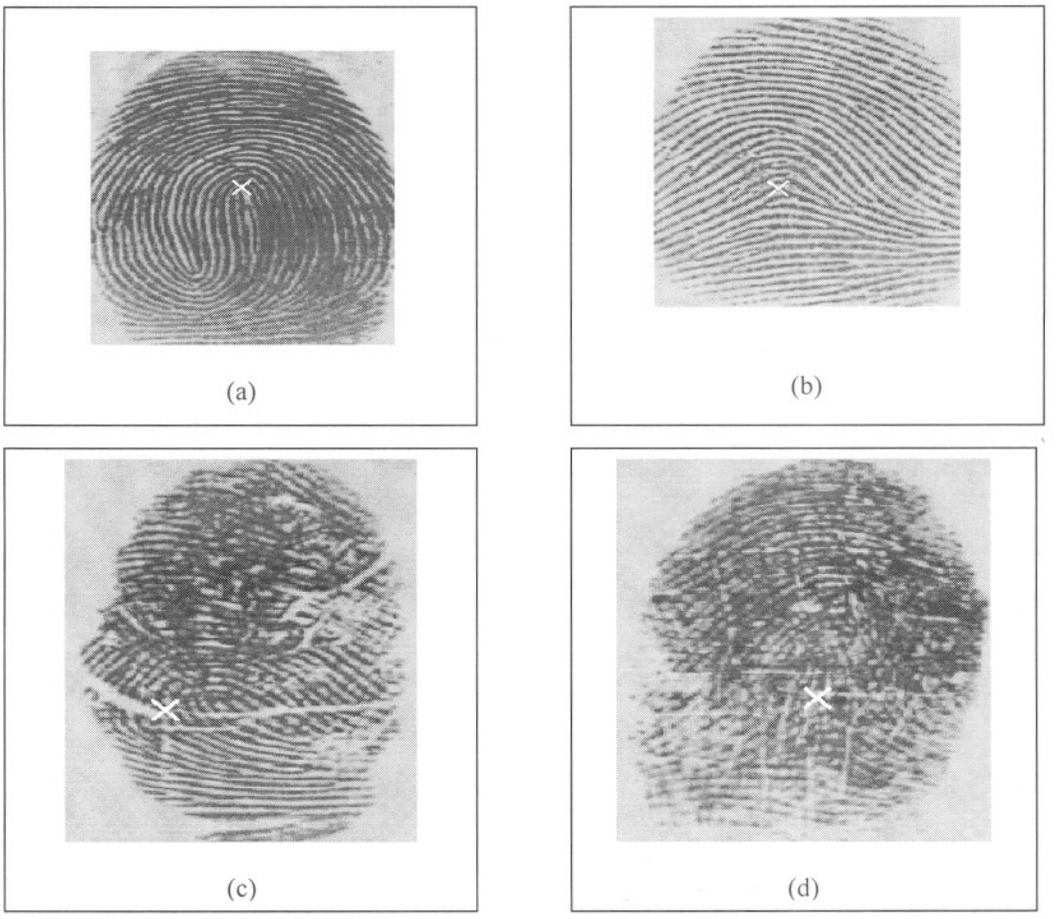


Fig.2.9: Examples of the results of our reference point location algorithm good quality fingerprints (a) and (b). The algorithm fails on very poor quality fingerprints such as (c) and (d).

Where

$$T_i = \text{i div k}, \tag{2.23}$$

$$\theta_i = (\text{i mod k}) \times (2\pi/\text{k}), \tag{2.24}$$

$$r = \sqrt{(x - x_c)^2 + (y - y_c)^2}, \tag{2.25}$$

$$\theta = \tan^{-1}(y - y_c)/(x - x_c), \tag{2.26}$$

b is the width of each band, k is the number of sectors considered in each band, and i=0, . . . , ($B \times k$ –1), where B is the number of concentric bands considered around the reference point for feature extraction.

2.7.3 Filtering

Fingerprints have local parallel ridges and valleys, and well-defined local frequency and orientation (see **Fig.2.10**). Properly tuned Gabor filters [48, 49] can remove noise, preserve the true ridge and valley structures, and provide information contained in a particular orientation in the image. A minutia point can be viewed as an anomaly in locally parallel ridges and it is this information that we are attempting to capture using the Gabor filters.

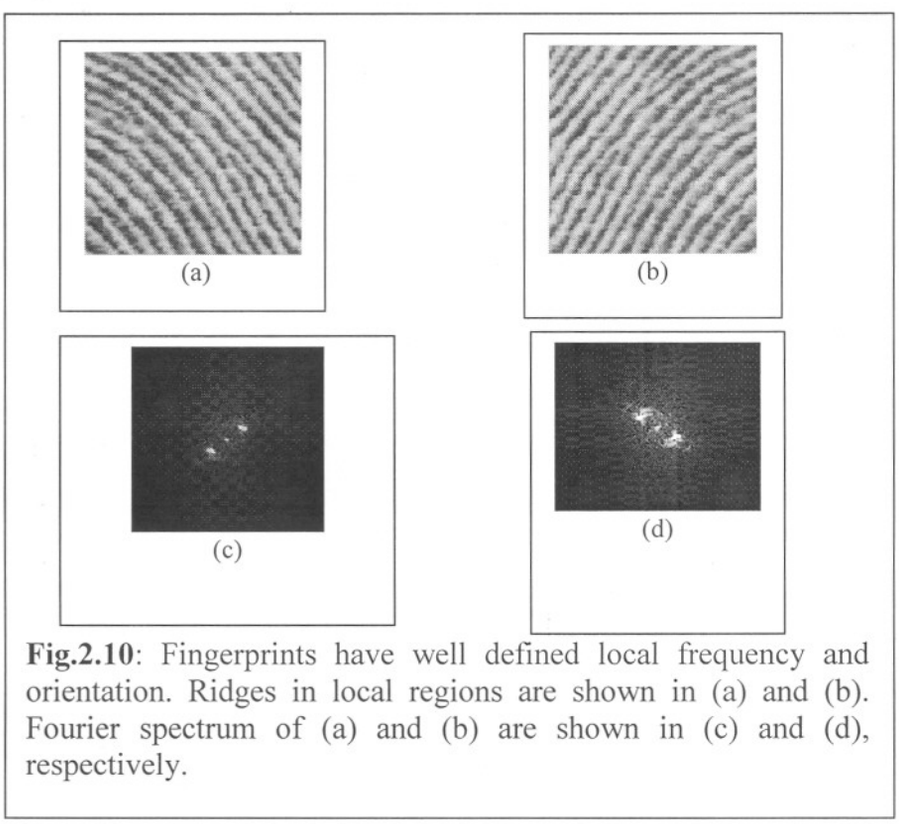


Fig.2.10: Fingerprints have well defined local frequency and orientation. Ridges in local regions are shown in (a) and (b). Fourier spectrum of (a) and (b) are shown in (c) and (d), respectively.

Before filtering the fingerprint image, it is needed to normalize the gray level intensities in the region of interest in each sector separately to a constant mean and variance. Normalization is performed to remove the effects of sensor noise and gray level background due to finger pressure differences. Let $I(x, y)$ denote the gray value at pixel (x, y), M_i and V_i, the estimated mean and variance of grey levels in sector S_i respectively, and $N_i(x, y)$, the normalized gray-level value at pixel (x, y). For all the pixels in sector S_i, the normalized image is defined as:

$$N_i(x,y) = \begin{cases} M_0 + \sqrt{\dfrac{V_0 \times I(x,y) - M_i)^2}{V_i}}, & \text{if I(x,y)>M}_i \\ M_0 - \sqrt{\dfrac{V_0 \times (I(x,y) - M_i)^2)}{V_i}}, & \text{otherwise,} \end{cases} \tag{2.27}$$

Where M_0 and V_0 are the desired mean and variance values, respectively.

Normalization is a pixel-wise operation which does not change the clarity of the ridge and valley structures. If normalization is performed on the entire image, then it cannot compensate for the intensity variations in different parts of the image due to the finger pressure differences. A separate normalization of each individual sector alleviates this problem.

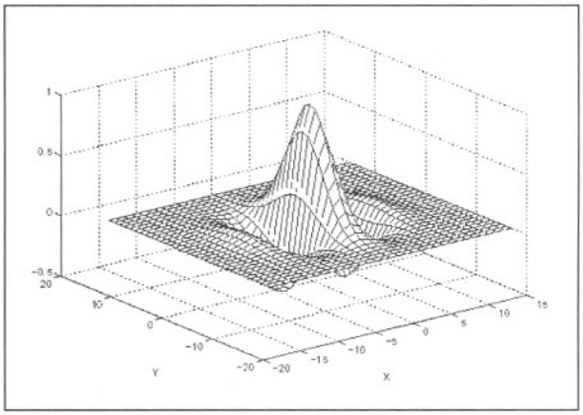
An even symmetric Gabor filter has the following general form in the spatial domain:

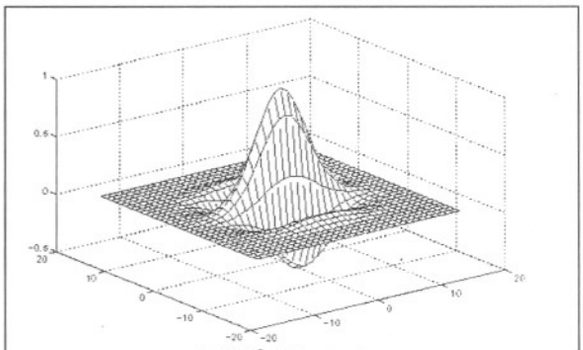
$$G(x,y;f,\theta) = \exp\left\{\frac{-1}{2}\left[\frac{x'^2}{\delta_x^2} + \frac{y'^2}{\delta_y^2}\right]\right\}\cos(2\pi f x'), \tag{2.28}$$

$$x' = x\sin\theta + y\cos\theta, \tag{2.29}$$

$$y' = x\cos\theta - y\sin\theta, \tag{2.30}$$

Where f is the frequency of the sinusoidal plane wave along the direction θ from the x-axis, and δ_x and δ_y are the space constants of the Gaussian envelope along x and y axes, respectively. The spatial characteristics of Gabor filters can be seen in **Fig.2.11.**





(b) 90° Orientation

Fig.2.11: Gabor filters (mask size = 33 × 33, f = 0.1, δ_x = 4.0, δ_y = 4.0). Only 0° and 90° oriented filters are shown here.

2.7.4 Feature Vector

It is difficult to rely on features that are extracted based on explicit detection of structural features in fingerprints, especially in poor quality images. Features based on statistical properties of images are likely to degrade gracefully with the image quality deterioration. For this study, it has been used *grayscale variance-based features*. The average absolute deviation of the gray levels from the mean value in an image sector is indicative of the overall ridge activity in that sector which we claim to be useful for fingerprint classification and verification. Similar features were successfully used earlier by Jain and Farrokhnia [50] for texture classification and segmentation.

Let $F_{i\theta}(x, y)$ be the θ-direction filtered image for sector S_i. Now, $\forall i \in \{0,1,2,....,79\}$ and $\theta \in \{0^\circ, 22.5^\circ, 45^\circ, 67.5^\circ, 90^\circ, 112.5^\circ, 135^\circ, 157.5^\circ\}$, the feature value, $V_{i\theta}$, is the average absolute deviation from the mean defined as:

$$V_{i\theta} = \frac{1}{n_i}\left(\sum_{n_i} |F_{i\theta}(x, y) - P_{i\theta}|\right), \tag{2.31}$$

Where n_i is the number of pixels in S_i and $P_{i\theta}$ is the mean of pixels values of $F_{i\theta}(x, y)$ in sector S_i. The average absolute deviation of each sector in each of the eight filtered images defines the components of our 640-dimensional *feature vector*.

Chapter Three

GRID-MAPPING FEATURE EXTRACTION TECHNIQUE

Contents **Page No.**

3.1 Introduction:

Traditionally, minutiae, core, delta, crossover, bifurcation, ridge ending, pore, island, enclosure, bridge features of a fingerprints are considered to identify a person (**Fig.1.4**). However, these features are accurately recognized when the features are prominent, during scanning of the fingerprint images. But in some cases due to some reasons it is not always available good fingerprint images. In such cases it is very difficult to recognize these features with proper technique to get accurate information about the fingerprint. *In this research, we propose an alternative crude method Grid-Mapping Feature (**GMF**), which can be applied when all the traditional methods fail to recognize very poor fingerprint images.* The technique to get these features is described bellow.

3.2 Fundamental Steps of GMF Extraction

In this section, we discuss about how to extract features the off-line fingerprint image. In *chapter two*, the traditional features of fingerprint are described elaborately. The following basic steps are consider to get *the GMF extraction of poor fingerprint images* (**Fig.3.1(a)**).

1. Fingerprint Image Acquisition.
2. Fingerprint Image Pre-processing
   (i) Filtering, (ii) Clipping (iii) Edge Detection and (iv) Thinning.
3. GMF Extraction Technique.

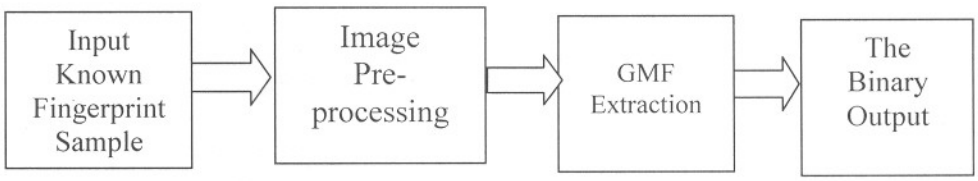


Fig.3.1(a): Block Diagram to Extract Feature of Fingerprint Image Using GMF Technique.

3.2.1 Fingerprint Image Acquisition:

The fingerprint image acquired by the off-line process is known as the "inked" fingerprints, while the image acquired by the on-line process is known as "live-scan" fingerprints. There are lots off devices for live-scan of fingerprints. These are described in the following sections:

3.2.1.1 Optical Fingerprint Sensors

The optical method is one of the most common methods [51]. At the heart of the optical scanner, a CCD-Camera (charged coupled device) is used. A CCD-Camera is simply an array of light sensitive diodes called photosites. In general the finger will be placed on a

glass plate and the CCD camera takes the picture. The CCD system has an array of LEDs (light-emitting diodes) to illuminate the ridges and valleys of the finger. The advantage of optical systems is the very low price; the disadvantage of optical systems is that they are quite easy to fake. Another problem is latent fingerprints, which are remaining fingerprints from the previous finger those were placed on the sensor surface. Leading manufacturers of optical fingerprint scanners are Delsy, Dermalog, Smiths Heimann Biometrics (spin-off of Jenoptik / Rheinmetall)

3.2.1.2 Thermal Electric Sensors

The thermal-electric [51] method is less usual. Currently there exists only the Atmel FingerChip™ with thermal-electric technology on the market. The FingerChip utilizes a unique method for imaging the entire finger by "sweeping" it across the sensor. Sweeping captures successive images (slices), and then uses special software to reconstruct the fingerprint image. This method allows the FingerChip™ to return a large, high quality, 500 dots per inch image of the fingerprint with 256 grayscales.

The sensor measures the temperature differential between the skin ridges and the air caught in the fingerprint valleys. This method provides a high quality image even on poor quality fingerprints such as ones that are dry or worn with little depth between the peaks and valleys of the fingerprint. The thermal technology also operates well under extreme environmental conditions, like extreme temperatures, high humidity, dirt, oil and water contamination. In addition to providing a small form factor, the sweeping method also has the benefit of self-cleaning the sensor, which avoids latent fingerprints. Latent fingerprints are prints left behind on a non-sweeping sensor which not only can cause problems with future reads but also leaves an image that can be copied and possibly used to gain access to a system. In fact, the sweeping method utilizing thermal based technology allows the FingerChip to be resistant of many other technologies. The FingerChip works in low-temperature and high humidity environments, too. Another benefit of this technology is a very big image of good quality, and an always clean sensor. The disadvantage is, that the image quality depends a little bit from the user's skill in using the scanner. The second disadvantage is the heating of the sensor array

which increases the power-consumption of the sensor. The heating of the sensor is necessary to avoid the possibility of a thermal equilibrium between the sensor and the fingerprint surface. In high volume the design of the scanner leads to lower prices because the manufacturing process needs less silicon.

3.2.1.3 Capacitive Sensors

The capacitive method [51] is one of the most popular methods. Like the other scanners the capacitive fingerprint scanner generates an image of the ridges and valleys that make up a fingerprint. The capacitive sensor uses capacitors with electrical current for measuring the fingerprint. A capacitive sensor is made up of an array of tiny cells. Each cell includes two conductor plates, covered with an insulating layer. The main advantage of capacitive sensors is that they require a real fingerprint. Capacitive sensors have problems with wet and dry fingers. With wet fingers the users sometimes get black images, whilst dry fingers make the image pale. Leading manufacturers of capacitive sensors are Infineon, Veridicom, Sony and ST Microelectronics.

3.2.1.4 E-Field Sensors

The E-Field sensor [51] works with an antenna array and measures the electric field beyond the surface layer of the skin where the fingerprint begins. The E-field technology claims that it is capable of working for everyone, under tough real-world conditions such as dry, worn, or dirty skin. E-Field technology creates a field between the finger and the adjacent semiconductor that mimics the shape of the ridges and valleys of the finger's epidermal layer. An under-pixel amplifier is used to measure the signals. The sensors operate together to a clearer image that accurately corresponds to the pattern of the fingerprint and results in a clearer image than optical or DC capacitive technologies produce. This allows E-Field Technology to acquire fingers that other technologies cannot. With E-Field technology, antenna arrays measure the skin's subsurface features by generating and detecting linear field geometries the live layer of skin cells is originated beneath the skin's surface. This is in contrast to the spherical or tubular field geometries generated by simple capacitive sensor, which only read the very top surface of the skin. As a result, fingers that are difficult or impossible to acquire using capacitive

sensors can be successfully acquired with E-field Technology. Recently there exists also a swipe sensor with E-field technology, which is due to be announced for release within the next few months. One disadvantage is the low resolution of the images and the small image area, what is leading to a higher Equal-Error-Rate (EER).

3.2.1.5 Touch-less Sensors

A touch-less sensor [51] works similar to an optical sensor. In general there is a precision glass optic with a distance of 2-3 inches from the fingerprint while the finger is scanned. The fingerprint is put on an area with a hole. One disadvantage can be considered is that dust and dirt may fall through the hole onto the glass optic with the possible result of bad images. Another point is that the scanned fingerprints are spherical which leads to more complex matching algorithms.

3.2.1.6 Surface Pressure Sensor

The principle of pressure sensing [51] is that when a finger is placed over the sensor area, only the ridges of the valleys in contrast have no contact with the sensor fingerprint come in contact with the sensor piezo array cells. A main difference for the further recognition and matching to other sensors we have is pressure sensors generate a 1-bit binary image. A 1-bit image has less information than an 8-bit gray-scale image. Ridge a surface pressure sensor can be considered as allowing clearly sensing both dry and wet fingers. Furthermore it has a large sensing area, which enables it to capture a complete fingerprint for better accuracy of recognition that leads to a lower EER (Equal Error Rate).

3.2.1.7 The Off-Line Fingerprint Collection Process

In this research, we use off-line fingerprint to investigate *grid-mapping feature* extraction method. In the inked fingerprint acquisition technique, ink is applied to the finger and then pressed onto a paper to form an impression. The impression onto the paper is then scanned at 500-dpi resolution by a standard grayscale scanner. These images are stored in a file for further analysis.

3.2.2 Fingerprint Image Preprocessing

In this section we discuss about how to enhance the fingerprint images quality. In order to improve the quality of the poor fingerprint images the following steps are considered (**Fig.3.1(b)**):

(i) Filtering,

(ii) Clipping,

(iii) Edge Detection,

(iv) Thinning.

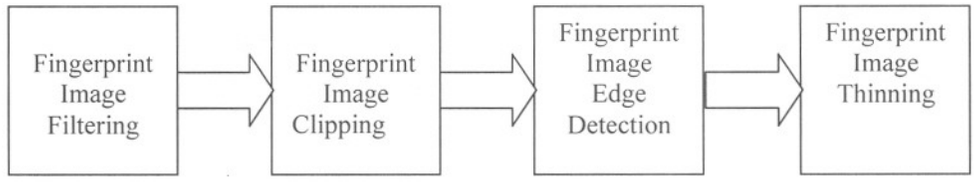


Fig.3.1(b): Block Diagram of Fingerprint Image Preprocessing

3.2.2.1 Filtering

When we are using fingerprint scanner, we may have some unwanted scratch upon our scanned image. A fingerprint expert may be needed to remove such unwanted scratch from the fingerprint image. Sometimes few part of ridge lines may be unavailable. For automatic enhancement process we need to develop some kind of algorithm to remove such kind of breaking line. In this work, noisy fingerprint images are filtered by using the Laplacian spatial filtering technique.

3.2.2.1.1 Basics of Spatial Filtering

The mechanics of spatial filtering [52] are shown in **Fig.3.2**. The process consists simply of moving the filter mask from point to point in a fingerprint image. At each point (x, y), the response of the filter mask at that point is calculated using a predefined relationship. For linear spatial filtering, the response is given by a sum of products of the filter coefficients and the corresponding fingerprint image pixels in the area spanned by the

filter mask. For the 3 × 3 mask illustrated in **Fig.3.2**, the result, R, of linear filtering with the filter mask at a point (x, y) in the image is

$$R = \omega(-1,-1)f(x-1, y-1) + \omega(-1,0)f(x-1, y) + \ldots\ldots\ldots + \omega(0,0)f(x, y) + \ldots\ldots\ldots$$

$$\ldots\ldots + \omega(1,0)f(x+1, y) + w(1,1)f((x+1, y+1), \qquad (3.1)$$

where it has been seen that the sum of products of the mask coefficients with the corresponding pixels directly under the mask. It has been noted that the coefficient $\omega(0,0)$ coincides with fingerprint image value $f(x, y)$, indicating that the mask is centered at (x, y) when the computation of the sum of products takes place. For a mask of size $m \times n$, we assume that $m = 2a + 1$ and $n = 2b + 1$, where a and b are nonnegative integers.

In General, linear filtering of a fingerprint image f of size $M \times N$ with a filter mask of size $m \times n$ is given by the expression:

$$g(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s,t) f(x+s, y+t) \qquad (3.2)$$

where, from the previous paragraph, $a = (m - 1)/2$ and $b = (n - 1)/2$. To generate a complete fingerprint image this equation must be applied for $x = 0, 1, 2, \ldots\ldots\ldots, M\text{-}1$ and $y = 0, 1, 2, \ldots\ldots\ldots, N\text{-}1$. In this way, it is assured that the mask process all pixels in the fingerprint image.

When interest lies on the response, R, of an $m \times n$ mask at any point (x, y), and not on the mechanics of implementing mask convolution, it is common practice to simplify the notation by using the following expression:

$$R = \omega_1 z_1 + \omega_2 z_2 + \ldots\ldots\ldots + \omega_{mn} z_{mn}$$

$$= \sum_{i=1}^{mn} \omega_i z_i \qquad (3.3)$$

where the ω's are mask coefficients, the z's are the values of the fingerprint image gray levels corresponding to those coefficients, and mn is the total number of coefficients in the mask. For the 3 × 3 general mask shown in **Fig.3.3**, the response at any point (x, y) in the fingerprint image by

$$R = \omega_1 z_1 + \omega_2 z_2 + \ldots\ldots\ldots\ldots + \omega_9 z_9$$

$$= \sum_{i=1}^{9} \omega_i z_i \qquad (3.4)$$

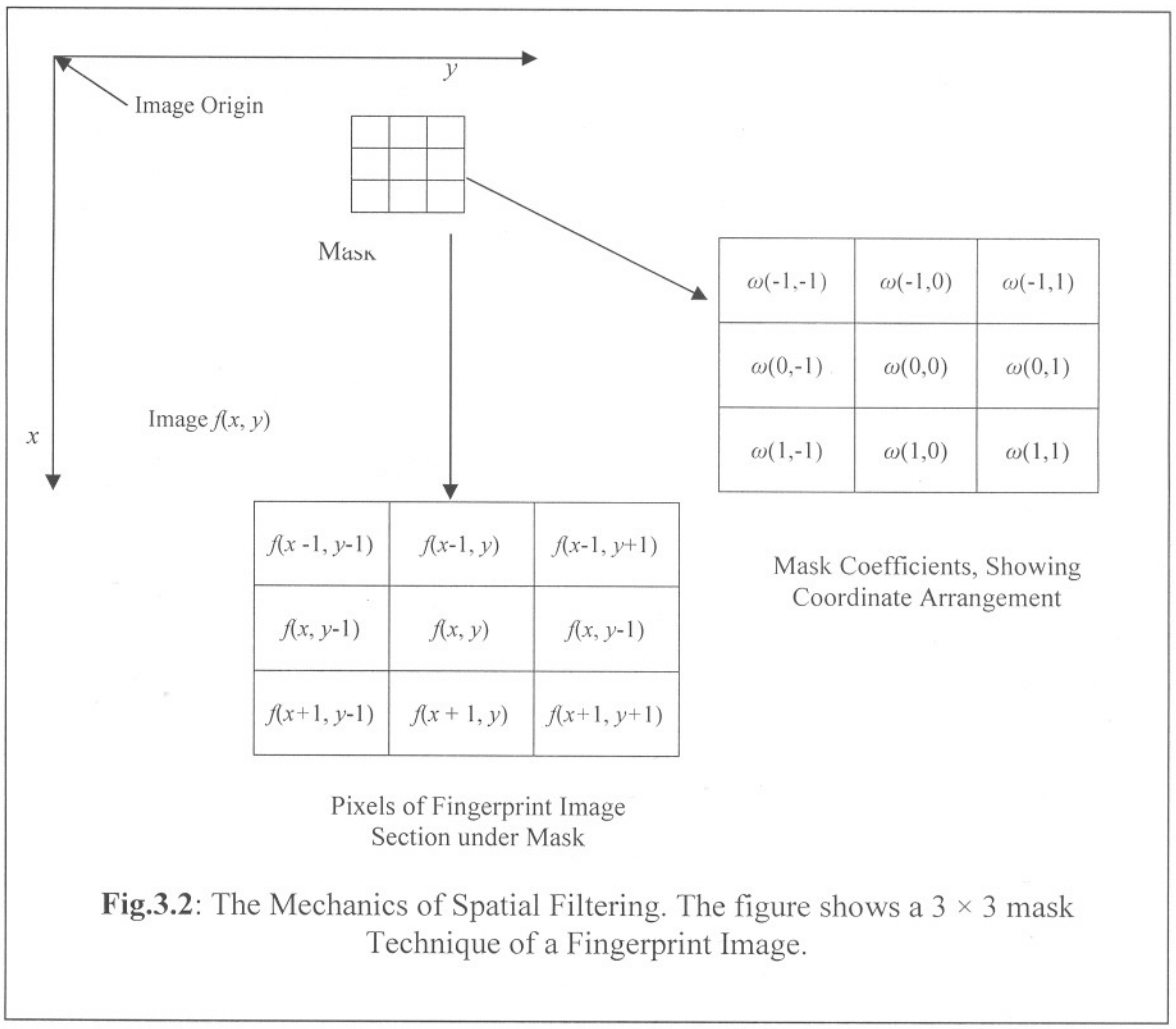


Fig.3.2: The Mechanics of Spatial Filtering. The figure shows a 3 × 3 mask Technique of a Fingerprint Image.

ω_1	ω_2	ω_3
ω_4	ω_5	ω_6
ω_7	ω_8	ω_9

Fig.3.3: Another representation of a general 3 × 3 spatial filter mask.

3.2.2.1.2 Smoothing Spatial Filters

Smoothing filters [52] are used for blurring and for noise reduction. The output (response) of a smoothing, linear spatial filter is simply the average of pixels contained in the neighbourhood of the filter mask. **Fig.3.4** shows two 3 × 3 smoothing filters. Use of the first filter yields the standard average of the pixels under the mask. This can the best seen by substituting the coefficients of the mask into **Eq.(3.3)**

$$R = \frac{1}{9}\sum_{i=1}^{9} z_i$$

which is the average of the gray levels of the pixels in the 3 × 3 neighbourhood defined by the mask. It is noted that, instead of being 1/9, the coefficients of the filter are all 1's. The idea here is that it is computationally more efficient to have coefficients valued 1. At the end of the filtering process the entire fingerprint image is divided by 9. An $m \times n$ mask would have a normalizing constant equal to $1/mn$. A spatial averaging filter in which all coefficients are equal is sometimes called a box filter.

$\frac{1}{9}\times$

1	1	1
1	1	1
1	1	1

$\frac{1}{16}\times$

1	2	1
2	4	2
1	2	1

Fig.3.4: Two 3 × 3 smoothing (averaging) filter masks. The constant multiplier in front of each mask is equal to the sum of the values of its coefficients, as is required to compute an average.

With reference to **Eq.(3.2)**, the general implementation for filtering an $M \times N$ fingerprint image with a weighted averaging filter of size $m \times n$ (m and n odd) is given by the expression:

$$g(x,y) = \frac{\sum_{s=-a}^{a}\sum_{t=-b}^{b}\omega(s,t)f(x+s,y+t)}{\sum_{s=-a}^{a}\sum_{t=-b}^{b}\omega(s,t)} \quad (3.5)$$

The denominator in **Eq.(3.5)** is simply the sum of the mask coefficients and, therefore, it is a constant that needs to be computed only once. Typically, this scale factor is applied to all the pixels of the output fingerprint image after the filtering process is completed.

3.2.2.1.3 Sharpening Spatial Filters

The principal objective of sharpening [52] is to highlight fine detail in a fingerprint image or to enhance detail that has been blurred, either in error or as a natural effect of a particular method of image acquisition.

Foundation of the Method:

The derivatives of a digital function are defined in terms of difference. There are various ways to define these differences. However, we require that any definition we use for a first derivative:

(i) must be zero in flat segments (areas of constant gray levels values),

(ii) must be nonzero at the onset of a gray level step or rump,

(iii) must be nonzero along rumps.

Similarly, any definition of a second derivative

(i) must be zero in flat areas,

(ii) must be nonzero at the onset of a gray level step or rump,

(iii) must be nonzero along rumps of constant slope.

A basic definition of the first-order derivative of a one-dimensional function $f(x)$ is the difference

$$\frac{\delta f}{\delta x} = f(x+1) - f(x) \quad (3.6)$$

We used a partial derivative here in order to keep the notation the same as when we consider an image function of two variables, $f(x, y)$, at which time we will be dealing with partial derivatives along the two spatial axes.

Similarly, we define a second-order derivative as the difference

$$\frac{\delta^2 f}{\delta x^2} = f(x+1) + f(x-1) - 2f(x) \quad (3.7)$$

The Laplacian (Second Derivatives for Image Enhancement):

In this section, we consider in some detail the use of two-dimensional, second-order derivatives for fingerprint image enhancement [52]. The approach basically consists of defining a discrete formulation of the second-order derivative and then constructing a filter mask based on that formulation. We are interested in isotropic filters, whose response is independent of the direction of the discontinuities in the fingerprint image to which the filter is applied.

Development of the method:

It can be shown that the simplest isotropic derivative operation is the *Laplacian*, which, for a function (fingerprint image) $f(x, y)$ of two variables, is defined as

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{3.8}$$

Because derivatives of any order are linear operations, the Laplacian is a linear operator. In order to be useful for digital image processing, this equation needs to be expressed in discrete form. There are several ways to define a digital Laplacian using neighbourhoods. Taking into account that we now have two variables, we use the following notation for the partial second-order derivative in the x-direction:

$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y) \tag{3.9}$$

and, similarly in the y-direction, as

$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y) \tag{3.10}$$

The digital implementation of the two-dimensional Laplacian in **Eq. (3.8)** is obtained by summing these two components:

$$\nabla^2 f = [f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1)] - 4f(x, y) \tag{3.11}$$

This equation can be implemented using the mask shown in **Fig.3.5(a)**, which gives an isotropic result for rotations in increments of 90^0. The diagonal directions can be incorporated in the definition of the digital Laplacian by adding two more terms to **Eq.(3.11)**, one for each of the two diagonal directions. The form of each new term is the same as either **Eq.(3.9)** or **Eq.(3.10)**, but the coordinates are along the diagonals. Since each diagonal term also contains a $-2f(x, y)$ term, the total subtracted from the difference

terms now would be $-8f(x, y)$. The mask used to implement this new definition is shown in **Fig.3.5(b)**. The mask yields isotropic results for increments of 45^0. The other two masks shown in **Fig.3.4** also are used frequently in practice. The **Fig.3.5(c,d)** shows the equivalent results, but the difference in sign must be kept in mind when combining (by addition or subtraction) a Laplacian-filtered image with another image. Thus, the basic way in which we use the Laplacian for image enhancement is as follows:

$$g(x,y) = \begin{cases} f(x,y) - \nabla^2 f(x,y) & \text{if the center coefficient of the Laplacian mask is negative.} \\ f(x,y) + \nabla^2 f(x,y) & \text{if the center coefficient of the Laplacian mask is positive.} \end{cases} \quad (3.12)$$

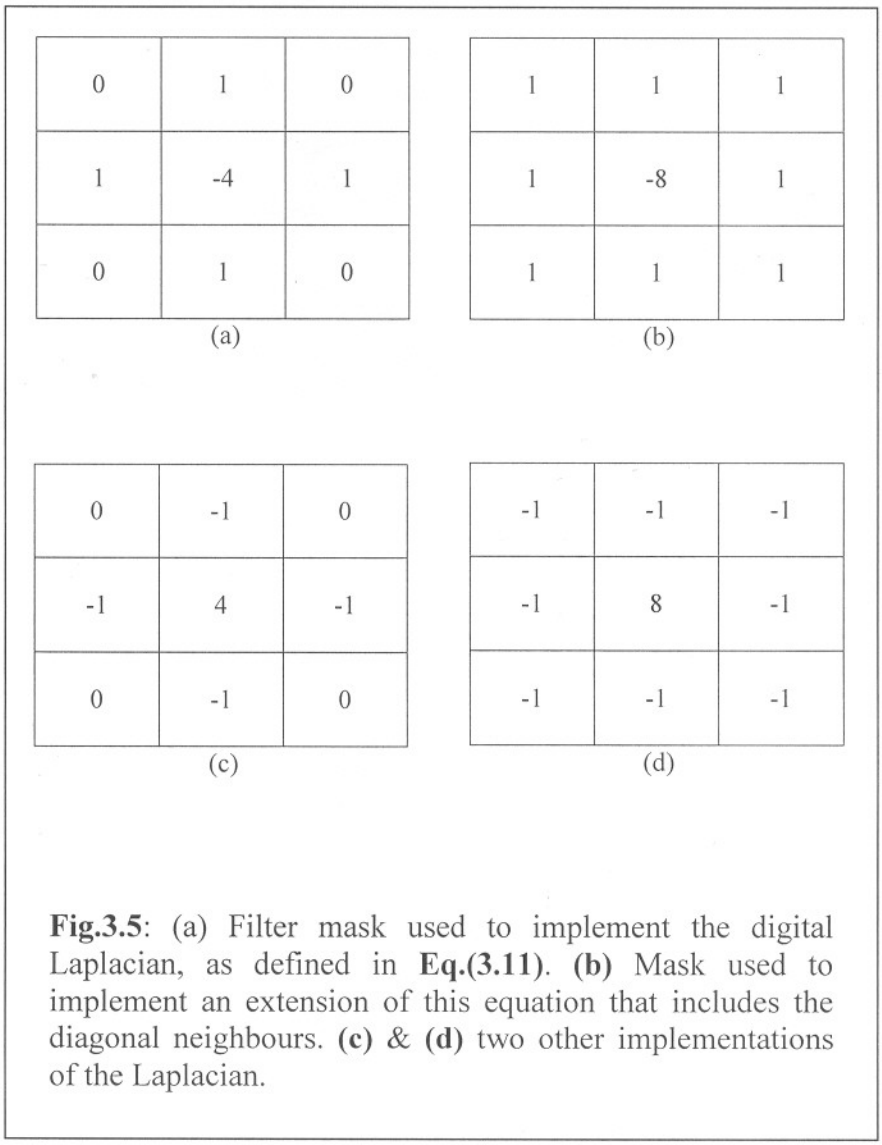


Fig.3.5: (a) Filter mask used to implement the digital Laplacian, as defined in **Eq.(3.11)**. **(b)** Mask used to implement an extension of this equation that includes the diagonal neighbours. **(c)** & **(d)** two other implementations of the Laplacian.

3.2.2.2 Clipping

Clipping is another important pre-processing step in image processing and pattern recognition system. It is quite clear to all that we are not interested in whole portion of the scanned image. We are only interested into the portion of fingerprint data. So we need to clip the fingerprint data. The scanned image size may be large. We can say that the necessity of clipping in pre-processing stage is:

a. Accurate pattern matching.
b. Faster processing.
c. Position independent pattern matching.

Now the problem is how to clip the fingerprint data from the scanned image. Many researchers do the clipping process in various ways. Some one use boundary detection algorithm, others develop their own algorithm. In our work we just develop our simple technique for clipping purpose. Our clipped image should be a rectangle. We need to determine the coordinate position of four boundary points. For this purpose we define the fingerprint image into four regions, left, right, top and bottom. First of all to determine *TopMost* of top region we start scanning from co-ordinate position (0, 0) to right of the image. Whenever we find any black pixels we mark it as *TopMost* and stop scanning more. If we find no pixel in first row we move to scan through the second row, and so on.

Now we want to determine the *BottomMost* point. We start from (x, y) where x and y be the maximum value of coordinate position. Starting from (x, y) we move right to left. Whenever we find any black pixel we mark it as *BottomMost*. If any pixel is not found in first row then we go through second row, and so on (**Fig.3.6**).

In order to determine the *LeftMost* we again start from (0, 0) coordinate position and scan through top to bottom. The first black pixel found in the scanning is marked as *LeftMost*. If no black pixel in first column then we will scan the second column (i.e., from left to right).

The *RightMost* is found in the same way, except we start scanning from (x, 0) position and instead of scanning next column we move to previous column (i.e., from right to left).

These processes can be summarized in the following algorithmic steps:

- **Algorithm to determine *TopMost* :**
  1. Start scanning from coordinate (0, 0).
  2. For each row of pixels (top to bottom) perform step 3.
  3. For each pixel from left to right in current row do
     3.1 If the current pixel is a black then
     mark it as *TopMost* and break
     3.2 Move to next pixel
  4. Store *TopMost.*

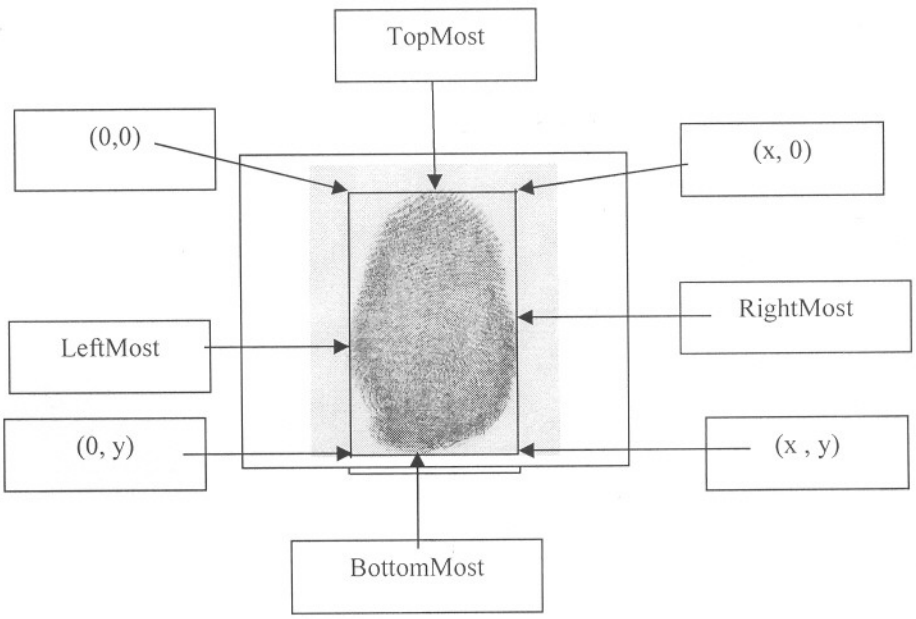


Fig.3.6: Shows the Process of Detection of Edges for Clipping the Fingerprint Image.

- **Algorithm to determine *BottomMost* :**
  1. Start scanning from coordinate (x, y).
  2. For each row of pixels (bottom to top) perform step 3.
  3. For each pixel from right to left in current row do
     3.1 If the current pixel is a black then
     mark it as *BottomMost* and break.
     3.2 Move to next pixel.
  4 Store *BottomMost*.
- **Algorithm to determine *LeftMost* :**

1. Start scanning from coordinate $(0, y)$.
2. For each column of pixels (left to right) perform step 3.
3. For each pixel from top to bottom in current column do
   3.1 If the current pixel is a black then
   mark it as *LeftMost* and break.
   3.2 Move to next pixel.
4. Store *LeftMost*.

- **Algorithm to determine *RightMost* :**
  1. Start scanning from coordinate $(x, 0)$.
  2. For each column of pixels (right to left) perform step 3.
  3. For each pixel from top to bottom in current column do
     3.1 If the current pixel is a black then
     mark it as *RightMost* and break.
     3.2 Move to next pixel.
  4. Store *RightMost*.

Now we have *TopMost, BottomMost, LeftMost* and *RightMost* position in our hand and we need to determine the coordinates of the angular points of desired rectangle area. Let the coordinates of *LeftTop, RightTop, LeftBottom* and *RightBottom* be:

- *LeftTop(x, y)* ≡ *LeftMost x , TopMost y*
- *RightTop(x, y)* ≡ *RightMost x , TopMost y*
- *LeftBottom(x, y)* ≡ *RightMost x , BottomMost y*
- *RightBottom(x, y)* ≡ *LeftMost x , BottomMost y*

The image between the clipping area *LeftTop, RightTop, LeftBottom* and *RightBottom* is stored for future processing. The screen shot of the clipping is shown in **Fig.3.7**.

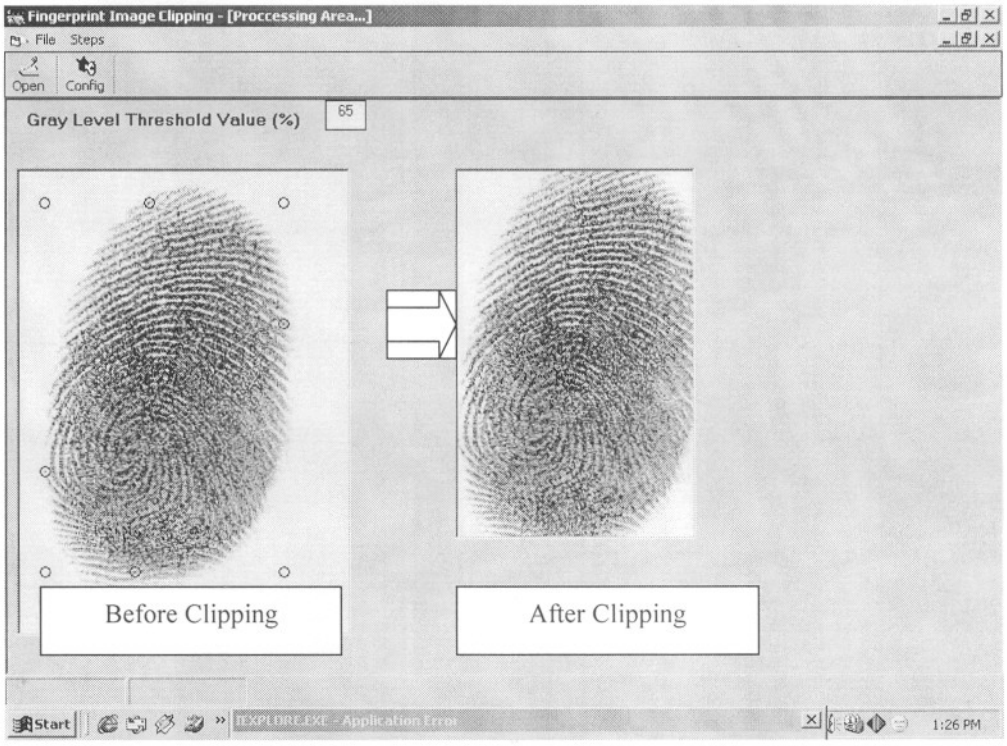


Fig.3.7: Shows the Fingerprint Image Clipping.

3.2.2.3 Edge Detection

In this section, we discuss about how to detect edge [52] of a fingerprint image. It is one of the vital issues that ensure to extract the feature of a fingerprint image properly. The edge detection is one of the gray-level discontinuities of image segmentation. Other two gray-level discontinuities of image segmentation are point and line detection. Although point and line detection certainly are important in any discussion on segmentation, edge detection is by far the most common approach for detecting meaningful discontinuities in

gray level. In this section we discuss approaches for implementing first and second-order digital derivatives for the detection of edges in a fingerprint image.

Basic Formulation

An edge is a set of connected pixels that lie on the boundary between two regions. However, an edge is a local concept whereas a region boundary, owing to the way it is defined, is a more global idea. A reasonable definition of edge requires the ability to measure gray-level transitions in a meaningful way. An ideal edge has the properties of the model shown in **Fig.3.8(a)**. In practice, optics, sampling and other image acquisition imperfections yield edges that are blurred, with the degree of blurring being determined by factors such as the quality of the image acquisition system, the sampling rate, illumination conditions under which the image is required. As a result, edges are more closely modeled as having a ramplike profile, such as the one shown in **Fig.3.8(b)**.

The slope of the ramp is inversely proportional to the degree of the blurring in the edge. An *edge point* now is any point contained in the *ramp*, and an edge would then be a set of such points that are connected. The *thickness* of the edge is determined by the length of the ramp, as it transitions from an initial to the final gray-level. This length is determined by the slope, which, in turn, is determined by the degree of burring. This makes sense: blurred edges tend to be thick and sharp edges tend to be thin.

We are led to the conclusion that, to be classified as a meaningful edge point, the transition in gray-level associated with that point has to be significantly stronger than the background at that point. Since we are dealing with local computations, the method of choice to determine whether a value is significant or not is to use a threshold. Thus, we define a point in a fingerprint image as being an edge point if its two-dimensional first-order derivative is greater than a specified threshold. A set of such points that are connected according to a predefined criterion of connectedness is by definition an edge. The term edge segment generally is used if the edge is short in relation to the dimensions of the fingerprint image.

A key problem in segmentation is to assemble edge segments into longer edges. An alternate definition if we elect to use the second-derivative is simply to define the edge points in a fingerprint image as the zero crossings of its second derivative. The definition of an edge in this case is the same as above. The first-order derivatives in a fingerprint image are computed the gradient. Second-order derivatives are obtained using the Laplacian.

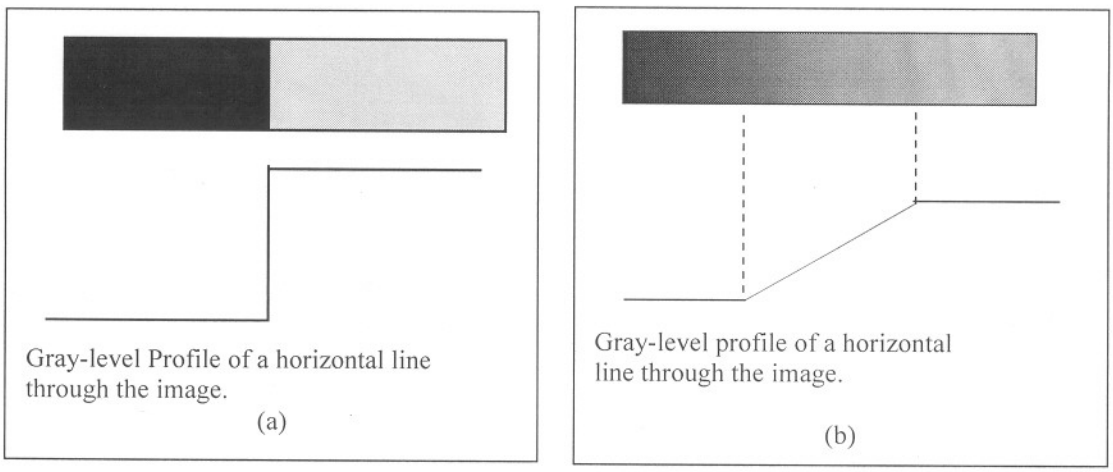


Fig.3.8: (a) Model of an Ideal Digital Edge. (b) Model of a ramp edge.

Gradient Operators:

First-order derivatives [52] of a fingerprint image are based on various approximations of the 2-D gradient. The gradient of an image $f(x, y)$ at location (x, y) is defined as the vector:

$$\nabla F = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \tag{3.13}$$

An important quantity in edge detection is the magnitude of this vector, denoted ∇f, where

$$\nabla f = mag(\nabla F) = \left[G_x^2 + G_y^2\right]^{1/2} \tag{3.14}$$

This quantity gives the maximum rate of increase of $f(x, y)$ per unit distance in the direction of ∇F. It is a common practice to refer to ∇f also as the *gradient*. The *direction* of the gradient vector also is an important quantity. Let $\alpha(x, y)$ represent the direction angle of the vector ∇F at (x, y). Then, from vector analysis,

$$\alpha(x, y) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{3.15}$$

where the angle is measured with respect to the x-axis. The direction of an edge at (x, y) is perpendicular to the direction of the gradient vector at that point. Computation of the gradient of a fingerprint image is based on obtaining the partial derivatives $\partial f / \partial x$ and $\partial f / \partial y$ at every pixel location. Let the 3 × 3 area shown in **Fig.3.9(a)** represent the gray levels in a neighbourhood of a fingerprint image. The simplest way to implement a first-order partial derivative at point z_5 is to use following *Roberts* cross-gradient operator:

$$G_x = (z_9 - z_5) \tag{3.16}$$

and

$$G_y = (z_8 - z_6) \tag{3.17}$$

The derivatives can be implemented for an entire fingerprint image by using the masks shown in **Fig.3.9(b)**. Masks of size 2 × 2 are difficult to implement because they do not have a clear centre. An approach using masks of size 3 × 3 is given by

$$G_x = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3) \tag{3.18}$$

and

$$G_y = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7) \tag{3.19}$$

In this formulation, the difference between the first and third rows of the 3 × 3 image region approximates the derivative in the x-direction, and the difference between the third

and first columns approximates the derivative in the y-direction. The masks shown in **Fig.3.9(d)** and **(e)**, called the *Prewitt* operators, can be used to implement these two

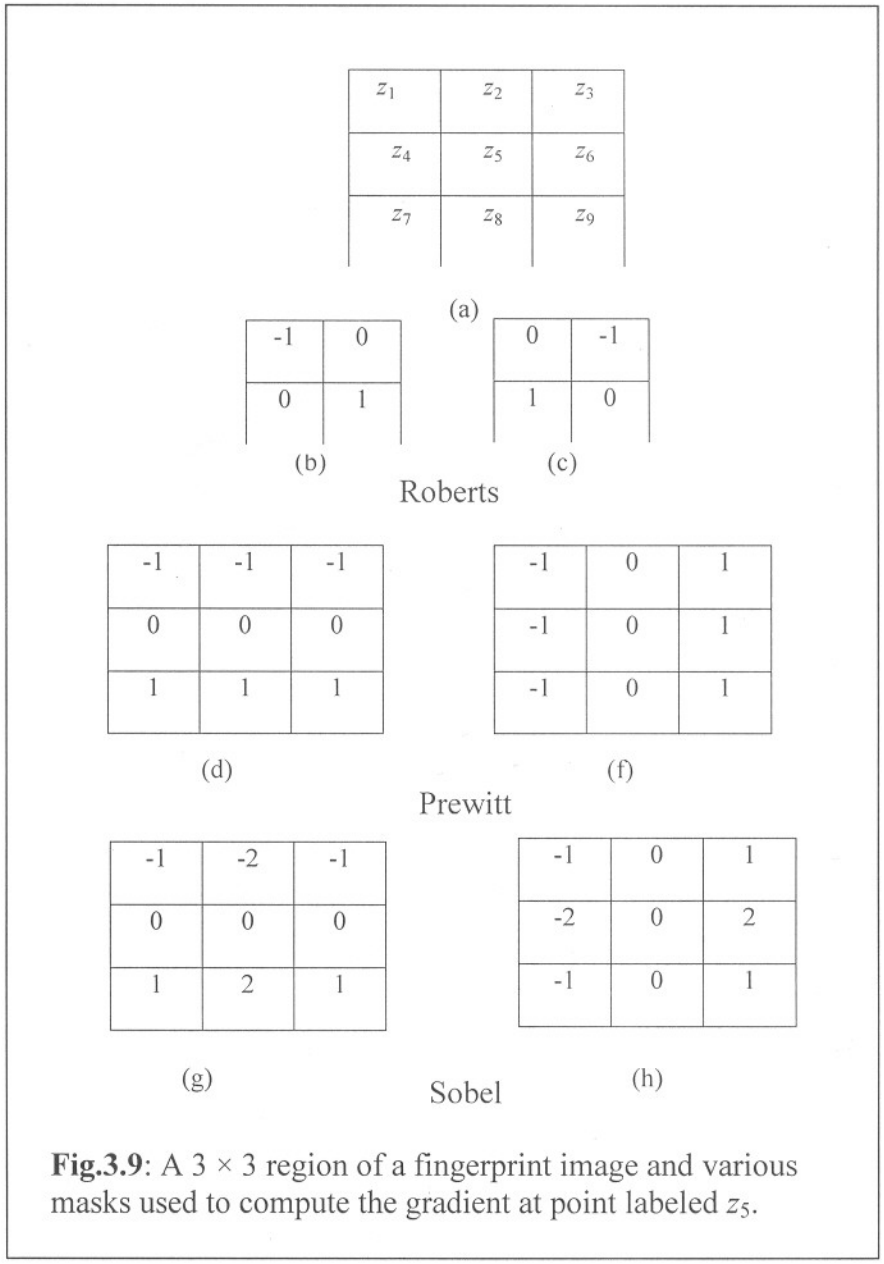


Fig.3.9: A 3 × 3 region of a fingerprint image and various masks used to compute the gradient at point labeled z_5.

equations. Now adding weight of 2 in the centre coefficient of **Eq.(3.18)** and **(3.19)**, We get

$$G_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3) \tag{3.20}$$

and

$$G_y = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7) \tag{3.21}$$

A weight value of 2 is used to achieve some smoothing by giving more importance to the center point. **Fig.3.9(f)** and **(g)**, called the *Sobel* operators, are used to implement these two equations. The Prewitt and Sobel operators are among the most used in practice for computing digital gradients.

The masks just discussed are used to obtain the gradient components G_x and G_y. An approach used frequency is to approximate the gradient by absolute values:

$$\nabla f \approx |G_x| + |G_y|. \tag{3.22}$$

The Laplacian

The Laplacian [52] of a 2-D function $f(x, y)$ is a second-order derivative defined in **Eq.(3.8)**. Digital approximations to the Laplacian are introduced in **Section 3.2.2.1.3.** For a 3 × 3 region, one of the two forms encountered most frequently in practice is

$$\nabla^2 f = 4z_5 - (z_2 + z_4 + z_6 + z_8) \tag{3.23}$$

where the z's are defined in **Fig.3.9(a)**. A digital approximation including the diagonal neighbours is given by

$$\nabla^2 f = 8z_5 - (z_1 + z_2 + z_3 + z_4 + z_5 + z_6 + z_7 + z_8 + z_9). \tag{3.24}$$

Masks for implementing these two equations are shown in **Fig.3.10**. In these masks the implementations of **Eq.(3.23)** and **(3.24)** are isotropic for rotation increments of 90^0 and 45^0, respectively.

0	-1	0
-1	4	-1
0	-1	0

(a)

-1	-1	-1
-1	8	-1
-1	-1	-1

(b)

Fig.3.10: Laplacian masks used to implement Eq.(3.23) and (3.24)

A screen shot of different fingerprint edge detection technique is shown in **Fig.3.11**.

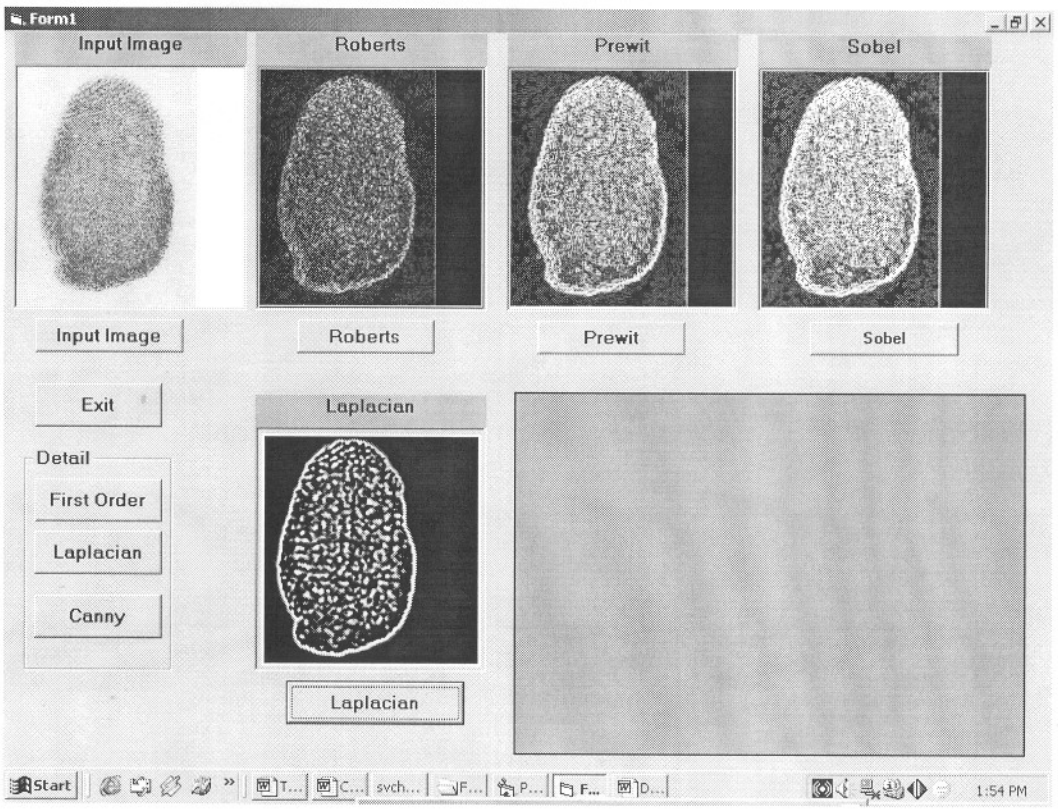


Fig.3.11: Fingerprint Edge Detection.

3.2.2.4 Thinning:

Thinning [52] is another important step to extract feature of fingerprint images. This approach representing the structural shape region is reduced like a graph. This reduction may be accomplished by obtaining the *skeleton* of the region via a thinning algorithm. Thinning algorithms iteratively delete edge points of a region subject to the constraints that deletion of these points:

(1) does not remove end points,

(2) does not break connectedness, and

(3) does not cause excessive erosion of the region.

In this section we present an algorithm for thinning binary regions [Gonzales]. Region points are assumed to have value 1 and background points to have value 0. The method consists of successive passes of two basic steps applied to the *contour points* of the given region, where a contour point is any pixel with value 1 and having at least one 8-neighbour valued 0. With reference to the *8-neighbourhood* definition shown in **Fig.3.12**, step 1 flags a contour point p for deletion if the following conditions are satisfied:

(a) $2 \leq N(p_1) \leq 6$;

(b) $S(p_1) = 1$;

(c) $p_2 . p_4 . p_6 = 0$; (3.25)

(d) $p_4 . p_6 . p_8 = 0$;

where $N(p_1)$ is the number of nonzero neighbors of p_1; that is,

$$N(p_1) = p_2 + p_3 + \ldots\ldots\ldots + p_8 + p_9$$

p_9	p_2	p_3
p_8	p_1	p_4
p_7	p_6	p_5

Fig.3.12: Neighborhood arrangement used by the thinning algorithm.

and $S(p_1)$ is the number of 0-1 transitions in the ordered sequence of $p_2, p_3, \ldots\ldots, p_8, p_9, p_2$. For example, $N(p_1) = 4$ and $S(p_1) = 3$ in **Fig. 3.13**.

In step 2, conditions (a) and (b) remain the same, but conditions (c) and are changed to

(c′) $p_2 . p_4 . p_8 = 0$;

(d′) $p_2 . p_6 . p_8 = 0$.

0	0	1
1	p_1	0
1	0	1

Fig.3.13: Illustration of Conditions (a) and (b) in Eq.(3.25). In this case $N(p_1) = 4$ and $S(p_1) = 3$.

Step 1 is applied to every border pixel in the binary region under consideration. If one or more of conditions (a)-(d) are violated, the value of the point in question is not changed. If all conditions are satisfied the point is flagged for deletion. However, the point is not deleted until all border points have been processed. This delay prevents changing the structure of the data ring execution of the algorithm. After step 1 has been applied to all border points, those that were flagged are deleted (changed to 0). Then, step 2 is applied to the resulting data in exactly the same manner as step 1. Thus one iteration of the thinning algorithm consists of

(1) Applying step 1 to flag border points for deletion;

(2) Deleting the flagged points;

(3) Applying step 2 to flag the remaining border points for deletion; and

(4) Deleting the flagged points.

This basic procedure is applied iteratively until no further points are deleted, at which time the algorithm terminates, yielding the skeleton of the region. Condition (a) is violated when contour point p_1 has only one or seven 8-neighbors valued 1. Having only one such neighbor implies that p_1 is the end point of a skeleton stroke and obviously should not be deleted. Deleting p_1 if had seven such neighbors would cause erosion into the region. Condition (b) is violated when it is applied to points on a stroke 1 pixel thick. Hence this condition prevents disconnection of segments of a skeleton during the thinning operation. Conditions (c) and (d) are satisfied simultaneously by the minimum

set of values: ($p4$ = 0 or p_6 = 0) or (p_2 = 0 and p_8 = 0). Thus with reference the neighborhood arrangement in **Fig.3.12**, a point that satisfies these conditions, as well as conditions (a) and (b), is an east or south boundary point a northwest corner point in the boundary. In either case, p_1 is not part of the skeleton and should be removed. Similarly conditions (c') and (d') are satisfied simultaneously by the following minimum set of values: (p_2 = 0 or p_8 = 0) or (p_4 = 0 and p_6 = 0). These correspond to north or west boundary or southeast corner point. It is to be noted that the northeast corner points have p_2 = 0 and p_4=0 and thus satisfies the condition (c) and (d) as well as (c´) and (d´). The same is true for southwest corner points which have p_6 = 0 and p_8 = 0. A screen shot of fingerprint image thinning is shown in **Fig.3.14**.

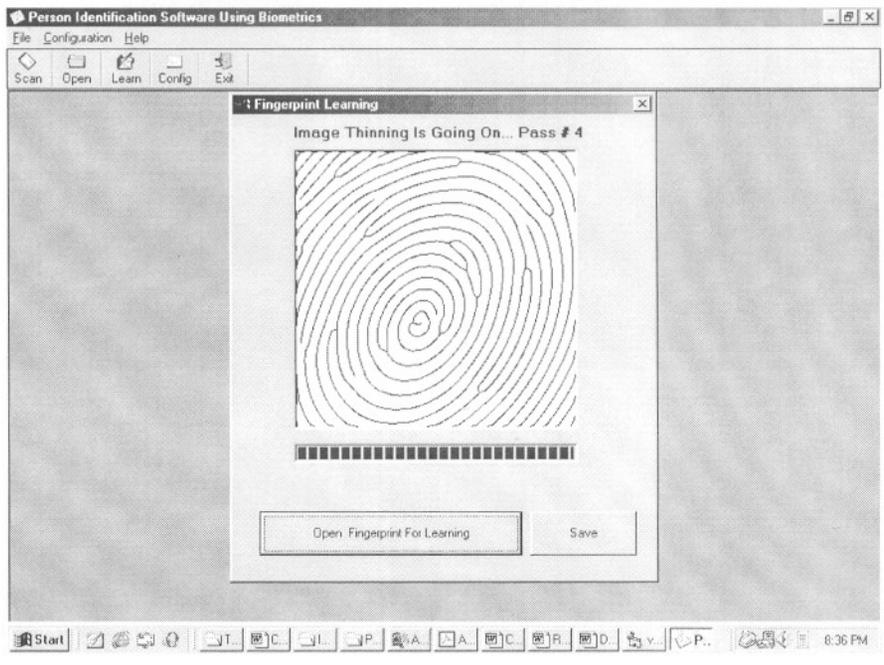


Fig.3.14: Fingerprint Image Thinning

3.3 Grid Mapping Feature (GMF) Extraction:

From the above discussion, we have acknowledged about that the overall process to improve the quality of poor fingerprint images. In this section *GMF* extraction process has been described. The poor fingerprint images are processed by using filtering, clipping

and edge detection. The following process has been adopted to extract feature for fingerprint images. The fingerprint image is transformed to 300X260-pixel image by using image-scaling process. Then the fingerprint is processed through grid mapping with fixed square/rectangle cells, such as 16x16 or 32X32 square/rectangle areas (**Fig.3.15**). If each small square/rectangle cell contains more than 45~55% black pixels, then its digital value is considered as 1 otherwise 0. An extraction algorithm is used to extract grid mapping features from the gray scale fingerprint image by examining the neighborhood pixels. These binary values are used to train the Multilayer neural network using BackPropagation Algorithm. The screen shot of the GMF Extraction is shown in **Fig.3.16**.

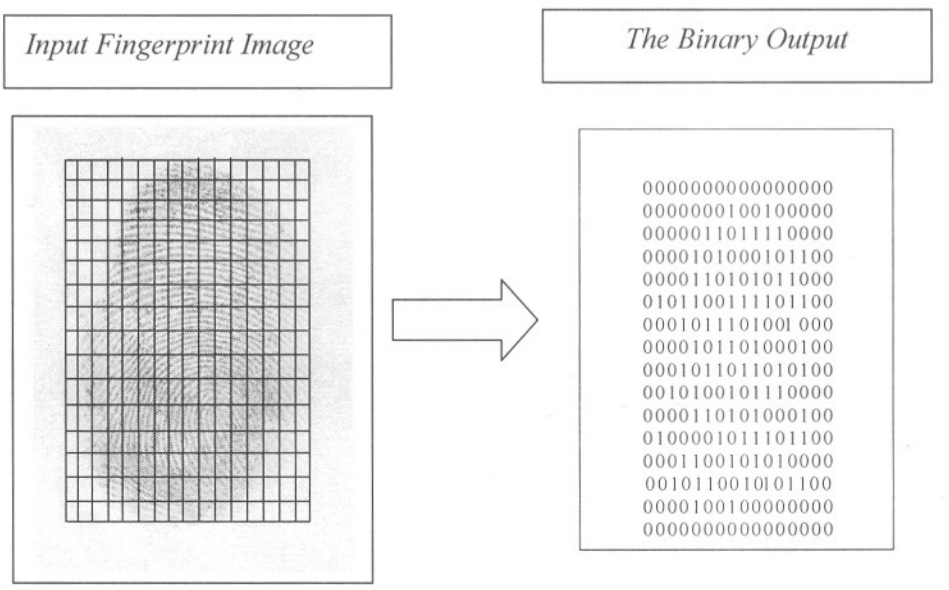


Fig.3.15: GMF extraction from a sample fingerprint image of 16 X 16 Feature Matrixes.

3.4 GMF Extraction Algorithm:

In this section, we describe the algorithm to extract the feature of the fingerprint images.

Step 1: Input the image of a fingerprint.

Step2: Define the number of m x m matrices.

Step3: Calculate the binary value for each cell.

A. Determine each pixel's RGB value and get the sum of RGB value.
B. Check whether or not the sum of RGB value is near the RGB value of BLACK pixel.
C. If the sum of RGB value is near the RGB value of BLACK pixel then increase the Index counter.
D. Repeat steps A, B, C for each pixel of the cell.
E. If the number of the black-pixel is above 45~55% then put the binary value 1 otherwise 0.
F. Return binary value of the cell.

Step 4: Put the cell value into a file.

Step 5: Repeat step 3 & 4 for each cell.

Step 6: Exit.

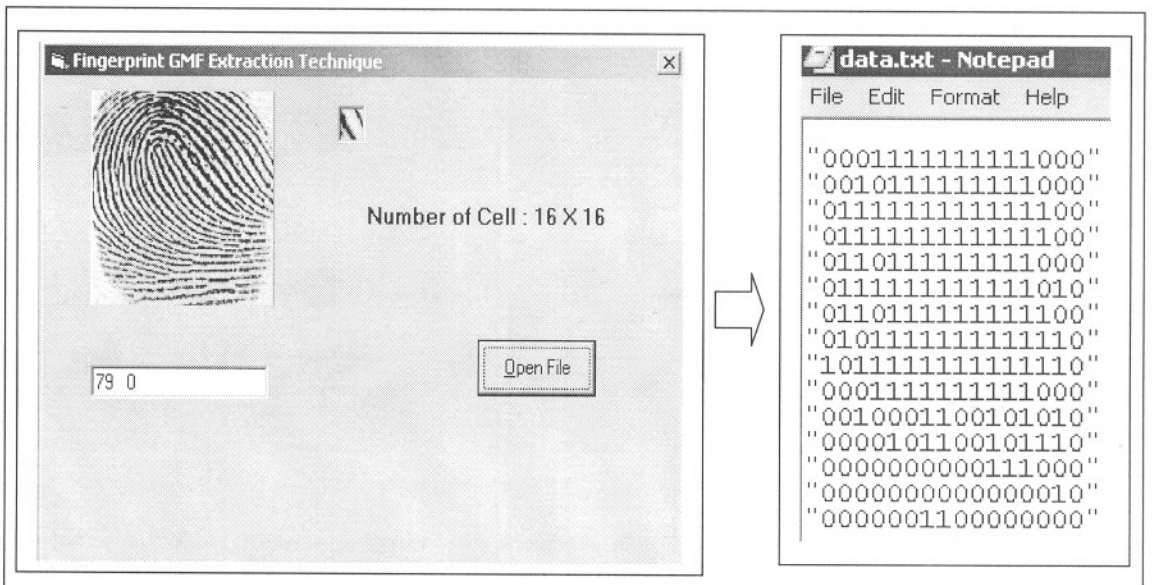


Fig.3.16: Screen shot of GMF extraction from a sample fingerprint image of 16 X 16 Feature Matrixes.

It is clear that the GMF extraction process is to be quite easy but lengthy. We also observed that the output of the process is the binary number. So it can be easily use to train any types of matching algorithm to find the exact person. Specially, for the criminal investigation, it may be very much effective to extract features of the poor fingerprint images. The dataflow diagram of GMF extraction technique is shown in **Fig.3.17**.

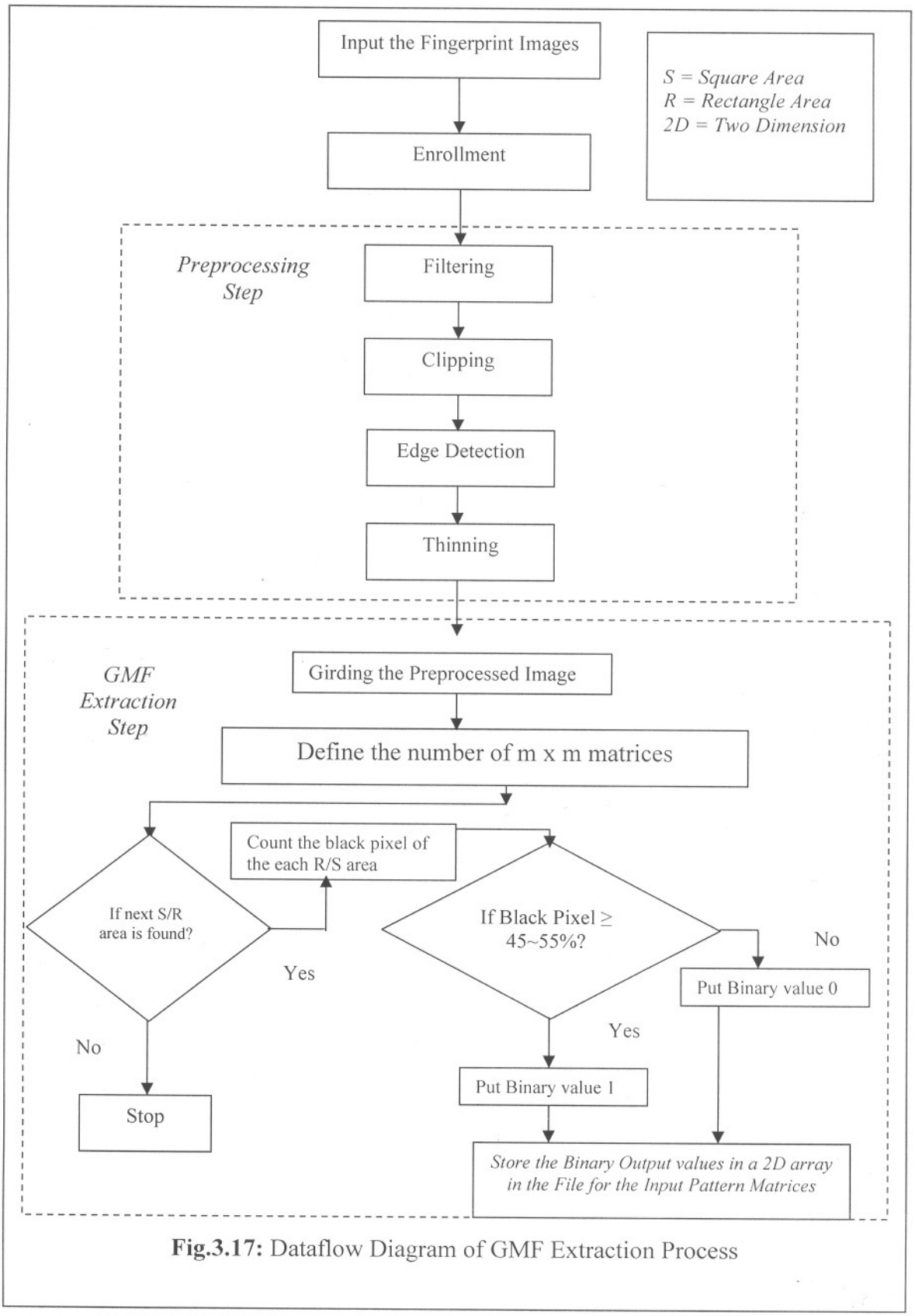


Fig.3.17: Dataflow Diagram of GMF Extraction Process

Chapter Four

FINGERPRINT CLASSIFICATION TECHNIQUES

Contents **Page No.**

4.1 Introduction:

Several approaches have been developed for off-line fingerprint classification. These approaches can be broadly categorized into *five main* categories: (i) knowledge-based, (ii) structure-based, (iii) frequency-based, (iv) syntactic-based and (v) *special gray-level value-based.* The knowledge-based [10] fingerprint classification technique uses the locations of singular points (core and delta) to classify a fingerprint into the five above-mentioned classes. A knowledge-based approach tries to capture the knowledge of a human expert by deriving rules for each category by hand-constructing the models and therefore, does not require training. A structure-based approach [10] uses the estimated orientation field in a fingerprint image to classify the fingerprint into one of the five classes. A frequency-based approach [10] uses the frequency spectrum of the fingerprints for classification. A syntactic approach uses a formal grammar to represent and classify fingerprints. *Finally, we propose a new category approaches special gray-level value-based, which is based on the special value of the pixels of the fingerprint images for classification.*

Fingerprint classification literature survey [10] is shown in **Table-4.1**. The number of classes is denoted by C, the classification accuracy is denoted by Acc, and the reject rate is denoted by RR. The classification accuracies reported by the different authors are on different databases with different number of fingerprints and therefore, they cannot be

directly compared. Most of the work in fingerprint classification is based on supervised learning and discrete class assignment using knowledge-based features [10].

Table-4.1: Fingerprint classification literature survey.

Author	C	Features	Method	Acc. (RR)
Kawagoe and Tojo 1984	7	Singular points	Rule-based	91.5% (0%)
Blue et al. 1994	5	Orientation field	Neural network	92.8% (0%)
Wilson et al. 1994	5	Orientation field	Neural network	90.2% (10%)
Candela et al. 1995	6	Orientation field	Neural network	92.2% (0%)
Pal and Mitra 1996	5	Orientation field	Neural network	82+% (0%)
Fitz and Green 1996	3	FFT	Nearest-neighbor	85% (0%)
Karu and Jain 1996	5	Singular points	Rule-based	85% (0%)
Senior 1997	4	Ridge lines	Hidden Markov Model	90% (0%)
Chong et al. 1997	5	Ridge lines	Rule-based	96.5% (0%)
Hong and Jain 1999	5	Singular points and ridge lines	Rule-based	87.5% (0%)

*In this work we propose a fingerprint classification algorithm (**Fig.4.1**) based on GMF of fingerprint representation which is derived from special gray-level values of pixels of the fingerprint images. It is more capable of tolerating poor image quality, which is a major difficulty in fingerprint classification.*

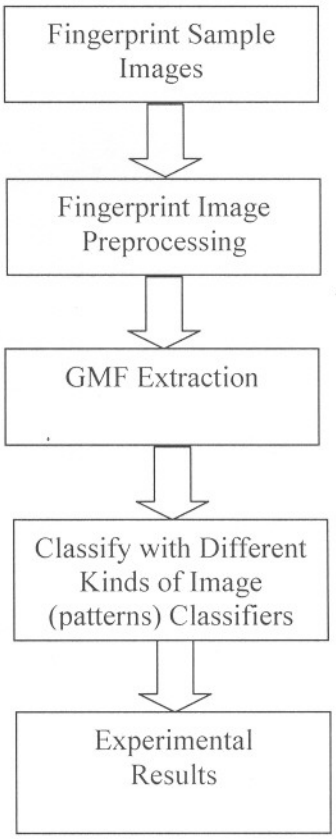


Fig.4.1: Dataflow Diagram of Off-line Fingerprint Classification Algorithm

4.2 Classification Techniques

There are different kinds of fingerprint classification techniques presently used world wide. Out of them most common and recent invent fingerprint classification techniques are as follows:

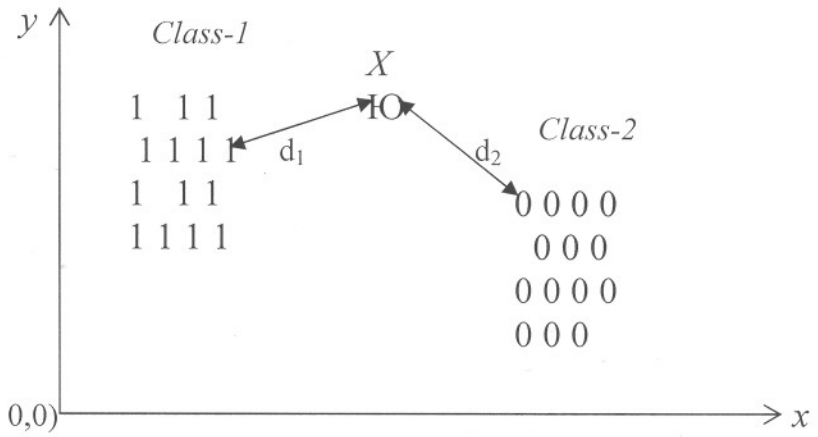
- Minimum Distance Classifier
- K- Nearest Neighbour Classifier
- Support Vector Machine Classifier
- Neural Network as a Classifier

In this research we emphasis on Neural Network as a Classifier. Before we discuss about the artificial neural network, we start from the minimum distance classifier.

4.3 Minimum Distance Classifier

Fingerprint classification by distance functions is one of the earliest concepts in fingerprint recognition. For this technique, we have considered the diagram of **Fig.4.2**. This is a two class representation in the fingerprint features (patterns) space and we decide what class in between the two classes an unclassified feature (pattern), *X*, belongs to. Nearest neighboring class techniques [54, 55, 57], in essence, make a decision based on the shortest distance to the neighbouring class samples, they assign it to whichever class it appears to be closest to not an unreasonable assumption. Formally, that defines a discriminant function $\int(X)$ by:

$$\int(X) = closest(class1) - closest(class2) \tag{4.1}$$



- *X* (HO) Unclassified Class of Fingerprint
- d_1 Shortest Distance of Class-1
- d_2 Shortest Distance of Class-2

Fig.4.2 Classification by Comparison to the Nearest Neighbour Classes

Several methods are used to measure these distances. Some of them describe bellow.

4.3.1 Euclidean Distance

The shortest distance [55, 56] in *n*-dimensional feature space, which is the usual distance between two feature vectors (points) $a_i = (a_1, a_2, ..., a_n)$ and $b_i = (b_1, b_2, ..., b_n)$, is

defined by,

$$d_{euc}(a,b) = \sqrt{\sum_{i=0}^{n}(b_i - a_i)^2} \tag{4.2}$$

4.3.2 Hamming Distance

The most basic measure, and one that is widely used because of its simplicity, is the Hamming distance [55] measure. This distance is measured by evaluating the difference between each component of one vector with the corresponding component of the other, and summing these differences provides an absolute value for the variation between the two vectors. It is defined by

$$d_H(a,b) = \sum_{i=1}^{n}(|a_i - b_i|) \tag{4.3}$$

4.3.3 Mahalanobis Distance

The Mahalanobis distance [57] is measured by the following equation

$$d_M = \left((x-\mu_i)^T \Sigma^{-1}(x-\mu_i)^{1/2}\right) \tag{4.4}$$

where x is the feature vectors in the n-dimensional and the $\mu_i = E[x]$ is the mean value of the ω_i class and Σ_i the $n \times n$ covariance matrix defined as

$$\Sigma_i = E[(x-\mu_i)(x-\mu_i)^T] \tag{4.5}$$

The covariance matrix is symmetric and it is always be diagonalized by a unitary transform

$$\Sigma = \Phi\Lambda\Phi^T \tag{4.6}$$

where $\Phi^T = \Phi^{-T}$ and Λ is the diagonal matrix whose elements are the eigenvalues of Σ. Φ has as its columns the corresponding eigenvectors of Σ

$$\Phi = [v_1, v_2, ..., v_n) \tag{4.7}$$

Combining (4.4) and (4.5), we obtain

$$(x-\mu_i)^T \Phi\Lambda^{-1}\Phi^T(x-\mu_i) = c^2 \tag{4.8}$$

where $d_M = c$ (constant distance). Let $x' = \Phi^T x$. The coordinates of x' are equal to $v_k^T x, k = 1,2,3,....., n$ that is, the projections of x onto the eigenvectors. In other words,

they are the coordinates of x with respect to a new coordinate system whose axes are determined by $v_k, k = 1,2,3,\ldots\ldots\ldots, n$. Equation (4.8) now can be written as

$$\frac{(x_1' - \mu_{in}')^2}{\lambda_1} + \ldots\ldots + \frac{(x_l' - \mu_{in}'}{\lambda_n} = c^2 \qquad (4.9)$$

This is the equation of a hyperellipsoid in the new coordinate system. **Fig.4.3** shows the $n = 2$ case. The centre of mass of the ellipse is at μ_i, and the principal axes are aligned with the corresponding eigenvectors and have lengths $2\sqrt{\lambda_k c}$, respectively. Thus, all points having the same distance from a specific point are located on an ellipse.

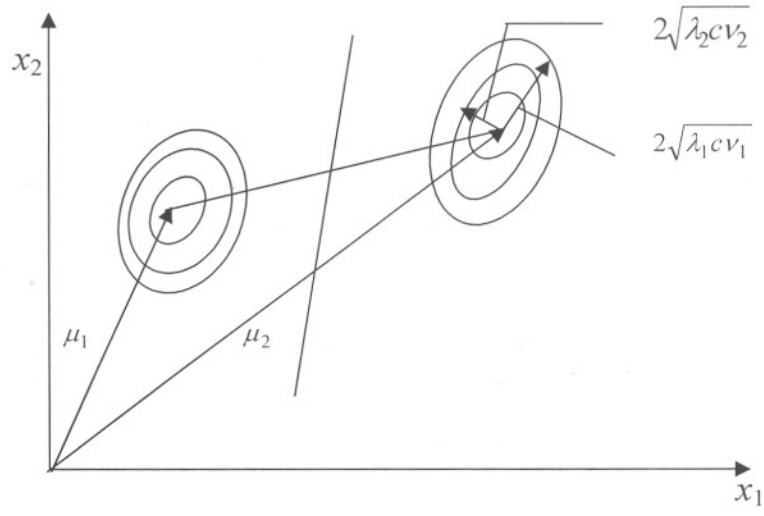


Fig.4.3: The Mahalanobis Distance from the mean points of each class

4.4 K- Nearest Neighbour Classifier

In this method the number of nearest neighbour points [57] of *GMF features* of a fingerprint image $k_N = k$ is to be fixed and the size of the volume around feature vector x is to be adjusted each time, to include k points. Thus, in low-density areas the volume is

to be large and in high-density areas it is to be small. We can also consider more general types of regions, besides the hypercube. The estimator can now be written as-

$$\hat{p}(x) = \frac{k}{NV(x)} \tag{4.10}$$

where the dependence of the volume $V(x)$ on x is explicitly shown. Again it can be shown that asymptotically ($\lim k = +\infty$, $\lim N = +\infty$, $\lim(k/n) = 0$) this is an unbiased and consistent estimate and it is known as the *k Nearest Neighbour* (kNN) density estimate. So that the results concerning on the finite k and finite total number of features N (**Fig.4.4**). The algorithm for the k-nearest neighbour is summarized [57] as follows. Given an unknown feature vector of a fingerprint image x and a distance measure, then:

- *Step-1*: Out of the N training vectors, identify the k nearest neighbours, irrespective of class label. k is chosen to be odd for a two class problem, and in general not to be a multiple of the number of classes M.
- *Step-2*: Out of these k samples, identify the number of the vectors, k_i, that belong to class $\omega_i, i = 1,2,3,.....,M$. Obviously, $\sum_i k_i = k$.
- Assign x to the class ω_i with the maximum number k_i of samples.

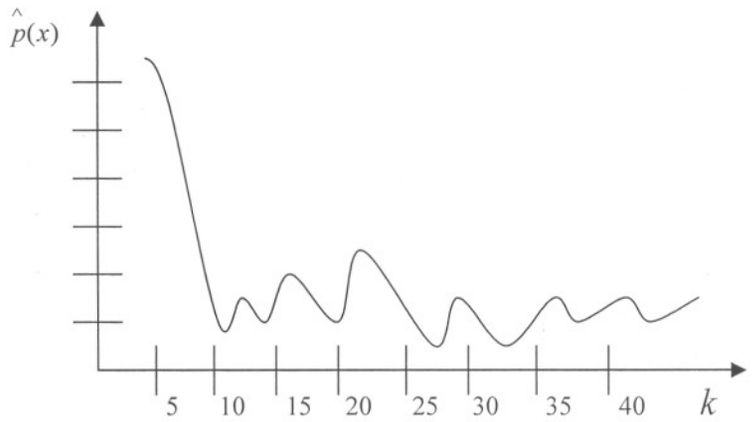


Fig.4.4: k versus classification probability for the k-Nearest Neighbour classifier

4.5 Support Vector Machine (SVM) Classifier

In this section an alternative rationale for designing linear classifiers is to be adopted for GMF of a fingerprint image. It starts with the two-class linearly separable task and then it extends the method to more general cases where data are not separable [57]. Let x_i, i = 1, 2, 3, ..., N, be the feature vectors of the training set, X. These belong to either of the two classes ω_1 and ω_2 which are assumed to be linearly separable. The goal, once more, is to design a hyperplane

$$g(x) = \omega^T(x) + \omega_0 \tag{4.11}$$

that classifies correctly all the training vectors (**Fig.4.5**). $g(x)$ is a linear function of x and the respective decision surfaces are *hyperplanes*.

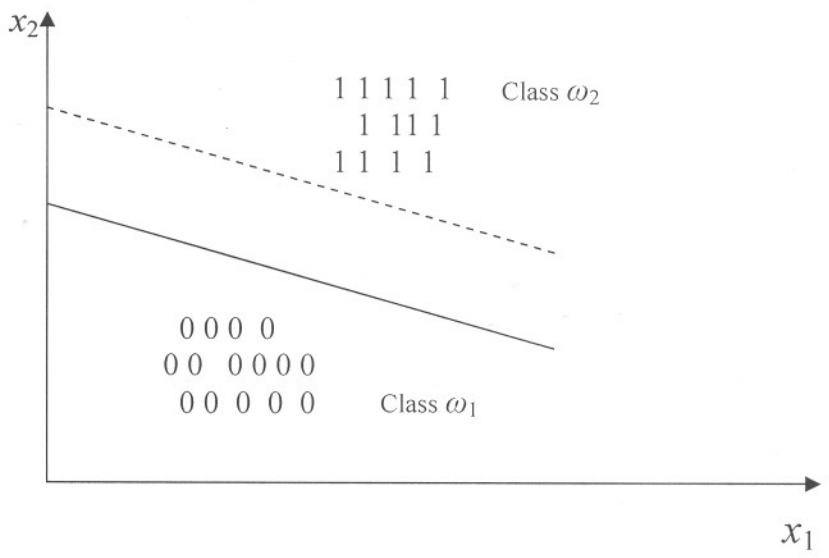


Fig.4.5: An example of a linearly separable two pattern class problem with two possible liner classifiers

Let us now quantify the term *margin* that a hyperplane leaves from both classes. Every hyperplane is characterized by its direction (determined by ω) and its exact position in space (determined by ω_0). Since we want to give no preference to either of the classes,

then it is reasonable for each direction to select that hyperplane which has the same distance from the respective nearest points in ω_1 and ω_2. This is illustrated in **Fig.4.6**.

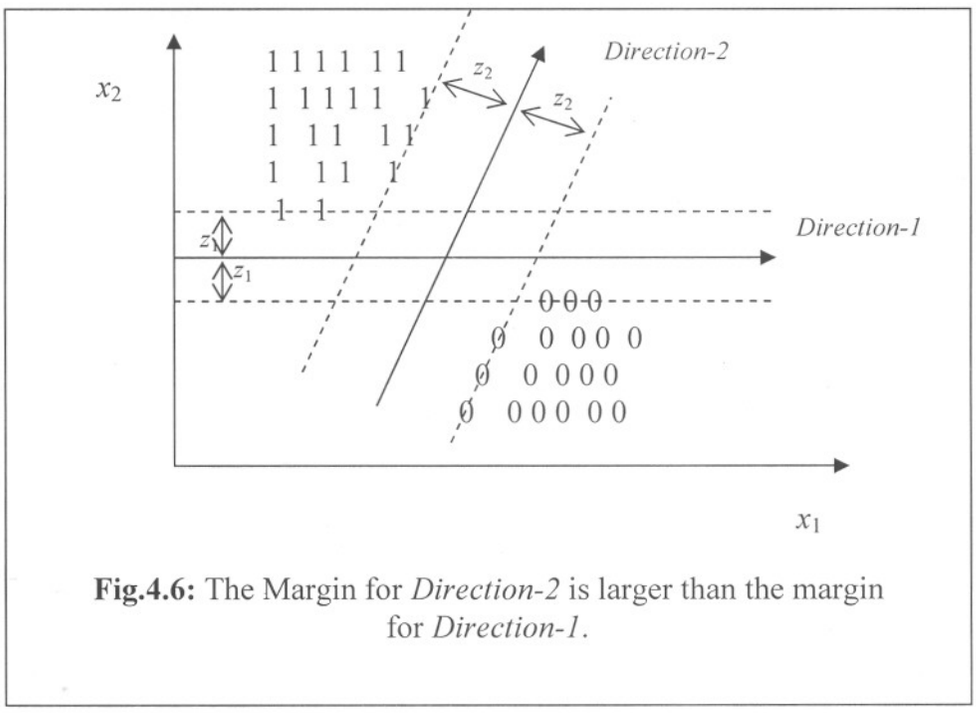


Fig.4.6: The Margin for *Direction-2* is larger than the margin for *Direction-1*.

The hyperplanes shown with dark lines are the selected ones from the infinite set in the respective direction. The margin for *direction-1* is $2z_1$ and the margin for *direction-2* is $2z_2$. Our goal is to search for the direction that gives the maximum possible margin. However, each hyperplane is determined within a scaling factor. The distance of a reference point from a hyperplane [57] is given by

$$z = \frac{|g(x)|}{\|\omega\|} \tag{4.12}$$

We can now scale ω, ω_0 so that the value of g(x), at the nearest points in ω_1, ω_2. This is equivalent [57] to

1. Having a margin of $\frac{1}{\|\omega\|} + \frac{1}{\|\omega\|} = \frac{2}{\|\omega\|}$

2. Requiring that $\omega^T x+\omega_0 \geq 1, \quad \forall x \in \omega_1$

$$\omega^T x+\omega_0 \leq -1, \quad \forall x \in \omega_2$$

For each x_i, we denote the corresponding class indicator by y_i (+1 for ω_1, -1 for ω_2.) Our task can now be summarized as: Compute the parameters ω, ω_0 of the hyperplane so as to

$$\text{minimize} \quad J(\omega) \equiv \frac{1}{2}\|\omega\|^2 \tag{4.13}$$

$$\text{subject to} \quad y_i(\omega^T x_i + \omega_0) \geq 1, \qquad i = 1,2,3,\ldots\ldots, N \tag{4.14}$$

Obviously, minimizing the norm makes the margin maximum. This is a nonlinear (quadratic) optimization task subject to a set of linear inequality constraints. The Karush-Kuhn_Tucker (KKT) conditions that the minimizer of Eq.**(4.13)** and **(4.14)** has to satisfy are

$$\frac{\partial}{\partial \omega} L(\omega, \omega_0, \lambda) = 0 \tag{4.15}$$

$$\frac{\partial}{\partial \omega_0} L(\omega, \omega_0, \lambda) = 0 \tag{4.16}$$

$$\lambda_i \geq 0, \quad i = 1,2,3,\ldots\ldots, N \tag{4.17}$$

$$\lambda_i[y_i(\omega^T x_i + \omega_0) - 1] = 0, \qquad i = 1,2,3,\ldots\ldots, N \tag{4.18}$$

where λ is the vector of the Langrange multipliers, λ_i, and $L(\omega, \omega_0, \lambda)$ is the Langrangian function defined as

$$L(\omega, \omega_0, \lambda) = \frac{1}{2}\omega^T\omega - \sum_{i=1}^{N} \lambda_i[y_i(\omega^T x_i + \omega_0) - 1] \tag{4.19}$$

Combining (4.15), (4.16), and (4.19) results in

$$\omega = \sum_{i=1}^{N} \lambda_i y_i x_i \tag{4.20}$$

$$\text{and} \qquad \sum_{i=1}^{N} \lambda_i y_i = 0 \tag{4.21}$$

The Langrange multipliers can be either zero or positive. Thus, the vector parameter ω of the optimal solution is a linear combination of $N_S \leq N$ feature vectors which are associated with $\lambda_i \neq 0$. That is,

$$\omega = \sum_{i=1}^{N_S} \lambda_i y_i x_i \tag{4.22}$$

These are known as *support vectors* and the optimum hyperplane classifier as a *support vector machine* (SVM).

4.6 Neural Network as a Classifier

Neural Network is a powerful tool used in modern intelligent systems. Nowadays, many applications that involve pattern recognition, Fingerprint identification, feature mapping, clustering, classification and etc. use Neural Networks [58, 59, 60] as an essential component. In recent decades, several types of neural networks have been developed. Error Back Propagation, Kohonen feature map and Hopfield network are some of basic networks that have been developed and are used in many applications.

One common scheme for classifying neural networks is based on whether the learning is done with the aid of a teacher (supervised learning, which is described in *chapter-5*) or without such assistance (unsupervised learning, which is described in *chapter-6*). Our intention is to summarize concepts related to neural networks and give detail to improve some understanding of what can be accomplished with neural net models and how these models are developed and are used to classify different patterns (Fingerprint images).

In this thesis general Multi Layer Perceptron or Error BackPropagation Neural Network is presented. The sample application uses this Multi Layer Perceptron and classifies the features of fingerprint images. In ***chapter-5***, Multi Layer Perceptron or Back propagation Neural Network is described widely, and in ***chapter-8***, a fingerprint image classifying network has been developed and to implement to classify fingerprint images from twenty different person's fingerprint images into their corresponding ID's, using *Minimum Distance Error Rate Back propagation Neural Network* [53].

Chapter Five

NEURAL NETWORK MODELS FOR FINGERPRINT IDENTIFICATION

Contents **Page No.**

5.1 Introduction

Artificial neural network is a very much powerful as well as well-established method to use as pattern learning, matching and classifying technique in the field of computation. Neural Networks are biologically motivated and statistically based. Neural networks are known for their ability to make rapid memory associations rather than for high-precision computation processing. Neural nets turn out to be especially good at sensor-related pattern recognition problems such as vision and speech-processing. They are also excellent for a variety of pattern matching problems as we consider for fingerprint matching as sample, in this research.

A neural network is a network wherein each node models a very simplified neuron. Each arc is treated as a connection, and the node computes by responding to its inputs. A general definition of neural computing is given bellow:

"Neural computing is the study of networks of adaptable, nodes which, through a process of learning from task examples, store experiential knowledge and make it available for use". This definition is such that the neural nets of the living brain are included in the field of study. The nodes of the brain are adaptable, they acquire, knowledge through changes in the function of the node by being exposed to examples. Admittedly, very little is known about the details of how all these happen, but the above definition deliberately provides room for a consideration of biological findings.

Neural networks [61] are information processing systems. In general, they can be thought of as "**black-box**" devices that accept inputs and produce outputs. Some of the operations that neural network performs given bellow:

(i). *Classification*: An input pattern is passed to the network, and the network produces a representative class as output.

(ii). *Pattern Matching*: An input pattern is passed to the network and the network produces the corresponding output pattern.

(iii). *Pattern Completion*: An incomplete pattern is passed to the network and the network produces an output pattern that has the missing portions of the input pattern filled in.

(iv). *Noise Removal*: A noise corrupted input pattern is passed to the network and the network removes some (or all) of the noise and produces a cleaner version of the input pattern as the output.

(v). *Optimization*: An input pattern representing the initial values for a specific optimization problem is presented to the network and the network produces a set of variables that represents a solution to the problem.

(vi). *Control*: An input pattern represents the current state of a controller and the desired response for the controller and the output is the proper command sequence that will create the desired response.

5.2 The Neural Networks

Artificial Neural Network (ANN) are basically of two [58, 62] types (**Fig.5.1**), (1). Supervised Neural Network(SNN) and (2). Unsupervised Neural Network (UNN) and another one is Reinforcement Learning. In this research work, we emphasis on both Supervised and Unsupervised neural network. Specially, we use supervised neural network for training, learning and testing for the fingerprint recognition. An algorithm has been developed for this purpose, which is described in ***section- 5.6.***

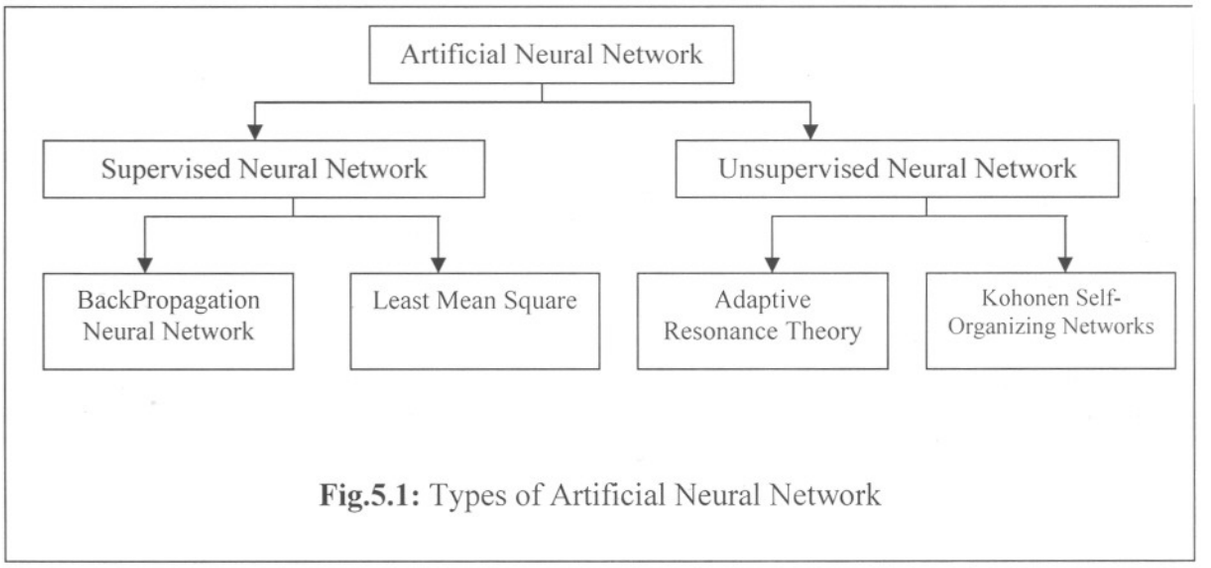


Fig.5.1: Types of Artificial Neural Network

5.2.1 Supervised Neural Network (SNN):

Supervised Neural Network (SNN) is used to implement the off-line fingerprint identification system. In supervised neural network [58, 62], the training data consist of many pairs of input/output training patterns. Therefore, the learning is benefited from the assistance of a teacher (**Fig.5.2**). Given a new training pattern, say, (m+1)th, the weights may be updated as follows:

$$w_{ij}^{(m+1)} = w_{ij}^{m} + \Delta w_{ij}^{m} \qquad (5.1)$$

To train the supervised neural network for fingerprint identification system we use the BackPropagation algorithm.

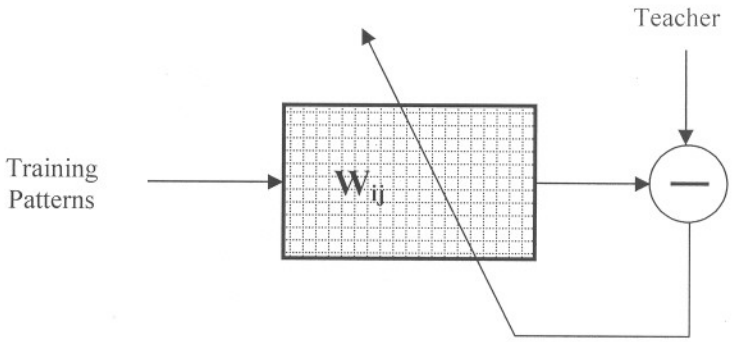


Fig.5.2: Block Diagram of a Supervised Neural Network

5.2.2 Unsupervised Neural Network (UNN)

For an unsupervised learning rule [58, 59, 60, 62], the training sets consist of input training patterns only. Therefore, the network is trained without benefit of any teacher (**Fig.5.3**). The network learns to adapt based on the experiences collected through the previous training patterns. Here is a typical schema of an unsupervised system:

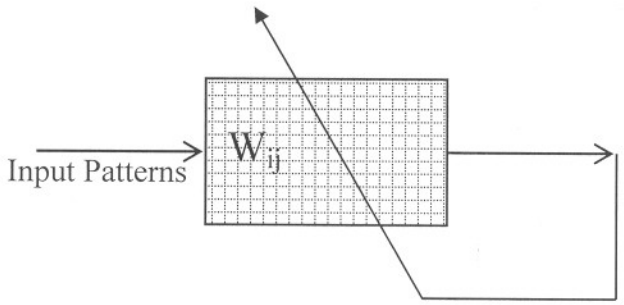


Fig.5.3: Block Diagram of an Unsupervised Neural Network

Typical examples are the Hebbian learning rule, the competitive learning rule, Adaptive Resonance Theory (ART) and Kohonen Self-Organizing Networks. A simple version of Hebbian learning rule is that when unit *i* and unit *j* are simultaneously excited, the strength of the connection between them increases in proportion to the product of their activations.

5.2.3 Reinforcement Learning

The last type of learning is reinforcement learning where only grades of good or bad are presented to the network during training. The recall mode is the process of finding how well the network has learned. Often the network will be presented with a different set of data from the training data. The outputs produced by the neural network are compared against the desired outputs to determine the network's performance.

5.3 The learning rule

The learning rule for multilayer perceptrons is called the "generalized delta rule" or the "BackPropagation rule"[60], and was suggested in 1986 by *Rumelhart*, *McClelland* and *Williams* [91]. It signaled the renaissance of the whole subject, it was later found that *Parker* had published similar results in 1982, and then *Werbons* was shown to have done the work in 1974. Such is the nature of science, however; groups working in diverse fields cannot keep up with all the advances in other areas, and so there is often duplication of effort. However, *Rumelhart* and *McClelland* [91] are credited with reviving the perceptron since they have not only developed the rule independently to the earlier claims, but used it to produce multilayer networks that they investigated and characterized [91].

The operation of the multilayer network is compared with the desired response enables the weights to be altered so that the network can produce a more accurate output next time. The learning rule provides the method for adjusting the weights in the network. However, the use of sigmoid function means that enough information about the output is

available to units in earlier layers, so that these units can have their weights adjusted so as to decrease the error next time.

The learning rule is a little more complex, however we can best understand it by considering how the net behaves as patterns are taught to it. When we show the untrained network an input pattern (known fingerprint of a person), it will produce any random output. We need to define an error function that represents the difference between the network's current output and the correct output that we want it to produce. Because we need to know the correct pattern (fingerprint of a person), this type of learning is known as supervised learning. In order to learn successfully we want to make the output of the net approach the desired output, that is, we want to reduce the value of this error function continuously. This is achieved by adjusting the weights on the links between the units, and the generalized delta rule does this by calculating the value of the error function for that particular input, and then back-propagation (hence the name!) the error from one layer to the previous one. Each unit in the net has its weights adjusted so that it reduces the value of the error function. For units actually on the output, their output and the desired output is known, so adjustment of the weights is relatively simple. But for the units in the middle layer, the adjustment is not so obvious. Intuitively, we might guess that the hidden units that are connected to outputs with a large error should have their weights adjusted a lot, while units that feed almost correct outputs should not be altered much. In fact, the mathematics shows that the weights for a particular node should be adjusted in direct proportion to the error in the units to which it is connected: that is why back-propagating these errors through the net allow the weights between all the layers to be correctly adjusted. In this way the error function is reduced and the network learns.

5.3.1 The BackPropagation Algorithm

The notation used for this algorithm [60, 62] is as follows: E_p is the error function for pattern p, t_{pj} represents the target output for pattern p on node j, whilst o_{pj} represents the actual output at that node. ω_{ij} is the weight from node i to node j.

Let us define the error function to be proportional to the square of the difference between the actual and desired output, for all the patterns to be learnt.

$$E_p = \frac{1}{2}\sum_j (t_{pj} - o_{pj})^2 \tag{5.2}$$

The ½ makes the maths a bit simpler, and brings this specific error function into line with other similar measures. The activation of each unit j, for pattern p, can be written as

$$net_{pj} = \sum_i \omega_{ij} o_{pi} \tag{5.3}$$

i.e. simply the weighted sum, as in the single-layer perceptron.

The output from each unit j is the threshold function f_j acting on the weighted sum. In the perceptron, this was the step function; in the multilayer perceptron, it is usually the sigmoid function, although any continuously differentiable monotonic function can be used.

$$o_{pj} = f_i(net_{pj}) \tag{5.4}$$

We can write

$$\frac{\partial E_p}{\partial \omega_{ij}} = \frac{\partial E_p}{\partial net_{pj}} \frac{\partial net_{pj}}{\partial \omega_{ij}} \tag{5.5}$$

by the chain rule.

Looking at the second term in **Eq.(5.5)**, and substituting in **Eq.(5.3)**,

$$\frac{\partial net_{pj}}{\partial \omega_{ij}} = \frac{\partial}{\partial \omega_{ij}} \sum_k \omega_{kj} o_{pk} = \sum_k \frac{\partial \omega_{kj}}{\partial \omega_{ij}} o_{pk} = o_{pi} \tag{5.6}$$

Since $\frac{\partial \omega_{kj}}{\partial \omega_{ij}} = 0$ except when it equals 1.

We can define the change in error as a function of the change in the net inputs to a unit as

$$-\frac{\partial E_p}{\partial net_{pj}} = \delta_{pj} \tag{5.7}$$

and so **Eq.(5.5)** becomes

$$-\frac{\partial E_p}{\partial \omega_{ij}} = \delta_{pj} o_{pi} \tag{5.8}$$

Decreasing the value of E_p therefore means making the weight changes proportional to $\delta_{pj} o_{pi}$, i.e.

$$\Delta_p \omega_{ij} = \eta \delta_{pj} o_{pi} \tag{5.9}$$

We now need to know what δ_{pj} is for each of the units if we know this then we can decrease E. Using **Eq.(5.7)** and the chain rule, we can write

$$\delta_{pj} = -\frac{\partial E_p}{\partial net_{pj}} = \frac{\partial E_p}{\partial o_{pj}} \frac{\partial o_{pj}}{\partial net_{pj}} \tag{5.10}$$

Considering the second term, from **Eq.(5.4)**,

$$\frac{\partial o_{pj}}{\partial net_{pj}} = f_j'(net_{pj}) \tag{5.11}$$

Considering now the first term in **Eq.(5.10),** from **Eq.(5.2)**, we can differentiate E_p with respect to o_{pj}, giving

$$\frac{\partial E_p}{\partial o_{pj}} = -(t_{pj} - o_{pj}) \tag{5.12}$$

Thus $$\delta_{pj} = f_j'(net_{pj})(t_{pj} - o_{pj}) \tag{5.13}$$

This is useful for the output units, since the target and output are both available, but not for the hidden units, since their targets are not known.

So, if unit j is not an output unit, we can write, by the chain rule again, that

$$\frac{\partial E_p}{\partial o_{pj}} = \sum_k \frac{\partial E_p}{\partial net_{pk}} \frac{\partial net_{pk}}{\partial o_{pj}} = \sum_k \frac{\partial E_p}{\partial net_{pk}} \frac{\partial}{\partial o_{pj}} \sum_i \omega_{ik} o_{pi} \tag{5.14}$$

$$\frac{\partial E_p}{\partial o_{pj}} = -\sum_k \delta_{pk} \omega_{jk} \tag{5.15}$$

using **Eq.(5.3)** and **Eq.(5.7)** and noting that the sum drops out since the partial differential is non-zero for only one value, just as in **Eq.(5.6)**. Substituting **Eq.(5.15)** in **Eq.(5.10)**, we get finally

$$\delta_{pj} = f_j'(net_{pj}) \sum_k \delta_{pk} \omega_{jk} \tag{5.16}$$

This equation represents the change in the error function, with respect to the weights in the network. This provides a method for changing the error function so as to be sure of reducing it. The function is proportional to the errors δ_{pk} in subsequent units, so the error

has to be calculated in the output units first (given by **Eq.(5.13)**) and then passed back through the net to the earlier units to allow them to alter their connection weights. It is the passing back of this error value that leads to the networks being referred to as back-propagation networks. **Eq.(5.13)** and **Eq.(5.16)** show how we can train our multilayer networks.

In this regard is used the sigmoid function as the non-linear threshold function is that it is quite like the step function, and so should demonstrate behaviour of a similar nature. The sigmoid function is define as

$$f(net) = 1/(1+e^{-k*net}) \tag{5.17}$$

and has the range $0 < f(net) < 1$. k is a positive constant that controls the "spread" of the function large values of k squash the function until as $k \rightarrow \infty$, $f(net) \rightarrow$ *Heaviside function*. It also acts as an automatic gain control, since for small input signals the slope is quite steep and so the function is changing quite rapidly, producing a large gain. For large inputs, the slope and thus the gain is much less. This means that the network can accept large inputs and still remain sensitive to small changes. A major reason for its use is that it has a simple derivative, however, and this makes the implementation of the back-propagation system much easier. Given that the output of a unit, o_{pj} is given by

$$o_{pj} = f(net) = 1/(1+e^{-k*net}) \tag{5.18}$$

the derivative with respect to that unit, is given by

$$f'(net) = ke^{-k*net}/(1+e^{-k*net})^2 \tag{5.19}$$

$$\text{Or,} \quad f'(net) = kf(net)(1-f(net)) = ko_{pj}(1-o_{pj}) \tag{5.20}$$

The derivation is therefore a simple function of the outputs.

5.3.2 The Overall Feed forward Structure

The feed forward network is composed of hierarchy of processing units, organized in a series of two or more mutually exclusive sets of neuron or layers. The first or input layer acts as a holding site for the values applied to the network. The last or output layer is the

point at which the final stare of the network is read. Between these two extremes lie zero or more layers for the hidden units. Links, or weights, connect each unit in one layer to only those in the next higher layer. There is an implied directionality in these connecting weight, is fed forward to provide a portion of the activation for the units in the next-higher layer.

Fingerprint presents a unique task to identify a person more exactly. The method adopted for fingerprint classification need not be limited to neural networks only. In fact any standard statistical method of fingerprint classification could be equally utilized for fingerprint identification. However the well-known BackPropagation algorithm [59, 60] that can be used to train the MLP is a widely used method for fingerprint classification.

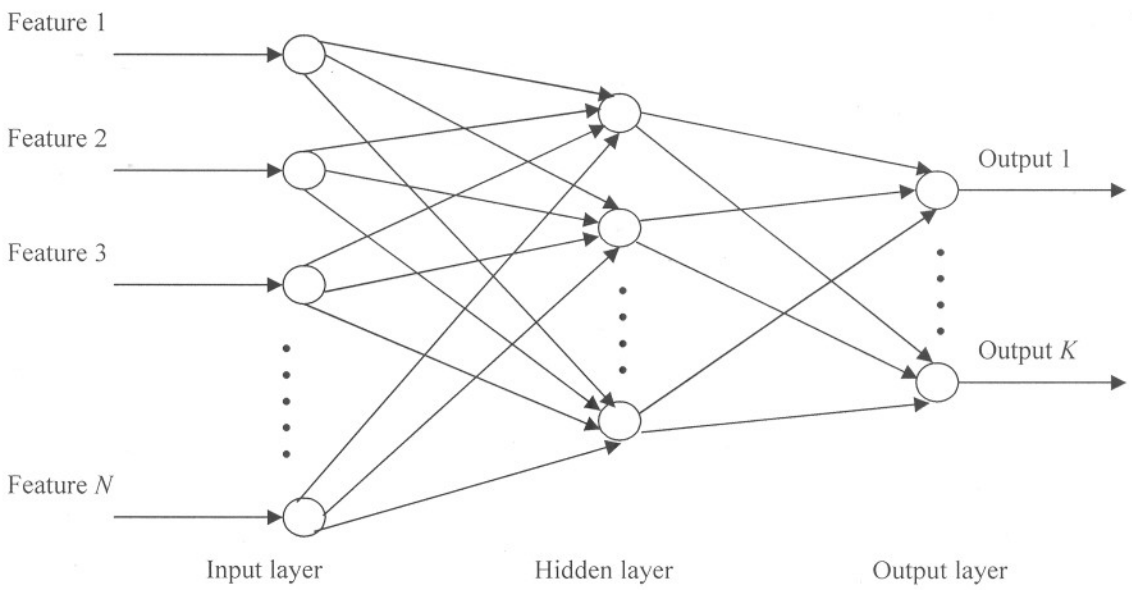


Fig.5.4: A Multilayer Perceptron (MLP)

The structure of a three layer neural network can be used for fingerprint identification task shown in **Fig.5.4**. There is an input layer, which gates the inputs, namely, fingerprint features vectors. The output layer of the network produces the output, which classify the

fingerprints of different persons. One hidden layer is used between the input and output layer.

5.3.1 Role of Input Layer

The role of input layer [58] is somewhat fictitious, in that layer units are used only to 'hold' input values and distribute these values to all units in the next layer. Thus, the input layer units do not implement a separate mapping or conversion of the input data, and their weights are insignificant. The feed forward network must have the ability to learn pattern mapping. The network may be made to function as a pattern associator through training. Training is accomplished by presenting the pattern to be classified to the network and determining its output.

The actual output of the network is compared with a 'target' and a error measure is calculated. The error measure in then propagated backward through the network and is used to determine weight changes within the network. This process is repeated until the network reaches a desired state of response. Although this is an idealized description of training, this does not imply that an arbitrary network will converge to the response.

5.3.2 Role of Hidden Layer

There are several interesting and related interpretations that may be attached to the units in internal layers. These are the following:

- The internal layers *remap* the inputs and results of other (previous) internal layers to achieve a more linearly separable or 'classifiable' representation of the data. In fact, in this case suitable external preprocessing of the input can have the same effect.
- The internal layers may allow attachment of semantics to certain combinations of layer inputs.

5.3.5 Role of Output Layer

The role of output layer is very much important for matching with target of the pattern classes. It has the following tasks:

- Compare the present output with the target output.
- If the error rate is very high compare with tolerance rate, then propagate those values to the feedback to the network.
- If the output has acceptable tolerance rate then store those weighted and threshold values in a file.

5.4 The Multilayer Perceptron Algorithm

The algorithm for the multilayer perceptron that implements the BackPropagation training rule is shown below [60, 62]. It requires the units to have thresholding non-linear functions that are continuously differentiable, i.e. smooth everywhere. We have assumed the use of the sigmoid function, $j(net) = 1/(1+e^{-k*net})$ since it has a simple derivative.

1. *Initialize weights and thresholds*. Set all weights and thresholds to small random values.
2. *Present input and desired output*. Present input $X_p = x_0, x_1, x_2, ..., x_{n-1}$ and target output $T_p = t_0, t_1, t_2, t_3, ..., t_{m-1}$ where n is the number of input nodes and m is the number of output nodes. Set ω_0 to $-\theta$, the bias, and x_0 to be always 1. For pattern association, X_p and T_p represent the patterns associated. For classification, T_p is set to zero except for one element set to 1 that corresponds to the class that X_p is in.
3. *Calculate actual output*. Each layer calculates

$$y_{pj} = f\left[\sum_{i=0}^{n-1} \omega_i x_i\right] \tag{5.21}$$

   and passes that as input to the next layer. The final layer outputs values o_{pj}.
4. *Adapt weights*. Start from the output layer, and work backwards.

$$\omega_{ij}(t+1) = \omega_{ij}(t) + \eta \delta_{pj} o_{pj} \tag{5.22}$$

$\omega_{ij}(t)$ represents the weights from node i and to node j at time t, η is a gain term, and δ_{pj} is an error term for pattern p on node j.

For output units

$$\delta_{pj} = ko_{pj}(1-o_{pj})(t_{pj}-o_{pj}) \tag{5.23}$$

For hidden units

$$\delta_{pj} = ko_{pj}(1-o_{pj})\sum_{k}\delta_{pk}\omega_{jk} \tag{5.24}$$

where the sum is over the k nodes in the layer above node j.

5.5 Minimum Distance Error Rate BackPropagation (MDER-BP) Algorithms

This proposed Algorithm [53] is similar to BackPropagation algorithm, but the major change is to calculate error rate. Here the *Euclidean distance* is used to calculate the error rate, which is the difference between the present output and the target output. The proposed algorithm is as follows:

The learning of this network has been accomplished by error Back Propagation Neural Network (**Fig.5.5**). The weight vectors W_{ij} and W_{jk} are the weighted values between layers i and j, j and k respectively. The network is provided with the input patterns and also the desired respective output patterns. The input patterns a are connected to hidden *Processing Elements* (PEs) through the weights W_{ij}.

In the *hidden layer*, each PE computed the weighted sum according to the equation, which is given by

$$net_{aj} = \sum W_{ij} O_{ai} \tag{5.25}$$

where O_{ai} is the input of unit i for pattern number a. The threshold, uh_j of each PE is then added to its weighted sum to obtain the activation $active_j$ of that PE i,e,

$$active_j = net_{aj} + uh_j \tag{5.26}$$

where uh_j is the hidden *threshold weight* for jth PEs. This activation determines whether the output of the respective PE is either 1 or 0 (fires or not) by using a sigmoid function,

$$O_{aj} = \frac{1}{1+e^{-k_1 * active_j}} \tag{5.27}$$

where k_1 is called the *spread factors*, these O_{aj} then serve as the input to the output computation. Signal O_{aj} are then fanned out to the output layer according to the relation,

$$net_{ak} = \sum W_{jk} O_{aj} \tag{5.28}$$

and the output threshold weight uo_k for k-th output PEs is added to it to find out the activation $active_{(k)}$

$$active_k = net_{ak} + uo_k \tag{5.29}$$

The actual output O_{ak} is computed using the same sigmoid function,

$$O_{ak} = \frac{1}{1 + e^{-k_2 * active_k}} \tag{5.30}$$

Here another *spread factor* k_2 has been employed for the output units.

In the *second stage*, after completing the *feed-forward propagation,*

an error is computed by comparing the output O_{ak} with the respective target t_{ak} ,i.e

$$\delta_{ak} = \sqrt{\sum_{k=0}^{k-1} (t_{ak} - O_{ak})^2} \tag{5.31}$$

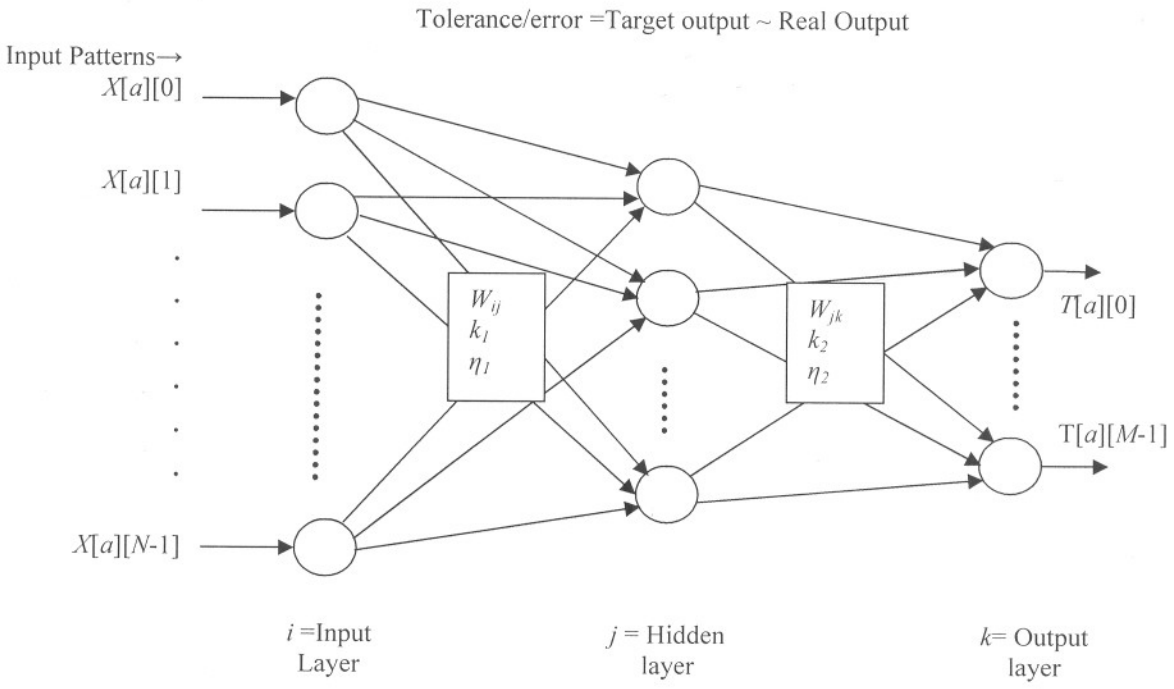


Fig.5.5 MDER- BackPropagation Neural Network

This error is then used to adjust the weight vector W_{jk} using the equation,

$$\Delta W_{jk} = \eta_2 k_2 \delta_{ak} O_{aj} O_{ak} (1 - O_{ak}) \tag{5.32}$$

where, $\int'(activeo_k) = k_2 O_{ak}(1 - O_{ak})$ the derivation of sigmoid function and η_2 is the *learning factor* of the network.

The weight vector W_{jk} is then adjusted to $W_{jk} + \Delta W_{jk}$. For the threshold weight of the output PE, similar equation is applied,

$$\Delta uo_k = \eta_2 k_2 \delta_{ak} O_{ak} (1 - O_{ak}) \tag{5.33}$$

and the *new threshold weight* equaled $uo_k + \Delta uo_k$.

In the *next step*, this error and the adjusted weight vector W_{jk} are fedback to the hidden layer to adjust the weight vector W_{ij} and threshold weight uh_j. In this layer change in weight vector W_{ij} is computed by using equation,

$$\Delta W_{ij} = \eta_1 k_1 O_{ai} O_{aj} (1 - O_{ak}) \sum \delta_{ak} W_{jk} \tag{5.34}$$

where, $\int'(activeh_j) = k_1 O_{ai}(1 - O_{aj})$ and η_1 is the *learning factor* of the network. The weight vector W_{ij} is then adjusted to $W_{ij} + \Delta W_{ij}$. For the threshold weights of the hidden PEs, similar equation is applied

$$\Delta uh_j = \eta_1 k_1 (1 - O_{aj}) \sum \delta_{ak} W_{jk} \tag{5.35}$$

and *new threshold weights* are calculated $uh_j + \Delta uh_j$.

The properties of sum-squared error equation dictate that as output approaches its maximum or minimum value, adjustments to individual weights become less pronounced. This is a testament to the stability of the *MDER-BackPropagation algorithm* [53]. The significance of the training process is that, as the network trains, the nodes in the intermediate layers organize themselves such that different nodes learn to recognize different features of the total input space.

Minimum Distance Error Rate (MDER) Calculation:

In this algorithm [53], the GMF features of fingerprint images are applied as an input vectors for training and testing the network for verification of the fingerprint images. Either it corresponds to the correctly recognized the fingerprint of a person or it fail to recognize the fingerprint of a person. The technique used for this purpose is the expression for Euclidean error-distance calculation is given below:

$$\delta_{ak}(t_{ak}, O_{ak})_{euc} = \sqrt{\sum_{k=0}^{k-1}(t_{ak} - O_{ak})^2} \tag{5.36}$$

Where,

t_{ak} and O_{ak} are the two vectors, *target* and *output* respectively.

$\delta_{ak}(t_{ak}, o_{ak})_{edu}$ is the error-distance between the vectors.

k is the dimensionality of the vectors.

In this case, t_{ak} is the target vector of all the patterns and O_{ak} is the output of one particular fingerprint of a person. If the pattern (fingerprint) is corresponding to the minimum error-distance, then it is recognized.

5.6 Application of MDER-BP Algorithms for Fingerprints Identification

The method of MDER-BP is proposed for simulation to apply for fingerprints recognition. The proposed method is actually modified BackPropagation Neural Network with a moderation in error calculations. The practical use of MDER-BP method has been described in ***Chapter-8***.

Chapter Six

UNSUPERVISED NEURAL NETWORK MODELS FOR FINGERPRINT VERIFICATION

Contents **Page. No.**

6.1 Introduction

Unsupervised Neural Networks are the unlabeled instances and they are processed blindly or heuristically. These methods are used to verify and to recognize the fingerprint patterns of a particular person. The desired response is not known; thus, explicit error information cannot be used to improve the network behavior. This task is more abstract and less defined which result in less computational complexity and less accuracy than supervised learning algorithm. Examples are Adaptive Resonance Theory (**ART**), Kohonen Self-Organizing Networks, and Associative Memory etc.

6.2 Adaptive Resonance Theory (ART)

The adaptive resonance theory (ART) is developed to model a massively parallel architecture for a self-organizing neural pattern recognition network, based on biological and behavioural data [58, 59]. The major feature of ART, proposed by Stephen Grossberg and Gail Carpenter, is the ability to switch modes between plastic (the learning state

where the internal parameters of the network can be modified) and stable (a fixed classification set), without detriment to any previous learning. The network also displays many behavioural type properties, such as sensitivity to context, that enables the network to discriminate irrelevant information or information that is repeatedly shown to the network. A key to solving the stability-plasticity dilemma is to add a feedback mechanism facilitates the learning of new information without destroying old information, automatic switching between stable and plastic modes, and stabilization of the encoding of the classes done by the nodes. The results from this approach are two neural-network architectures that are particularly suited for pattern-classification problems in realistic environments. These network architectures are referred to as ART1, ART2 and ART3. ART1, ART2 and ART3 differ in the nature of their input patterns. ART1 networks require that the input vectors be binary. ART2 and ART3 networks are suitable for processing analog, or gray-scale, patterns.

6.3 ART Network Description

The basic features of the ART architecture [58, 59] are shown in **Fig.6.1**. Patterns of activity that develop over the nodes in the two layers of the attentional subsystem are

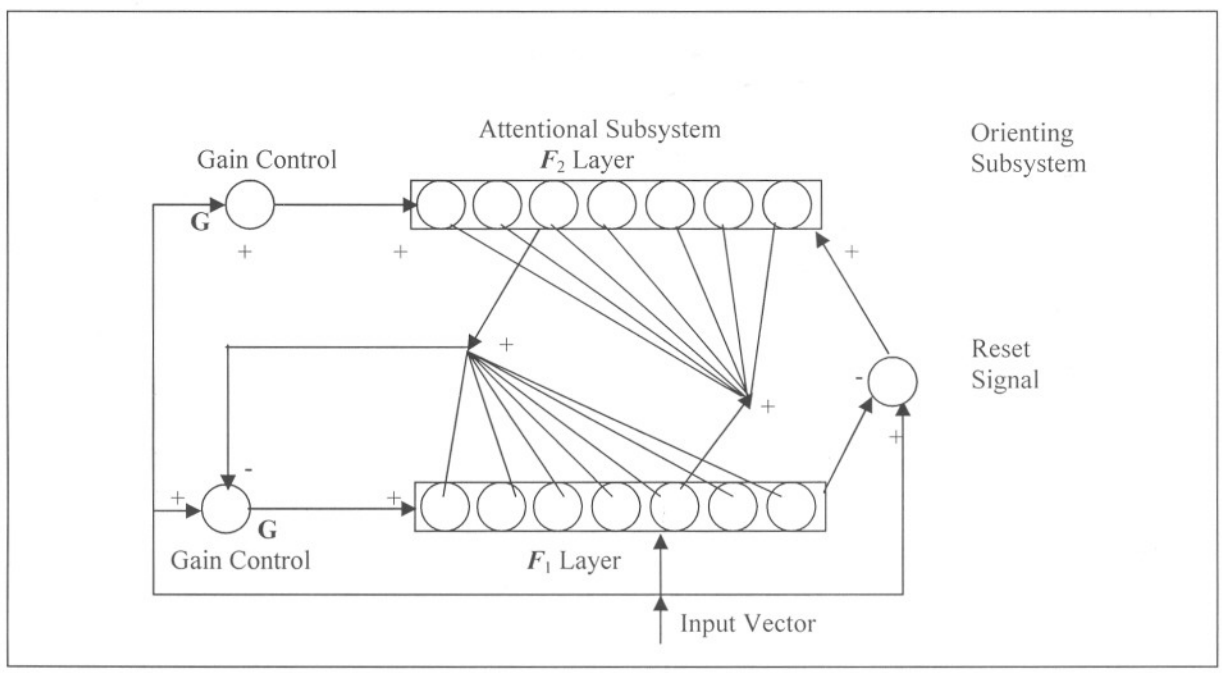


Fig.6.1: The ART System Architecture

called short-term memory (**STM**) traces because they exist only in association with a single application with a input vector. The weight associated with the bottom-up and top-down connections between F_1 and F_2 are called long-term memory (**LTM**) traces because they encode information that remains a part of the network for an extended period.

6.3.1 Pattern Matching in ART

To illustrate the processing that takes place, we shall describe a hypothetical sequence of events that might occur in an ART network [59]. The scenario is a simple pattern-matching operation during which an ART network tries to determine whether an input pattern is among the patterns previously stored in the network. **Fig.6.2** illustrates the operation. In **Fig.6.2(a)**, an input pattern, ***I***, is presented to the units on F_1 in the same manner as in other networks: one vector component goes to each node. A pattern activation, ***X***, is produced across F_1. The processing node by the units on this layer is a somewhat more complicated form of that done by the input layer of the Counter Propagation Network (CPN). The same input pattern excites both the orienting subsystem, ***A***, and the gain control, ***G***. The output pattern, ***S***, results in an inhibitory signal that is also sent to ***A***. The network is structured such that this inhibitory signal exactly cancels the excitatory effect of the signal from ***I***, so that ***A*** remains inactive. ***G*** supplies an excitatory signal to F_1. The same signal is applied to each node on the layer and is therefore known as a nonspecific signal. The need for this signal will be made clear later. The appearance of ***X*** on F_1 results in an output pattern, ***S***, which is sent through connections to F_2. Each F_2 unit receives the entire input vector, ***S***, from F_1. F_2 units calculate the net-input values in the usual manner by summing the products of the input values and the connection weights. In response to inputs from F_1, a pattern of activity, ***Y***, develops across the nodes of F_2. F_2 is a competitive layer that performs a contrast enhancement on the input signal like the competitive layer. The gain control signals to F_2 are omitted here for simplicity.

In **Fig.6.2(b)**, the pattern of activity, ***Y***, results in an output pattern, ***U***, from F_2. This output pattern is sent as an inhibitory signal to the gain control system. The gain control is configured such that if it receives any inhibitory signal from F_2, it ceases activity. ***U***

also becomes a second input pattern for the F_1 units. $\boldsymbol{U}$ is transformed by long-term memory (**LTM**) traces on the top-down connections from F_2 to F_1. We shall call this transformed pattern $\boldsymbol{V}$. Notice that there are three possible sources of input to F_1, but that only two appear to be used at any one time. The units on F_1 (and F_2 as well) are constructed so that they can become active only if two out of the possible three sources of input are active. This feature is called the **2/3 rule** and it plays an important role in ART, which we shall discuss more fully later in this section.

Because of the 2/3 rule, only those F_1 nodes receiving signals from both $\boldsymbol{I}$ and $\boldsymbol{V}$ will remain active. The pattern that remains on F_1 is $\boldsymbol{I} \cap \boldsymbol{V}$, the intersection of $\boldsymbol{I}$ and $\boldsymbol{V}$ will remain active. In **Fig.6.2(b)**, the patterns mismatch and a new activity pattern, $\boldsymbol{X}^*$, develops on F_1. Since the new output pattern, $\boldsymbol{S}^*$, is different from the original pattern, $\boldsymbol{S}$, the inhibitory signal to $\boldsymbol{A}$ no longer cancels the excitation coming from $\boldsymbol{I}$.

In **Fig.6.2(c)**, A has become active in response to the mismatch of patterns on F_1. $\boldsymbol{A}$ sends a nonspecific reset signal to all of the nodes on F_2. These nodes respond according to their present state. If they are inactive, they do not respond. If they are active, they become inactive and they stay that way for an extended period of time. This sustained inhibition is necessary to prevent the same node from winning the competition during the next matching cycle. Since $\boldsymbol{Y}$ no longer appears, the top-down output and the inhibitory signal to the gain control also disappear.

In **Fig.6.2(d)**, the original pattern, $\boldsymbol{X}$, is reinstated on F_1, and a new cycle of pattern matching begins. This time a new pattern, $\boldsymbol{Y}^*$, appears on F_2. The nodes participating in the original pattern, $\boldsymbol{Y}$, remain inactive due to the long term effects of the reset signal from $\boldsymbol{A}$.

This cycle of pattern matching will continue until a match is found, or until F_2 runs out of previously stored patterns. If no matching is found, the network will assign some uncommitted node or nodes on F_2 and will begin to learn the new pattern. Learning takes place through the modification of the weights, or the LTM traces. It is important to understand that this learning process does not state or stop, but rather continues even while the pattern matching process takes place. Anytime signals are sent over

connections, the weights associated with those connections are subject to modification. Why then do the mismatches not result in loss of knowledge or the learning of incorrect associations?

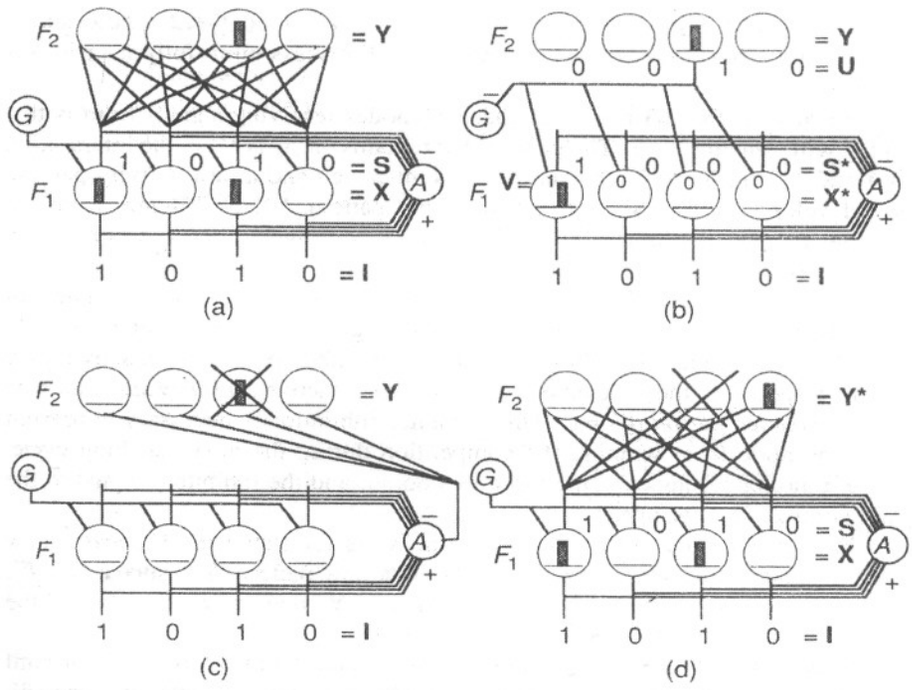


Fig.6.2: A Pattern-matching cycle in an ART network is shown.

The reason is that the time required for significant changes to occur in the weights is very long with respect to the time required for a complete matching cycle. The connections participating in mismatches are not active long enough to affect the associated weights seriously.

When a match does occur, there is no reset signal and the network settles down into a resonant state as described earlier. During this stable state, connections remain active for a sufficiently long time so that weights are strengthened. This resonant state can arise

only when a pattern match occurs, or during the enlistment of new units on F_2 in order to store a previously unknown pattern.

6.4 Adaptive Resonance Theory-1 (ART1)

As it was mentioned in the **section 6.2** of this chapter, ART1 network [58] requires binary input vector, that is, they must have components of the set {0,1}. This restriction may appear to limit the utility of the network but there are many problems in transforming data that can be cast into binary format.

6.4.1 Architecture of ART1

The neural network for ART1 model consists of the following:

(a) A layer of neuron called F_1 layer (input layer or comparison layer).
(b) A node for each layer as a gain control unit.
(c) A layer of neurons called F_2 layer (output layer or recognition layer).
(d) Bottom-up connection from F_1 to F_2 layer.
(e) Top-down connection from F_2 to F_1 layer.
(f) Inhibitory connection (negative weights) from F_2 layer to gain control.
(g) Excitatory connection (positive weights) from gain control to a layer.
(h) Inhibitory connection from F_1 layer to reset node.
(i) Excitatory connection from reset node to F_2.

The ART1 architecture shown in **Fig.6.3** consists of two layers of neurons called comparison layer and the recognition layer. Usually, the classification decision is indicated by a single neuron in the recognition layer that fires. The neurons in the comparison layer respond to input features in the pattern. The synaptic connections (weights) between these two layers are modifiable in both the directions. According to learning rules, the recognition layer neurons have inhibitory connections that allow for competition. These two layers constitute attentioned system. The network architecture also consists of three additional modules labeled *Gain1*, *Gain2*, and reset as shown in **Fig.6.3**. In the attentioned subsystem, if the match of input pattern with any of the prototype stored occurs, resonance is established.

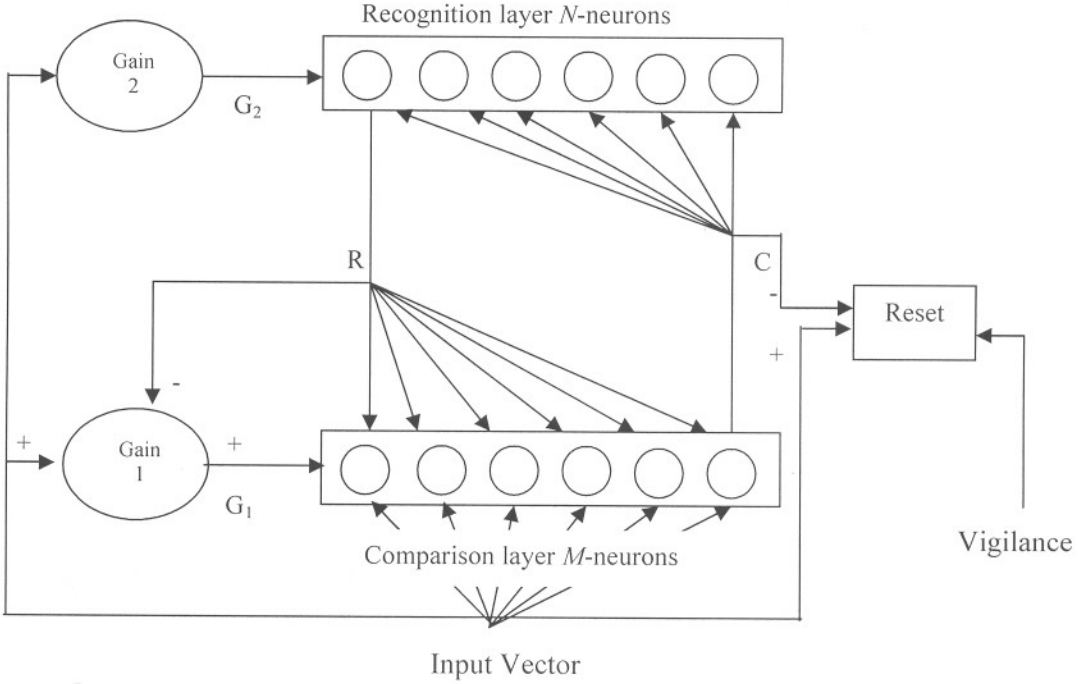


Fig.6.3: Adaptive Resonance Theory-1 (ART1) Network

The orienting subsystem is responsible for sending mismatch between bottom-up and top-down patterns on the recognition layer. The recognition layer response to an input vector is compared to the original input vector through a mechanism called *vigilance*. When vigilance falls below a threshold, a new category must be created and the input vector must be stored into that category. The recognition layer follows the winner which takes all paradigms. The recognition layer is shown in **Fig.6.4**.

6.2.2 Special Features of ART1 Models

One special feature of an ART1 model [58, 59] is that a two-third rule is necessary to determine the activity of the neuron in the F_1 layer. There are three input sources to each neuron in the F_1 layer. They are the external input, the output of the gain control, and the output of F_2 layer neurons. The gain control unit and 2/3 rule together ensure proper response from the input layer neurons.

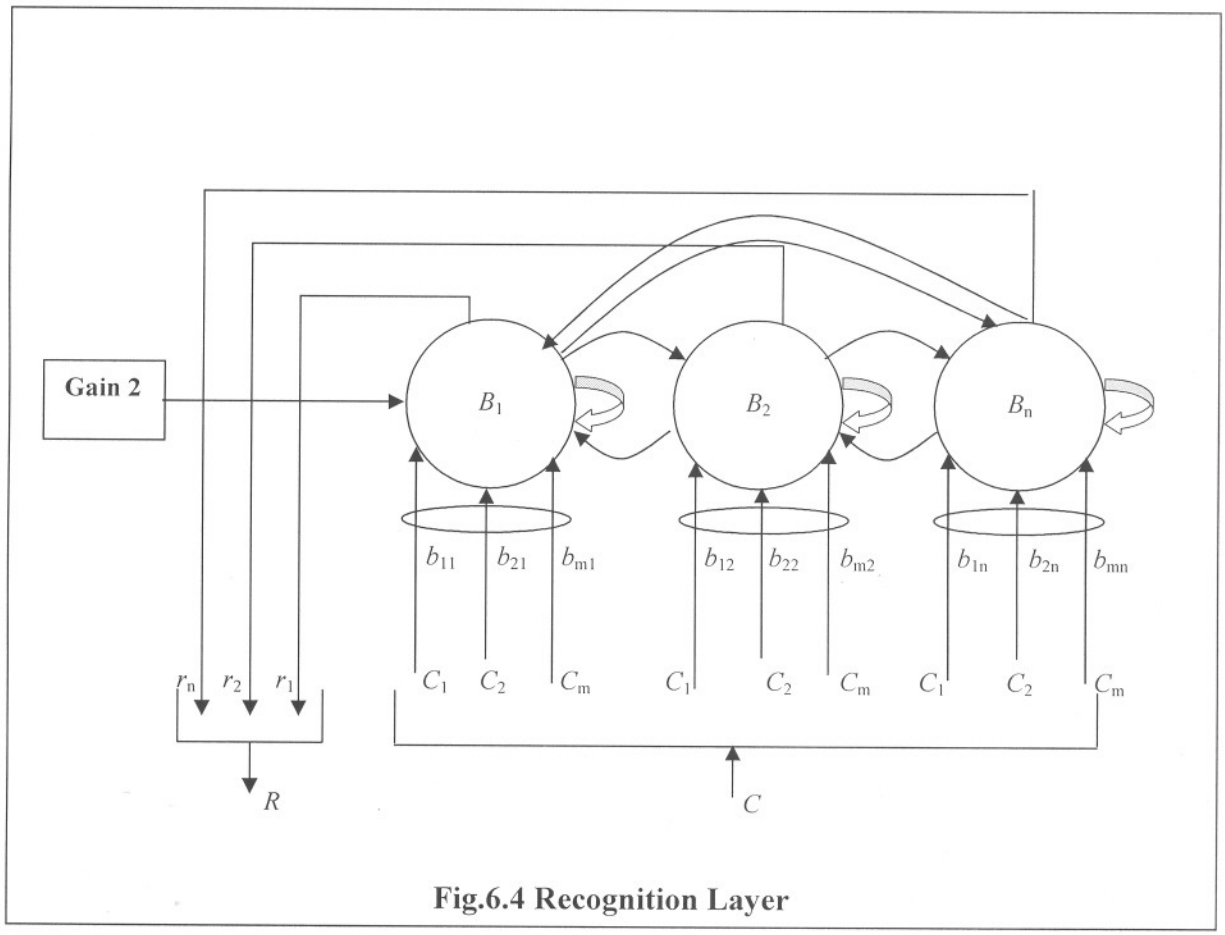


Fig.6.4 Recognition Layer

A second feature is that the vigilance parameter is used to determine the activity of the reset unit, which is activated whenever there is no match found among existing patterns during classification.

Let us consider **Fig.6.4**

$$net_j = \sum_{i=1}^{N} b_{ij} C_i \tag{6.1}$$

$$r_j = f(net_j) = \begin{cases} 1 & for \quad net_j > net_i, \quad for \quad all \quad i \neq j \\ 0 & otherwise \end{cases} \tag{6.2}$$

Where C_i is the output of the ith comparison layer neuron, f is a step function and thus, r_j results in a binary value. M is the number of neurons in the comparison layer. **Fig.6.5** shows the comparison layer.

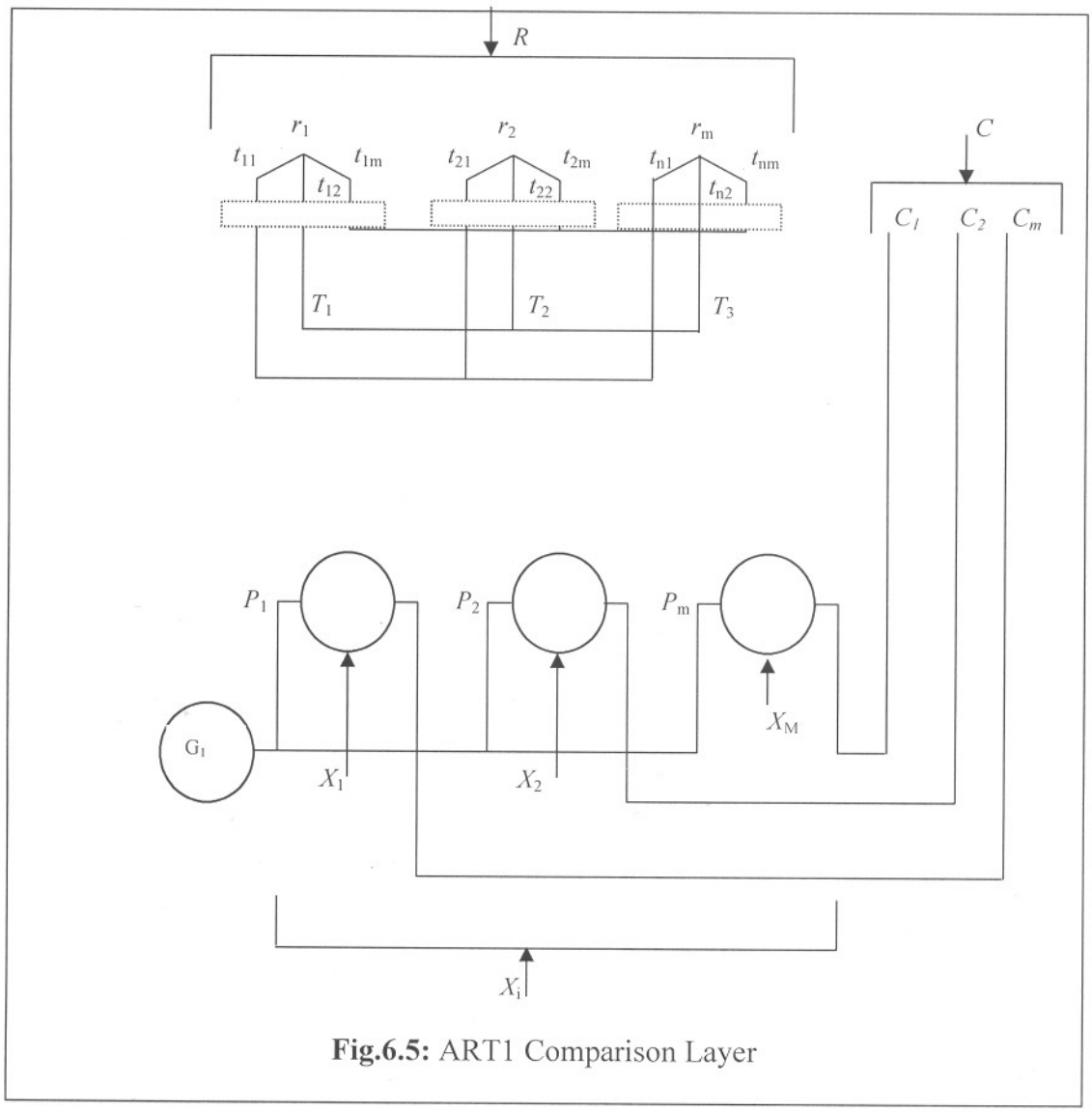


Fig.6.5: ART1 Comparison Layer

As shown in **Fig.6.5**, each neuron i in the comparison layer receives the following three inputs:

(a) A component or the input pattern X, X_i.

(b) The gain signal G_1 is a scalar (binary value), thus, the same value is input to each neuron.

(c) A feedback signal from the recognition layer is a weighted sum of the recognition layer outputs. Thus,

$$P_i = \sum_{j=1}^{N} t_{ji} r_j \qquad for \quad i = 1,2,....M \tag{6.3}$$

Where r_j is the output of the jth recognition layer neuron and N is the number of neurons in the recognition layer, T_j is the weight vector associated with the recognition layer neuron j, vector C represents the output of comparison layer, gain G_1 is 1 when the R vector is 0 and the logical OR of the components of the input vector X is 1 as

$$G_1 = (\bar{r}_1 \mid \bar{r}_2 \mid .... \mid \bar{r}_n)(X_1 \mid X_2 \mid X_3 \mid .... \mid X_M) \tag{6.4a}$$

Gain G_2 is 1 when the logical OR of the components of the input vector X is 1 as

$$G_2 = (X_1 \mid X_2 \mid ..... \mid X_m) \tag{6.4b}$$

The steps of ART1 operation are given in **Fig.6.5 to 6.9**.

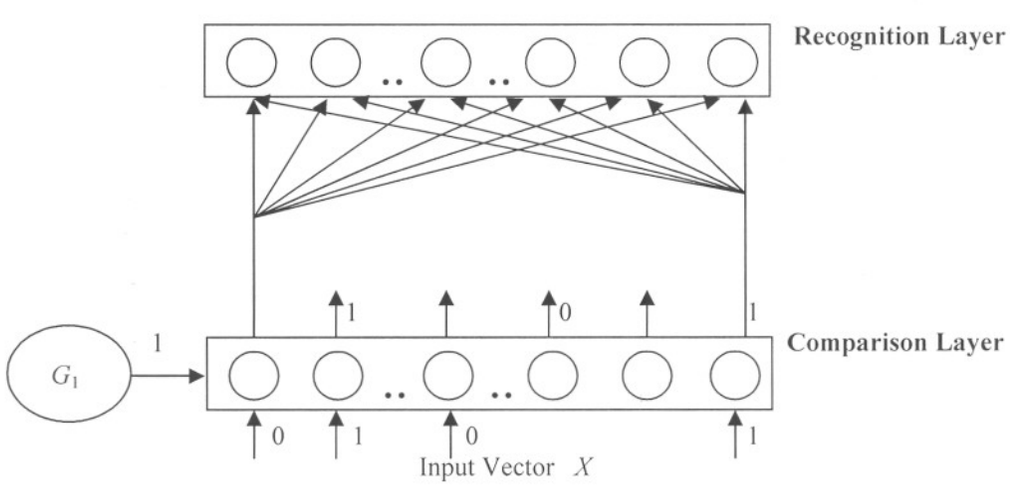


Fig.6.6: Step-1. $G_1 = 1$ The input vector is passed through the comparison layer to the recognition layer.

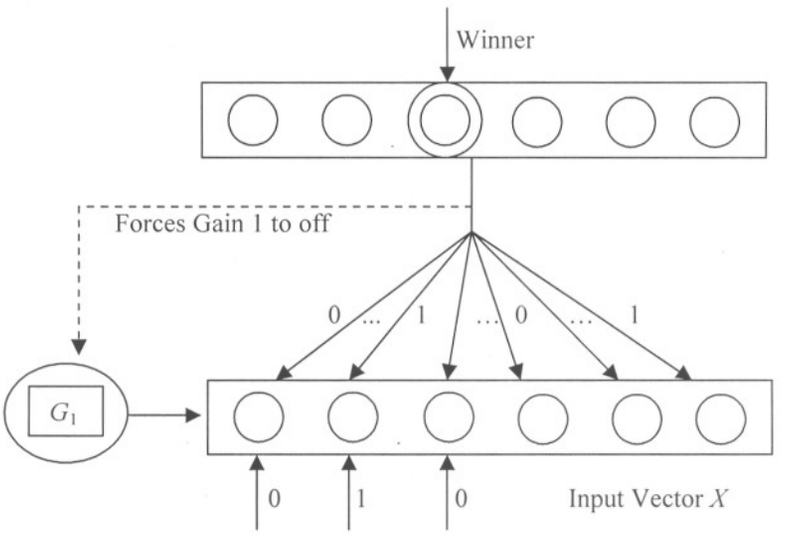


Fig.6.7 Step 2. The best neuron of the recognition layer has been selected as winner; the winner sends its signal through its top-down weights.

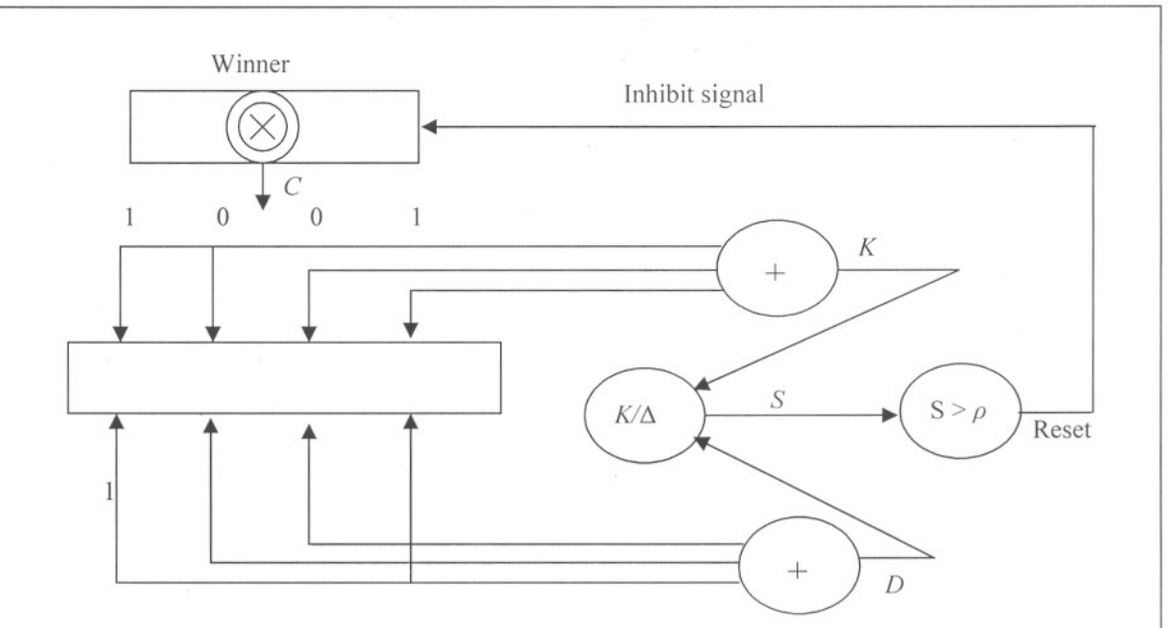


Fig.6.8 Step 3. The input vector X and P vector in recognition layer compared. Vigilance failed. Winning neuron is inhibited.

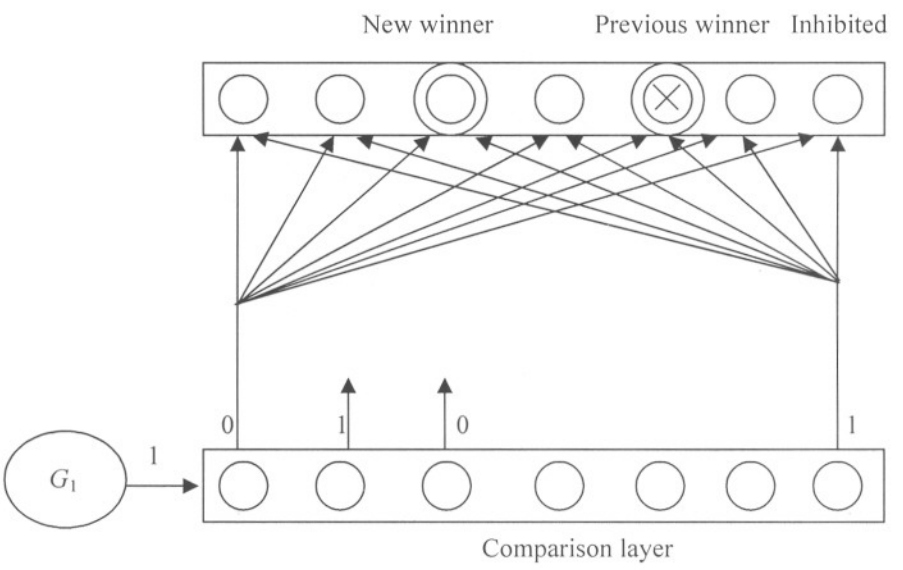


Fig.6.9 Step 4. Previous winning neuron is disabled. New winner is selected.

6.4.3 ART1 Algorithm

To begin with, we must determine [58] the size of the F_1 and F_2 layers as

$$\text{Number of units in } F_1 = M$$
$$\text{Number of units in } F_2 = N \tag{6.5}$$

Other parameters must be chosen according to the following constraints.

$$A \geq 0$$
$$C \geq 0$$
$$D \geq 0$$
$$\max(D,1) < B < D + 1$$
$$\text{L} > 1, \text{ typically L} = 2$$
$$0 < \rho \leq 1 \tag{6.6}$$

The parameter B must be chosen to satisfy the above constraint to implement 2/3 rule successfully to distinguish between top-down and bottom-up patterns.

Initialization. Top-down weights must be initialized as

$$[td]_{M \times N} = \text{top-down weight matrix} \tag{6.7}$$

$$td_{ij}^{0} > \frac{B-1}{D} \tag{6.8}$$

Bottom-up weights must be initialized as

$$[bu]_{N \times M} = \text{bottom-up weights} \tag{6.9}$$

$$0 < bu_{ij}^{0} < \frac{L}{L-1+M} \tag{6.10}$$

The activities on F_2 are initialized to zero but according to our chosen model, F_1 activities are initialized to

$$x_i(0) = \frac{-B}{1+C} \tag{6.11}$$

All input patterns must be binary $I_i \in \{0,1\}$. The norm of the vector is equal to the sum of the components.

$$|I| = \sum_{i=1}^{M} I_i \tag{6.12}$$

$|I|$ will be equal to number of non zero components of the vector. **Algorithm 6.1** illustrates the steps to be followed [58, 59].

Algorithm 6.1 (ART1 Algorithm):

Step 1: Apply an input vector ***I*** to F_1 and F_1 activities are calculated as

$$X_i = \frac{I_i}{1+A(I_i+B)+C} \tag{6.13}$$

Step 2: Calculate the output vector for F_1

$$S_i = h(X_i) = \begin{cases} 1 & if \quad X_i > 0 \\ 0 & if \quad X_i \le 0 \end{cases} \tag{6.14}$$

Step 3: Propagate S forwards to F_2 and calculate the activities according to

$$\{T\}_{N\times 1} = [bu]_{N\times M}\{S\}_{M\times 1} \tag{6.15}$$

Step 4: Only the winning F_2 node has a nonzero output.

$$u_j = \begin{cases} 1 & T_j = \max\{T_k\}\forall k \\ 0 & otherwise \end{cases}$$

$$\{u\}_{N\,x\,1} \tag{6.16}$$

We shall assume that winning node is J.

Step 5: Propagate the output from F_2 back to F_1. Calculate the net inputs from F_2 to F_1 as

$$\{V\}_{M\times 1} = [td]_{N\times M}\{u\}_{N\times 1} \tag{6.17}$$

Step 6: Calculate new activities according to

$$X_i = \frac{I_i + DV_i - B}{1 + A(I_i + DV_i) + C} \tag{6.18}$$

Step 7: Determine new output values $\{S\}$ as in step 2.

Step 8: Determine the degree of match between input pattern and the top-down template as

$$\frac{|S|}{|I|} = \frac{\sum_{i=1}^{M} S_i}{\sum_{i=1}^{M} I_i} \tag{6.19}$$

Step 9: If the above value is $< \rho$ mark J as inactive, zero the outputs of F_2 and return to ***step 1*** using the original pattern and if the above value is greater than or equal to ρ then continue.

Step 10: Update bottom-up weights on J only as

$$[bu]_{J,i} = \begin{cases} \dfrac{L}{L-1+|S|} & if \quad i \quad is \quad active \\ 0 \quad if \quad i \quad is \quad inactive \end{cases} \tag{6.20}$$

Step 11: Update the top-down weights coming from J only to all F_1 units.

$$[t,d]_{i,J} = \begin{cases} 1 \quad if \quad i \quad is \quad active \\ 0 \quad if \quad i \quad is \quad inactive \end{cases} \tag{6.21}$$

Step 12: Remove the input pattern. Restore all inactive F_2 units. Return to ***step 1*** with a new input pattern.

End ART1 Algorithm.

6.5 Adaptive Resonance Theory-2 (ART2)

6.5.1 Architecture of ART2:

ART2 networks [58, 59] self organize stable recognition categories in response to arbitrary sequences of analog (Grey-scale, continuous-valued) input patterns, as well as

binary input patterns. On the surface, it looks that the main difference between ART1 and ART2 is that ART2 accepts input vectors whose components can have any real numbers as their value. But in execution, ART2 network is considerably different from network. The capability of recognizing analog patterns represents a significant enhancement to the system. ART2 also recognizes the underlying similarity of identical patterns superimposed on constant backgrounds having different levels. **Fig.6.10** shows the ART2 architecture where the comparison layer of F_1 layer is spilt into several sublayers. Additionally, the orienting subsystem has also accommodated real-valued data. ART2 includes the following:

(a) Allowance for noise suppression.

(b) Normalization, i.e. constant to enhance the significant parts of the pattern.

(c) Comparison of top-down and bottom-up signals needed to reset the mechanism, and

(d) Dealing with real-valued data that may be arbitrarily close to one another.

The learning law of ART2 is much simpler even though network is complicated. **Carpenter** and **Grossberg** (1987), the developers of ART2 architecture have been developing various architectures of ART2 and in this chapter, it will be described one such architecture [58, 59].

6.5.2 ART2 Algorithm [58, 59]:

The size of the F_1 and F_2 layers is determined as

Number of units in F_1 layer = M

Number of units in F_2 layer = N

The parameters are chosen according to the following constants

$$a, b > 0$$

$$0 \le d \le 1; cd/(1-d) \le 1; 0 \le \theta \le 1; 0 \le \rho \le 1; e << 1 \qquad (6.22)$$

Top-down weights are initialized to zero as $[td]$ = [0]. The top-down weight matrix consists of M rows and N columns.

Bottom-up weights are initialized as

$$[bu] = \begin{bmatrix} \alpha & \alpha & \alpha \\ \alpha & \alpha & \alpha \\ \alpha & \alpha & \alpha \end{bmatrix} \tag{6.23}$$

where

$$a \leq \frac{0.5}{\{(1-d)\sqrt{M}\}}$$

The steps to be followed are illustrated in **Algorithm 6.2**.

Algorithm 6.2 (ART2 Algorithm)

Step 1: Initialize all layer and sublayer outputs to zero vectors and establish a cycle counter initialized to a value 1.

Step 2: Apply an input pattern $\boldsymbol{I}$ to the $\boldsymbol{W}$ layer of $\boldsymbol{F_1}$. The output of this layer is

$$W_i = I_i + au_i \tag{6.24}$$

Step 3: Propagate forward to $\boldsymbol{X}$ sublayer as

$$X_i = \frac{W_i}{e + \|W\|} \tag{6.25}$$

Where $\|W\|$ denotes the second norm of $\{W\}$ given by

$$\|W\| = \sqrt{W_1^2 + W_2^2 + W_3^2 + ... + W_M^2} \tag{6.26}$$

Step 4: Propagate forward to the $\boldsymbol{v}$ sublayer as

$$v_i = f(X_i) + bf(q_i) \tag{6.27}$$

Where,
$$f(X) = \begin{cases} 0 & 0 \leq \theta \\ X & X > \theta \end{cases} \tag{6.28}$$

It is to be noted that the second term is zero on the first pass through as q is zero at that time.

Step 5: Propagate to u sublayer as

$$u_i = \frac{v_i}{(e + \|v\|)} \tag{6.29}$$

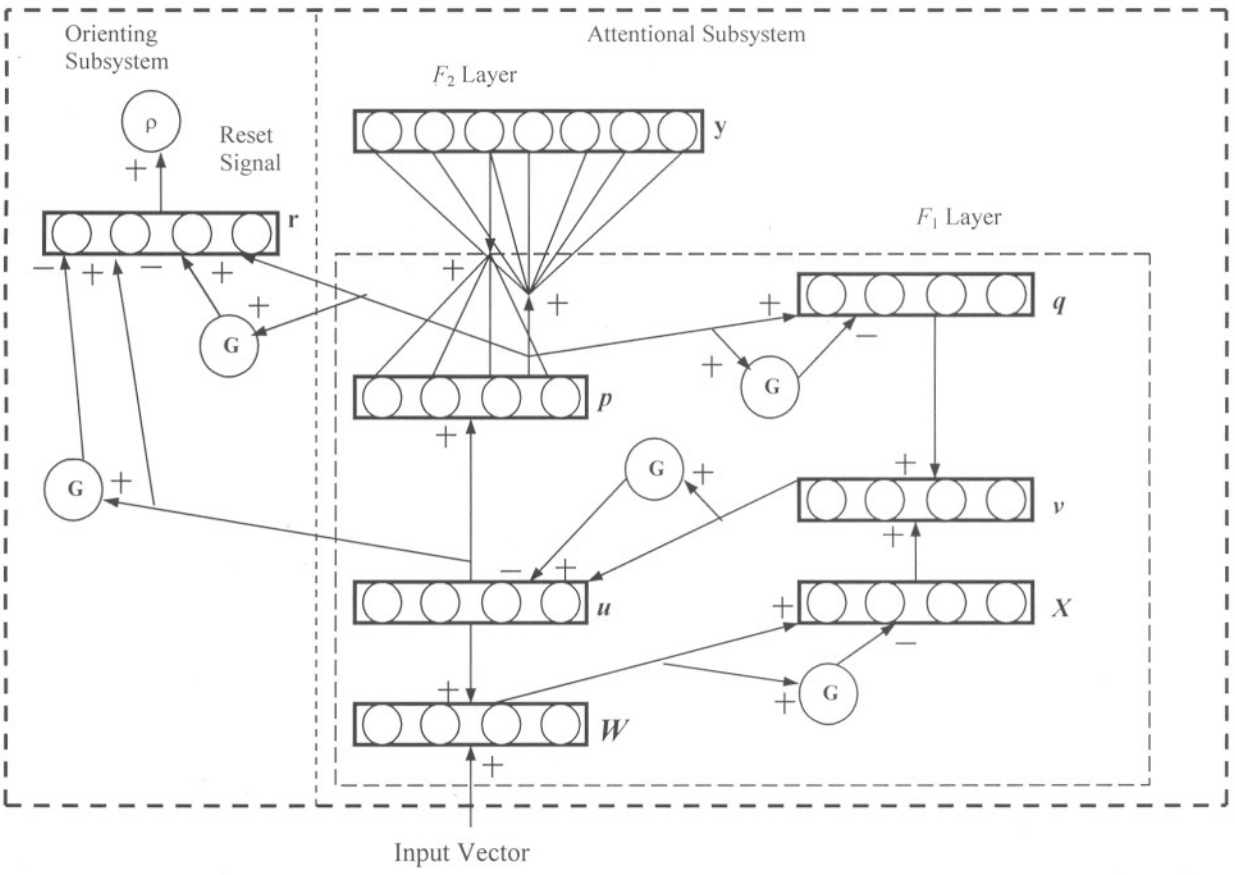


Fig.6.10: The Overall Structure of the ART-2 Network

Step 6: Propagate forward to P sublayer as

$$P_i = u_i + d(td_{i,J}) \tag{6.30}$$

Where Jth node on F_2 is the winner if the competition layer F_2 is inactive, i.e. $P_i = u_i$. Similarly, the network is still in its initial configuration $P_i = u_i$ because $td(i,j) = 0$.

Step 7: Propagate to qth sublayer as

$$q_i = \frac{p_i}{(e + \|P\|)} \tag{6.31}$$

Step 8: Repeat steps 2 to 7 as necessary to stabilize values on F_1.

Step 9: Calculate the output of rth layer as

$$r_i = \frac{(u_i + CP_i)}{(e + \|u\| + \|CP\|)} \tag{6.32}$$

Step 10: Determine whether a reset signal to F_2. Mark any active F_2 node as ineligible for competition, reset the cycle counter to one and return to ***step 2***. If there is no reset, and the cycle counter is 1, increment the cycle counter and continue with ***step 11***. If there is no reset and the cycle counter is greater than one then skip to ***step 14*** as resonance has been established.

Step 11: Propagate the output of P sublayer to the F_2 layer and calculate the net inputs to F_2, as

$$T_j = \sum P_i bu(j,i) \tag{6.33}$$

Step 12: Only the winning F_2 node has normalized output as

$$g(T_j) = \begin{cases} d & [T_j = \max_k \{T_k\}] \\ 0 & otherwise \end{cases} \tag{6.34}$$

Any node marked as ineligible by previous reset signals does not participate in the competition.

Step 13: Repeat ***steps 6*** to ***10***.

Step 14: Modify bottom-up weights on the winning F_2 unit as

$$bu(J,i) = \frac{u_i}{(1-d)} \tag{6.35}$$

Step 15: Modify top-down weights coming from the winning F_2 unit as

$$td(i,J) = \frac{u_i}{(1-d)} \tag{6.36}$$

Step 16: Remove the input vector. Restore all inactive F_2 units. Return to step 1 with a new input pattern.

6.6 Applications

The ART2 method is widely used in several pattern recognition problems. In this work, the Adaptive Resonance Theory 2 (ART-2) network is used to implement the off-line fingerprint verification system. ART-2 is designed for *GMF of fingerprint images* of continuous-valued input vectors or clustering *binary input vectors*. The feedback

mechanism of ART-2 network facilitates the learning the new information without destroying the old information.

Fig.6.10 shows the basic structure of ART-2 network. The two major subsystems are the attentional subsystem and the orienting subsystem. F_1 and F_2 represent two layers of nodes in the attentional subsystem. Nodes on each layer are fully interconnected to the nodes on the other layer. A plus sign indicates an excitatory connection and a minus sign indicates an inhibitory connection. The F_1 layer is divided into six subsystems, W, X, u, v, p and q. The gain control unit G and the reset unit ρ control the processing of the input data vector and creation of the clusters. The ART2 Algorithm has been described in ***section 6.5.2***.

The parameters of F_1 sub-layers are not changed during the verification, only the control parameter ρ have to be set properly to the authentic and the fake fingerprints were set to right clusters. The three methods of setting the control parameter were tested in this work:

(i). **Manual setting** (**Manual**): Control parameter ρ is set to the fixed value (ρ = **0.99**) manually, for all verification.

By the following equation we find the two automatic setting values:

$$g(T_j) = \begin{cases} dT_j = \max\{T_k\} \\ 0\, otherwise \end{cases}$$

(ii). **Automatic setting** (AS_1): $g(T_j)$ Minimum value.

(iii) **Automatic setting** (AS_2): $g(T_j)$ Maximum value.

The experimental results were shown in **Table-1** by changing the control parameters.

6.6.1 Experimental Results and Discussion

To test the fingerprint verification system, fingerprints are taken from 20 people. 20 authentic fingerprints of each person are collected. Sometimes the person is not satisfied with his/her own fingerprint. The quality of fingerprint depends on how his/her fingers are inked and the way he/she placed his/her finger on the paper. If the quality of the fingerprint is not good, then the fingerprint can be classified as fakes. For the evaluation of such cases, the person marked his/her authentic fingerprint by a mark from the scale 1-5 (1 means the best form of the fingerprints). The summary of the experimental results,

Table -6.1: The Results of Fingerprint Verification with Different Control Parameters.

Person ID	Authentic Fingerprint Classification						Authentic Fingerprint Verification (%)		
	Authentic Recognition			Fake Recognition					
	Manual	AS_1	AS_2	Manual	AS_1	AS_2	Manual	AS_1	AS_2
1	16	17	17	4	3	3	80	85	85
2	17	15	16	3	5	4	85	75	80
3	18	17	18	2	3	2	90	85	90
4	19	17	18	1	3	2	95	85	90
5	17	16	16	3	4	4	85	80	80
6	17	17	17	3	3	3	85	85	85
7	16	17	18	4	3	2	80	85	90
8	18	16	17	2	4	3	90	80	85
9	19	17	16	1	3	4	95	85	80
10	18	18	17	2	2	3	90	90	85
11	18	19	18	2	1	2	90	95	90
12	17	17	15	3	3	5	85	85	75
13	17	18	18	3	2	2	85	90	90
14	16	17	16	4	3	4	80	85	80
15	14	16	17	6	4	3	70	80	85
16	17	16	18	3	4	2	85	80	90
17	16	15	17	4	5	3	80	75	85
18	18	17	16	2	3	4	90	85	80
19	16	15	17	4	5	3	80	75	85
20	17	17	15	3	2	5	85	85	75
Average % of Fingerprint Recognition ⇨							85.25	83.5	84.25
Grand Average % of Fingerprint Recognition ⇨							84.33		

are presented in **Table-6.1**. The percentage of fingerprint verification was evaluated without the respect of fingerprint quality mark mentioned above. But in most cases, the fingerprints classified as fakes are the ones labeled by the person as poor fingerprints. We have observed that the best result can be achieved if the control parameter ρ was set automatically during the training process as the minimum value of activation level of unit $g(T_j)$ (Automatic setting (AS_1)).

6.6.2 Conclusion

The unsupervised learning method, ART-2 network for fingerprint verification is implemented to observe its power of verification and pattern matching. In this experimental work, we have seen that this network can be used for poor fingerprint images verifier and produces satisfactory results with respect to the training set size. A small number of the network parameters have been set manually to train the ART-2 network. These parameters have remained at most the same in the verification process too. *Finally, we compare this result of fingerprint recognition with other neural network methods which are described in Chapter-8.*

Chapter Seven

FINGERPRINT MATCHING ALGORITHMS

Contents **Page No**

7.1 Introduction

This chapter focuses on the different kinds of algorithms those are used to match fingerprint images. The existing popular fingerprint matching techniques can be widely classified into two categories. These are Minutiae-based and Correlation-based [10]. The minutiae-based [66] techniques typically match the two minutiae sets from two fingerprints by first aligning the two sets and then counting the number of minutiae that match. A typical minutiae extraction technique performs the following sequential operations on the fingerprint image: (a) fingerprint image enhancement, (b) binarization (segmentation into ridges and valleys), (c) thinning, and (d) minutiae detection [10]. Several commercial [63] and academic [2, 64] algorithms follow these sequential steps for minutiae detection.

Alternative techniques for minutiae detection directly operate on the gray scale fingerprint image itself and detection minutiae by adaptively tracing the gray scale ridges in the fingerprint images [4, 65]. The alignment between the input and the template

fingerprint images can be obtained using one or more of the fingerprint features. ***Correlation-based*** techniques match the global pattern of ridges and furrows to see if the ridges align. The simplest technique is to align the two fingerprint images and subtract the input from the template to see if the ridges correspond. However, such a simplistic approach suffers from many problems including the errors in estimation of alignment, non-linear deformation in fingerprint images, and noise. An auto-correlation technique has been proposed by Sibbald [13] that computes the correlation between the input and the template at fixed translation and rotation increments.

Above mentioned two fingerprints matching techniques are very much effective for good quality of fingerprint images. *The main difficulties in the minutiae-based and Correlation-based approach are that it is very difficult to reliably extract minutiae and global patterns in a poor quality of fingerprint images. In this case, we propose an alternative method called* ***GMF technique*** *to extract features from very poor fingerprint images, which is described in* ***chapter-3****. To implement this feature for matching an alternative algorithm has been proposed called* ***Minimum Distance Error Rate BackPropagation Algorithm*** [53], *which is described in* ***chapter-5*** *and shows the practical application in* ***chapter-8****.*

7.2 Matching Algorithms

In this section we discuss about different kinds fingerprint matching algorithms, which are presently used widely all over the world. These are:

- Hough Transform Based Matching (Algorithm Hough)
- String Distance Based Matching (Algorithm String)
- 2D Dynamic Programming Based Matching (Algorithm Dynamic)
- Filterbank Based matching (Algorithm Filter)
- Minimum Distance Error Rate BackPropagation Algorithm

7.2.1 Hough Transform Based Matching

The fingerprint matching problem can be regarded as template matching [64]: given two sets of minutia features, compute their matching score. The two main steps of the algorithm are:

(i) Compute the transformation parameters $\delta_x, \delta_y, \theta$ and s between the two images, where δ_x and δ_y are translations along $x-$ and $y-$ directions, respectively, θ is the rotation angle, and s is the scaling factor;

(ii) Align two sets of minutia points with the estimated parameters and count the matched pairs within a bounding box;

(iii) Repeat the previous two steps for the range of allowed transformations. The transformation that results in the highest matching score is believed to be the correct one. The final matching score is scaled between 0 and 99.

Details of the algorithm can be found in [64].

7.2.2 String Distance Based Matching

Each set of extracted minutia features is first converted into polar coordinates with respect to an anchor point. The two-dimensional (2D) minutia features are, therefore, reduced to a one-dimensional (1D) string by concatenating points in an increasing order of radial angle in polar coordinates. The string matching algorithm is applied to compute the edit distance [10] between the two strings. The edit distance can be easily normalized and converted into a matching score. This algorithm [2] can be summarized as follows:

(i) Rotation and translation are estimated by matching ridge segment (represented as planar curve) associated with each minutia in the input image with the ridge segment associated with each minutia in the template image. The rotation and translation parameters that result in the maximum number of matched minutiae pairs within a bounding box was used to define the estimated transformation and the corresponding minutiae are labeled as anchor minutiae, A_1 and A_2, respectively.

(ii) Convert each set of minutia into a 1D string using polar coordinates anchored at A_1 and A_2, respectively.

(iii) Compute the edit distance between the two 1D strings. The matched minutiae pairs are retrieved based on the minimal edit distance between the two strings.

(iv) Output the normalized matching score (in the range of 0-99) which is the ratio of the number of matched-pairs and the number of minutiae points.

7.2.3 Two Dimensional (2D) Dynamic Programming Based Matching

This matching algorithm is a generalization of the above mentioned string-based algorithm. The transformation of a 2D pattern into a 1D pattern usually results in a loss of information. Chen and Jain [67] have shown that fingerprint matching using 2D dynamic time warping can be done as efficiently as 1D string editing while avoiding the above mentioned problems with algorithm String. The 2D dynamic time warping algorithm can be characterized by the following steps:

(i) Estimate the rotation between the two sets of minutia features as in Step 1 of algorithm String.

(ii) Align the two minutia sets using the estimated parameters from Step 1.

(iii) Compute the maximal matched minutia pairs of the two minutia sets using 2D dynamic programming technique. The intuitive interpretation of this step is to warp one set of minutia to align the other so that the number of matched minutiae is maximized.

(iv) Output the normalized matching score (in the range of 0-99) which is based on only those minutiae that lie within the overlapping region. A penalty term is added to deal with unmatched minutia features.

7.2.4 Filterbank Based matching

This fingerprint matching algorithm is based on finding the Euclidean distance between the corresponding FingerCodes [10]. The translation invariance in the FingerCode is established by identifying the reference point. However, FingerCodes are not rotationally invariant. The approximate rotation invariance is achieved by cyclically rotating the features in the FingerCode itself. A single-step cyclic rotation of the features in the

FingerCode described by **Eq.(7.1)-(7.3)** corresponds to a feature vector which would be obtained if the image was rotated by 22.5^0. A rotation by R steps corresponds to a $R \times 22.5^0$ rotation of the image. A positive rotation implies counterclockwise rotation while a negative rotation implies clockwise rotation. The FingerCode obtained after R steps of rotation is given by

$$V_{i\theta}^{R} = V_{i'\theta'} \tag{7.1}$$

$$i' = (i + k - R) \bmod k + (i \quad div \quad k) \times k \tag{7.2}$$

$$\theta' = (\theta + 180 + 22.5^0 \times (-R)) \bmod 180^0 \tag{7.3}$$

where $V_{i\theta}^{R}$ is the rotated FingerCode, $V_{i'\theta'}$ is the original FingerCode, $k(=16)$ is the number of sectors in a band, $i \in [0,1,2,.....,79]$, and $\theta \in [0^0, 22.5^0, 45^0, 67.5^0, 90^0, 112.5^0, 135^0, 157.5^0]$. Details of the algorithm can be found in [10].

7.2.5 MDER-BP Algorithm for Matching

This algorithm is described in ***chapter-5***. The proposed algorithm is effective for fingerprint images Learning/Training and Matching/Testing and as well as the identification of the person. It is very much efficient method for poor fingerprint images especially for the criminals fingerprints investigation. This algorithm is very much suitable for off-line fingerprint identification system. In **chapter-8** the practical implementation of **Minimum Distance Error Rate BackPropagation (MDER-BP) Algorithm** [53] has been described widely.

Chapter Eight

IMPLEMENTATION OF FINGERPRINT IDENTIFICATION SYSTEM USING MDER-BP NEURAL NETWORK

Contents Page No.

8.1 Introduction

It was mentioned earlier that fingerprint verification system is one of the most popular biometric security method that provides a way to identify a person. In this work, we heed to ensure the identification of the fingerprints of only the right people. There are many commercial systems designed for person identification. The most popular systems are based on fingerprint, facial image, retina colour and signature recognition techniques etc [1, 10]. Fingerprint is the rigid and furrow patterns on the tip of a finger. It is a distinctive feature and remains invariant over a person's lifetime, excepts for cuts, burn and bruises. Fingerprint verification involves matching two fingerprint images in order to verify a

person's claimed identity. The two fundamental features on which fingerprint identification is based are:

(i) Fingerprint details are permanent and

(ii) Fingerprints of an individual are unique.

Fingerprint authentication requires acquiring and digitizing a fingerprint image. The digital image of the fingerprint includes several unique features in terms of rigid bifurcation and rigid ending, collectively referred to as minutiae [1]. According to the method of acquisition of fingerprint data, there are two types of fingerprint verification system (i) On-line and (ii) Off-line.

The fundamental difference between the on-line and off-line fingerprint verification system is the nature of the fingerprint samples used in each system. In the *on-line* fingerprint verification system, data about the rigid bifurcation and rigid ending of fingerprint and pressure of the fingertip are collected as a function of time by using on-line scanner [1, 6]. The fingerprint samples are analyzed in a dynamic environment. This allows on-line systems to collect real-time information.

An off-line fingerprint verification system is fed with a 2D static image [52] of a fingerprint as input from a digital camera or a scanner. The system analyzes digital image of a fingerprint using either the raw pixel data or some sort of representation of the pixel data. Only data that can be obtained directly from the fingerprint image is used in this system. There is no additional information given to it for fingerprint verification purpose.

There are several approaches for fingerprint verification process. Many researchers proposed and worked upon various approaches. Traditionally, minutiae, core, delta, crossover, bifurcation, ridge ending, pore, island, enclosure, bridge features of a fingerprints are considered to identify a person (**Fig.1.4** & **Chapter-2**). However, these features are accurately recognized when the features are prominent during scanning the fingerprint images. But in some cases due to different reasons good fingerprint images are not always available. In such cases it is difficult to recognize these features with

proper technique to get accurate information about the fingerprint. *In this research work in chapter-3, we proposed an alternative crude method Grid-Mapping Feature (**GMF**)* extraction [90].

8.2 Fundamental Steps of Fingerprint Verification System

Like all other verification systems the fingerprint verification system has two fundamental stages. They are the *Training stage* and the *Matching stage*. There are several approaches proposed by various researchers in order to accomplish the *Training* and *Matching* process. The *Training* and *Matching* stages are subdivided into following phases (**Fig.8.1**):

1. Fingerprint Sample Collection.
2. Image Pre-processing
   ((i) Filtering, (ii) Clipping (iii) Edge Detection and (iv) Thinning.)
3. GMF Extraction.
4. Training/Matching Stages [53].

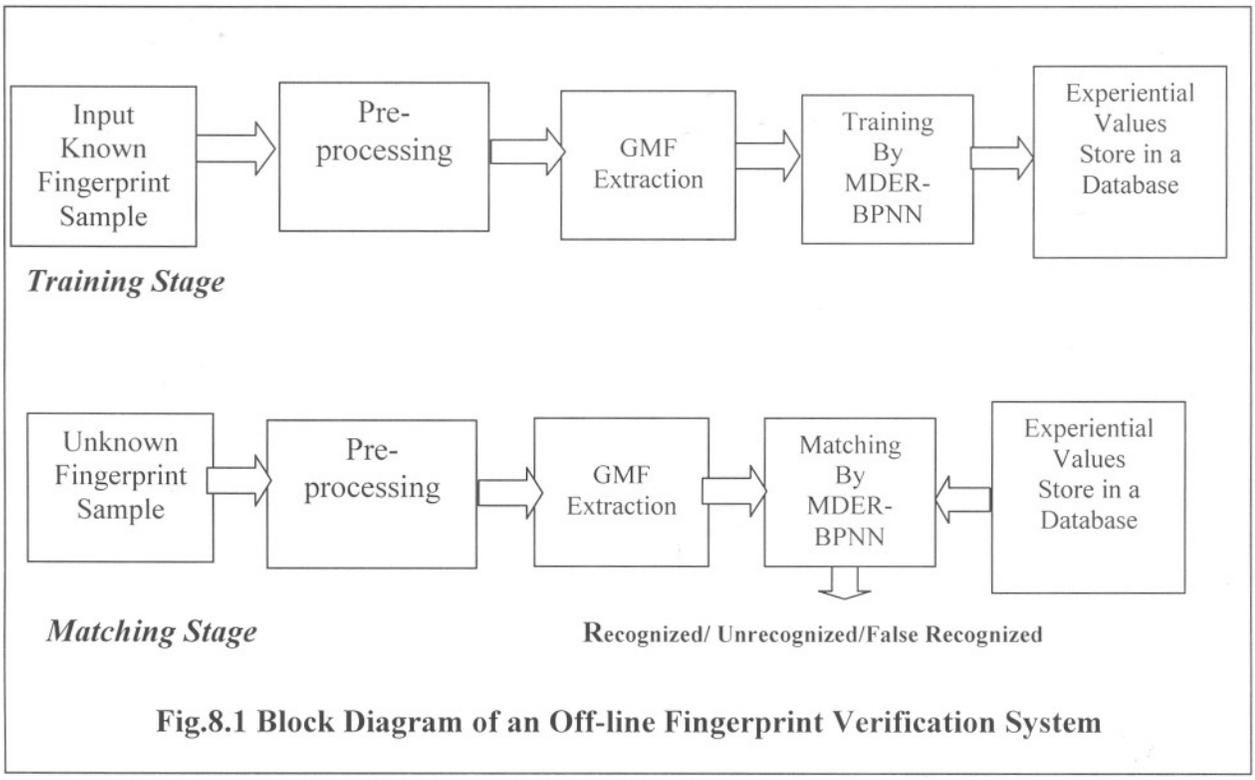


Fig.8.1 Block Diagram of an Off-line Fingerprint Verification System

8.3 Fingerprint Sample Collection

The fingerprint image acquired by the off-line process is known as the "***inked***" fingerprints while the image acquired by the on-line process is known as "***live-scan***" fingerprints. In the inked fingerprint acquisition method ink is applied to the finger and then pressed onto a paper to form an impression [53]. The paper is then scanned at 500-dpi resolution by a standard grayscale scanner.

8.4 Image Pre-Processing

Image pre-processing includes several stages of image processing. It includes Noise reduction, Edge detection, Clipping and Thinning [52]. When we are using fingerprint scanner we may have some unwanted scratch upon our scanned image. A fingerprint expert may need to remove such unwanted scratches from the image. Sometimes some part of ridge lines may be absent. For automatic enhancement process we need to develop some kind of algorithm to remove such kind of breaking line.

An *edge* [52] is the boundary between two regions with relatively distinct gray level properties. The idea underlying most edge-detection techniques is on the computation of a local derivative operator such as 'Roberts', 'Prewitt' or 'Sobel' [52] operators. In practice, the set of pixels obtained from the edge detection algorithm seldom characterizes a boundary completely because of noise breaks in the boundary lines and other effects that introduce spurious intensity discontinuities. Thus, edge detection algorithms typically are followed by linking and other boundary detection procedures designed to assemble edge pixels into meaningful boundaries.

In order to improve the quality of the poor fingerprint images the following steps are considered (**Fig.8.2 (a)**):

(i) Filtering,
(ii) Clipping,
(iii) Edge Detection,
(iv) Thinning.

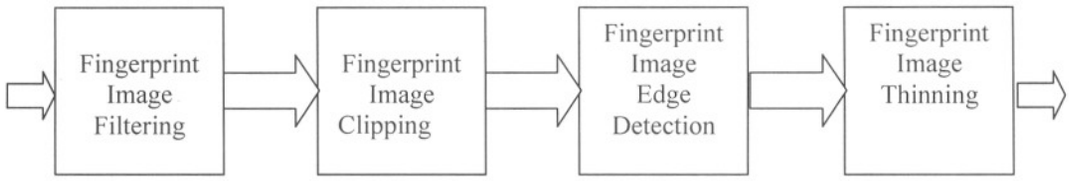
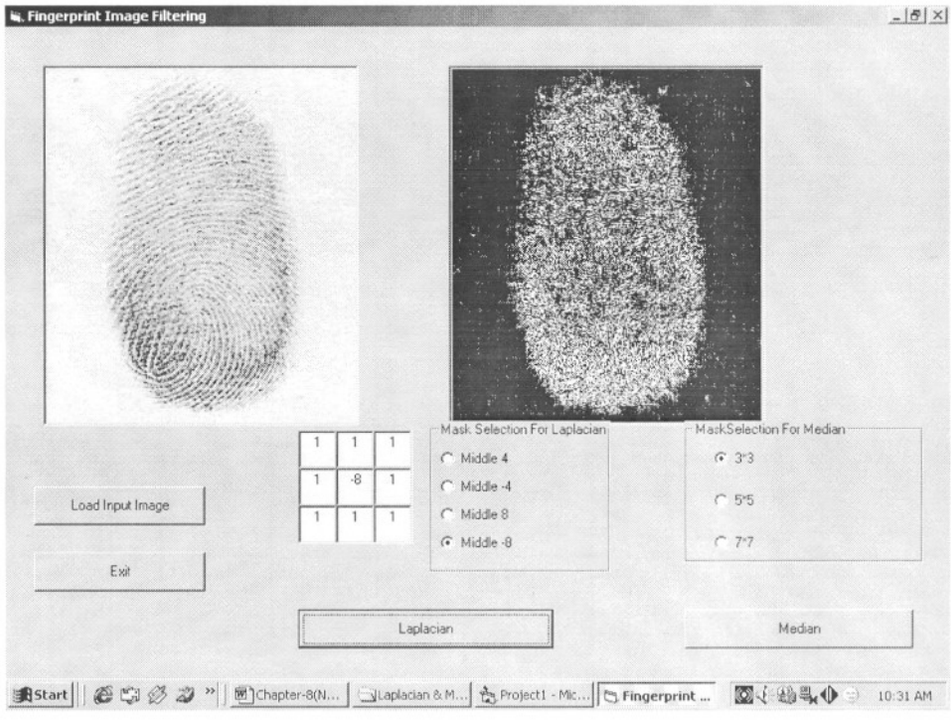


Fig.8.2(a): Block Diagram of Fingerprint Image Preprocessing

8.4.1 Fingerprint Filtering

When we are using fingerprint scanner, we may have some unwanted scratch upon our scanned image. A fingerprint expert may be needed to remove such unwanted scratch from the fingerprint image. Sometimes few part of ridge lines may be unavailable. For automatic enhancement process we need to develop some kind of algorithm to remove such kind of breaking line. In this work, noisy fingerprint images are filtered by using the Laplacian spatial filtering [52] technique (**Fig.8.2(b)**). In ***chapter-3***, fingerprint filtering process has been described elaborately.



8.2(b): Screen Shot of Fingerprint Filtering

8.4.2 Fingerprint Clipping

Clipping is another important pre-processing step in image processing and pattern recognition system. It is quite clear to all that we are not interested in whole portion of the scanned image. We are only interested into the portion of fingerprint data. So we need to clip the fingerprint data. The scanned image size may be large. We can say the necessity of clipping in pre-processing stage is:

a. Accurate pattern matching.
b. Faster processing.
c. Position independent pattern matching.

Now the problem is how to clip the fingerprint data from the scanned image. Many researchers do the clipping process in various ways. Some use boundary detection algorithm, others develop their own algorithm. In our work we just develop our simple technique for clipping purpose. Our clipped image should be a rectangle. We need to determine the coordinate position of four boundary points.

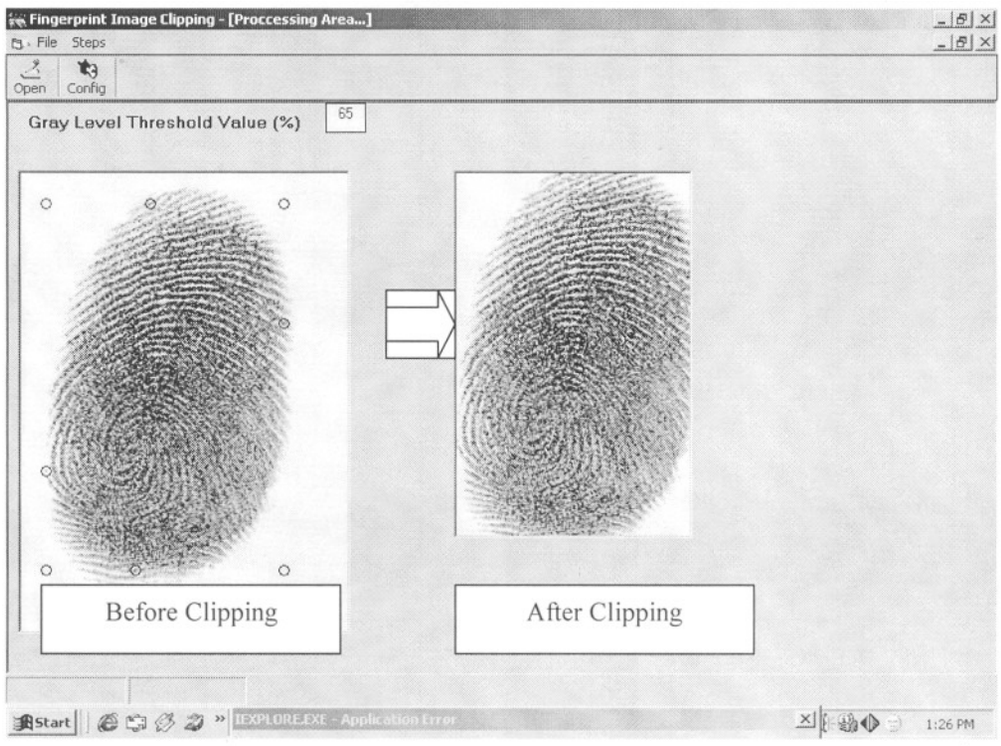


Fig.8.3: Screen Shot the Fingerprint Image Clipping.

For this purpose we define the fingerprint image into four regions, left, right, top and bottom. First of all to determine ***TopMost*** of top region we start scanning from co-ordinate position (0, 0) to right of the image. Whenever we find any black pixels we mark it as ***TopMost*** and stop scanning more. If we find no pixel in first row we move to scan through the second row, and so on.

Now we want to determine the ***BottomMost*** point. We start from (x, y) where x and y be the maximum value of coordinate position. Starting from (x, y) we move right to left. Whenever we find any black pixel we mark it as ***BottomMost***. If any pixel is not found in first row then we go through second row, and so on (**Fig.3.6**).
In order to determine the ***LeftMost*** we again start from (0, 0) coordinate position and scan through top to bottom. The first black pixel found in the scanning is marked as ***LeftMost***. If no black pixel in first column then we will scan the second column (i.e., from left to right).

The ***RightMost*** is found in the same way, except we start scanning from $(x, 0)$ position and instead of scanning next column we move to previous column (i.e., from right to left). **Fig.8.3** shows the screen shot of fingerprint image clipping. In ***chapter-3***, fingerprint clipping technique has been shown widely.

8.4.3 Fingerprint Edge Detection

In this section, we discuss about how to detect edge of a fingerprint image. It is one of the vital issues that ensure to extract the feature of a fingerprint image properly. The edge detection [73] is one of the gray-level discontinuities of image segmentation. Other two the gray-level discontinuities of image segmentation are point and line detection. Although point and line detection certainly are important in any discussion on segmentation, edge detection [52] is by far the most common approach for detecting meaningful discontinuities in gray level. In this section we discuss approaches for implementing first and second-order digital derivatives for the detection of edges in a fingerprint image.

An edge [52] is a set of connected pixels that lie on the boundary between two regions. However, an edge is a local concept whereas a region boundary, owing to the way it is defined, is a more global idea. A reasonable definition of edge requires the ability to measure gray-level transitions in a meaningful way. An ideal edge has the properties of the model shown in **Fig.3.8(a)**.

In practice, optics, sampling and other image acquisition imperfections yield edges that are blurred, with the degree of blurring being determined by factors such as the quality of the image acquisition system, the sampling rate, illumination conditions under which the image is required. As a result, edges are more closely modeled as having a ramplike profile, such as the one shown in **Fig.3.8(b)**.

The slope of the ramp [52] is inversely proportional to the degree of the blurring in the edge. An edge point now is any point contained in the ramp, and an edge would then be a set of such points that are connected. The thickness [52] of the edge is determined by the length of the ramp, as it transitions from an initial to the final gray-level. This length is determined by the slope, which, in turn, is determined by the degree of burring. This makes sense: blurred edges tend to be thick and sharp edges tend to be thin.

We are led to the conclusion that, to be classified as a meaningful edge point, the transition in gray-level associated with that point has to be significantly stronger than the background at that point. Since, we are dealing with local computations, the method of choice to determine whether a value is significant or not is to use a threshold. Thus, we define a point in a fingerprint image as being an edge point if its two-dimensional first-order derivative is greater than a specified threshold. A set of such points that are connected according to a predefined criterion of connectedness is by definition an edge. The term edge segment generally is used if the edge is short in relation to the dimensions of the fingerprint image.

A key problem in segmentation is to assemble edge segments into longer edges. An alternate definition if we elect to use the second-derivative is simply to define the edge

points in a fingerprint image as the zero crossings of its second derivative. The definition of an edge in this case is the same as above. The first-order derivatives in a fingerprint image are computed the gradient. Second-order derivatives are obtained using the Laplacian.

A key problem in segmentation is to assemble edge segments into longer edges. An alternate definition if we elect to use the second-derivative is simply to define the edge points in a fingerprint image as the zero crossings of its second derivative. The definition of an edge in this case is the same as above. The first-order derivatives in a fingerprint image are computed the gradient. Second-order derivatives are obtained using the Laplacian. **Fig.8.4** shows the screen shot of fingerprint image edge detection.

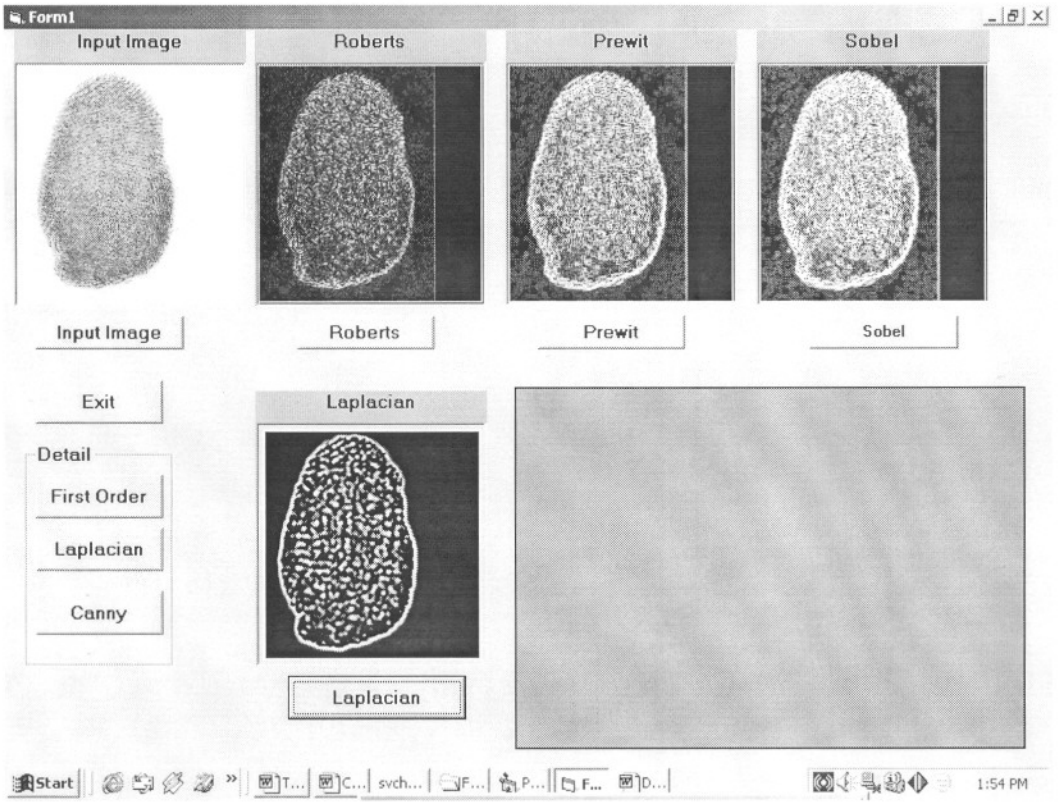


Fig.8.4: Screen Shot of Fingerprint Edge Detection.

8.4.4 Fingerprint Image Thinning

Thinning is another important step to extract feature of fingerprint images. This approach representing the structural shape region is reduced like a graph. This reduction may be accomplished by obtaining the *skeleton* of the region via a thinning algorithm. Thinning algorithms iteratively delete edge points of a region subject to the constraints that deletion of these points:

(1) does not remove end points,

(2) does not break connectedness, and

(3) does not cause excessive erosion of the region.

In ***chapter-3***, fingerprint image thinning process has been described elaborately. The screen shot of fingerprint image thinning is shown in **Fig.8.5**.

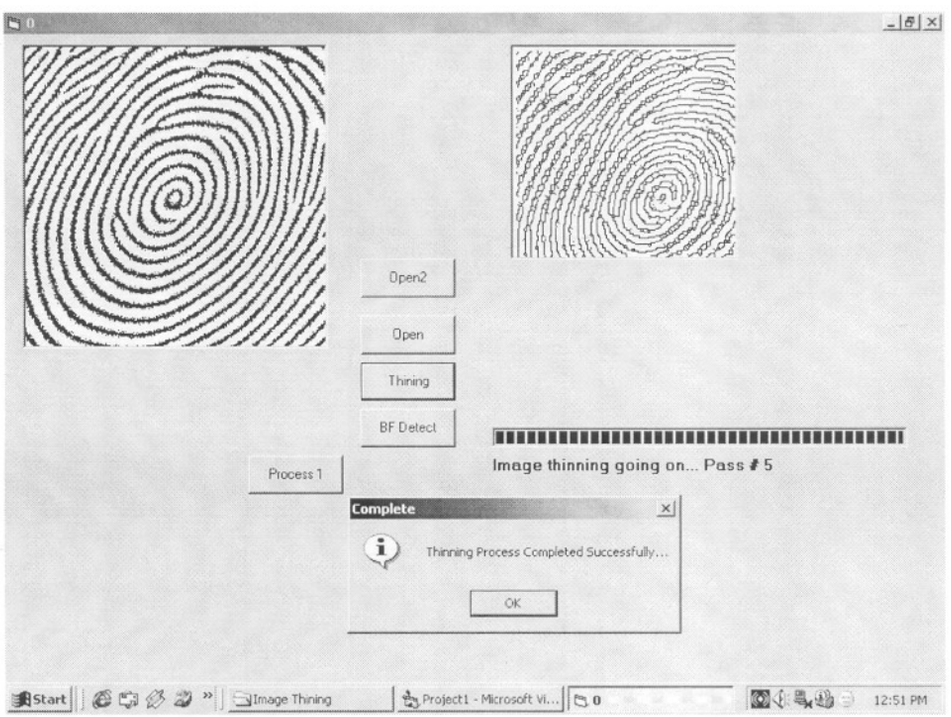


Fig.8.5: Screen Shot of Fingerprint Image Thinning

8.5 Grid-Mapping Feature (GMF) Extraction

The poor fingerprint images are process by using filtering, clipping and edge detection. The following process has been adopted to extract feature for fingerprint images. The fingerprint image is transformed to 300X260-pixel image by using image-scaling process. Then the fingerprint is processed through grid mapping with fixed square/rectangle cells, such as 16x16 or 32X32 square/rectangle areas (**Fig.3.15**). If each small square/rectangle cell contains more than 45~55% black pixels, then its digital value is considered as 1 otherwise 0. An extraction algorithm is used to extract grid mapping features from the gray scale fingerprint image by examining the neighborhood pixels. These binary values are used to train the Multilayer neural network using MDER-BackPropagation Algorithm.

In ***chapter-3*** fingerprint ***GMF Extraction*** technique has been described elaborately. The screen shot of the GMF Extraction process is shown in **Fig.8.6**.

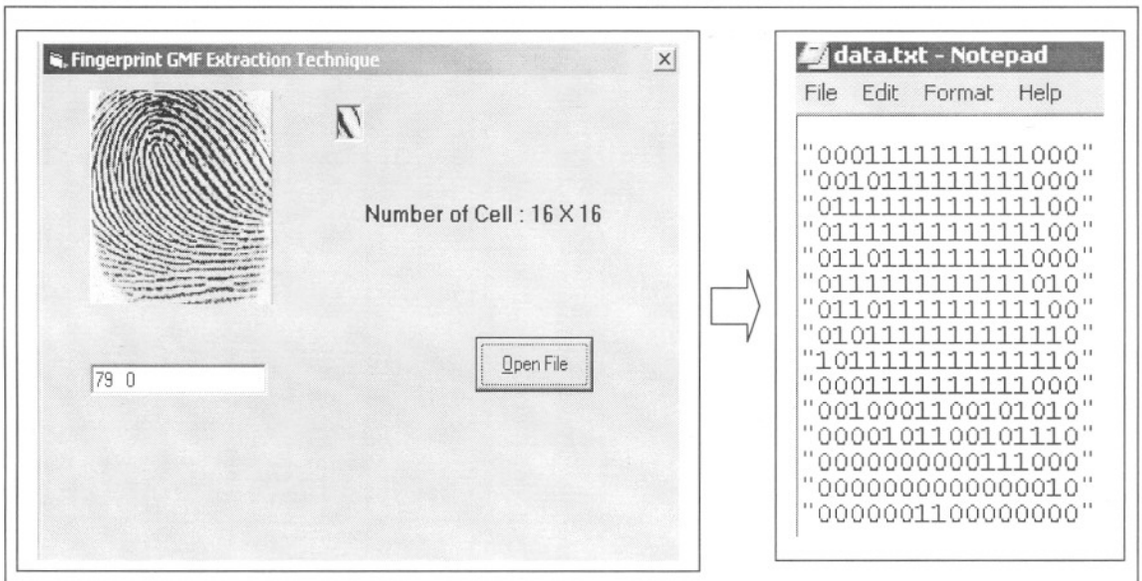


Fig.8.6: Practical Screen Shot of GMF extraction from a sample fingerprint image of 16×16 Feature Matrixes.

8.6 Neural Network Models for Fingerprint Identification

Neural Network is a powerful tool to use in modern artificial intelligent systems. Nowadays, many applications that involve pattern recognition, feature mapping, clustering, classification, ***fingerprint matching*** etc. use Neural Networks as an essential component. In recent decades, several types of S***upervised and Unsupervised Neural Networks*** [62] have been developed. **Error BackPropagation Neural Network**, **Kohonen Self-Organizing Network** [59, 60]**, Hopfield Network** [60]**, Adaptive Resonance Theory (ART)** [58, 59] are some of basic networks that have been developed and are used in many applications. In ***chapter-5***, some supervised neural network models have described. In ***chapter-6***, the unsupervised models of neural network have been described widely.

In this thesis, we take 20 fingerprints of a finger from 20 different persons. In ***chapter-6***, we use all the 20 samples of fingerprint images for each person. But in ***chapter-8***, here we use only 5 samples of fingerprint images from 20 different persons.

8.6.1 Topology

In this work, the three-layered, feed-forward ***Minimum Distance Error Rate BackPropagation Neural Network*** (MDER-BPNN), [53] which is fully interconnected by layers, has been considered. Thus, there are no connections that bypass one layer to go directly to a later layer. The MDER-BPNN is called a mapping network because it is able to compute some functional relationship between its input and output. **Fig.8.7** shows the three-layer MDER-BPNN architecture. The input vector is represented by $X[a][i]$, where a is the fingerprint of a person and $[i]$ is the **GMF** pattern matrixes, i,e, 16×16 array (i = 0, 1, 2, 3, - - - - , 255), formed 20×256 pattern matrixes (taking one sample from one person). For example, thus $X[7][120]$ represent 120 th input **GMF** pattern element for person "7".

The target output is represented by $T[a][k]$, where $[a]$ is the fingerprint of a person (in this research, we consider fingerprints of 20 persons, i.e., a = 0, 1. 2, 3, 4, 5, ... , 18, 19) and $[k]$ is the target output pattern matrices of a person ID, i.e., k = 0, 1, 2, 3, 4).

Target Output of a person Fingerprint:

/*Person One*/	**/*Person Two*/**	.....	**/*Person Twenty*/**
T[0][0]=0;	T[1][0]=0;		T[19][0]=1;
T[0][1]=0;	T[1][1]=0;		T[19][1]=1;
T[0][2]=0;	T[1][2]=0;		T[19][2]=0;
T[0][3]=0;	T[1][3]=0;		T[19][3]=0;
T[0][4]=0;	T[1][4]=1;		T[19][4]=1;

8.6.2 Training the fingerprint with Minimum Distance Error Rate-BPNN

This Algorithm is similar to BackPropagation algorithm [60, 62], but the major change is to calculate error rate. Here the *Euclidean distance* is used to calculate the error rate, which is the difference between the present output and the target output. The proposed algorithm is as follows:

The learning of *MDER-BP* [53] Algorithm has been accomplished by error Back Propagation Neural Network (**Fig.8.7**). The weight vectors W_{ij} and W_{jk} are the weighted values between layers i and j, j and k respectively. The network is provided with the input patterns and also the desired respective output patterns. The input patterns a **(GMF**16×16 **pattern matrices)** are connected to hidden processing elements (PEs) through the weights W_{ij}.

In the *hidden layer*, each PE computed the weighted sum according to the equation, which is given by

$$net_{aj} = \sum W_{ij} O_{ai} \qquad (8.1)$$

where O_{ai} is the input of unit i for pattern number a. The threshold, uh_j of each PE is then added to its weighted sum to obtain the activation $active_j$ of that PE i,e,

$$active_j = net_{aj} + uh_j \qquad (8.2)$$

where uh_j is the hidden *threshold weight* for jth PEs. This activation determines whether the output of the respective PE is either 1 or 0 (fires or not) by using a sigmoid function,

$$O_{aj} = \frac{1}{1 + e^{-k_1 * active_j}} \qquad (8.3)$$

where k_1 is called the *spread factors*, these O_{aj} are then serve as the input to the output computation. Signal O_{aj} are then fan out to the output layer according to the relation,

$$net_{ak} = \sum W_{jk} O_{aj} \tag{8.4}$$

and the output threshold weight uo_k for k-th output PEs is added to it to find out the activation $active_{(k)}$

$$active_k = net_{ak} + uo_k \tag{8.5}$$

The actual output O_{ak} is computed using the same sigmoid function,

$$O_{ak} = \frac{1}{1 + e^{-k_2 * active_k}} \tag{8.6}$$

Here another *spread factor* k_2 has been employed for the output units.

In the *second stage*, after completing the *feed-forward propagation,*

an error is computed by comparing the output O_{ak} with the respective target t_{ak} ,i.e

$$\delta_{ak} = \sqrt{\sum_{k=0}^{k-1} (t_{ak} - O_{ak})^2} \tag{8.7}$$

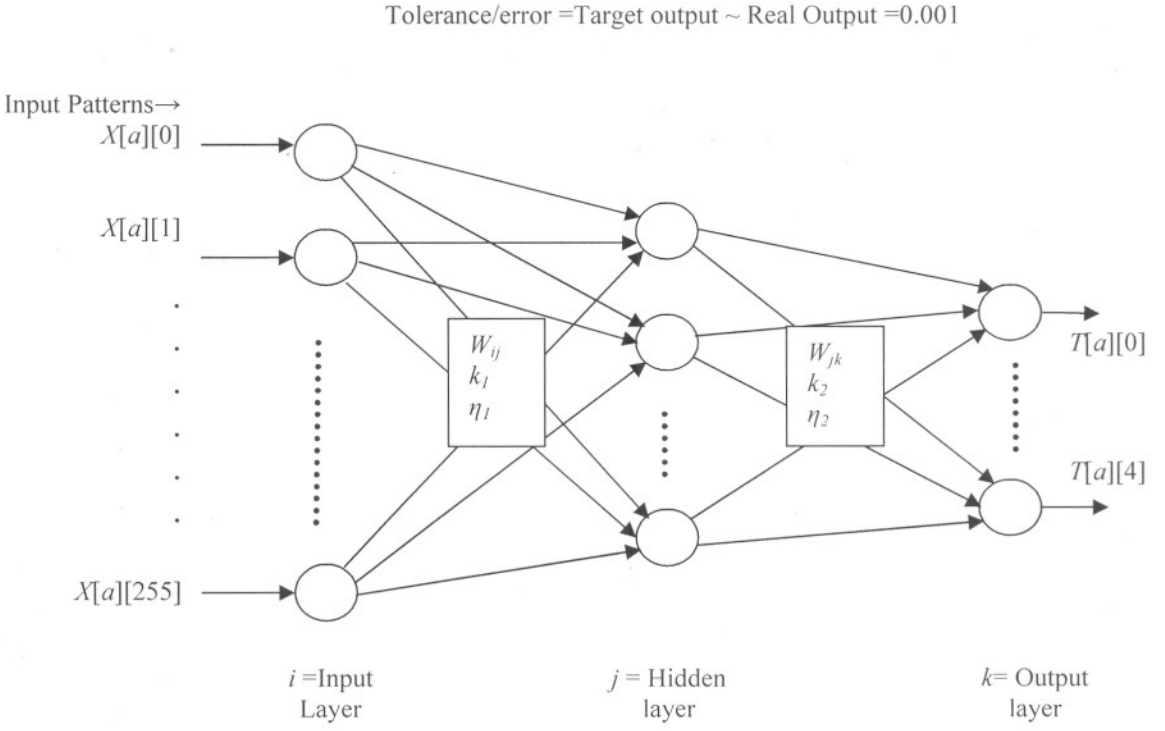


Fig.8.7 MDER- BackPropagation Neural Network

This error is then used to adjust the weight vector W_{jk} using the equation,

$$\Delta W_{jk} = \eta_2 k_2 \delta_{ak} O_{aj} O_{ak} (1 - O_{ak}) \tag{8.8}$$

where, $\int'(activeo_k) = k_2 O_{ak}(1 - O_{ak})$ the derivation of sigmoid function and η_2 is the *learning factor* of the network.

The weight vector W_{jk} is then adjusted to $W_{jk} + \Delta W_{jk}$. For the threshold weight of the output PE, similar equation is applied,

$$\Delta uo_k = \eta_2 k_2 \delta_{ak} O_{ak} (1 - O_{ak}) \tag{8.9}$$

and the *new threshold weight* equaled $uo_k + \Delta uo_k$.

In the *next step*, this error and the adjusted weight vector W_{jk} are feedback to the hidden layer to adjust the weight vector W_{ij} and threshold weight uh_j. In this layer change in weight vector W_{ij} is computed by using equation,

$$\Delta W_{ij} = \eta_1 k_1 O_{ai} O_{aj} (1 - O_{ak}) \sum \delta_{ak} W_{jk} \tag{8.10}$$

where, $\int'(activeh_j) = k_1 O_{ai}(1 - O_{aj})$ and η_1 is the *learning factor* of the network. The weight vector W_{ij} is then adjusted to $W_{ij} + \Delta W_{ij}$. For the threshold weights of the hidden PEs, similar equation is applied

$$\Delta uh_j = \eta_1 k_1 (1 - O_{aj}) \sum \delta_{ak} W_{jk} \tag{8.11}$$

and *new threshold weights* are calculated $uh_j + \Delta uh_j$.

The properties of sum-squared error equation dictate that as output approaches its maximum or minimum value, adjustments to individual weights become less pronounced. This is a testament to the stability of the *MDER-BackPropagation algorithm*. The significance of the training process is that, as the network trains, the nodes in the intermediate layers organize themselves such that different nodes learn to recognize different features of the total input space. Finally, these experiential weighted and threshold values are stored in a file. The screen shot of fingerprint training process is shown in **Fig.8.8**.

Minimum Distance Error Rate (MDER) Calculation:

In this algorithm, the GMF features of fingerprint images are applied as an input vectors for training and testing the network for verification of the fingerprint images. Either it corresponds to the correctly recognized the fingerprint of a person or it fail to recognize the fingerprint of a person. The technique used for this purpose is the expression for Euclidean error-distance calculation is given below:

$$\delta_{ak}(t_{ak}, O_{ak})_{euc} = \sqrt{\sum_{k=0}^{k-1}(t_{ak} - O_{ak})^2} \tag{8.12}$$

Where,

t_{ak} and O_{ak} are the two vectors, *target* and *output* respectively.

$\delta_{ak}(t_{ak}, o_{ak})_{edu}$ is the error-distance between the vectors.

k is the dimensionality of the vectors.

In this case, t_{ak} is the target vector of all the patterns and O_{ak} is the output of one particular fingerprint of a person. If the pattern (fingerprint) is corresponding to the minimum error-distance, then it is recognized.

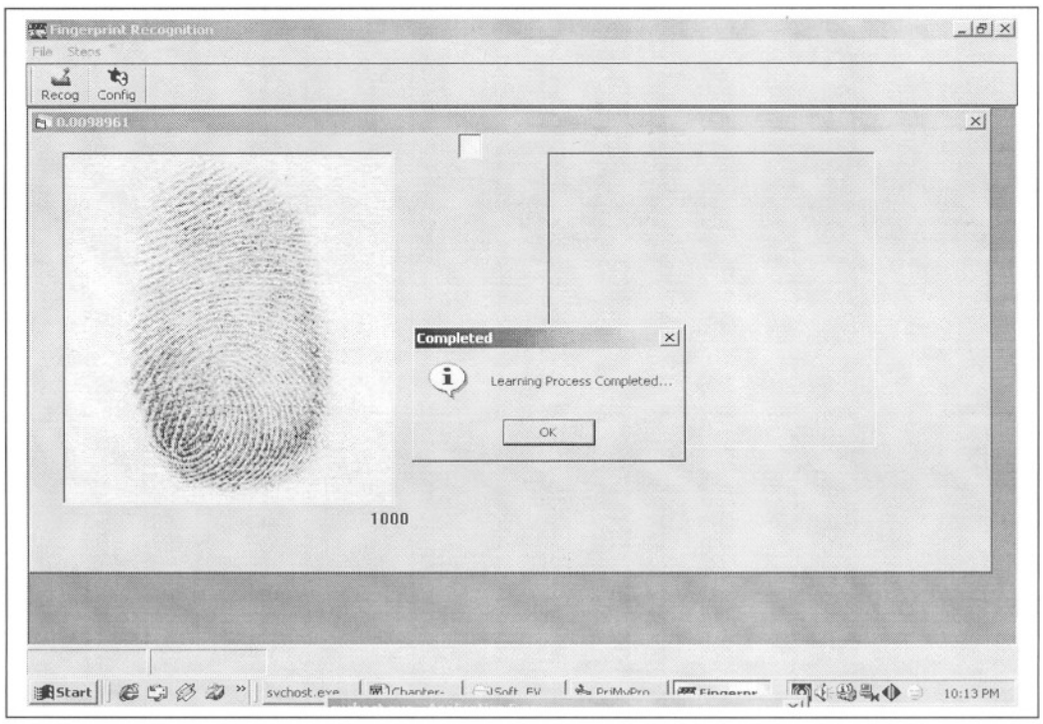


Fig.8.8 The Screen Shot of Fingerprint Training/Learning Process

8.6.3 Matching the Fingerprints

In the ***Matching Stage*** the weighted and threshold values are initialized in a file. In the same way *Minimum Distance Error Rate BackPropagation Neural Network* [53] is connected and the fingerprints are tested comparing with the present outputs and previous experiential values. Lastly, we get the result is the form whether the network has ***recognized*** or ***unrecognized*** or ***false recognized*** the fingerprint of a person.

In this work, the algorithm is implementing in the Microsoft Visual Basic 6.0 programming language. All programs are written in the Microsoft Visual Basic 6.0 programming language in order to interface the work nicely. Different techniques have been considered for finding weights and threshold values. These are

(i) When spread factors (k_1 and k_2) are varied then learning rates (η_1 and η_2) and hidden units are held constant.

(ii) When learning rates (η_1 and η_2) are varied then spread factors (k_1 and k_2) and hidden units are held constant.

(iii) When hidden units are varied then spread factors (k_1 and k_2) and learning rates (η_1 and η_2) are held constant.

In this thesis, several hidden units were considered. Lastly the weights and threshold values for 3, 5, 7, 9, 11, 13, 15 and 17 hidden units are stored for further used, to recognize the fingerprint images.

8.7 Experimental Results

In this fingerprint verification system, fingerprints are taken from 20 different people. From each person 5 fingerprints of a finger (Thumb) are collected. From the fingerprint file each fingerprint is separated from its neighbors to produce a fingerprint database file. After extracting the features from each fingerprint, the ***GMF pattern matrix*** is applied to the input of ***MDER-BackPropagation Neural Network*** for training purpose. The network ***spread factors*** (k_1 and k_2), ***learning rates*** (η_1 and η_2) and ***hidden units*** are varied. The ***three conditions*** of this experiment regarding the variable parameters have been mentioned above. During the recognition period, the error tolerance level is set to

0.001. After completing the training, the updated weights and threshold values are stored in a file, which are used in the fingerprint matching process. To verify fingerprint, new fingerprint image is taken from a person and features are extracted to form a feature matrix. The feature matrix is then applied to the input of the MDER-BackPropagation Neural Network [53] to observe whether the system ***recognized*** the fingerprint or ***unrecognized*** not or show ***false recognition***. The screen shot of ***recognized*** or ***unrecognized*** or ***false recognized*** the fingerprints of a person are shown in **Fig.8.9**, **Fig.8.10** and **Fig.8.11** respectively.

The accuracy of the system has been described by the following equation:

$$\% \text{ of Accuracy of the System} = \frac{\text{Total No. of Recognized Fingerprint Samples}}{\text{Total No. of Fingerprint Samples}} \text{ X } 100$$

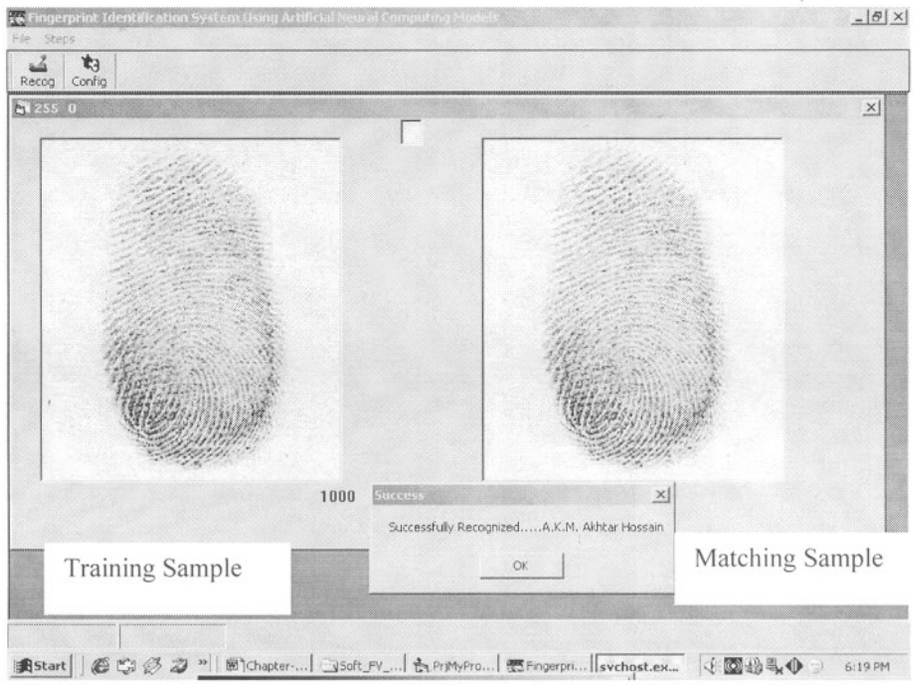


Fig.8.9: The screen Shot of Recognized the Fingerprint of a Person

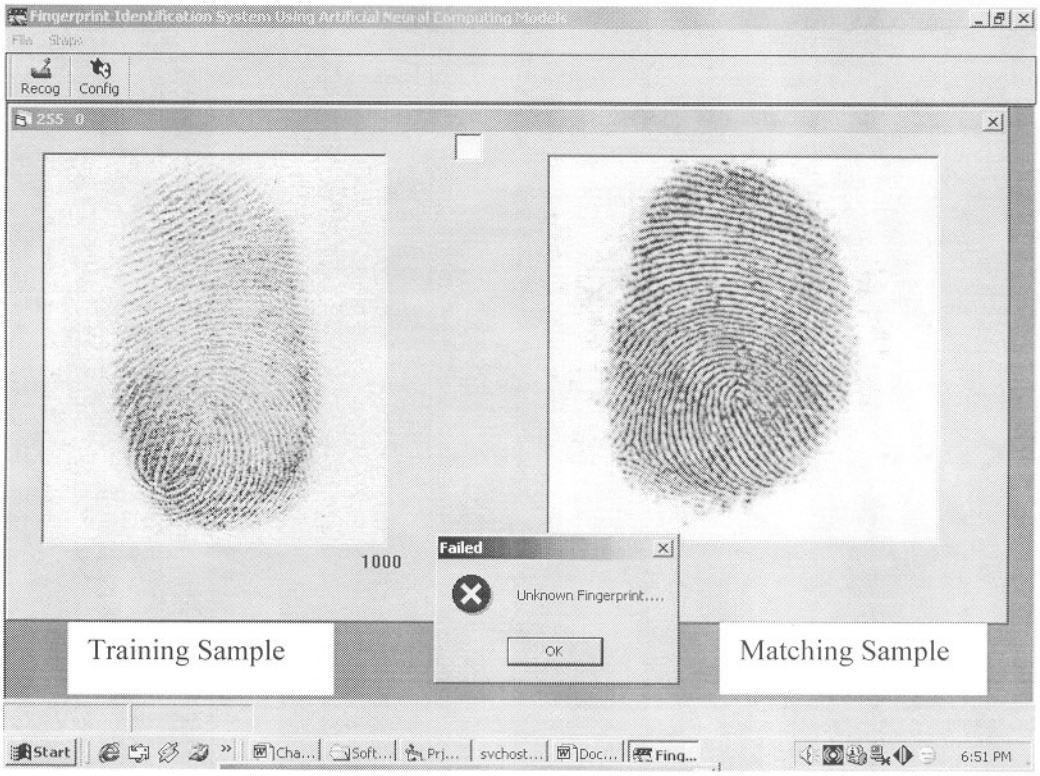


Fig.8.10: The screen Shot of Unrecognized Fingerprint of a Person

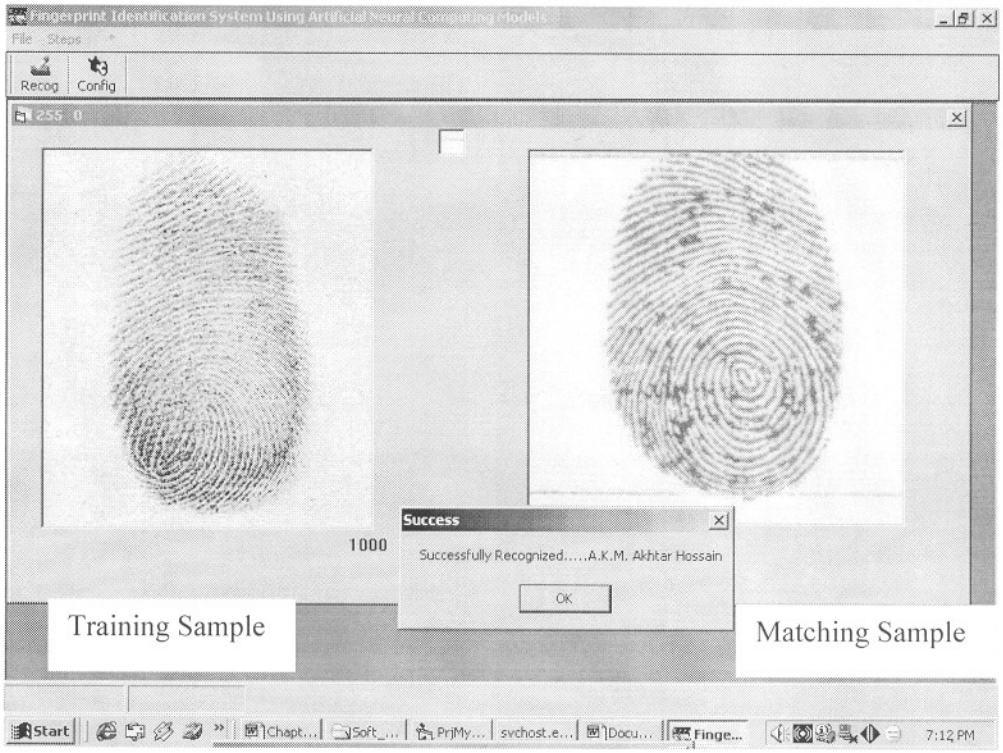


Fig.8.11: The screen Shot of False Recognized Fingerprint of a Person

In the following **Table-8.1**, shows the results adopting the first condition as mentioned above (*(i) When spread factors (k_1 and k_2) are varied then learning rates (η_1 and η_2) and hidden units are held constant*).

. Table-8.1: The Experimental Results with varying spread factors (k_1 and k_2).

Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.3$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) Hidden Units, $j = 9$.						
No. of person	No. of fingerprint samples from each person	Total no. of fingerprint samples	No. of recognized samples	No. of unrecognized samples	No. of false recognized samples	Accuracy of the system
20	5	100	80	12	08	80%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.4$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) Hidden Units, $j = 9$.						
20	5	100	83	10	07	83%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.5$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) Hidden Units, $j = 9$.						
20	5	100	87	08	05	87%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) Hidden Units, $j = 9$.						
20	5	100	91	05	04	91%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.7$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) Hidden Units, $j = 9$.						
20	5	100	86	10	04	86%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.8$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) Hidden Units, $j = 9$.						
20	5	100	80	13	07	80%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.9$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) Hidden Units, $j = 9$.						
20	5	100	78	14	08	78%

In the following **Table-8.2**, shows the results adopting the second condition as mentioned above (*(ii) When learning rates (η_1 and η_2) are varied then spread factors (k_1 and k_2) and hidden units are held constant*).

Table-8.2: The Experimental Results with varying Learning Rates (η_1 and η_2).

Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) **Learning Rates, $\eta_1 = \eta_2 = 0.3$** and (iii) Hidden Units, $j = 9$.						
No. of person	No. of fingerprint samples from each person	Total no. of fingerprint samples	No. of recognized samples	No. of unrecognized samples	No. of false recognized samples	Accuracy of the system
20	5	100	75	15	10	75%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) **Learning Rates, $\eta_1 = \eta_2 = 0.4$** and (iii) Hidden Units, $j = 9$.						
20	5	100	82	11	08	82%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) **Learning Rates, $\eta_1 = \eta_2 = 0.5$** and (iii) Hidden Units, $j = 9$.						
20	5	100	86	08	06	86%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) **Learning Rates, $\eta_1 = \eta_2 = 0.6$** and (iii) Hidden Units, $j = 9$.						
20	5	100	87	08	05	87%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) **Learning Rates, $\eta_1 = \eta_2 = 0.7$** and (iii) Hidden Units, $j = 9$.						
20	5	100	89	06	04	89%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) **Learning Rates, $\eta_1 = \eta_2 = 0.8$** and (iii) Hidden Units, $j = 9$.						
20	5	100	91	06	03	91%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) **Learning Rates, $\eta_1 = \eta_2 = 0.9$** and (iii) Hidden Units, $j = 9$.						
20	5	100	87	09	06	87%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) **Learning Rates, $\eta_1 = \eta_2 = 1.0$** and (iii) Hidden Units, $j = 9$.						
20	5	100	84	07	07	84%

In the following **Table-8.3**, shows the results adopting the second condition as mentioned above (*(iii) When hidden units are varied then spread factors (k_1 and k_2) and learning rates (η_1 and η_2) are held constant*).

Table-8.3: The Experimental Results with varying Hidden Units (j).

Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) **Hidden Units, $j = 3$.**						
No. of person	No. of fingerprint samples from each person	Total no. of fingerprint samples	No. of recognized samples	No. of unrecognized samples	No. of false recognized samples	Accuracy of the system
20	5	100	79	11	10	79%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) **Hidden Units, $j = 5$.**						
20	5	100	82	11	08	82%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) **Hidden Units, $j = 7$.**						
20	5	100	86	08	06	86%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) **Hidden Units, $j = 9$.**						
20	5	100	91	05	04	91%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) **Hidden Units, $j = 11$.**						
20	5	100	90	06	04	90%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) **Hidden Units, $j = 13$.**						
20	5	100	87	08	05	87%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) **Hidden Units, $j = 15$.**						
20	5	100	85	10	05	85%
Learning /Matching Parameters: (i) Spread Factors, $\kappa_1 = \kappa_2 = 0.6$, (ii) Learning Rates, $\eta_1 = \eta_2 = 0.8$ and (iii) **Hidden Units, $j = 17$.**						
20	5	100	84	10	06	84%

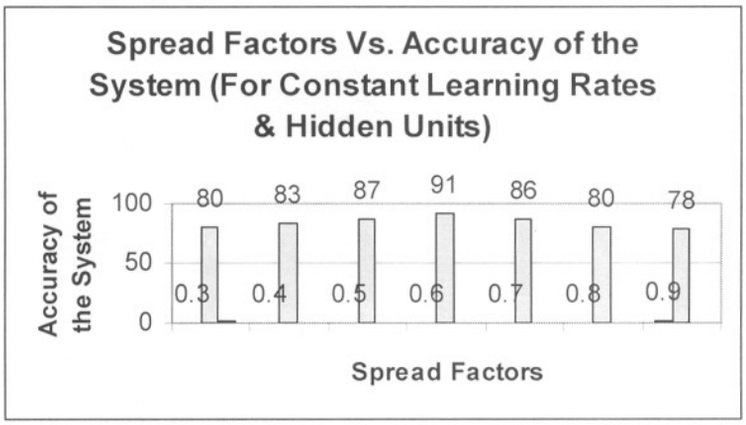


Fig.8.12: The graphical representation of Spread Factors Vs. Accuracy of the System based on Table-8.1

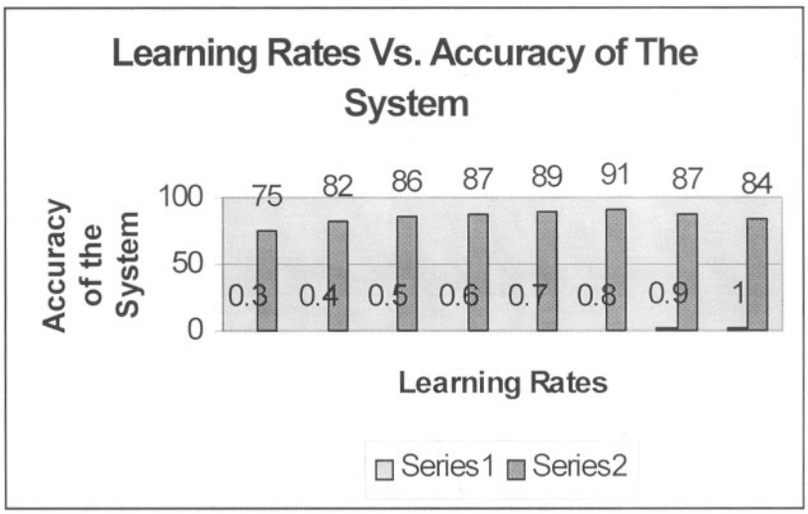


Fig.8.13: The graphical representation of Learning Rates Vs. Accuracy of the System based on Table-8.2

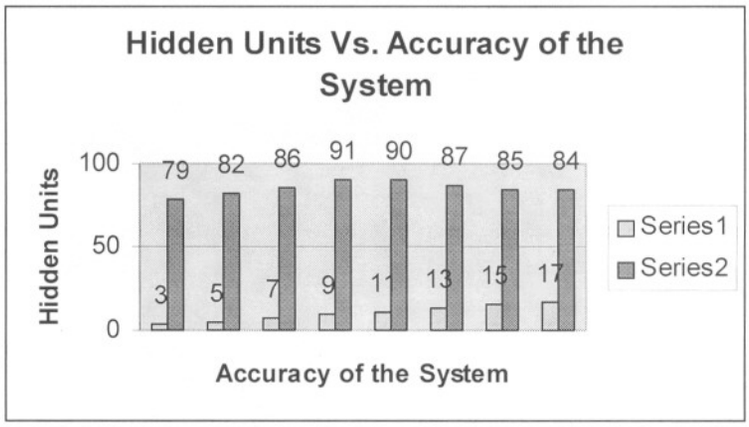


Fig.8.14: The graphical representation of Learning Rates Vs. Accuracy of the System based on Table-8.3

Chapter Nine

DISCUSSION AND CONCLUSION

Contents **Page No.**

9.1 Discussion regarding the work

The general discussion of this work is based on the experimental results of ***chapter-8*** and ***chapter-6*** and also other concepts proposed in the theorem. In this work, we consider the samples of poor fingerprint images. For fingerprint feature extraction and for training and matching techniques two theorems are proposed.

(1) The Grid Mapping Feature (GMF) extraction method for poor fingerprint images.

(2) Another one is the Minimum Distance Error Rate BackPropagation (MDER-BP) Algorithm for pattern (fingerprint) learning and matching.

In ***chapter-3***, The GMF extraction process has been described sequentially. The basic problem of this GMF extraction process is that the gray-scale values of the same fingerprints are different due to the uneven inking of the finger as well as the pressure when the finger is pressed on to the paper. During the scanning period the properties of scanner may be different. Some of the fingerprints are too much poor to extract correct GMF from those images. The 16 × 16 pattern matrices of GMF extraction of a fingerprint image may be unfit to get correct result for these poor fingerprint images. We should take other pattern matrices like 32 × 32. But we do not work with 32 × 32 pattern matrices of GMF technique due to bulky inputs of the network as well as slow speed of our computer. If the processor speed were very high (such as 4GHz) then in it could work with these properties of poor fingerprints.

In **chapter-8**, the practical implementation of proposed two methods has been simulated for finding experimental results. Form the observation of the experimental results we have found different types of concepts regarding the network. *The **GMF** inputs are heavily depended on the parameters like learning rates, spread factors, hidden units and iterations, of the **MDER-BP** Neural Network.* In this research, we varied only one parameter while other two parameters remained constant. These results were shown in Table-8.1, 8.2, and 8.3.

(1) For the experimental results from **Table-8.1** we have seen that the spread factors gives the highest efficiency of the system 91% when spread factors value k_1 and $k_2 = 0.6$ while the value of learning rates η_1 and $\eta_2 = 0.8$ and the number of hidden units $j = 9$.

(2) From the **Table-8.2** we have observed that the learning rates give the highest efficiency of the system 91% when learning rates value η_1 and $\eta_2 = 0.8$ while the value of spread factors k_1 and $k_2 = 0.6$ and the number of hidden units $j = 9$.

(3) From the **Table-8.3** we have observed that the hidden units give the highest efficiency 91% when number of hidden units $j = 9$ while the value of spread factors k_1 and $k_2 = 0.6$ and the value of learning rates η_1 and $\eta_2 = 0.8$.

From this study we have observed that the accuracy of the system is the same for three different conditions while the parameters are of the same value. The graphical presentations from the experimental results, which were found in Table-8.1, 8.2 and 8.3, were shown in **Fig.8.12**, **8.13** and **8.14** respectively.

9.2 Merits and Demerits of this system

9.2.1 Merits:

The system significantly recognized the right person with his/her poor fingerprint images. The main advantage of the neural computing system is that, it can distribute its experiential values all over the network. So it is more secured to store its learning

information in a database while the other existing system could not support this kind of facility.

9.2.2 Demerits:

The system is much more complicated in comparison with other exiting fingerprint verification system. It is also heavily depended on its parameters so that lot of manual change of values can be executed during learning period. The scale of gray level of the same person's fingerprint images often can cause different values of GMF extraction. This is the basic problem to identify the correct person. The system sometimes recognized false person, which can occur in some cases due to the error rate of the system shown in the experimental results.

9.3 Conclusions

The proposed ***Grid Mapping Feature-based*** fingerprint matching system gives an acceptable accuracy in off-line identification system for fingerprint images with poor features. It is also proved that the proposed ***MDER-BP Algorithm*** has also shown its capability for training and matching the ***poor fingerprint*** images. It can be concluded that the accuracy of the system for three different cases is as follows:

(i) When spread factors (k_1 and k_2) are varied then learning rates (η_1 and $\eta_2 = 0.8$) and hidden units ($j = 9$) are held constant. In this condition, we observed from the experimental results shown in **Table-8.1** and **Fig.8.12** that the accuracy of the system increased up to k_1 **and** $k_2 =$ **0.6** and after that the accuracy of the system deceased with increase of k_1 and k_2.

(ii) When learning rates (η_1 and η_2) are varied then spread factors (k_1 **and** k_2 **=0.6**) and hidden units ($j = 9$) are held constant. In this condition, we observed from the experimental results shown in **Table-8.2** and **Fig.8.13** that the accuracy of the system increased up to η_1 **and** $\eta_2 =$ **0.8** and after that the accuracy of the system deceased with increase of η_1 and η_2.

(iii) When hidden units (j) are varied then spread factors (k_1 **and** k_2 = **0.6**) and learning rates (η_1 **and** η_2 = **0.8**) are held constant. In this condition, we observed result from the experimental findings shown in **Table-8.3** and **Fig.8.14** that the accuracy of the system increased up to *Hidden Units* (j) = **9** and after that the accuracy of the system deceased with increase of *Hidden Units* (j).

Several factors are responsible for getting correct result through neural computing techniques. The convergence of the solution depends heavily on initialization with random numbers and accuracy of the results depend on (i) spread factors (ii) learning rates, (iii) iterations and (iv) hidden units. Finally, it is concluded that the performance of the system for recognition of fingerprint using the *MDER-BP Algorithm* shows better efficiency with respect to *Adaptive Resonance Theory-2* (in chapter-6) for the *poor fingerprint images*.

9.3 Future Guideline

The fingerprint images possess different gray-level scale for a single person, because sometimes a person inked his/her finger with more ink or some times press his/her finger onto the paper slightly. This creates different quality of the fingerprint images with different gray-level scale. The orientation of the fingerprint is also a great problem to identify the exact person. In future the researchers can consider the orientation of the static images in a unified coordinated system of the oriented texture. It may be more authentic if core and delta features of fingerprints are considered to merge with GMF extraction technique. Off course, it is complicated to identify correctly from huge number of poor fingerprint images. The nearest neighbourhood method which has been derived in chapter-2 (section-2.5) can also be used by the researchers for person identification with good fingerprint images.

References

[1] D.Maltoni, D. Maio, A.K. Jain, S. Prabhakar, "Handbook of Fingerprint Recognition" Springer, New York, 2003.

[2] A. K. Jain, L. Hong, S. Pankanti, and Ruud Bolle, "an Identity authentication System Using Fingerprints," *Proceedings of the IEEE,* Vol. 85, No. 9, pp. 1365-1388, 1997.

[3] A. K. Jain, S. Prabhakar, and L. Hong, "A Multichannel Approach to Fingerprint Classification", *IEEE Trans. Pattern Ana. and Machine Intell.,* Vol. 21, No. 4, pp. 348-359, 1999.

[4] D. Maio and D. Maltoni, "Direct Gray-Scale Minutiae Detection in Fingerprints," *IEEE Trans. Pattern Anal. and Machine Intell.,* Vol. 19, No. 1, pp. 27-40, 1997.

[5] G. T. Candela, P. J. Grother, C. I. Watson, R. A. Wilkinson, and C. L. Wilson, "PCASYS: A Pattern-Level Classification Automation System for Fingerprints," *NIST Tech. Report NISTIR 5647,* August 1995.

[6] L. Hong and A. K. Jain, "Classification of Fingerprint Images," *11th Scandinavian Conference on Image Analysis,* June 7-11, Kangerlussuaq, Greenland, 1999.

[7] M. Eshera and K. S. Fu, "A Similarity Measure Between Attributed Relational Graphs for Images Analysis," in *Proc. 7th Int'l. Conf. Pattern Recognition,* Montreal, Canada, July 30-August 3, 1984.

[8] S. Gold and A. Rangarajan, "A Graduated Assignment Algorithm for Graph Matching," *IEEE Trans. Pattern Anal. Machine Intell.,* Vol. 18, No. 4, pp. 377-388, 1996.

[9] A. K. Hrechak and J. A. McHugh, "Automated Fingerprint Recognition Using Structural Matching," *Pattern Recognition,* Vol. 23, pp. 893-904, 1990.

[10] S. Prabhakar, "Fingerprint Classification and Matching Using a Filterbank", Ph.D. Thesis, Michigan State University, USA, 2001.

[11] A. Ranade and A. Rosenfeld, "Point Pattern Matching by Relaxation," *Pattern Recognition,* Vol. 12, No. 2, pp. 269-275, 1993.

[12] E. C. Driscoll, C. O. Martin, K. Ruby, J. J. Rissel, and J. G. Watson, "Method and Apparatus for Verification Identity Using Image Correlation," *US Patent No. 5067162,* 1991.

[13] A. sibbald, "Method and Apparatus for Fingerprint Characterization and Recognition Using Auto-correlation Pattern," *US Patent 5633947,* 1997.

[14] A. Senior, "A Hidden Markov Model Fingerprint Classifier," *Proceedings of the 31st Asilomar conference on Signals, systems and Computers,* pp. 306-310, 1997.

[15] L. O'Gorman, "Fingerprint Verification," In *Biometrics: Personal Identification in a Networked Society,* A. K. Jain, R. Bolle, and S. Pankanti (editors), Kluwer Academic Publishers, pp. 43-64, 1999.

[16] A. P. Fitz and R. J. Green, "Fingerprint Classification Using Hexagonal Fast Fourier Transform," *Pattern Recognition,* Vol. 29, No. 10, pp. 1587-1597, 1996.

[17] B. G. Sherlock and D. M. Monro, "A Model for Interpreting Fingerprint Topology," *Pattern Recognition,* Vol. 26, No. 7, pp. 1047-1055, 1993.

[18] M. Kawagoe and A. Tojo, "Fingerprint Pattern Classification," *Pattern Recognition,* Vol. 17, No. 3, pp. 295-303, 1984.

[19] A. Alamansa and L. Cohen, "Fingerprint Image Matching by Minimization of a Thin-Plate Energy Using a Two-Step Iterative Algorithm with Auxiliary Variables," *Workshop on the Application of Computer Vision,* palm Springs, California, December 4-6, 2000.

[20] Federal Bureau of Investigation. www.fbi.gov

[21] A. Lumini, D. Maio and D. Maltoni, "Continuous vs Exclusive Classification for Fingerprint Retrieval", Pattern Recognition Letters, Vol. 18, No. 10, pp. 1027-1034, October 1997.

[22] DigitalPersona White Paper, Guide To Fingerprint Identification, 2005.

http://www.comptalk.com/white_papers/Guide%20to%20Fingerprint%20Authentication.pdf

[23] J Edgar Hoover, Federal Bureau of Investigation, Department of Justice, Classification of Fingerprints, US Government Printing Office 1939.

[24] J. Osterburg, T. Parthasarathy, T. E. S. Raghavan, and S. L. Sclove, "Development of a Mathematical Formula for the Calculation of Fingerprint Probabilities Based on Individual Characteristics", Journal of the American Statistical Association, Vol. 72, No. 360, pp. 772-778, 1977.

[25] D. A. Stoney and J. I. Thornton, "A Critical Analysis of Quantitative Fingerprint Individuality Models", *Journal of Forensic Sciences,* Vol. 31, No. 4, pp. 1187-1216 Oct. 1986.

[26] A. R. Roddy and J. D. Stosz, "Fingerprint Features-Statistical Analysis and System Performance Estimates", Proc. IEEE, Vol. 85, No. 9, pp. 1390-1421, 1997.

[27] F. Galton, *Finger Prints,* London: McMillan, 1892.

[28] T. Roxburgh, "On Evidential Value of Fingerprints", Sankhya: Indian Journal of Statistics, Vol. 1, pp. 189-214, 1933.

[29] C. Kingston, "Probabilistic Analysis of Partial Fingerprint Patterns", *Ph.D Thesis,* University of California, Berkeley, 1964.

[30] K. Peason, "Galton's Work on Evidential Value of Fingerprints", *Sankhya: Indian Journal of Statistics,* Vol. 1, No. 50, 1933.

[31] E. R. Henry, Classification and Use of Fingerprints, London: Routledge, pp. 54-58, 1900.

[32] V. Balthazard, "De I'identification par les empreintes ditails", *Comptes Rendus des Academies des Sciences,* No. 152, Vol. 1862, 1911.

[33] B. Wentworth and H. H. Wilder, Personal Identification, R. G. Badger, Boston, 1918.

[34] H. Cummins and Charles Midlo, *Fingerprints, Palms and Soles: An Introduction to Dermatoglyphics.* Dover Publications, Inc., New York, 1961.

[35] S. R. Gupta, "Statistical Survey of Ridge Characteristics", *Int. Criminal Police Review,* Vol. 218, No. 130, 1968.

[36] L. Amy, "Recherches sur L'identification des Traces Papillaries", Annales de Medicine Legale, Vol. 28, No. 2, pp. 96-101, 1948.

[37] C. Champod and P. A. Margot, "Computer Assisted Analysis of Minutiae Occurrences on Fingerprints", *Proc. International Symposium on Fingerprint Detection and Identification,* J. Almog and E. Spinger, editors, Israel National Police, Jerusalem, pp. 305, 1996.

[38] S. L. Sclove, "The Occurrence of Fingerprint Characteristics as a Two Dimensional Process", *Journal of American Statistical Association,* Vol. 74, No. 367, pp. 588-595, 1979.

[39] D.A. Stoney, "A Quantitative Assessment of Fingerprint Individuality", University of California, Berkeley, Ph.D. Thesis, 1985.

[40] M. Trauring, "Automatic Comparison of Finger-ridge Patterns", *Nature, pp.* 938-940, 1963.

[41] S. B. Meagher, B. Buldowle, and D. ziesig, "50K Fingerprint Comparison Test", United States of America vs. Byron Mitchell, U.S. District Court Eastern

District of Philadelphia. Government Exhibits 6-8 and 6-9 in Daubert Hearing before Judge J. Curtis Joyner, July 8-9, 1999.

[42] M. R. Stiles, "Government's post-Daubert Hearing Memorandum," United States District Court for the Eastern District of Pennsylvania, USA vs Mitchell, Criminal case No. 96-00407, http://www.usao-edpa.com/Invest/Mitchell/704postd.htm, 2000.

[43] J. L. Wayman, "Daubert Hearing on Fingerprinting: When Bad Science Leads to Good Law: The Disturbing Irony of the Daubert Hearing in the Case of U. S. V. Byron C. Mitchell", http://www.engr.sisu.edu/biometrics/publications_daubert.html

[44] A. R. Rao, "A Taxonomy for Texture Description and Identification", Springer-Verlag, New York, 1990.

[45] A. C. Bovik, M. Clark, and W. S. Geisler, "Multichannel Texture Analysis Using Localized Spatial Filters," *IEEE Trans. Pattern Anal, and Machine Intell,* Vol. 12, No. 1, pp. 55-73, January 1990.

[46] A. K. Jain and F. Farrokhnia, "Unsupervised Texture Segmentation Using Gabor Filters, "*Pattern Recognition*, Vol. 24, No. 12, pp. 1167-1186, 1991.

[47] D. A. Stoney, "Distribution of Epidermal Ridge Minutiae," *American Journal of Physical Anthropology,* Vol. 77. pp. 367-376, 1988.

[48] J. G. Daugman, "High Confidence Recognition of Persons by a Test of Statistical Independence," *IEEE Trans. Pattern Anal. and Machine Intell.,* Vol. 15, No. 11, pp. 1148-1161, 1993.

[49] J. G. Daugman, "Uncertainty Relation for Resolution in Space, Spatial Frequency, and Orientation Optimized by Two-Dimensional Visual Cortical Filters," *J. Opt. Socamer. A,* Vol. 2, pp. 1160, 1985.

[50] A. K. Jain and F. Farrokhnia, "Unsupervised Texture Segmentation Using Gabor Filters, "*Pattern Recognition*, Vol. 24, No. 12, pp. 1167-1186, 1991.

[51] Gerik Alexander Von Graevenitz, "Introduction to Fingerprint Technology", Bergdata Biometrics GmbH, Bonn, Germany.

[52] Gonzalez R.C. & Woods R.E., "Digital Image Processing" Second Edition, PP-116-134, 534-572, ISBN:81-7808-629-8, Pearson Education-2002.

[53] **A. K. M. Akhtar Hossain** and **S.K. Ahmed Kamal**, "Fingerprint Verification System Using Minimum Distance Error Rate BackPropagation Algorithm", Scientific Journal founded in 1997, Bulletin of Donetsk National University, 004.932.721, ISSN 1817-2237, Ukraine, PP-439-444, **1/2005.**

[54] ***A.K.M. Akhtar Hossain***, "Pattern Classification by Neural Networks", Dept. of Applied Physics and electronics, University of Rajshahi, Bangladesh, M.Sc. Thesis, June, 1995.

[55] J.T. Tou and R.C. "Gonzalez, "Pattern Recognition principle", pp 5-215.

[56] Earl Gose, Richard Johnsonbaugh and Steve Jost, "Pattern Recognition and Image Analusis", Prentice Hall of India Private Limited, New Delhi, ISBN:-81-203-1484-0, 2000.

[57] Sergios Theodoridis and Konstantinos Koutroumbas, "Pattern Recognition", Elsevier academic press, USA, pp-35-150, ISBN:0-12-685875-6, 2003.

[58] S. Rajasekaran and G.A. Vijayalakashmi Pai, "Neural Networks, Fuzzy Logic, and Genetic Algorithms Synthesis and Applications", Prentice Hall of India Private Limited, New Delhi, ISBN:81-203-2186-3, 2003.

[59] J. A. Freeman and D. M. Skapura, "Neural Networks", Addison-Wesley Longman Inc, California, ISBN:0-201-52444-9, pp.89-124, 1991, First ISE Reprint 1999.

[60] R. Beale and T. Jacson, "Neural Computing: An Introduction", PP-68-75, IOP Publishing Ltd. ISBN: 0-85274-262-2, 1990.

[61] James L. Noyes, "Artificial Intelligence with Common Lisp, Fundamentals of Symbolic and Numeric Processing", PP-4-15, 418-453, Galgotia publications (P) Ltd. 1993.

[62] Bart Kosko, "Neural Networks and fuzzy systems, A Dynamic systems Approach to matching Intelligence", PP-111-220, Prentice Hall of India Private Limited, New Delhi, ISBN:81-203-0868-9, 2003.

[63] L. O'Gorman, "Fingerprint Verification," In *Biometrics: Personal Identification in a Networked Society,* A. K. Jain, R. Bolle, and S. Pankanti (editors), Kluwer Academic Publishers, pp. 43-64, 1999.

[64] N. Ratha, K. Karu, S. Chen, and A. K. Jain, "A Real-Time Matching System for Large Fingerprint Databases," *IEEE Trans. Pattern Anal. Machine Intell.,* Vol. 18, No. 8, pp. 799-813, 1996.

[65] X. Jiang, W. Y. Yau and W. Ser, "Minutie Extraction by adaptive Tracing the Gray Level Ridge of the Fingerprint Image", *IEEE International Conference on Image Processing,* Japan, 1999.

[66] X. Jaing, W. Y. Yau, "Fingerprint Minutiae Matching Based on the Local and Global Structures," *Proc. 15th International Conference on Pattern Recognition,* Vol. 2, pp. 10421045, Barcelona, Spain, September 2000.

[67] S. Chen and A. K.Jain, "A Fingerprint Matching Algorithm Using Dynamic Programming", *Technical Report,* Department of Computer Science and Engineering, Michigan State University, 1999.

[68] S. B. Meagher, B. Buldowle, and D. ziesig, "50K Fingerprint Comparison Test", United States of America vs. Byron Mitchell, U.S. District Court Eastern District of Philadelphia. Government Exhibits 6-8 and 6-9 in Daubert Hearing before Judge J. Curtis Joyner, July 8-9, 1999.

[69] M. R. Stiles, "Government's post-Daubert Hearing Memorandum," United States District Court for the Eastern District of Pennsylvania, USA vs Mitchell,

Criminal case No. 96-00407, http://www.usao-edpa.com/ Invest/Mitchell/704postd.htm, 2000.

[70] N. Duta, A. K. Jain and M-P Dubuisson-Jolly, "Automatic Construction of 2D Shape Models", *IEEE Trans. Patt Anal. and Machine Intell.* Vol. 23, No. 5, May 2001.

[71] N. L. Segal, *Entwined Lives: Twins and What They Us About Human Behavior,* Plume, New York, 2000.

[72] N. Ratha, J. H. Connell, and R. M. Bolle, "An Analysis of Minutiae Matching Strength", *Proc. 3rd International Conference on Audio- and Video- Based Biometric Person Authentication,* pp. 223-228, Sweden, June 6-8, 2001.

[73] M. R. Verma, A. K. Majumdar, and B. Chatterjee, "Edge Detection in Fingerprints," *Pattern Recognition,* Vol. 20, No. 5, pp. 513-523, 1987.

[74] R. Cappelli, A. Erol, D. Maio, and D. Maltoni, "Synthetic Fingerprint-image Generation", Proc. International Conference on Pattern Recognition (ICPR), Barcelona, Vol. 3, pp. 475-478, September 2000.

[75] R. Cappelli, D. Maio, and D. Maltoni, "Fingerprint Classification based on Multi-space KL", *Proc. Working on Automatic Identification Advances Technologies (AutoID'99),* Summit (NJ), pp. 117-120, October 1999.

[76] R. Cappelli, D. Maio and D. Maltoni, "Modelling Plastic Distortion in Fingerprint Images", Proc. Second International Conference on Advances in Pattern Recognition (ICAPR2001), Rio de Janeiro, pp. 369-376, March 2001.

[77] R. Collobert and S. Bengio, "SVMTorch: Support Vector Machines for LargeScale Regression Problems", *Journal of Machine Learning Research,* Vol. 1, pp. 143-160, 2001.

[78] R. O. Dyda, P. E. Hart, and D. G. Stock, *Pattern Classification,* 2nd Edition, John Wiley & Sons, November 2000.

[79] S. A. Cole, "What Counts for Identity?" *Fingerprint Whorld,* Vol. 27, No. 103, pp. 7-35, 2001.

[80] S. Prabhakar and A. K. Jain, "Decision-level Fusion in Fingerprint Verification" to appear in *Pattern Recognition,* 2001.

[81] S. Prabhakar, A. K. Jain, J. Wang, S. Pankanti, and R. Bolle, :Minutiae Verification and Classification for Fingerprint Matching", *Proc. 15th International Conference on Pattern Recognition (ICPR),* Vol. I, pp. 25-29, Barcelona, September 3-8, 2000.

[82] X. Jaing, W. Y. Yau, "Fingerprint Minutiae Matching Based on the Local and Global Structures," *Proc. 15th International Conference on Pattern Recognition,* Vol. 2, pp. 10421045, Barcelona, Spain, September 2000.

[83] Y. Yao, P. Frasconi, and M. Pontil, "Fingerprint Classification with Combination of Support Vector Machines", *Proc. 3rd International Conference on Audio- and Video-Based Biometric Person Authentication,* pp. 253-258, Sweden, June 6-8, 2001.

[84] ***A. K. M. Akhtar Hossain*** and **S. K. Ahmed Kamal**, "Grid Mapping *Features based Fingerprint Verification System*" Proceedings of 8th International Conference on Computer and Information Technology, Islamic University of Technology (IUT), Gazipur, Dhaka, Bangladesh, PP-1052-1057, ISBN 984-32-2873-1, **ICCIT-2005.**

[85] http://www.biometrika.it/eng/wp_fingintro.html

[86] http://www.biometrika.it/eng/wp_biointro.html

[87] http://www.newscientist.com/lastword/article.jsp?id=lw474

[88] http://www.prip.tuwien.ac.at/~hanbury/praktika

[89] **A. K. M. Akhtar Hossain & S.K. Ahmed Kamal**, "An Approach To Extract Fingerprint Feature Using Grid-Mapping Technique And To Match Through BackPropagation Neural Network", Scientific Journal founded in 1997, Bulletin of Donetsk National University, 004.932.721, ISSN 1817-2237, PP-321-328, Ukraine, **2/2005**.

[90] ***A. K. M. Akhtar Hossain*** and **S. K. Ahmed Kamal**, "*Minutiae Features based Fingerprint Verification System using BackPropagation Neural Network*", Rajshahi University Studies, Journal of Science, Part-B, Bangladesh, ISSN 1681-0708, **2005**.

[91] Rumelhart, D.E., G.R. Hinton and R.J. Williams (**1986**), Learning Tnternal Representations by Error Propagation, in Parallel Distributed Processing, Vol. 1, Rumelhart, D.E., J.L. Eds. MIT Press Cambridge MA.

Rajshahi University Library
Documentation Section
Document No. D - 2513
Date 24/5/06